A POPULATION-BASED CASE-CONTROL STUDY OF RISK FACTORS FOR CONNECTIVE TISSUE DISEASES

by

William Baldwin Teel

A dissertation submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

University of Washington

1997

Approved by _____
Chairperson of Supervisory Committee

_____

_____

_____

_____

Program Authorized to Offer Degree Epidemiology

Date 7/18/97

Doctoral Dissertation

In presenting this dissertation in partial fulfillment of the requirements for the Doctoral degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of this dissertation is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for copying or reproduction of this dissertation may be referred to University Microfilms, 1490 Eisenhower Place, P.O. Box 975, Ann Arbor, MI 48106, to whom the author has granted "the right to reproduce and sell (a) copies of the manuscript in microform and/or (b) printed copies of the manuscript made from microform."

Signature 

Date 7/25/97

University of Washington

Abstract

A POPULATION-BASED CASE-CONTROL STUDY OF RISK FACTORS FOR CONNECTIVE TISSUE DISEASES

by William Baldwin Teel

Chairperson of the Supervisory Committee
Professor Thomas D. Koepsell, M.D., M.P.H.
Department of Epidemiology

This dissertation describes the results of a population-based case-control study of risk factors for five connective tissue diseases: systemic lupus erythematosus, Sjogren's syndrome, systemic sclerosis / CREST syndrome, polymyositis / dermatomyositis and mixed connective tissue disease.

Cases were female residents of King County, Washington who were diagnosed by a rheumatologist between January 1, 1983 and December 31, 1991 as having a disease of interest. Two control groups were used. The 'new' control group was identified by random digit telephone dialing and was frequency matched to the case group on age at diagnosis and calendar year of diagnosis. Mailed questionnaires were used for concurrent data collection on the cases and on the 'new' control group, with a telephone administered questionnaire for non-respondents. The 'pre-existing' control group had been used for three other epidemiologic studies, and was identified and had in-person interviews before 1992, when mass media attention to the breast implant issue intensified.

Of the 532 women identified with a connective tissue disease, 427 (80.3%) returned a questionnaire or completed a telephone interview. The response rate among 'new' controls was 74.3% and 79.0% among the 'pre-existing' controls. Six cases (1.4%) had breast implants prior to their CTD diagnosis: 4/161 (2.5%) of cases with Sjogren's

syndrome; 2/191 (1.1%) with systemic lupus erythematosus; 0/55 with progressive systemic sclerosis or CREST syndrome; 0/17 with polymyositis or dermatomyositis; and 0/3 with mixed connective tissue disease. In the 'new' control group 24/1577 (1.5%) women reported having implants prior to their reference date, as did 16/1672 (1.0%) women in the 'pre-existing' control group. Using the 'new' control group, after adjusting for age, calendar year, and race (white, black, other), the odds ratio for any CTD was 0.9 (95% CI: 0.4, 2.3), for systemic lupus erythematosus it was 0.8 (95% CI: 0.2, 3.4), and for Sjogren's syndrome it was 1.6, (95% CI: 0.5, 4.7). Using the 'pre-existing' control group, and with similar adjustment, for any CTD the odds ratio was 1.3 (95%CI: 0.4, 4.1), for systemic lupus erythematosus it was 1.6 (95% CI: 0.3, 7.3), and for Sjogren's syndrome the odds ratio was 1.8 (95% CI: 0.4, 7.8).

We found little association between breast implants and the connective tissue diseases studied

TABLE OF CONTENTS

LIST OF TABLES

ACKNOWLEDGMENTS

The author wishes to thank and acknowledge the kind assistance of a plethora of people that helped and / or guided me in the course of this research.

My dissertation committee for guidance, constructive criticism, and generally making the process of research actually interesting and fun: Thomas Koepsell,M.D., MPH, Janet Daling, Ph.D., Carin Dugowson, M.D., MPH, Bruce Psaty, Ph.D., M.D., and Richard Kronmall, Ph.D.

Lynda Voigt, Ph.D. for assistance and consulting on random digit dialing and frequency matching.

Dr. Don Martin, Ph.D. for assistance with the frequency matching algorithm. Ms. Milda Tautvydas, M.F.A. and Ms. Cheri Anderson for research assistance.

This study could not have happened without the willing and helpful cooperation of the area rheumatologists, their staff, and their associated institutions. We thank: Dr. John Baldwin, Dr. Carl Brodie, Dr. Paul Brown, Dr. Patrick Campbell, Dr. Jeffrey Carlin, Dr. Daniel Furst, Dr. Gregory Gardner, Dr. Joyce Gauthier, Dr. Bruce Gilliland, Dr. Jan Hilson, Dr. Richard Jimenez, Dr. Christi Kenyon, Dr. Mart Mannik, Dr. Philip Mease, Dr. Peter Mohai, Dr. Bernard Mullen, Dr. Richard Neiman, Dr. J. Lee Nelson, Dr. Ina Oppliger, Dr. Maurice Skeith, Dr. Andrew Solomon, Dr. David Stage, Dr. Gordon Starkebaum, Dr. Giancarlo Stone, Dr. Mark Wener, Dr. Peter Simkin, Dr. Wayne Tsuji, Dr. Pamela Sheets, Dr. Steven Overman, Dr. Scott Pollock, Dr. Kenneth Wilske, Dr. Eric Sasso, Dr. Robert Willkens, Dr. Wayne Wallis, Dr. Pedro Trujillo.

This study was funded under a contract between the Dow Corning Corporation and the University of Washington. Under the contract provisions, the University retained full

control over the research design, rights to the study data, and unrestricted right to publish study results regardless of the findings.

DEDICATION

The author wishes to dedicate this dissertation to:

My parents Ann and Pat Teel, although they are not quite sure why I moved from the computer field to epidemiology; their loan at a critical time was quite useful, allowing me to continue working.

Professor Thomas D. Koepsell, M.D.,MPH who has provided me an incredible amount of guidance, support, encouragement, and a model of calmness throughout this research far in excess of what I ever envisioned or ever heard of as the responsibility of a dissertation chairman.

INTRODUCTION

This study was designed and conducted to investigate if breast implants are a risk factor for five connective tissue diseases: systemic lupus erythematosus, Sjogren's syndrome, systemic sclerosis / CREST syndrome, polymyositis / dermatomyositis and mixed connective tissue disease.

Secondary objectives were to evaluate if exogenous female hormones, hair dyes and other exposures are risk factors for the connective tissue diseases of interest in this study. The third objective was to obtain an estimate of the incidence rates for the connective tissue diseases studied.

CHAPTER 1: A POPULATION-BASED CASE-CONTROL STUDY OF BREAST IMPLANTS AS A RISK FACTOR FOR CONNECTIVE TISSUE DISEASES IN KING COUNTY, WASHINGTON.

Summary

Background. Although silicone breast implants were removed from the market in 1992, controversy about their long-term health effects persists. We conducted a population-based case-control study to investigate the association between the risk of connective tissue diseases and the use breast implants in women.

Methods. Cases were female residents of King County, Washington who were diagnosed by a rheumatologist between January 1, 1983 and December 31, 1991 as having one of five connective tissue diseases (CTDs): systemic lupus erythematosus, progressive systemic sclerosis / CREST syndrome, Sjogren's syndrome, polymyositis / dermatomyositis, or mixed connective tissue disease. Two control groups were used. The 'new' control group was identified by random digit telephone dialing and was frequency matched to the case group on age at diagnosis and calendar year of diagnosis. Mailed questionnaires were used for concurrent data collection on the cases and on the 'new' control group, with a telephone administered follow-up questionnaire for non-respondents. The 'pre-existing' control group had been used for three other epidemiologic studies. This control group was identified and had in-person interviews before 1992, when mass media attention to the breast implant issue intensified.

Results. Of the 532 women identified with a connective tissue disease, 427 (80.3%) returned a questionnaire or completed a telephone interview. The response rate among 'new' controls was 74.3% and 79.0% among the 'pre-existing' controls. Six cases (1.4%) had breast implants prior to their CTD diagnosis: 4/161 (2.5%) of cases with Sjogren's syndrome; 2/191 (1.1%) with systemic lupus erythematosus; 0/55 with progressive systemic sclerosis or CREST syndrome; 0/17 with polymyositis or dermatomyositis; and 0/3 with mixed connective tissue disease. In the 'new' control group 24/1577 (1.5%) women reported having implants prior to their reference date, as did 16/1672 (1.0%)

women in the 'pre-existing' control group. Using the 'new' control group, after adjusting for age, calendar year, and race (white, black, other), the odds ratio for any CTD was 0.9 (95% CI: 0.4, 2.3), for systemic lupus erythematosus it was 0.8 (95% CI: 0.2, 3.4), and for Sjogren's syndrome it was 1.6, (95% CI: 0.5, 4.7). Using the 'pre-existing' control group, and with similar adjustment, for any CTD the odds ratio was 1.3 (95%CI: 0.4, 4.1), for systemic lupus erythematosus it was 1.6 (95% CI: 0.3, 7.3), and for Sjogren's syndrome the odds ratio was 1.8 (95% CI: 0.4, 7.8).

Conclusions. We found little association between breast implants and the connective tissue diseases studied. We could not rule out a weak effect of breast implants on risk of these diseases or an effect on other CTDs.

Keywords: breast implants, epidemiology, case-control study, systemic lupus erythematosus, progressive systemic sclerosis, CREST, Sjogren's syndrome, polymyositis, dermatomyositis.

Background

There is a long history of attempts to enlarge or reconstruct the female breast artificially[1], culminating in 1962 with the invention of the silicone-elastomer envelope containing silicone gel. Since then, it has been estimated between 850,000 to 1.5 million women in the United States have received breast implants for either post-mastectomy reconstruction or cosmetic augmentation[2,3]. Since 1967, there have been over 150 case reports describing adverse effects in women with breast implants[4] including capsular contracture, mastitis, breast cancer, and a host of connective tissue diseases. Several studies suggest that the human body forms a capsule to isolate the implant from the surrounding breast tissue[5]. There is conflicting evidence that antibodies to silicone and / or silicone elastomers are produced by the human immune system[6,7,8].

In January, 1992, the United States Food and Drug Administration while noting the scarcity of evidence from any analytic epidemiologic studies, called for a moratorium on the use of silicone breast implants for the US market[3], until new information on the safety of silicone breast implants could be reviewed. Extensive media attention and

litigation followed. Since then, fifteen epidemiologic studies[9-23] have been published. With one exception, no study has found a significant association between breast implants and an increased risk for well-defined connective tissue diseases. Most have not been population-based, had only small sample size, or focused only on one disease, or had a poor response rate. The study[23] with a response rate of 25% used 'self-report' of diseases found a significant small increase for any connective tissue disease that was driven by a category of "other CTD including mixed". We conducted a population-based case-control study to investigate whether women with breast implants were more likely than women without breast implants to develop one of several connective tissue diseases -- systemic lupus erythematosus, progressive systemic sclerosis or CREST syndrome, Sjogren's syndrome, polymyositis / dermatomyositis, and mixed connective tissue disease.

Methods

Study Setting

King County, which includes the city of Seattle, is the largest metropolitan area in Washington state and has a population of approximately 1.5 million people[24]. It is bounded on the west by Puget Sound and on the east by the Cascade mountains.

Identification of Cases and Control Subjects

We attempted to identify all cases of the connective tissue diseases of interest in women who were residents of King County at the time of their initial diagnosis between January 1, 1983 and December 31, 1991. The start date of the study period was selected based on the lack of controls prior to 1983 in the 'pre-existing' control group, as well as the unavailability of older archival records at certain rheumatologic practices. The end date was chosen to minimize potential selection bias due to the extensive media attention around the breast implant issue. After December, 1991, women with breast implants who had symptoms of CTDs might differentially have sought medical attention compared to women with the same symptoms who did not have breast implants.

All rheumatologists, including those whose practices bordered the north and the south of King County, were contacted and invited to participate in the study. Six of the 42 rheumatologists declined; two estimated that less than 3% of their practices populations lived in King County, and had no way other than manual chart review to identify the King County patients. Three other rheumatologists had small practices or had been in practice only during the last few years of the study period. For each practice, the best method to identify cases of interest was determined in consultation with the rheumatologist and staff. The methods included analysis of computer billing records, review of microfiche billing records, manually reviewing every active chart and every chart in archival storage, and combinations of the various methods.

Cases were screened for eligibility through medical record review by trained research assistants who used a standardized abstraction form. Case definitions followed American College of Rheumatology criteria[25,26,27] for systemic lupus erythematosus, Sjogren's syndrome, and progressive systemic sclerosis. For mixed connective tissue disease and polymyositis / dermatomyositis, criteria from the literature[28,29] were used. Two levels of diagnostic certainty were used. Definite cases met all disease criteria and had been so diagnosed by the rheumatologist. For probable cases, either the physician made a specific diagnosis but the case fell one feature short of the requisite number of disease criteria, or the disease criteria were met, but the rheumatologist only called the disease "probable". Over 96% of all provisionally qualifying cases were re-abstracted by the first author to confirm study eligibility.

We utilized two control groups of King County women. The 'pre-existing' control group had been identified by random-digit dialing for three other epidemiologic studies of diseases of women[30,31,32] and interviewed before the media attention to the breast implant issue. The 'pre-existing' control group was interviewed between 1986 and 1991 and had reference dates based upon the designs of the three studies[30,31,32] for which these controls had originally been identified. Controls with reference years in 1991 were excluded because selection criteria in that year had included not having had a

hysterectomy. In addition, there were no 'pre-existing' controls for women over age 40 years in 1983, over age 45 years in 1984, and over age 65 years in 1985 and 1986.

The 'new' control group was also selected by random digit dialing in 1994 - 1996. Random telephone numbers were generated using unrestricted random sampling of all working non-cellular prefixes in King County and adding a random 4-digit number to create a telephone number.[33] Each number was called at least 12 times at different times of the day and week over a three-week period before it was abandoned. A total of 19,000 telephone numbers were called using a two-step method of recruitment. Women were screened for King County residence and age, and if qualified were invited to participate in the study. If they agreed to participate, the second step involved frequency matching the potential controls to the distribution of age (in five year categories) and year of diagnosis in the case group. A computer program randomly selected a subset of the potential controls and assigned a randomly generated reference date to be similar to the distribution of diagnosis dates among comparably aged cases.

There were 532 cases who met study disease criteria as definite or probable CTD case. Nine cases (1.7%) were not mailed study materials because their rheumatologist refused permission or advised against contacting these patients. Of the remaining 523 women, 427 (81.6%) women returned a questionnaire or participated in a telephone interview. Forty-seven women (8.8%) refused to participate, 31 (5.8%) were not locatable, and 18 (3.4%) did not participate for other reasons, which included 3 cases with hearing problems, 7 deceased women with no identifiable next-of-kin, 3 whose physician did not give permission for a telephone interview, and 5 women who were too ill to participate.

In the course of telephone sampling to identify the 'new' control group, 86.1% of households provided enough screening information to determine whether the household included a potentially eligible woman, and 1,702 (86.4%) of the 1,971 women who were invited to participate agreed to do so , yielding an overall response rate of

74.4%. Among controls selected to participate, 125 were later eliminated from analysis when they returned a questionnaire that indicated either they had not been King County residents at their reference dates, or had had a CTD diagnosis before their reference date. In the pre-existing control group the overall response rate was 79.0%.

Data Collection

Data were collected by mailed self-administered questionnaire for both the case and 'new' control groups. The survey included questions on marital status and sociodemographic characteristics, diabetes, reproductive history, breast cancer, breast implants, hair dye, and work history. Three repeat mailings to non-respondents were used to increase response rate. If no response was received after 4 mailings, a telephone interview was sought, using the same questions. All cases and 'new' controls who reported breast implants were mailed an additional two-page questionnaire asking for additional details about their breast implant history, including manufacturer, type of implant, adverse reactions, subsequent breast implant surgery. The 'pre-existing' control women were also asked if they had ever been diagnosed with a connective tissue disease of interest. Data on implants in this control group were obtained during an in-person interview. Those controls in either group who stated that they had had a CTD diagnosis before their reference date were excluded.

In addition, a small study was undertaken to ascertain whether there might be underreporting of breast implants by control study participants. One hundred women who had had breast implant-related surgical procedures (CPT codes[34]: 19325,19340, 19370, 19371, 19396) at the University of Washington (UW) were randomly selected and contacted as though they were potential controls. Due to restrictions imposed by available record information, the sub-study only had access to phone numbers and medical record numbers, but not to names or other identifiers on these women. When these numbers were called, 52 age eligible women associated with the 100 phone numbers agreed to participate. For the other 48 phone numbers, 7 were business numbers, 12 were disconnected, 12 were refusals, 2 did not speak English, 8 had no

females, 3 were computer or fax lines, and 4 were answering machines. We were unable to verify whether women who declined to participate were the same women who had had breast implant surgery. Of the 52 women who agreed to participate at the phone screen, 48 (92.3%) returned a completed questionnaire or completed a phone interview, of whom 38 indicated that they had had breast implants. We later sought to review UW medical records for the 10 women who returned a questionnaire but indicated that they did not have breast implants. None had UW medical charts, but 2 of the women did have daughters with UW medical charts, and their daughters had had breast implant surgery at an age less than 18. We also attempted to review medical charts of the four women that had initially agreed to participate but never returned a questionnaire, and/or declined to participate in the telephone interview. Only one woman had an UW medical chart, and this woman did have breast implants. We concluded from this sub-study that most women with breast implants and without CTDs are likely to report their breast implant status in a telephone interview or mailed questionnaire.

All data were double entered. Data analysis utilized logistic and polychotomous logistic regression (STATA 5.0.[35]) The odds ratio was used to estimate the relative risk[39].

All instruments and methods were approved by multiple human subject institutional review boards, including those of the University of Washington, Virginia Mason Medical Center, and Group Health Cooperative.

Results

The prevalence of breast implants in the case and control groups is shown in Table 1. In the 'new' control group, 24 women (1.5%) reported having had breast implants placed before their reference date, as did 16 women (1.0%) in the 'pre-existing' control group. The prevalence of implants in the two control groups was similar and not statistically significantly different with or without adjustment for age and reference year. Among the cases, 4 women (2.5%) with Sjogren's syndrome indicated that they had had breast implants before their date of diagnosis, as did 2 women (1.1%) with systemic lupus

erythematosus. None of the cases with other CTDs reported having had breast implants.

Table 2 compares selected characteristics in the three study groups. For most of the selected characteristics the controls and cases were fairly similar. The main difference was that cases of systemic lupus erythematosus were more likely to be black. Race for this reason was controlled for in all analyses.

Because no cases of progressive systemic sclerosis, polymyositis / dermatomyositis, or mixed connective tissue disease reported breast implants, these diagnoses were not examined individually in subsequent analyses, although they were included as part of the 'any CTD' category. Table 3 shows that, using the 'new' control group as the comparison group, adjusted for age, reference year, and race (black, white, other), the odds ratio and 95% confidence interval for any CTD was 0.9 (0.4, 2.3), for systemic lupus erythematosus it was 0.8 (0.2, 3.4), and for Sjogren's syndrome it was 1.6 (0.5, 4.7). Using the 'pre-existing' control group, the odds ratio and 95% confidence interval for any CTD was 1.3 (0.4, 4.1), for systemic lupus erythematosus it was 1.6 (0.3, 7.3), and for Sjogren's syndrome it was 1.8 (0.4, 7.8). We also examined the corresponding odds ratios among only cases that had a definite diagnosis of CTDs. Most odds ratios decreased slightly, with wider confidence intervals that uniformly included 1.0.

Cases who had resided in King County at time of diagnosis were included regardless of current residence in King County. Controls were also required to be residents of King County as of their reference dates, but also had to reside in the county at the time of the telephone sampling in order to be included. To determine whether long-term residence in King County might confound the results, we also analyzed the results when restricted to women who resided in King County both at reference date and on the date when they reported exposure information. The results of this analysis were similar to those shown in Table 3. Using the 'new' controls as the comparison group, the adjusted odds ratios and 95% confidence intervals for any definite or probable CTD 1.1 (0.4, 2.8), for

systemic lupus erythematosus it was 1.0 (0.2, 4.4) and for Sjogren's syndrome it was 1.6 (0.6, 5.3). When restricted to cases with definite CTDs the adjusted odds ratios and 95% confidence interval for any CTD was 0.9 (0.3, 2.7), for systemic lupus erythematosus it was 0.6 (0.1, 4.5), and for Sjogren's syndrome it was 1.6 (0.5, 5.4).

Discussion

The results of this study provide little support for the hypothesis that breast implants place a woman at increased risk of any of the five connective tissue diseases studied. Because of the rarity of breast implants in both cases and controls, it was not possible to rule out a small increase. These results are, however, consistent with 15 other recent epidemiologic studies of well defined CTDs[9-23]. Only one (a large cohort study[23]) has found a significantly increased risk for any CTD. The estimated relative risk in that study was small (1.24 95% CI: 1.08, 1.41) and was driven by a category of self-reported "other CTD including mixed CTD". In addition, the study had only a 25% response rate, opening the possibility of selection bias. Three meta-analyses[36-38] that combined risk estimates from the breast implant case-control and cohort studies reached similar conclusions.

The present study used a case-control design, because the diseases of interest are so rare. Availability of the 'pre-existing' control group was fortuitous, because these women had been interviewed before extensive media attention about the breast-implant issue began in 1992. Comparing them to the 'new' control group, which was surveyed in 1994-96, showed that the differences in self reported prevalence of breast implants before and after 1992 were small and well within the range expected due to chance. The 'new' control group allowed data collection methods and the time interval between reference date and survey date to be the same for cases and controls. For this reason, we regard the 'new' control group findings as the primary findings of this study, although the similarity in findings between the two control groups provides a form of internal replication.

As in any epidemiologic study the results need to be viewed in terms of chance, selection bias, measurement error and confounding as possible alternative explanations[39]. When evaluating the role that chance might play in this study, there is a possibility that a real difference exists in CTD risk in relation to breast implants, but escaped detection. Given the rarity of these diseases and of exposure to breast implants in the general population, based on the 95% confidence limits, this study could not rule out an increased odds ratio up to 2.3 or 4.2 for any CTD, depending upon which control group is used.

Selection bias could result if there were differential participation of cases or controls on the basis of exposure status. The incidence of the CTDs studied in this setting was similar to published values in other populations[40,41]. In the case group, the medical records of all potential cases were reviewed to make sure they met the criteria for disease status, year of diagnosis, age, and King County residency, irrespective of breast implant status. Similarly, the controls were selected as potential participants without knowledge of their breast implant status. The same questionnaires were used for both the cases and the 'new' controls, and a subset of the questions had been used in the in-person interview with the 'pre-existing' controls. Recall bias could result if there were differential reporting of exposure in the case or control group. The sub-study suggested reasonably complete reporting among controls whose exposure status was known from other sources. Any increased reporting of implants among cases would tend to increase the estimate of the odds ratio, yet it was close to 1.0. The overall response rate was similar in the case group (80%) and the two control groups: (74%) overall for the 'new' controls and 79% for the 'preexisting' controls. It is nonetheless possible that cases with breast implants were more or less likely to participate.

The role of confounding due to unmeasured covariates is another possible source of bias in any observational epidemiologic study. This study controlled for age and reference year, which were frequency matched between the case and new control groups. In addition it controlled for race, since black and Asian women were less likely

to obtain implants and more likely to have a CTD. Forty additional covariates were investigated, and none was found to change the estimate of the breast implant odds ratio importantly. Other covariates that are now known or suspected to be associated with breast implants were not measured in this study. These include the number of sexual partners, alcohol consumption and smoking.[42] It is known that estrogen replacement therapy (ERT) is a risk factor for SLE[43], but controlling for ERT, or oral contraceptive use in the analysis did not change the breast implant odds ratio.

It is known that the human body can form antibodies to silicone or silicone elastomers and forms a capsule around implants as a foreign body reaction causing capsular contracture.[4,44-47] We also looked for an increased risk of CTDs in relation to silicone exposure, from syringe lubricants in insulin-dependent diabetics and did not find one.

In conclusion, the results of this study do not support a significantly increased risk for the five well-defined connective tissue diseases studied among women who had been exposed to breast implants. Since the prevalence of breast implants is so low, it was not possible to rule out a small increase in CTD risk. Our findings also do not exclude the possibility that a different CTD syndrome not fitting any of the diagnostic criteria we used may be associated with breast implants.

CHAPTER 2: A POPULATION-BASED CASE-CONTROL STUDY OF EXOGENOUS FEMALE HORMONES AS RISK FACTOR FOR SYSTEMIC LUPUS ERYTHEMATOSUS IN KING COUNTY, WASHINGTON

Summary

Background. The incidence of systemic lupus erythematosus (SLE) is higher for women than men; sex hormones maybe related to the etiology of this disease. A recent study found hormone replacement therapy to be associated with an increased risk of developing SLE. We conducted a population-based case-control study to investigate the use of exogenous female hormones as a potential risk factor for SLE.

Methods. Cases were female residents of King County, Washington who were first diagnosed between January 1, 1983 and December 31, 1991 by a rheumatologist as having systemic lupus erythematosus. Two control groups were used. The 'new' control group was identified via random digit dialing and was frequency matched to the case group on calendar year and age at diagnosis. Mailed questionnaires were used for concurrent data collection on the cases and on the 'new' control group, with telephone follow-up for non-respondents. The 'pre-existing' control group included King County women who had been interviewed in person for three other epidemiologic studies.

Results. Of the 252 SLE cases identified, 191 (75.8%) returned questionnaires or completed telephone interviews. The response rate among the 'new' controls was 74.3% and 79% among the 'pre-existing' controls. Using the 'new' control group, after adjusting for age, calendar year, race, and past oral contraceptive use, the odds ratio (95% confidence interval) for estrogen replacement therapy in postmenopausal women was 1.30, (0.73, 2.13). Using the 'pre-existing' control group and the same adjustments the odds ratio was 0.97 (0.49, 1.94). Using the 'new' control group, after adjusting for age, calendar year, and race the odds ratio for ever use of oral contraceptives in premenopausal women was 1.51 (0.98, 2.36), for past use it was 1.71 (1.08, 2.71), and for current use it was 0.66 (0.33, 1.32). Using the 'preexisting' control group and the

same adjustments, the odds ratio for ever use of oral contraceptives was 0.57 (0.42, 0.76), for past use it was 0.66 (0.37, 1.17), and for current use it was 0.34 (0.14, 0.80).

Conclusions. We found little association between estrogen replacement therapy in postmenopausal women and the risk for developing systemic lupus erythematosus. In premenopausal women, current use of oral contraceptives may reduce the risk of SLE. Whether SLE risk is increased among past oral contraceptive users is unclear.

Keywords: exogenous female hormones, estrogen replacement therapy, oral contraceptives, epidemiology, case-control study, systemic lupus erythematosus.

Background

The etiology of systemic lupus erythematosus (SLE), an autoimmune disease, is largely unknown. A number of studies suggest that women are at increased risk of developing SLE as the incidence of SLE in women of child bearing age is some 6-10 times higher than that of men of comparable age.[48,49] Recent attention has focused on steroid sex hormones as a possible explanation for these gender differences. Three epidemiologic studies[50,51,52] did not find an increased risk for SLE in women who used oral contraceptives. One recent study suggested that the use of postmenopausal estrogens is associated with a significant increase in risk of developing SLE[53]. This same study found an increased risk for SLE in oral contraceptive users[54]. Given the sparseness of epidemiologic research about the risks of exogenous female hormones on the risk of developing SLE, we used information from a population-based case-control study to investigate whether women who used exogenous female hormones (oral contraceptives, estrogen replacement therapy) were more likely to develop systemic lupus erythematosus (SLE) than women who had not used such hormones. The parent study investigated breast implants as a risk factor for connective tissue diseases[55].

Methods

Study Setting

King County, which includes the city of Seattle, is the largest metropolitan area in Washington state and has a population of approximately 1.5 million people[24]. It is bounded on the west by Puget Sound and on the east by the Cascade mountains.

Identification of Cases and Control Subjects

We attempted to identify all cases of SLE in women who were residents of King County, Washington at the time of their initial diagnosis between January 1, 1983 and December 31, 1991. All rheumatologists, including those whose practices bordered the north and the south of King County, were contacted and invited to participate in the study. Six of the 42 rheumatologists declined; two estimated that less than 3% of their practices' populations lived in King County. Three other rheumatologists had small practices or had been in practice only during the last few years of the study period. For each participating practice, the best method to identify cases was determined in consultation with the rheumatologist and his or her staff. The methods included analysis of computer billing records; review of microfiche billing records; manual review of every active chart and every chart in archival storage; and in some instances combinations of the various methods.

Cases were screened for eligibility by trained research assistants who used a standardized chart abstraction form for review of the medical records. The case definition followed American College of Rheumatology (ACR) criteria[25] for systemic lupus erythematosus. Two levels of diagnostic certainty were used. 'Definite' cases met the ACR criteria and agreed with the rheumatologists' diagnosis. For 'probable' cases, either the physician made a specific diagnosis but the case fell one feature short of the requisite number of four of the eleven disease criteria, or the disease criteria were met,

but the rheumatologist only called the disease "probable". Over 96% of all provisionally qualifying cases were re-abstracted by the first author to confirm study eligibility.

We used two control groups of King County women. The 'preexisting' control group had been identified via random-digit dialing for three previous epidemiologic studies of diseases of women,[30,31,32] had been interviewed between 1986 and 1991, and had reference dates that were frequency matched to the distribution of cases in the other three studies. 'Preexisting' controls with reference years in 1991 were excluded from the present analyses because their selection criteria included their not having had a hysterectomy. In addition, there were no controls for women over age 40 years in 1983, over age 45 years in 1984, and over age 65 years in 1985 and 1986.

The 'new' control group was also selected by random digit dialing in 1994 - 1996. Random phone numbers[33] were generated using unrestricted random sampling of all working non-cellular prefixes in King County and appending a random 4-digit number to create a telephone number. Each number was called at least 12 times at different times of the day and week over a three-week period before it was abandoned. About of 19,000 telephone numbers were called using a two-step method of recruitment. First, women were screened for King County residency and age, and if qualified were invited to participate in the study. If they agreed to participate, the second step involved frequency matching the potential controls on the age (in five year categories) and year of diagnosis distribution in the case group. A computer program randomly selected a subset of the potential controls and assigned a randomly generated reference date to be similar to the distribution of diagnosis dates among comparably aged cases.

There were 252 cases who met study disease criteria as definite or probable SLE cases. Seven cases (2.8%) were not mailed study materials because their rheumatologist refused permission or advised against contacting these patients. Of the remaining 245 women, 191 (78.0%) women returned a questionnaire or participated in a telephone interview. Twenty-seven women (11.1%) refused to participate, 19 (7.8%)

women were not locatable, 5 (2.0%) women did not participate for other reasons, including 2 women with hearing problems, 2 deceased women with no next-of-kin, and 1 woman who was too ill to participate.

In the course of telephone sampling to identify the 'new' control group, 86.1% of households provided enough screening information to determine whether the household included a potentially eligible woman, and 1,702 (86.4%) of the 1,971 women who were invited to participate agreed to do so, yielding an overall response rate of 74.4%. Among controls selected to participate, 125 were later eliminated from analysis when they returned a questionnaire that indicated either they had not been a King County resident at their reference date, or had had a diagnosis of SLE before their reference date. In the pre-existing control group the overall response rate was 79.0%.

Data Collection

Data were collected by mailed self-administered questionnaires for both the case and 'new' control groups. The survey included questions on marital status and sociodemographic characteristics, diabetes, reproductive history including oral contraceptive use and estrogen replacement therapy, breast cancer, breast implants, hair dye, and work history. Three repeat mailings to non-respondents were used to increase response rate. If no response was received after 4 mailings, a telephone interview was sought, using the same questions. The case identification and data collection methods were pilot tested.

Women's menstrual status was ascertained by asking: "What statement below best describes your menstrual periods in (reference month and year) (check one). (a) I was still having my regular periods. (b) I was pregnant or nursing at that time. (c) They had never started. (d) My periods had begun to stop. (e) My periods had stopped permanently." If either (d) or (e) were selected, 3 additional questions were asked: (1) How old were you when your periods stopped? (2) Did they stop naturally? (3) If they

did not stop naturally, how did they stop?" A few women older than age 55 indicated that they were still having their menstrual periods; however other responses on the questionnaire suggested that they were actually postmenopausal and were experiencing occasional vaginal bleeding due to estrogen therapy or for other reasons. For these women, responses to other items on the questionnaire were used to infer a menopause date.

To ascertain a woman's use of non-contraceptive exogenous hormones, the following question was asked: "Before (reference month and year), had you ever-used any estrogens (such as Premarin), progesterone, (such as Provera), or any female hormones other than for birth control? These could include vaginal creams, suppositories, pills, injections, or skin patches." YES or NO were the only available answers.

Women were also asked the following question to determine oral contraceptive use: "In (reference month and year) what best describes your use of oral contraceptives or birth control pills?" The provided answers were: never-used, current-use or past-use. If current or past-use of oral contraceptives was marked the following question was asked: "For how long had you used birth control pills prior to (reference month and year)?"

Over 40 variables were examined as potential confounders and individually added to the statistical model to ascertain if they altered the odds ratios for estrogen use or oral contraceptive use. Year of diagnosis and age (in 5 year age categories) were included in the model as well. Data analysis utilized logistic regression (STATA 5.0.[35]).

All instruments and methods were approved by multiple human subject institutional review boards, including those of the University of Washington, Virginia Mason Medical Center, and the Group Health Cooperative.

Results

Table 4 compares selected characteristics of premenopausal and postmenopausal women in the three study groups: cases, the 'preexisting' controls, and the 'new' controls. The case and control groups within the same menstrual status category were fairly similar. The main differences were a higher proportion of non-white women (black, other) in the cases as compared to the controls. Race for this reason was controlled for in all analyses.

Table 5 shows that using the 'new' control group, after adjusting for age, calendar year, race (white, black, other), and past-use of oral contraceptives, the odds ratio and associated 95% confidence intervals for estrogen replacement therapy in postmenopausal women was 1.30 (0.73, 2.13) in probable or definite cases. When restricted to definite SLE cases it was 1.51 (0.82, 2.70). Using the 'preexisting' control group and the same adjustments, the odds ratios were 0.97 (0.49, 1.91) for all cases and 0.94 (0.46, 1.92) for definite SLE cases. There was a slight confounding of estrogen use by oral contraceptive use, as when not controlling for past oral contraceptive use the odds ratio for estrogen use was slightly higher in both control groups.

Table 5 also shows the odds ratios for past-use of oral contraceptives adjusted for estrogen use in postmenopausal women. Using the 'new' control group, after adjusting for age, calendar year, race (white, black, other), and estrogen use, the odds ratio and associated 95% confidence intervals for past-use of oral contraceptives in post menopausal women was 1.14 (0.61, 2.13). Using the 'preexisting' control group and the same adjustments the odds ratio was 0.53 (0.24, 1.17). When restricted to 'definite' cases of SLE the odds ratios and associated confidence intervals were essentially the same for both control groups.

Table 6 shows the odds ratios and associated 95% confidence intervals for duration of use of oral contraceptives adjusted for estrogen use, age at reference date, year of diagnosis, and race (white, black, other) in postmenopausal women. None of the

category-specific associations were significant in either category of definite or probable SLE for either control group. The $\chi 2$ tests for trend were not significant.

Table 7 shows the odds ratios for oral contraceptive use in premenopausal women. Using the 'new' control group, after adjusting for age, calendar year, and race (white, black, other), the odds ratio (95% confidence intervals) for ever-use of oral contraceptives in premenopausal women was 1.51 (0.98, 2.36), for past-use it was 1.71 (1.08, 2.71), and for current-use of oral contraceptives the odds ratio was 0.66 (0.33, 1.32). When restricted to definite SLE cases, and using the same adjustments, the odds ratio (95% confidence interval) for ever-use of oral contraceptives was 1.54 (0.98, 2.42), for past-use it was 1.70 (1.06, 2.74), and for current-use it was 0.71 (0.35, 1.43). Using the 'preexisting' control group and the same adjustments the odds ratio (95% confidence intervals) for ever-use of oral contraceptives was 0.57 (0.42, 0.76), for past-use it was 0.66 (0.37, 1.17), and for current-use of oral contraceptives it was 0.34 (0.14, 0.80). When restricted to definite SLE, the odds ratio (95% confidence interval) for ever-use of oral contraceptives was 0.62 (0.35, 1.09), for past-use it was 0.64 (0.35, 1.15) and for current-use it was 0.36 (0.15, 1.17).

Tables 8 and 9 show the odds ratios and associated 95% confidence intervals for duration of use of oral contraceptives adjusted for age at reference date, year of diagnosis, and race (white, black, other) in premenopausal women for the 'new' and 'preexisting' control groups and for probable and definite cases. Only in the 'new' control group did past-use of oral contraceptives for greater than 5 years and less than or equal to 10 years show a significantly increased odds ratio of 1.85 (1.09, 3.14). The $\chi 2$ test for trend in either the definite or probable case groups was not significant. When using the 'preexisting' control group, the $\chi 2$ test for trend was significant for a decrease in the odds ratios for duration of OC use in both the definite and probable categories of SLE diagnosis, for both ever-use and current-use of oral contraceptives.

Discussion

The present study used a case-control design, since SLE is so rare. The risk of developing SLE in postmenopausal women was not found to be significantly increased by use of hormone replacement therapy or the past use of oral contraceptives. In premenopausal women the risk of SLE appeared to be significantly increased by past use of oral contraceptives when compared to the experience of OC use in the 'new' control group. However, when duration of use was examined as a measure of dose, no dose-response gradation in risk was found. The odds ratio for past use of oral contraceptives using the 'new' control group was not large, and the lower end of the confidence interval was only slightly larger than 1. When restricted to definite cases of SLE the increased odds ratio persisted for past use of oral contraceptives in premenopausal women.

Using the 'preexisting control group, there was a significant trend for decreasing odds ratios for SLE with increasing duration of OC use in the ever-use and current-use categories. For ever use and past use of oral contraceptives there were significant decreased risks of OC use in the different duration categories. If there is an increased risk of SLE for oral contraceptive use, these results suggest that it is slight.

The only other study to have investigated postmenopausal estrogen replacement therapy[50] found significantly increased risk for developing SLE for ever-use 2.1 (1.1, 4.0), past-use 1.8 (0.8, 4.1), and current-use 2.5 (1.2, 5.0), and also found a significant trend for duration of estrogen use. The present study had only information on ever-use / never-use and did not have any duration of use information.

Three other studies[51,52,53] that have looked at oral contraceptive use as a risk factor for SLE did not find a significantly increased risk. These studies were not population-based and used friends, hospital registration, or medical outpatients with other chronic illness as controls. There is mention of an increase in risk for OC use[54] in the same research

that found an increased risk for estrogen replacement therapy[50], but no details were provided.

As in any epidemiologic study the results need to be viewed in terms of chance, selection bias, measurement error and confounding as possible alternative explanations of the results[36]. When evaluating the role that chance might play in this study, one is concerned with the possibility that a real difference in exogenous hormone exposure exists between the cases and the controls, but escaped detection. Given the rarity of the disease in the study area, based on the 95% confidence limits, this study could not rule out a true relative risk of 1.45 or less for postmenopausal hormone use or a true relative risk of 2.36 or less of ever-use OCs in premenopausal women.

Selection bias could result if there was differential selection of cases or controls on the basis of exposure status. In the case group, the medical charts of all potential cases were re-reviewed to make sure they met the criteria for disease status, year of diagnosis, age, and King County residency, irrespective of exogenous hormone use. Similarly, the controls were selected to participate without knowledge of hormone use. Measurement bias could result if information was obtained differently between the case and control groups. The same questionnaires were used for both the cases and the new controls, and a subset of the questions for the in-person interview had been used with the pre-existing controls. In the 'pre-existing' control group, an in-person interview was used that used displays cards that showed different oral contraceptives pills, and estrogen pills. It id possible that the 'pre-existing' control group had a better recall of either use or non-use of exogenous hormones than either the cases or the 'new' control group, which might account for the difference in oral contraceptive risk found when using different control groups. Recall bias would result if there were differential reporting of an exposure in the case or control group. The incidence of SLE in this King County, Washington female population was similar to other published values in other populations.[41] The overall response rate was similar in the case group (80.3%) and the two control groups, 74.4% overall for the new controls, and 79% for the existing

controls. All 'new' controls were contacted within 3 months after their initial phone screen, whereas some of the cases were last seen by their rheumatologist in 1983 or 1984.

Misclassification of exposure status for post menopausal hormones or oral contraceptives use, or misclassification of menstrual status, could be responsible for the generally negative results. One could speculate that there is not likely to be misclassification of oral contraceptive use with respect to ever having used them, but misclassification could be possible for duration of oral contraceptives use. We used large years of duration of use and tried different cut points, but the significance of past-use of oral contraceptives in the 'new' control group did not change. Postmenopausal status was problematic as women taking hormone replacement therapy would continue to menstruate due to the effects of exogenous hormones. And some women in their 60's and 70's reported that their menstrual period continued, or did not report a date of menopause, thus making it problematic to determine when their menopause actually ended. When analysis was performed with uncorrected data the results were essentially unchanged from those presented.

The role of confounding due to unmeasured covariates is another possible source of bias in any observational epidemiologic study. This study controlled for age at reference date and reference year, which were frequency matched between the case and 'new' control groups. In addition it controlled for race, since black and Asian women had a higher incidence of SLE but were less likely to be represented in the control group. Forty additional covariates were investigated and none were found to change the estimate of the hormone odds ratios importantly.

In conclusion, the results of this study do not support a significantly increased risk for postmenopausal women with SLE to have been exposed to exogenous female hormones. We could not rule out a small increase in risk for the use of oral contraceptives in premenopausal women and development of systemic lupus

erythematosus. It could be that there is a slight protective effect in current-use of oral contraceptives and developing SLE. Further studies are required to substantiate these findings.

CHAPTER 3: THE INCIDENCE OF PRIMARY SJOGREN'S SYNDROME, SYSTEMIC LUPUS ERYTHEMATOSUS, POLYMYOSITIS / DERMATOMYOSITIS, AND SYSTEMIC SCLEROSIS / CREST SYNDROME OF WOMEN IN KING COUNTY, WASHINGTON,1983 THROUGH 1991.

Summary

Objective To determine the incidence of primary Sjogren's syndrome, systemic lupus erythematosus, systemic sclerosis / CREST syndrome, and polymyositis / dermatomyositis in female residents of King County Washington between January 1, 1983 and December 31, 1991 for women between the ages of 18 and 74.

Methods Case identification during the conduct of a population-based case-control study on risk factors for connective tissue diseases in women. Estimation of age and race specific female incidence rates from census data for whites, blacks, and all races combined.

Results A total of 252 systemic lupus erythematosus cases, 70 systemic sclerosis / CREST cases, 184 Sjogren's syndrome, and 28 cases of polymyositis / dermatomyositis cases were identified. For systemic lupus erythematosus the incidence rate per 100,000 person years (and age adjusted to 1990 US population) for whites was 3.98 (3.93), for blacks 20.57 (20.45), and for all races combined 5.44 (5.35). For systemic sclerosis / CREST the incidence rate (per 1,000,000 person years) for whites was 14.6 (16.2), for blacks 10.0 (9.4), and for all races combined 15.1 (16.5). For Sjogren's syndrome the incidence rate (per 100,000 person years) for whites was 3.86 (4.15), for blacks 5.02 (5.11), and for all races combined 3.96 (4.25). For polymyositis / dermatomyositis the incidence rate (per 1,000,000 person years) for whites was 5.4 (5.6), for blacks 10.0 (9.6), and for all races combined 6.0 (6.0).

Conclusion The incidence rate of systemic lupus erythematosus for black women was almost 5 times higher than the incidence rate in white women and much higher than most previously published studies. For systemic sclerosis /CREST syndrome there is a significant trend of increasing incidence with age. Incidence rates for primary Sjogren's

syndrome have not heretofore been published, and shows that the incidence rises quickly beyond the ages at which menopause commonly occurs.

Background

There are few data in the incidence rates for many connective tissue diseases, and incidence rates for primary Sjogren's syndrome have not been previously published. Many of the published rates are based on hospital records due to the expense and difficulty of doing truly community-based studies since connective tissue diseases rarely result in hospitalization or death, and registries rarely exist. Incidence rates are a useful measure of disease occurrence in populations, and more descriptive epidemiologic data may help generate etiologic hypotheses that might in turn lead to opportunities for prevention.

It is known that for the diseases studied in this paper, the incidence in women is some 3 to 10 times higher than in men of comparable ages[48,49]. Blacks are at increased risk for developing these diseases compared to whites[48,49].

The conduct of a population-based case-control study of risk factors for connective tissue diseases[56,57,58] provided us with the opportunity to estimate the incidence rates of systemic lupus erythematosus, progressive systemic sclerosis, Sjogren's syndrome, and polymyositis / dermatomyositis for female King County, Washington residents between the ages of 18 and 74 from the beginning of 1983 through the end of 1991.

Methods

Study population.

All women between the ages of 18 and 74 inclusive who were residents of King County, Washington constituted the study population. Cases of connective tissue diseases of interest were required to have been residents of King County, Washington at the time of their initial diagnosis. The date of diagnosis must have been between January 1, 1983 and December 31, 1991.

Sources for case ascertainment.

All rheumatologists in King County were approached and asked to participate. We also included rheumatologists working in the two counties bordering the north and south of King County who may have treated King County residents. In some practices and institutions, ICD-9 codes from computer billing records were used to identify potential cases of connective tissue diseases (CTDs) of interest. At other rheumatology practices without computerized billing records, all active and archival charts for the period of interest were reviewed manually. Of all the rheumatologists in King County, only four declined to participate. Three of the four had small practices or had only been in practice late in the study time period. Of the other practice, little information is known.

Trained research assistants were used to identify potentially eligible cases, and 96% of the final cases were reviewed by one of us (WBT) to verify eligibility for inclusion. For systemic lupus erythematosus, progressive systemic sclerosis / CREST syndrome, and Sjogren's syndrome, American College of Rheumatology (ACR) study criteria[25,26,27] were used. For polymyositis / dermatomyositis, criteria proposed by Bohan[29] were used. Two levels of diagnostic certainty were used - definite and probable. Definite cases met the above study criteria and were so diagnosed by the rheumatologist. For "probable" cases, either the physician called a case 'definite' but the patient's disease features were one item short of meeting disease criteria, or the case met the appropriate disease criteria but the rheumatologist only called the case 'probable'. Race was determined from response to a mailed questionnaire, or if the questionnaire was not returned from the medical chart. Population denominator information was derived from the 'Wonder system' of the Center for Disease Control's (CDC) Internet site at www.cdc.gov.

Data Analysis

All cases were double data entered into a data base from the chart abstraction form by a research assistant. A C++ program was used for data cleaning and to construct incidence rate tables. Data analysis consisted of creating race, age disease incidence

tables. Incidence rates were stratified by age in 10 year increments, and race (black, white, all races combined), and were expressed as cases per 100/000 person-years. Stata 5.0 was used[35] to do statistical tests.

Results

A total of 532 women with connective tissue diseases of interest were identified, including; 252 SLE cases, 70 systemic sclerosis / CREST cases, 184 cases of primary Sjogren's syndrome, 28 cases of polymyositis / dermatomyositis, and 4 cases of mixed connective tissue disease. Tables 10, 11, and 12, show the race and age specific disease incidence rates for the 4 diseases of interest, the number of cases for each disease in 10 year age categories, the population years at risk, and the age standardized incidence rates (direct method) to the 1990 United States population. Six cases had multiple diagnoses; 2 cases of SLE and polymyositis, 2 cases of systemic sclerosis / CREST and SLE, and 2 cases of systemic sclerosis and polymyositis.

Tables 10 through 12 describe the incidence rates for systemic lupus erythematosus (SLE). For SLE the incidence rate (per 100,000 person years) for whites was 3.98, for blacks 20.57, for all races 5.44. For black women, the age specific incidence rate was significantly higher than the white women's incidence rate (z test, $p < 0.001$). For all races the highest incidence of SLE was during the child bearing years, and declined in postmenopausal women, but the test for trend was not significant in any group. The over all incidence rate in black women is 20.57 some 5 times higher that the overall incidence rate in white women. The table also shows the age standardized incidence rates for the total 1990 US population.

Tables 10 through12 describes the incidence rates for systemic sclerosis and CREST syndrome. For systemic sclerosis / CREST the overall incidence rate (per 100,000 person years) for whites was 1.46, for blacks 1.00, and for all races 1.51. The overall incidence rates for white, black were not significantly different (z test, $p = 0.67$). In white women and all races combined there was a significant trend of increasing incidence of

systemic sclerosis and CREST with increasing age of women (p = 0.0001). For black women, the only cases occurred in younger women aged 18-39.

The incidence rates for Sjogren's syndrome are shown in Tables 10 through 12. For Sjogren's syndrome the overall incidence rate (per 100,000 person years) for white women was 3.86, for black women 5.02, and 3.96 for all races combined. For black and white women the incidence rates were not significantly different (z test, p = 0.75). In all racial groups the incidence rate increased with increasing age, and the incidence rates rose quickly beyond the ages at which menopause commonly occurs, and for white women and all races combined the trend was significant (p = 0.0001). The incidence rates were similar across all racial groups, with black women having a slightly higher incidence than white women.

Tables 10 through 12 describes the incidence rates for polymyositis / dermatomyositis. For polymyositis / dermatomyositis the incidence rate (per 100,000 person years) for whites was 0.54, for blacks 1.00, and all races 0.60. There did appear to be a age-related trend in the incidence rate for white women and all races combined (z test, p = 0.02 white, p = 0.3 all races). Again black women seemed to have a higher incidence rate than white women, but it was not significant (p = 0.75)

Table 13,14,15 compares the race specific incidence from other selected incidence rate studies for systemic lupus erythematosus, systemic sclerosis, and polymyositis / dermatomyositis to the results of this study. For black women this study found approximately a two fold increase of systemic lupus erythematosus compared to other incidence studies, with one exception, while the incidence rates for white women were quite comparable (Table 13). For systemic sclerosis, (Table 14) the situation is different, with the incidence rate for black women is about one-half as large as the other two studies, but the white incidence rate is higher. For polymyositis the incidence rates for black and white women is comparable to the incidence rates found in the other studies.

Discussion

The diseases discussed in this paper are all relatively rare in the general population. All of the cases in this study were incident cases diagnosed by a rheumatologist and were King County residents at the time of diagnosis.

The disease with the highest incidence rate was systemic lupus erythematosus. Women of childbearing age seemed to have higher incidence rates, but there was an appreciable incidence of SLE in the older age groups for all three groups. The incidence rates from this study are largely similar to other published incidence studies (Table 13). The results show a much larger incidence for systemic lupus erythematosus in the black and other race categories relative to the white population. The rates in the black category categories are much higher than other published rates (with one exception) for these race categories. For the black race category, this could result from two possible causes. Other studies used hospital based diagnosis and underestimated the incidence rates for minority populations (and probably for all races). In only one study[68] was the incidence rate for black women comparable to the incidence rates found in this study for SLE. The other possibility is that the US census data underestimated the size of the black population in King County.

The incidence rate for primary Sjogren's syndrome found in this study appears to be the first published estimate. We found a striking increase in incidence with age for all races. These incidence rates rise quickly beyond the ages at which menopause commonly occurs. There was also a higher incidence rate for black women when compared to white women, but is not so striking as the increase for systemic lupus erythematosus.

For systemic sclerosis and CREST syndrome the disease seems to have a higher incidence in white women, but the disease is much rarer than SLE, and the minority populations in King County rather small, so apparent racial differences may simply be due to chance. There is a steady increase in incidence with increasing age in white women, but again the number of cases in minority women are few due to both the small

population and the rarity of the disease. When comparing the incidence rates in this study to the rates obtained in other studies (Table 14) and populations, the rates seem comparable for white women, but for black women somewhat lower than incidence rates found in Michigan[73] and Allegheny, PA[72].

The incidence rates of polymyositis / dermatomyositis did not seem to vary consistently with age. The incidence was higher in black women, but the rarity of the disease and the small size of the non-white population makes it difficult to draw any firm conclusions. In comparing the incidence rates in this study to the results of other studies (Table 15) they appear to be quite comparable.

When restricted to 'definite' cases the incidence rates did not significantly change as there were fewer than 20 'probable' cases of the study diseases. Often, when reviewing the patient's medical charts, it seemed that the rheumatologists' use of "probable" in their patient records was rather casual, and varied on many patient encounters. The other reason that the incidence rates are reported together for both 'definite and 'probable' is that case ascertainment was not complete due to the non-participation by four of the area rheumatologists, and inclusion of the 'probable' cases gives a more likely reflection of the true underlying disease incidence rates.

In conclusion, this study shows that the incidence rates for the white population to be compatible with other studies, while the incidence rates for Black to be substantially higher than other studies for systemic lupus erythematosus. This suggests that other non-population based studies might have underestimated the impact of the diseases studied in minority populations. This study also presented the only published incidence rates for Sjogren's syndrome, and found that the Sjogren's syndrome incidence rates just behind SLE incidence among the four diseases studied here.

BIBLIOGRAPHY

1. Symmers W, St C. Silicone mastitis in "topless" waitresses and some other varieties of foreign-body mastitis. Br Med J 1968; 3:19-22.

2. Cook RR, Delongchamp RR, Woodbury MA, Perkins LL, HarrisonMC. The Prevalence of women with breast implants in the United States - 1989. J Clin Epidemoiol 1995;48:519-526.

3. Kessler DA. The basis of the FDA's decision on breast implants. N Eng J Med 1992;326:1713-1715.

4. Bridges AJ, Vasey FB. Silicone breast implants; history, safety, and potential complications. Arch Intern Med 1993;151:2638-2644.

5. Gott DM, Tinkler Silicone implants and connective tissue disease. Medical Devices Agency 1994.

6. Goldblum RM, Pelley RP, O'Donell AA, Pyron D, Heggers JP. Antibodies to silicone elastomers and reactions to ventriculoperitoneal shunts. Lancet 1992;340:510-513.

7. Rohrich RJ, Hollier LH, Robinson, Jr. JB. Determining the safety of the silicone envelope: In search of a silicone antibody. Plastic Reconstruct Surg. 1996; 98(3):455-458.

8. Marcus DM. An analytic review of silicone immunology. Arthritis Rheum. 1996;39(10):1619-1626.

9. Gabriel SE, O'Fallon WM, Kurland LT, Beard CM, Woods JE, Melton LJ. Risk of connective tissue diseases and other disorders after breast implantation. NEJM 1994;30(24):1697-1702.

10. Goldman JA, GreenBlatt J, Joines R, White L, Aylward B, Lamm SH. Breast implants, rheumatoid arthritis and connective tissue diseases in a clinical practice. J Clin Epidemiol 1995;48(4):571-582.

11. Giltay EJ, Bernelot HJ, Riley AH, Tan RG. Silicone breast prostheses and rheumatic symptoms: a retrospective follow up study. Ann Rheum Dis 1994;53:194-196.

12. Hochberg MC, Perlmutter DL, Medsgar TA, Nguyen K, Steen V, Weisman MH, White B, Wigley FM. Lack of association between augmentation mammoplasty and systemic sclerosis (scleroderma). Arthritis and Rheum 1996;39(7):1125-1131.

13. Wells KE, Cruse CW, Baker JL Jr, Daniels SM, Stern RA, Newman C, et al. The health status of women following cosmetic surgery. Plast Reconstr Surg 1994;93:907-912.

14. Englert HJ, Brooks P. Scleroderma and augmentation mammoplasty - a causal relationship? Aust NZ J Med 1994;24(1):74-80;

15. Schusterman MA, Kroll SS, Reece GP, Miller MJ, Ainslie N, Halabi S, Balch CM. Incidence of autoimmune disease in patients after breast reconstruction in silicone gel implants versus autogenous tissue: a preliminary report. Ann Plast Surg 1993;31(1):1-6.

16. Strom BL, Reidenberg MM, Freundlich B, Schinnar R. Breast silicone implants and risk of systemic lupus erythematosus. J Clin Epidemiol 994;47(10):1211-1214.

17. Dugowson CE, Daling J, Koepsell TD, Voigt L, Nelson JL. Silicone breast implants and risk for rheumatoid arthritis Rheum 1992;192:S66 (abstract).

18. Sanchez-Guerrero J, Colditz GA, Karlson EW, Hunter DJ, Speizer FE, Liang MH. Silicone breast implants and the risk of connective-tissue diseases and symptoms. NEJM 1995;332(25):1666-1670.

19. Wigley FM, Miller R, Hochberg MC, Stern V. Augmentation mammoplasty in patients with systemic sclerosis: data from the Baltimore Scleroderma Research Center and Pittsburgh Scleroderma Data Bank. Arthritis Rheum 1992:35(Suppl):S369.

20. Burns CJ, Laing TJ, Gillespie BW, Heeringa SG, Alcser KH, Mayes MD, Wasko MC, Cooper BC, Garabrant DH, Schottenfeld D. The epidemiology of scleroderma among women: assessment of risk from exposure to silicone and silica. J-Rheumatol. 1996; 23(11):1904-1911.

21. Sanchez-Guerrero J, Colddítz GA, Karlson EW, Hunter SJ, Speizer FE, Liang MH. Silicone breast implants and connective tissue diseases and symptoms. N Engl J Med 1995; 332(25):1666-1670.

22. McLaughin JK, Olsen JH, Mellemkjaer L. Correspondence Re: Breast implants, cancer and systemic sclerosis. J Nat Cancer Instit 1995;87(18)1415-1416.

23. Hennekens CH, Lee I-M, Cook NR, Hebert PR, Karlson EW, LaMotte F, Manson JE, Buring JE. Self-reported breast implants and connective-tissue diseases in female health professionals. JAMA 1996;275(8):616-621.

24. US Department of Commerce, Bureau of the Census. 1990 Census of Population and Housing: Summary Population and Housing Characteristics, Washington. US Government printing office, 1991.

25. Tan EM, Cohen AS, Fries JF, Masi AT, McShane DJ, Rothfield NF, Schaller JG, Talal N, Winchester RJ. The 1982 revised criteria for the classification of systemic lupus erythematosus. Arthrit Rheumat 1982;25(11):1271-1277

26. Subcommittee for Scleroderma Criteria of the American Rheumatism Association diagnostic and therapeutic Criteria committee. Preliminary criteria for the classification of systemic sclerosis (scleroderma). Arthrit Rheumat 1980; 23(3):581-90.

27. Vitali C, Bombardieri S, Moutsopoulos HM, Balestrieri G, Bencivelli W, Bernstein RM, Bjerrum KB, Braga S, Coll J, de-Vita-S, et. al. Preliminary criteria for the classification of Sjogren's syndrome. Arthrit Rheumat 1993;36(3):340-347.

28. Sharp GC. Diagnostic criteria for classification of MCTD. In Kasukawa R, Sharp GC, eds. Mixed connective tissue diseases and anti-nuclear antibodies. Amsterdam Elsevier. 1987;23-32.

29. Bohan A, Peter JB, Bowman RL, Pearson CM. Computer-assisted analysis of 153 patients with polymyositis and dermatomyositis. Medicine 1977;56:255-286.

30. Voigt LF., Koepsell TD, Nelson JL, Dugowson CE, Daling JR. Smoking, obesity and alcohol consumption and the risk of rheumatoid arthritis. Epidemiology 1994; 5:525-532.

31. Daling JR, Malone KE, Voigt LF, White E, Weiss NS. Risk of breast cancer associated with induced abortion. JNCI 1994; 86:1584-1592.

32. Stanford JL, Weiss NS, Voigt LF, Daling JR, Habel LA, Rossing MA. Combined estrogen and progestin hormone replacement therapy in relation to risk of breast cancer in middle-aged women. JAMA 1995; 274:137-142.

33. Waksberg J. Sampling methods for random digit dialing. J Am Stat Assoc 1978;73;40-46.

34. American Medical Association Physicians' Current Procedural Terminology CPT 1995 American Medical Association, Chicago, Ill.

35. Stata Corp. Stata Statistical Software: Release 5.0 1997 College Station, Tx.: Stata Corporation.

36. Wong O. A critical assessment of the relationship between silicone breast implants and connective tissue diseases. Regulatory Toxicology and Pharmacology 1996;23:74-85.

37. Hochberg MC, Perlmutter DL. The association of augmentation mammoplasty with connective tissue disease, including systemic sclerosis (Scleroderma): A meta-

analysis. In: Immunology of Silicones. M Potter, NR Rose, eds. New York: Springer 1996.

38. Perkins LL, Clark BD, Klein PJ, Cook RR. A meta-analysis of breast implants and connective tissue diseases. Ann Plast Surg. 1995;35:561-570.

39. Hennekens CH, Burring JE, Mayrent SL. Epidemiology in Medicine. 1987 Little, Brown and Company. Boston.

40. Oddis CV, Conte CG, Steen VD, et al. Incidence of polymyositis-dermatomyositis: a 20-year study of diagnosed cases in Allegheny County, PA 1963-1982. J Rheumatol 1990;17:1329-1334.

41. Michet Cj, McKenna CH, Elveback LR, Kaslow RA, Kurland LT. Epidemiology of systemic lupus erythematosus and other connective tissue diseases in Rochester, Minnesota, 1950 through 1979. Mayo Clin Proc 1985;60:105-113.

42. Cook LS, Daling JR, Voigt LF, deHart MP, Malone KE, Stanford JL, Weiss NS, Brinton LA, Gammon MD, Brogan D. Characteristics of women with and without breast implants. JAMA.1997;277(20):1612-1617.

43. Sanchez-Guerrero J. Liang MH, Karlson EW, Hunter DJ, Colditz GA. Postmenopausal estrogen therapy and the risk for developing systemic lupus erythematosus. Ann Int Med. 1995; 122(6):430-433.

44. Silverman BG, Brown SL, Bright RA, Kaczmarek RG, Arrowsmith-Lowe JB, Kessler DA. Reported complications of silicone gel breast implants: an epidemiologic review. Ann Intern Med. 1996;124(8):744-756.

45. Rosenberg-N-L. The neuromythology of silicone breast implants. Neurology. 1996;46(2):308-14.

46. Yoshida-S-H. Swan-S. Teuber-S-S. Gershwin-M-E. Silicone breast implants: immunotoxic and epidemiologic issues. Life-Sci. 1995;56(16):1299-1310.

47. Cuellar ML, Gluck O, Molina JF, Gutierrez S, Garcia C,. Espinoza-R. Silicone breast implant--associated musculoskeletal manifestations. Clin Rheumatol. 1995;14(6):667-672.

48. Massi AT, Kaslow RA. Sex effects in systemic lupus erythematosus: a clue to pathogenesis? Arthritis Rheum 1978;21:480-484.

49. 49 Silman AJ, Hochberg MC. Epidemiology of the Rheumatic Diseases Oxford University Press New York 1993.

50. Strom BL, Reidenberg MM, West S, Snyder ES, Freundlich B, Stolley PD. Shingles, allergies, family medical history, oral contraceptives, and other potential risk factors for systemic lupus erythematosus. Am J Epi 1994,140(7):632-642.

51. Hochberg MC, Kaslow RA. Risk factors for the development of systemic lupus erythematosys: A Case-Control Study. Clinical Research1983;31(4):732A.

52. Grimes DA, LeBolt SA, Grimes KR, Wingo PA. Systemic lupus erythematosus and reproductive function: a case-control study. Am J Obstet Gynecol. 1985;153(2):179-184.

53. Sanchez-Guerrero J. Liang MH, Karlson EW, Hunter DJ, Colditz GA. Postmenopausal estrogen therapy and the risk for developing systemic lupus erythematosus. Ann Int Med. 1995; 122(6):430-33.

54. Liang MH, Karlson EW. Female hormone therapy and risk of developing or exacerbating systemic lupus erythematosus or rheumatoid arthritis. Proc Assoc Am Physicians 1996;108(1):25-28.

55. Teel WB, Koepsell TD, Daling JR, Psaty BM, Dugowson C, Kronmall RA, Voigt LF. A population-based case-control study of breast implants as a risk factor for connective tissue diseases in King County, Washington. (This dissertation)

56. Teel WB, Koepsell TD, Daling JR, Psaty BM, Dugowson C, Kronmall RA, Voigt LF. A population-based case-control study of exogenous female hormones as risk factors for systemic lupus erythematosus in King County, Washington. (This Dissertation)

57. Teel WB, Koepsell TD, Daling JR, Psaty BM, Dugowson C, Kronmall RA, Voigt LF. A population-based case-control study of hair dyes as a risk factor for connective tissue diseases in King County, Washington. (In preparation)

58. Teel WB, Voigt LF, Martin D. A frequency matching program for epidemiologic case-control studies. (In preperation)

59. McCarty DJ, Manzi S, Medsger TA Jr,Ramsey-Goldman R, LaPorte RE, Kwoh CK. Incidence of systemic lupus erythematosus; Race and Gender Differences. Arthrit Rheumat 1995;38(9):1260-1270.56.

60. Siegel M, Lee S. The epidemiology of systemic lupus erythematosus. Semin Arthris Rheum 1973;3:1-54.

61. Siegel M, Holley HL, Lee SL. Epidemiologic studies on systemic lupus erythematosus: comparative data for new York City and Jefferson County, Alabama, 1956-1965. Arthritis Rheum 1970;13:802-811.

62. Michet C Jr., McKenna C, Elveback L, Kaslow A, Kurland L. Epidemiology of systemic lupus erythematosus and other connective tissue diseases in Rochester, Minnesota, 1950-1979. Mayo Clin Proc. 1985;60:105-113.

63. Hochberg MC. The incidence of systemic lupus erythematosus in Baltimore, Maryland, 1970-1977. Arthritis Rheum 1985;28:80-86.

64. Gudmundsson S, Steinsson K. Systemic lupus erythematosus in Iceland, 1975-1984: a nationwide epidemiologic study in an unselected population. J Rheumatol 1990;17:1162-1167.

65. Jonsson H, Nived O, Sturfelt G, Silman A. Estimating the incidence of systemic lupus erythematosus in a defined population using multiple sources of retrieval. Br J Rheumatol. 1990;29:185-188.

66. Hopkinson N, Doherty M, Powell R. The prevalence and incidence of systemic lupus erythematosus, UK, 1989-90. Br J Rheumatol. 1993;110-115.

67. Nossen J. Systemic lupus erythematosus on the Caribbean Island of Curacao: an epidemiologic investigation. Ann Rheum Dis. 1992;51:1191-1201.

68. Johnson AE, Gordon C, Palmer RG, Bacon PA. The prevalence and incidence of systemic lupus erythematosus in Birmingham, England: a relationship to ethnicity and country of birth. Arthritis Rheum 1995;38:551-557.

69. Oddis CV, Conte CG, Steen VD, Medsger DA Jr. Incidence of polymyositis-dermatomyositis: A 20-year study of hospital diagnosed cases in Allegheny County, PA. J Rheumatol 1990; 17(10):1329-1334.

70. Medsger TA Jr., Dawson WN, Masi AT. The epidemiology of polymyositis. Am J Med. 1970;48:715-23.

71. Benbassat J, Geffel D, Zlotnick A. Epidemiology of Polymyositis - Dermatomyositis in Israel, 1960-76. Isr J Med Sci 1980;16:197-200.

72. Steen VD, Oddis CV, Conte CG, Janoski J, Ziegler Casterline G, Medsger TA Jr. Incidence of systemic sclerosis in Allegheny County, Pennsylvania: a twenty-year study of hospital-Diagnosed Cases, 1963-1982. Arthritis Rheum. 1997;40(3):441-445.

73. Laing TJ, Gillespie BW, Toth MB, Mayes MD, Gallavan RH, Burns CJ, Johanns JR, Cooper BC, Keroack BJ, Wasko MC, Lacey JV, Schottenfeld D. Racial differences in scleroderma among women in Michigan. Arthritis Rheum. 1996;40(4):734-742.

74. Geirsson AJ, Steinsson K, Gudmundsson S, Vigfus S. Systemic sclerosis in Iceland. A nationwide epidemiologic study. Ann Rheum Dis. 1994;53:502-505.

APPENDIX A: TABLES

Table 1. Distribution of Breast Implant Status by Disease and Control Groups.

	Had Breast Implants				No		Percent
Case Group	**Silicone**	**Saline**	**Other**	**Total**	**Implants**	**Total**	**Implants**
SLE	2	0	0	2	189	191	1.1
SSc/CREST	0	0	0	0	55	55	0.0
Sjogren's syndrome	2	1	1	4	157	161	2.5
PM/DM	0	0	0	0	17	17	0.0
MCTD	0	0	0	0	3	3	0.0
All Cases	4	1	1	6	421	427	1.4
Control Group							
'New'	12	7	4	24*	1553	1577	1.5
'Pre-Existing'	16	0	0	16	1672	1688	1.0

*One Control did not indicate breast implant type
SLE = Systemic Lupus Erythematosus,
SSc = systemic sclerosis,
PM/DM = polymyositis, Dermatomyositis
MCTD = mixed connective tissue disease

Table 2. Distribution of Covariates Among Cases and Controls

	CASES						CONTROLS	
	SLE	SS/CREST	Sjogren's	PM/DM	MCTD	ANY CTD	PRE-EXISTING	NEW
Covariates	(N = 191)	(N = 55)	(N = 161)	(N = 17)	(N = 3)	(N = 427)	(N = 1688)	(N = 1577)
Age At Ref Date	39.2 ± 13.0	50.5 ± 14.7	50.6 ± 13.2	46.5 ± 14.7	31.7 ± 11.0	45.2 ± 14.4	48.1 ± 13.1	41.7 ± 14.1
Race - black (%)	13.8	3.6	5.0	11.8	33.3	10.9	3.2	1.8
- other (%)	17.0	3.6	6.9	0.0	33.3	9.2	7.7	4.5
- white (%)	69.2	92.7	88.1	88.2	33.3	79.9	89.1	93.7
Height	64.1 ± 2.5	64.1 ± 3.3	64.1 ± 2.6	64.7 ± 3.3	65.0 ± 1.7	64.1 ± 2.7	64.7 ± 2.6	64.6 ± 2.8
Weight	136.3 ± 8.8	133.0 ± 18.6	141.8 ± 31.3	156.8 ± 24.7	134.3 ± 8.1	138.7 ± 28.7	143.2 ± 30.4	140.8 ± 29.4
Body Mass Index	22.8 ± 4.5	22.3 ± 3.2	23.6 ± 5.2	26.3 ± 5.7	22.0 ± 1.7	23.2 ± 4.7	23.5 ± 4.9	23.2 ± 4.5
Education (years)	13.7 ± 2.9	13.6 ± 2.4	14.0 ± 3.1	14.1 ± 2.2	15.3 ± 1.2	13.9 ± 2.9	13.9 ± 2.5	14.1 ± 2.8
Ever Married (%)	79.1	83.6	89.4	76.5	33.3	83.1	91.0	81.4
Hair Dye Use (%)	50.3	43.4	65.2	58.8	66.6	55.5	62.6	45.7
Age at Menarche	12.8 ± 1.5	12.6 ± 1.6	12.7 ± 1.7	12.1 ± 1.0	13.3 ± 1.5	12.7 ± 1.6	12.7 ± 1.5	12.7 ± 1.4
Gravid	2.4 ± 2.0	2.1 ± 2.1	2.7 ± 1.9	3.1 ± 2.3	3.6 ± 3.2	2.7 ± 3.9	2.8 ± 2.1	2.5 ± 2.1
Miscarriages	0.43 ± 1.0	0.63 ± 1.7	0.43 ± 0.9	0.47 ± 0.8	0.33 ± 0.6	0.46 ± 1.1	0.39 ± 0.84	0.32 ± 0.8
No. of Ectopic pregnancies	0.02 ± 0.1	0.33 ± 1.7	0.06 ± 0.3	0	0	0.07 ± 0.6	0.03 ± 0.2	0.03 ± 0.2
No. of Abortions	0.30 ± 0.7	0.46 ± 1.7	0.18 ± 0.5	0.35 ± 0.8	0	0.29 ± 0.9	0.19 ± 0.6	0.28 ± 0.7
No. of Stillbirths	0.09 ± 0.4	0.33 ± 1.7	0.04 ± 0.3	0	0	0.10 ± 0.7	0.03 ± 0.2	0.3 ± 0.2
No. of Live Birth	1.46 ± 1.4	1.73 ± 2.0	1.91 ± 1.6	2.18 ± 1.8	1.33 ± 2.3	1.66 ± 1.6	2.17 ± 1.7	1.76 ± 1.6
BreastCancer(%)	1.1	0	3.1	0	0	1.6	2.3	1.3

Table 3. Breast Implant Odds Ratio by Disease and Reference Group and Diagnostic Certainty *

Definite and Probable Cases	**Any Connective Tissue Disease**	**Systemic Lupus Erythematosus**	**Sjogren's Syndrome**
Referent: New Controls	**0.9 (0.4, 2.3)**	**0.8 (0.2, 3.4)**	**1.6 (0.5, 4.7)**
Referent: Existing Controls	**1.3 (0.4, 4.1)**	**1.6 (0.3, 7.3)**	**1.8 (0.4, 7.8)**
Definite Cases			
Referent: New Controls	**0.7 (0.3, 2.1)**	**0.4 (0.1, 3.3)**	**1.4 (0.4, 4.8)**
Referent: Existing Controls	**1.2 (0.3, 4.2)**	**0.9 (0.1, 6.9)**	**2.0 (0.4, 8.8)**

* Odds Ratios adjusted for Reference year, Age, Race (white, black, other)

Table 4. Distribution of Covariates Among Cases and Controls by menopausal status (mean ± standard deviation, except as noted, ages in years)

	Pre menopausal			Post Menopausal		
Covariates	**SLE Cases** (n = 122)	**EXISTING Controls** (n = 786)	**NEW Controls** (n = 1016)	**SLE Cases** (n = 61)	**EXISTING Controls** (N = 900)	**NEW Controls** (N = 527)
Age At Ref Date	32.4 ± 8.2	37.3 ± 7.9	34.0 ± 8.3	52.2 ± 10.2	57.5 ± 8.6	56.5 ± 0.7
Height (inches)	64.2 ± 2.4	65.1 ± 2.7	64.9 ± 2.7	64.1 ± 2.8	64.4 ± 2.5	64.0 ± 3.2
Weight (lbs.)	133.8 ± 26.3	140.2 ± 29.3	138.9 ± 28.8	141.1 ± 33.7	145.8 ± 30.9	144.5 ± 30.3
Body Mass Index	22.6 ± 4.2	22.8 ± 4.8	22.4 ± 4.0	23.7 ± 5.3	24.2 ± 4.9	24.3 ± 5.0
Education (years)	14.2 ± 2.5	14.5 ± 2.3	14.6 ± 2.6	13.1 ± 3.3	13.3 ± 2.4	13.3 ± 3.1
Ever Married (%)	72.1	85.8	74.2	95.1	97.2	94.9
Hair Dye Use (%)	46.7	56.6	43.8	58.3	67.8	49.1
Age at Menarche	12.7 ± 1.5	12.7 ± 1.4	12.8 ± 1.4	13.0 ± 1.4	12.7 ± 1.5	12.5 ± 1.3
Gravidity	2.1 ± 2.0	2.3 ± 1.8	2.2 ± 1.9	2.9 ± 2.1	3.3 ± 2.2	3.0 ± 2.5
No. of Miscarriages	0.3 ± 1.0	0.3 ± 0.75	0.3 ± 0.7	0.7 ± 1.1	0.5 ± 0.9	0.4 ± 0.9
No. of Ectopic pregnancies	0.03 ± 0.18	0.03 ± 0.17	0.03 ± 0.18	0.0 ± 0.0	0.03 ± 0.18	0.03 ± 0.17
No. of Abortions	0.4 ± 0.8	0.3 ± 0.7	0.4 ± 0.8	0.1 ± 0.4	0.1 ± 0.4	0.1 ± 0.5
No. of Stillbirths	0.09 ± 0.36	0.01 ± 0.12	0.02 ± 0.16	0.10 ± 0.35	0.04 ± 0.21	0.05 ± 0.26
No. of Live Births	1.2 ± 1.3	1.6 ± 1.4	1.5 ± 1.3	2.0 ± 1.3	2.7 ± 1.7	2.3 ± 1.8

Table 4 (continued)

	Pre menopausal			Post Menopausal		
	SLE Cases	EXISTING Controls	NEW Controls	SLE Cases	EXISTING Controls	NEW Controls
Breast Cancer (%)	0.0	0.4	0.3	3.3	4.0	3.5
Race						
Black (%)	14.1	2.4	2.9	10.0	1.3	3.4
Other (%)	18.2	4.5	9.3	11.7	4.3	5.1
White (%)	67.8	93.1	87.9	78.3	94.3	91.4
Age menopause	–	–	–	39.0 ± 10.6	44.2 ± 8.3	43.4 ± 9.2
surgical (%)	–	–	–	63.6	45.6	42.2
natural (%)	–	–	–	29.1	49.4	41.5
Estrogen Use (%)	–	–	–	49.1	64.3	47.2
Oral contraceptives						
Ever-use (%)	72.1	85.2	69.0	38.3	41.4	33.6
Past-use (%)	60.7	75.1	52.9	38.3	41.4	33.6
Current-use (%)	11.5	10.0	16.2	–	–	–

Table 5. Exogenous hormone use in postmenopausal case and control women.

all adjusted for Race(white, black, other), reference year, age at reference date.

Estrogen use	**New Controls**	**Pre-Existing Controls**
Definite or Probable DX	1.45 (0.83, 2.54)	1.02 (0.52, 2.02)
Definite DX	1.69 (0.94, 3.03)	0.99 (0.49, 2.01)
Estrogen use	**New Controls**	**Pre-Existing Controls**
(adjusted for past OC use)		
Definite or Probable DX	1.30 (0.73, 2.13)	0.97 (0.49, 1.94)
Definite DX	1.51 (0.82, 2.70)	0.94 (0.46, 1.92)
Past Oral contraceptive use	**New Controls**	**Pre-Existing Controls**
Definite or Probable DX	1.09 (0.59, 2.02)	0.49 (0.22, 1.06)
Definite DX	1.06 (0.60, 1.89)	0.48 (0.22, 1.08)
Past Oral contraceptive use	**New Controls**	**Pre-Existing Controls**
(adjusted for past estrogen use)		
Definite or Probable DX	1.14 (0.61, 2.13)	0.53 (0.24, 1.17)
Definite DX	1.09 (0.61, 1.94)	0.53 (0.23, 1.19)

Table 6. Duration of postmenopausal women's past-use of Oral Contraceptive in relation to risk of SLE adjusted for estrogen, Race(white, black, other), reference year, age at reference date

Odds ratios and (95% confidence interval)

Definite or Probable DX	**New Controls**	**Pre-Existing Controls**
Never-used OCs	1.0	1.0
Years OC use <= 5	1.61 (0.61, 4.28)	0.42 (0.13, 1.33)
5 < Years OC use <= 10	1.43 (0.63, 3.21)	0.53 (0.19, 1.47)
10 < Years OC use <= 15	0.59 (0.17, 2.09)	0.38 (0.10, 1.49)
> 15 Years OC use	0.60 (0.17, 2.10)	0.90 (0.30, 2.70)
Test for Trend χ2 (1)	0.52 (p < 0.47)	0.67 (p < 0.41)
Definite DX		
Never-used OCs	1.0	1.0
Years OC use <= 5	1.51 (0.53, 4.29)	0.33 (0.09, 1.24)
5 < Years OC use <= 10	1.25 (0.52, 2.99)	0.58 (0.21, 1.64)
10 < Years OC use <= 15	0.43 (0.10, 1.92)	0.29 (0.06, 1.43)
> 15 Years OC use	0.68 (0.19, 22.39)	1.04 (0.35, 3.09)
Test for Trend χ2 (1)	0.68 (p < 0.41)	0.40 (p < 0.53)

Table 7. Premenopausal Oral Contraceptive use in relation to risk of SLE

Odds ratios (95% confidence intervals)

adjusted for Race(white, black, other), reference year, age at reference date

Definite or Probable DX	**New Controls**	**Pre-Existing Controls**
Never-use	1.0	1.0
Ever-use	1.51 (0.98, 2.36)	**0.57 (0.42, 0.76)**
Past-use	**1.71 (1.08, 2.71)**	0.66 (0.37, 1.17)
Current-use	0.66 (0.33, 1.32)	**0.34 (0.14, 0.80)**
Definite DX		
Never-use	1.0	1.0
Ever-use	1.54 (0.98, 2.42)	0.62 (0.35, 1.09)
Past-use	**1.70 (1.06, 2.74)**	0.64 (0.35, 1.15)
Current-use	0.71 (0.35, 1.43)	0.36 (0.15, 1.17)

Table 8. Duration of premenopausal use of Oral Contraceptives in relation to risk for SLE with 'NEW' Controls adjusted for Race(white, black, other), reference year, age at reference date

Odds ratios (95% confidence interval)

Definite or Probable DX	**Ever-use**	**Past-use**	**Current-use**
Never-used OCs	1.0	1.0	1.0
Years OC use <= 1	1.18 (0.65, 2.14)	1.44 (0.77, 2.72)	0.53 (0.11, 2.50)
1 < Years OC use <= 5	1.40 (0.86, 2.29)	**1.85 (1.09, 3.14)**	0.70 (0.28, 1.77)
5 < Years OC use <= 10	0.79 (0.42, 1.49)	0.95 (0.47, 1.91)	0.54 (0.15, 1.90)
10 < Years OC use	1.36 (0.67, 2.75)	1.85 (0.84, 4.09)	0.61 (0.13, 2.84)
Test for Trend $\chi 2$ (1)	0.14 ($p < 0.71$)	1.71 ($p < 0.19$)	1.60 ($p < 0.21$)
Definite DX			
Never-used OCs	1.0	1.0	1.0
Years OC use <= 1	1.12 (0.60, 2.08)	1.34 (0.69, 2.59)	0.57 (0.12, 2.71)
1 < Years OC use <= 5	1.33 (0.80, 2.21)	1.73, (0.99, 3.01)	0.73 (0.29, 1.84)
5 < Years OC use <= 10	0.84 (0.44, 1.59)	1.01 (0.50, 2.04)	0.57 (0.16, 2.00)
10 < Years OC use	1.47 (0.72, 2.99)	2.00 (0.90, 4.43)	0.66 (0.14, 3.06)
Test for Trend $\chi 2$ (1)	0.32 ($p < 0.57$)	2.06 ($p < 0.15$)	1.24 ($p < 0.27$)

Table 9. Duration of Premenopausal use of Oral Contraceptives in relation to risk of SLE 'Pre-Existing' Controls

adjusted for race(white, black, other), reference year, age at reference date

Odds ratios (95% confidence intervals)

Definite or Probable DX	**Ever-use**	**Past-use**	**Current-use**
Never-used OCs	1.0	1.0	1.0
Years OC use <= 1	**0.47 (0.23, 0.99)**	0.46 (0.21, 1.01)	0.91 (0.14, 6.04)
1 < Years OC use <= 5	0.57 (0.32, 1.03)	0.63 (0.34, 1.18)	0.60 (0.17, 2.03)
5 < Years OC use <= 10	**0.35 (0.17, 0.72)**	**0.36 (0.17, 0.77)**	0.40 (0.10, 1.68)
10 < Years OC use	0.54 (0.22, 1.31)	0.82 (0.33, 2.09)	---
Test for Trend $\chi 2$ (1)	5.47 ($p < 0.02$)	2.63 ($p < 0.11$)	6.74 ($p < 0.01$)
Definite DX			
Never-used OCs	1.0	1.0	1.0
Years OC use <= 1	**0.42 (0.19, 0.91)**	**0.39 (0.17, 0.91)**	0.98 (0.15, 6.62)
1 < Years OC use <= 5	**0.53 (0.29, 0.99)**	0.57 (0.30, 1.10)	0.64 (0.19, 2.20)
5 < Years OC use <= 10	**0.38 (0.18, 0.77)**	**0.38 (0.18, 0.83)**	0.43 (0.10, 1.81)
10 < Years OC use	0.57 (0.23, 1.41)	0.88 (0.34, 2.25)	---
Test for Trend $\chi 2$ (1)	4.18 ($p < 0.04$)	1.89 ($p < 0.17$)	5.91 ($p < 0.02$)

Table 10. CTD Incidence Rates / 100,000 for White women in King County Washington

		systemic lupus erythematosus		systemic sclerosis CREST		Sjogren's syndrome		Polymyositis Dermatomyositis	
AGE	Person Years	# Cases	Rate	# Cases	Rate	# Cases	Rate	# Cases	Rate
18-29	1,131,140	37	3.27	4	0.35	4	0.35	1	0.09
30-39	1,047,504	53	5.06	8	0.76	25	2.39	6	0.57
40-49	732,988	32	4.37	15	2.05	38	5.18	6	0.82
50-59	493,132	24	4.87	11	2.23	32	6.49	4	0.81
60-69	455,729	9	1.97	16	3.51	45	9.87	3	0.66
70-74	185,436	6	3.24	7	3.77	12	6.47	2	1.08
TOTAL	4,045,933	161	3.98	59	1.46	156	3.86	22	0.54
US 1990 adj.rate			3.93		1.62		4.15		0.56

Table 11. CTD Incidence Rates / 100,000 for Black women in King County Washington

		systemic lupus erythematosus		systemic sclerosis CREST		Sjogren's syndrome		Polymyositis Dermatomyositis	
AGE	Person Years	# Cases	Rate	# Cases	Rate	# Cases	Rate	# Cases	Rate
18-29	69,218	10	14.45	1	1.45	2	2.89	0	0.00
30-39	54,254	16	29.45	1	1.84	3	5.53	2	3.69
40-49	31,775	9	28.33	0	0.00	3	9.44	0	0.00
50-59	20,387	3	14.72	0	0.00	1	4.90	0	0.00
60-69	17,685	3	16.96	0	0.00	1	5.66	0	0.00
70-74	6,052	0	0.00	0	0.00	0	0.00	0	0.00
TOTAL	199,371	41	20.57	2	1.00	10	5.02	2	1.00
US 1990 adj.rate			20.45		0.94		5.11		0.96

Table 12. CTD Incidence Rates / 100,000 for Total King County Washington population.

		systemic lupus erythematosus		systemic sclerosis CREST		Sjogren's syndrome		Polymyositis Dermatomyositis	
AGE	Person Years	# Cases	Rate	# Cases	Rate	# Cases	Rate	# Cases	Rate
18-29	1,316,758	62	4.71	5	0.38	7	0.53	1	0.08
30-39	1,221,248	84	6.88	10	0.82	31	2.54	9	0.74
40-49	820,826	49	5.97	15	1.83	45	5.48	6	0.73
50-59	560,674	34	6.06	14	2.50	38	6.78	6	1.07
60-69	519,971	16	3.08	19	3.65	49	9.42	4	0.77
70-74	205,872	7	3.40	7	3.89	14	6.80	2	0.97
TOTAL	4,645,350	252	5.44	70	1.51	184	3.96	28	0.60
US 1990 Age Adjust			5.35		1.65		4.25		0.60

Table 13. Summary of selected systemic lupus erythematosus (SLE) incidence studies (per 100,000 population)

Study Location	Ref	Study Period	White	Black
Jefferson Co. Al.	60	1956-1965	1.06	2.7
NYC, NY	61	1956-1965	2.6	7.5
Rochester, Minn.	62	1950-1979	2.5	
Baltimore, MD	63	1970-1977	3.9	10.5
Iceland	64	1975-1984	5.9	
So. Sweden	65	1981-1986	5.4	
Nottingham, UK	66	1989-1990	6.1	
Curagao	67	1980-1989		7.9
Birmingham, UK	68	1991	4.3	25.8
Allegheny, Co. PA	59	1985-1990	5.0	12.5
Present Study		1983-1991	4.0	20.6

Table 14. Summary of selected systemic sclerosis /CREST incidence studies (per 1,000,000 population)

Study Location	Ref	Study Period	White	Black
Allegheny, PA	72	1963-1982	19.9	22.6
Rochester, Minn.	62	1950-1979	16.0	
Michigan	73	1980-1991	12.8	22.5
Iceland	74	1975-1990	7.0	
Present study		1983-1991	14.6	10.0

Table 15 . Summary of selected polymyositis / dermatomyositis incidence studies (per 1,000,000 population)

Study Location	Ref	Study Period	White	Black
Allegheny, PA	69	1963-1982	3.4	12.4
Shelby, TN	70	1947-1963	2.7	10.8
Israel	71	1960-1976	2.2	
Present study		1983-1991	5.4	10.0

APPENDIX B: C++ FREQUENCY MATCHING PROGRAM

```
/* Risk Factors for Connective Tissue Diseases Study
   Random Digit Dialing Frequency Matching Tool - RFCTDsRDDFMT
*/

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

enum Boolean {False, True};
enum toDo {Help, Initialize, bootStrap, selectSF, countCtrls, responseRate, freqMatch, quit};
enum sampleFractionMethod {calculateSF, modifySF, historicalSF};

const int toDoCmdNum = 7;
const int helpCmdNum = 5;
const int maxWaveInt = 9;
const int numStudyYears = 9;
const int numAgeCats = 12;
const int numAgeNowCats = 15;
const int studyBaseYear = 1983;
const int currentYear = 1995;
const int numYearsPerCell = 5;

int getRDDInfo();
void printMenu();
int getValidResponse(int);
int frequencyMatch();
int helpFunction();
int calculateInitialSampleFraction();
int selectSampleFraction();
```

```
int simulateSampleFraction();
Boolean calculatePermissibleCells(int, int, int, int *, int *);
int bootstrapSampleFraction();
int simBoot(Boolean);
int countControls();
Boolean sampleFractionSelected = False;
int calculateResponseRate();

float targetCtrlToCaseFMratio = 2.0;
int whatToDo; // used for state info of tasks
int freqMatchRatio = 3;  // 1 for filling cells to make sure all in
int modifySampleFractionFlag;
int sampleFractionCalculationMethod;
int numEstimatedControls;
int numCompletsEachAgeGroupNeeded[16]; // This needs to be global so we can access when do RDD frequency matching
float ProbCompleteQuest[16]; // NOTE: This can be an array if the prob changes by age to answer questionnaire
int intArray[1001];
float floatArray[1001];
int ageCatLow[16], ageCatHigh[16], CompletesNeeded[16], caseDead[16];
float sampleFraction[16], caseAgeSpecificMortalityRate[16], modSampFraction[16];
int caseYrDX_AgeAtDX[10][16];
int caseYrDX_AgeDXprojected[10][16];
int caseYrDX_AgeNow[10][16];
int controlYrKC_AgeNow[10][16], ProjControlYrKC_AgeNow[10][16];
int controlYrKC_AgeThen[10][16], controlsToDateThen[16];
int caseYrDX_AgeNowDead[10][16], ProjCaseYrDX_AgeNowDead[10][16];
float sampleFractionMatrix[10][16];

int projectedCases[10][16], projectedControls[10][16], projectedControlsThen[10][16];
float SFmatrix[10][16]; // used for simulation and bootstraping

float percentCasesAcertainedToDate, percentControlsAcertainedToDate;
long int iseed = -17;

FILE *infoFptr, *infoFptrOut, *selInfoFptr, *notSelInfoFptr;
```

```
typedef struct { int row, col; // just to make sure algorithm is correct  row (x) = yrs KC,  col (y) = age group
                 int ageCatNum[5];// actual number;
                 int ageCatNumThen[5];
                 int simTargetCtrlsNeeded;
                 int simCtrlsSel;
                 int ageCatSel[5];// projected number;
                 int simAgeCatNum[5]; // projected number
                 int simTotalCtrls; // projected number
                 int simTotalSel; // projected
                 int numCases, adjustedCtrls; // projected
                 float initialSF, adjustedSF;
               } sampFractCell;

sampFractCell SFMatrixInfo[10][16];

typedef struct { int row, col;
                 int ageCatCaseNum[5]; // actual number cases projected
                 int TotalCasesToDate;
                 int TotalCasesProj; //= sum(ageCatCaseNum[0...4]
                 int simTargetCtrls; //number of = num cases * response * fudge factor
                 int simCtrlsNeeded;
                 int simCtrlsSel;
                 float targetInflationFactor;
                 float fudgeFactor;
               } simulationStruct;

simulationStruct simMatrix[10][16];

typedef struct { int casesProj;
                 int targetCtrlNum;
                 int numCtrlSel;
                 int actualSel;
                 int numCtrlNeed;
                 int numCtrlReturn;
                 float sampleFraction;
                 float responseRate;
               } responseRateStruct;
```

```
responseRateStruct responseMatrix[10][16];

typedef struct { int indxi, indxj, ageThenIJ;
                 } permissibleCell;

permissibleCell permisCell[10]; // cells that a ctrl can possibley be used in

typedef struct { int numAdjustedCtrls;
                   float adjustedSF;
                 } simCell;

typedef struct {simCell simCellMatrix[10][16];
                 } simCellArray;

simCellArray *simCellPtr[1001];

typedef struct { int ctrlsLow, ctrlsHigh, casesEst;
                   float avgCtrls, sf, sfHigh, sfLow;
                 } bootMat;

bootMat bootMatrix[10][16];
bootMat bootMatrix2[10][16];
bootMat bootMatrix3[10][16];
bootMat bootMatrix100[10][16];


typedef struct { int i, j, k;
                   int age, year,yr1stKC, yrLastKC, birthYr;
                 } simBootStruct;

simBootStruct simBootInfo[3001];

int simBootOrder[3001];  // used to simulate and bootstrap SF
```

```
main()
   { Boolean done;

     printf("RFCTDs RDD Frequency Match Tool ver. 1.0 5/3/95 \n");
     done = False;
     do { printMenu();
          whatToDo = getValidResponse(toDoCmdNum);
          switch  (whatToDo)
                  {  case Help:  { helpFunction();
                                   } break;

                     case Initialize:  { getRDDinfo();
                                         } break;

                     case bootStrap: { calculateInitialSampleFraction();
                                       } break;

                     case selectSF:  { selectSampleFraction();
                                       } break;

                     case countCtrls: { countControls();
                                        } break;

                     case responseRate: { calculateResponseRate();
                                          } break;

                     case freqMatch: {frequencyMatch();
                                       } break;

                     case quit:   done = True;
                                  break;
                  }
        }
     while (done == False);

     return 0;
   }
```

```
void printMenu()
    {  printf("\nMENU \n  Enter one of the following integers for function to be executed ");
       printf("\n 0: Help ");
       printf("\n 1: Initialize / Setup ");
       printf("\n 2: Initial Simulation Sample Fraction ");
       printf("\n 3: Sample Fraction Exploration");
       printf("\n 4: Count Controls Selected so Far");
       printf("\n 5: Calculate Response Rate");
       printf("\n 6: Frequency Match");
       printf("\n 7: Quit");
       printf("\n Please Enter Response: ");
    }


int getValidResponse(int maxCmdInt) {
    int command;
    while (True) {
       command = getche();
       command  = command - 48;
       if ((command >= 0) && (command <= maxCmdInt))
          return (command);
       else {
          printf("\n illegal command: %i Please enter Valid Response ( 0 - 3 )", command);
          printMenu();
       }
    }
    return -1;
}


void printHelpFile(int filenum) {
    struct helpFileNames {char filenam[13];};

    helpFileNames helpFiles[9];
    // array of filenames array of chars
    char helpFileArr[13][10];
  }
```

```
void printHelpMenu()
    {  printf("\nMENU \n  Enter one of the following numbers for information about topic ");
       printf("\n 0: Help on Help ");
       printf("\n 1: Help on the format of the RDDINFO.DAT run control file");
       printf("\n 2: Calculate number of Controls Needed ");
       printf("\n 3: Frequency Match Options");
       printf("\n 4: Quit");
       printf("\n Please Enter Response: ");
    }

int helpFunction()
    { enum help {help, info, ctlneed, freqmtch, quit};
      Boolean done = False;
      int command;
      do
        { printf("\n HELP MENU: enter number of HELP Topic you wish displayed\n" );
          printHelpMenu();
          command = getValidResponse(helpCmdNum);
          switch (command)
             { case help: { printHelpFile(help);
                            break;
                          }
               case info: { printHelpFile(info);
                            break;
                          }
               case quit: { done = True;
                            break;
                          }
             }
        }
      while (done == False);
      return 0;
    }
```

```
int getRDDinfo()
   { const int numDiseases = 7;
     int result;
     int caseDX;
     char tempString[128];
     enum gender{female, male};
     int i, j; // for loop iteration var
     int numAgeCats;
     int ageAtDXindex;
     int casesToDate[16];
     float ProbRandPhNumResidential, ProbWoman18to74atNum, ProbAgreeReceiveLetter;
     //const float constMult = 1.99;
     float PopDistWomen[16];
     float constantMultiplier;
     float productVal[16];
     long int NumRDDcallsNeeded[16];
     int caseDX_caseYrDX[10][10];
     // char s1[10],s2[10], s3[10], s5[10];
     int caseDeadFlag, caseDXyear;
     int ageAtDX, caseAgeNow, ageNowIndex, caseDXyearIndex;
     long int caseID;

     printf("\n Get RDD info\n");
     // first oprn the info file
     result = 1;
     if ((infoFptr = fopen("../RDDINFO.DAT\0", "rt")) == NULL)
        { printf("ERROR: unable to open file rddInfo.dat \n");
          exit(0);
        }
     else
        { printf("File rddInfo.dat opened successfully\n"); // FILE opened OK!
          result = 0;  // return good status
          // now get the information on the distribution
          // calculate constant multiplier
          // constMultiplier = ProbRandPhNumResidential * ProbWoman18to74atNum * ProbContactAgreeReceiveLetter
          fgets(tempString, 128, infoFptr);
          //  Write print errmsg functionif (feof(infoFptr)) printErrMsg(10, ExitYes);
```

```
sscanf(tempString, "%i %f %f %f", &numAgeCats, &ProbRandPhNumResidential, &ProbWoman18to74atNum, &ProbAgreeReceiveLetter) ;
constantMultiplier = ProbRandPhNumResidential * ProbWoman18to74atNum * ProbAgreeReceiveLetter;
printf("\n calculated constant multiplier = %f",constantMultiplier);
// now get the lower age range categories
fgets(tempString, 128, infoFptr);
//if (feof(infoFptr)) printErrMsg(10, ExitYes);   fix this
sscanf(tempString, "%i %i %i %i %i %i %i %i %i %i %i %i %i %i %i",
                        &ageCatLow[0],&ageCatLow[1],&ageCatLow[2],&ageCatLow[3],&ageCatLow[4],&ageCatLow[5],&ageCatLow[6],&ageCatLow[7],
                        &ageCatLow[8],&ageCatLow[9],&ageCatLow[10], &ageCatLow[11],&ageCatLow[12], &ageCatLow[13], &ageCatLow[14]);
// now get the lower age range categories
fgets(tempString, 128, infoFptr);
//if (feof(infoFptr)) printErrMsg(10, ExitYes);
sscanf(tempString,"%i %i %i %i %i %i %i %i %i %i %i %i %i %i %i", &ageCatHigh[0], &ageCatHigh[1], &ageCatHigh[2],
                        &ageCatHigh[3], &ageCatHigh[4], &ageCatHigh[5], &ageCatHigh[6], &ageCatHigh[7],
                        &ageCatHigh[8], &ageCatHigh[9], &ageCatHigh[10], &ageCatHigh[11], &ageCatHigh[12], &ageCatHigh[13], &ageCatHigh[14]);
 // now get the number of  cases acertained so far in each age category
fgets(tempString, 128, infoFptr);
//if (feof(infoFptr)) printErrMsg(10, ExitYes);
//sscanf(tempString,"%i %i %i %i %i %i %i %i %i %i %i %i %i", &casesToDate[0],&casesToDate[1], &casesToDate[2],
//                        &casesToDate[3], &casesToDate[4], &casesToDate[5], &casesToDate[6],&casesToDate[7],
//                        &casesToDate[8], &casesToDate[9], &casesToDate[10], &casesToDate[11], &casesToDate[12]);
// now get the % of the cases acertained to date
fgets(tempString, 128, infoFptr);
//if (feof(infoFptr)) printErrMsg(10, ExitYes);
sscanf(tempString, "%f %f", &percentCasesAcertainedToDate, &percentControlsAcertainedToDate);
printf("\n Percent cases to date %f1.2  Percent ctrls to date %f1.2", percentCasesAcertainedToDate, percentControlsAcertainedToDate);
 // Get the population distribution of women per age group
fgets(tempString, 128, infoFptr);
//if (feof(infoFptr)) printErrMsg(10, ExitYes);
sscanf(tempString, "%f %f %f %f %f %f %f %f %f %f %f %f %f %f %f", &PopDistWomen[0], &PopDistWomen[1],
                        &PopDistWomen[2], &PopDistWomen[3], &PopDistWomen[4], &PopDistWomen[5], &PopDistWomen[6], &PopDistWomen[7],
                        &PopDistWomen[8], &PopDistWomen[9], &PopDistWomen[10], &PopDistWomen[11], &PopDistWomen[12], &PopDistWomen[13],
                        &PopDistWomen[14]);
// get the probability of control receiving questionnaire will complete it
fgets(tempString, 128, infoFptr);
//if (feof(infoFptr)) printErrMsg(10, ExitYes);
sscanf(tempString, "%f %f %f %f %f %f %f %f %f %f %f %f %f %f %f", &ProbCompleteQuest[0], &ProbCompleteQuest[1],
```

```
      &ProbCompleteQuest[2], &ProbCompleteQuest[3], &ProbCompleteQuest[4], &ProbCompleteQuest[5], &ProbCompleteQuest[6],
      &ProbCompleteQuest[7], &ProbCompleteQuest[8], &ProbCompleteQuest[9], &ProbCompleteQuest[10], &ProbCompleteQuest[11],
      &ProbCompleteQuest[12], &ProbCompleteQuest[13], &ProbCompleteQuest[14]); // NOTE: CAN meke array if the prob is different by age

// get the modify sample fraction method flag
fgets(tempString, 128, infoFptr);
//if (feof(infoFptr)) printErrMsg(10, ExitYes);
sscanf(tempString,"%i", &modifySampleFractionFlag);

switch (modifySampleFractionFlag)
   { case calculateSF: { break;}

     case modifySF: { fgets(tempString, 128, infoFptr);
                      //if (feof(infoFptr)) printErrMsg(10, ExitYes);
                      sscanf(tempString, "%f %f %f %f %f %f %f %f %f %f %f %f %f %f", sampleFraction[0], sampleFraction[1],
                             sampleFraction[2], sampleFraction[3], sampleFraction[4], sampleFraction[5], sampleFraction[6], sampleFraction[7],
                             sampleFraction[8], sampleFraction[9], sampleFraction[10], sampleFraction[11], sampleFraction[12],
                          sampleFraction[13], sampleFraction[14]);
                    }; break;

     case historicalSF: { break;
                        }
   }

// read in the dead case age specific rates seperate file or put into the rddinfo file
fgets(tempString, 128, infoFptr);
//if (feof(infoFptr)) printErrMsg(10, ExitYes);
sscanf(tempString, "%f %f %f %f %f %f %f %f %f %f %f %f %f %f %f", &caseAgeSpecificMortalityRate[0],
                           &caseAgeSpecificMortalityRate[1], &caseAgeSpecificMortalityRate[2], &caseAgeSpecificMortalityRate[3],
                           &caseAgeSpecificMortalityRate[4], &caseAgeSpecificMortalityRate[5], &caseAgeSpecificMortalityRate[6],
                           &caseAgeSpecificMortalityRate[7], &caseAgeSpecificMortalityRate[8], &caseAgeSpecificMortalityRate[9],
                           &caseAgeSpecificMortalityRate[10], &caseAgeSpecificMortalityRate[11], &caseAgeSpecificMortalityRate[12],
                           &caseAgeSpecificMortalityRate[13], &caseAgeSpecificMortalityRate[14]);

// close the input file
fclose(infoFptr);          // should check for error  on file close
// Now Have all the input stuff we need so make a table and check it out
```

```
/* Now Lets do this by reading in the number of cases found to date, take their age
   distribution and recalculate the numbers
   Can meke this a case later and have multiple options
*/
// initialize all the matrices
for (i = 0; i < numAgeNowCats; i++)
   casesToDate[i] = 0;
for (i = 0; i < numStudyYears; i++) // reinitialize the array will actually count cases to date
   { for (j = 0; j < numAgeNowCats; j++)
        { caseYrDX_AgeAtDX[i][j] = 0;
          caseYrDX_AgeNow[i][j] = 0;
          controlYrKC_AgeNow[i][j] = 0;
          caseYrDX_AgeNowDead[i][j] = 0;
          sampleFractionMatrix[i][j] = 0;
        }
   }
for ( i = 0; i < numDiseases; i++)    // initialize the array that holds the year and number of dxes
   { for (j = 0; j < numStudyYears; j++)
        { caseDX_caseYrDX[i][j] = 0;
        }
   }
// open up the case file and count up number of cases adjust for controls by adding 3 yrs to dx date so can get right number of controls
if ((infoFptr = fopen("../CASE.DAT\0", "rt")) == NULL)
 { printf("\n ERROR: unable to open file Case.dat \n");
    exit(0);
 }
else
 { printf("\n File case.dat opened successfully\n"); // FILE opened OK!
    result = 0;  // return good status

    if ((infoFptrOut = fopen("../caseSmry.dat\0", "wt")) == NULL)
       { printf("\n ERROR: unable to open case summry file CaseSmry.dat \n");
         exit(0);
       }
    else
       { printf("\n File case.dat opened successfully\n"); // FILE opened OK!
         result = 0;  // return good status
```

```
  }

while (feof(infoFptr) == 0)
  { fgets(tempString, 128, infoFptr);
    if (feof(infoFptr) != 0)
       break;
    // NOTE: CASE DEAD FLAG NOT IN DB YET!!!!!! 3/21/95
    sscanf(tempString, "%li %i %i %i", &caseID, &ageAtDX, &caseDXyear, &caseDX /*&caseDeadFlag, */ );
    // add count to specific age category
    caseDeadFlag = False;
    if ((ageAtDX < 18) || (ageAtDX > 74) || (ageAtDX == 999) || (caseDXyear == 999))
       { if (ageAtDX < 18) printf("\n ERROR illegal age at DX < 18 %s", tempString);
         if (ageAtDX > 74) printf("\n ERROR illegal age at DX > 74 %s", tempString);
         if (ageAtDX == 999) printf("\n ERROR illegal age at DX = 999 %s", tempString);
         if (caseDXyear == 999) printf("\n ERROR illegal case DXyear = 999 %s", tempString);
       }
     else
       { if (caseDX < 1 || caseDX > 6)
          printf("\n Warning illegal Case Diagnosis = %i", caseDX);
         // figure out which age@DX_YearDX each case belongs in
         caseDXyearIndex = caseDXyear - studyBaseYear;
         if ((caseDXyearIndex < 0) || (caseDXyearIndex > 9))
            printf("\nERROR: case DX year < 1983 or > 1991; %s", tempString); // check for nonsense input in data
         for (i = 0; i < numAgeNowCats; i++)
            { ageAtDXindex = -1;
              if ((ageAtDX >= ageCatLow[i]) && (ageAtDX <= ageCatHigh[i]))
                 { ageAtDXindex = i;
                   //casesToDate[i]++;
                   if (caseDeadFlag == True)
                     caseDead[i]++;  // keep track of num Dead Cases
                   break;
                 }
            }
          if (ageAtDXindex == -1)
            printf("\n ERROR: case age at DX > 74");
          else
            { caseYrDX_AgeAtDX[caseDXyearIndex][ageAtDXindex]++;  // update count in
```

```
                caseDX_caseYrDX[caseDX][caseDXyearIndex]++;  // update which dx in what year
              }
            caseAgeNow = ageAtDX + (currentYear - caseDXyear);
            for (i = 0; i < numAgeNowCats; i++)
               { ageNowIndex = -1;
                 if ((caseAgeNow >= ageCatLow[i]) && (caseAgeNow <= ageCatHigh[i]))
                   { ageNowIndex = i;
                     casesToDate[i]++;
                     break;
                   }
               }
            if ( ageNowIndex == -1)
              printf("\n ERROR: case age now > 86");
            else caseYrDX_AgeNow[caseDXyearIndex][ageNowIndex]++;
          }
      }
    fclose(infoFptr);          // should check for error  on file close
  }
// 1st calculate the projected number of total cases will have at study end
for (i=0; i < numAgeNowCats; i++)
   CompletesNeeded[i] = casesToDate[i] / percentCasesAcertainedToDate;
// 2nd calculate the product of constmult * prob complete * popDistrib
for (i=0; i < numAgeNowCats; i++)
   productVal[i] = constantMultiplier * PopDistWomen[i] * ProbCompleteQuest[i];
// 3rd calculate the number of primary RDD calls needed
for (i=0; i < numAgeNowCats; i++)
    NumRDDcallsNeeded[i] = CompletesNeeded[i] / productVal[i];
// 4th calculate the age specific proportion of dead cases
/* for now until get more case info use what is entered in the rddinfo file
for (i=0; i < numAgeNowCats; i++)
   { if (casesToDate[i] > 0)   // check for division by Zero (0)
       caseAgeSpecificMortalityRate[i] = ceil(caseDead[i] / casesToDate[i]);
   }
*/
// 5th print out the results to the file
if ((infoFptr = fopen("FreqMtch.out\0", "wt")) == NULL)
  { printf("ERROR: unable to open file freqMtch.out \n");
```

```
    exit(0);
  }
fprintf(infoFptr, "\n  Age  Completes Pop'n   Prob Complete Const Product  Num Prim ");
fprintf(infoFptr, "\nNeeded  Needed   Distb   Questionnaire  Mult         RDD need ");
fprintf(infoFptr, "\n-------------------------------------------------------------------");
for (i=0; i < numAgeNowCats; i++)
   { /* Do the table which given the number of complete control questionnaires needed
        and the population information will calculate the number of the primary RDD calls needed
     */
     fprintf(infoFptr, "\n %i-%i  %3i     %1.4f    %1.2f     %1.5f  %1.5f   %li",
             ageCatLow[i], ageCatHigh[i], CompletesNeeded[i], PopDistWomen[i], ProbCompleteQuest[i],
             constantMultiplier, productVal[i], NumRDDcallsNeeded[i]);
   }

// now we need to print out the caseDXyear_Age@DX
fprintf(infoFptr, "\n case DX Year _ Age@DX matrix ");
fprintf(infoFptr, "\n       18-19 20-24 25-29 30-34 35-39 40-44 45-49  50-54 55-59 60-64 65-69 70-74");

for (i = 0; i < numStudyYears; i++)
   { fprintf(infoFptr, "\n %i", studyBaseYear + i);
     for (j = 0; j < numAgeCats; j++)
        fprintf(infoFptr, "  %4i", caseYrDX_AgeAtDX[i][j]);
   }

// now we need to print out the  caseDXyear_AgeNow
fprintf(infoFptr, "\n case DX Year _ AgeNow matrix ");
fprintf(infoFptr, "\n       18-9 20-4 25-9 30-4 35-9 40-4 45-9 50-4 55-9 60-4 65-9 70-4 75-9 80-4 85-6");
for (i = 0; i < numStudyYears; i++)
   { fprintf(infoFptr, "\n %i", studyBaseYear + i);
     for (j = 0; j < numAgeNowCats; j++)
        fprintf(infoFptr, "  %3i", caseYrDX_AgeNow[i][j]);
   }

// now we need to print out the  caseDXyear_Age@DX projected matrix
 fprintf(infoFptr, "\n case DX Year _ Age@DX Projected at study End Matrix ");
 fprintf(infoFptr, "\n       18-9 20-4 25-9 30-4 35-9 40-4 45-9 50-4 55-9 60-4 65-9 70-4");
 for (i = 0; i < numStudyYears; i++)
```

```
        { fprintf(infoFptr, "\n %i", studyBaseYear + i);
          for (j = 0; j < numAgeCats; j++)
             { caseYrDX_AgeDXprojected[i][j] = ceil(caseYrDX_AgeAtDX[i][j] / percentCasesAcertainedToDate);
               fprintf(infoFptr, "  %3i", caseYrDX_AgeDXprojected[i][j]);
             }
        }

      fprintf(infoFptr, "\n case DX _ Case DX Year Matrix ");
      fprintf(infoFptr, "\n                          1983 1984 1985 1986 1987 1988 1989 1990 1991");
      fprintf(infoFptr, "\n Systemic Lupus Erythematosus  ");
      for (i = 0; i < numStudyYears; i++)
         fprintf(infoFptr, "  %3i", caseDX_caseYrDX[2][i]);
      fprintf(infoFptr, "\n Scleroderma / CREST          ");
      for (i = 0; i < numStudyYears; i++)
         fprintf(infoFptr, "  %3i", caseDX_caseYrDX[1][i]);
      fprintf(infoFptr, "\n Sjogren's Syndrome           ");
      for (i = 0; i < numStudyYears; i++)
         fprintf(infoFptr, "  %3i", caseDX_caseYrDX[3][i]);
      fprintf(infoFptr, "\n Mixed /Undifferentiated CTD   ");
      for (i = 0; i < numStudyYears; i++)
         fprintf(infoFptr, "  %3i", caseDX_caseYrDX[4][i] + caseDX_caseYrDX[5][i]);
      fprintf(infoFptr, "\n Polymyositis / Dermatomyositis");
      for (i = 0; i < numStudyYears; i++)
         fprintf(infoFptr, "  %3i", caseDX_caseYrDX[6][i]);

      fclose(infoFptrOut);
      fclose(infoFptr);
    }
  return result;
}


#define IM1 2147483563L
#define IM2 2147483399L
#define AM (1.0/IM1)
#define IMM1 (IM1-1)
```

```
#define IA1 40014L
#define IA2 40692L
#define IQ1 53668L
#define IQ2 52774L
#define IR1 12211
#define IR2 3791
#define NTAB 32
#define NDIV (1+IMM1/NTAB)
#define EPS 1.2e-7
#define RNMX (1.0-EPS)

float rand2(long *idum)
{
    int j;
    long k;
    static long idum2=123456789L;
    static long iy=0;
    static long iv[NTAB];
    float temp;

    if (*idum <= 0)
      { if (-(*idum) < 1)
            *idum = 1;
        else *idum = -(*idum);
        idum2=(*idum);
        for (j=NTAB+7;j>=0;j--)
            { k=(*idum)/IQ1;
              *idum=IA1*(*idum-k*IQ1)-k*IR1;
              if (*idum < 0) *idum += IM1;
              if (j < NTAB) iv[j] = *idum;
            }
        iy=iv[0];
      }

    k =(*idum)/IQ1;  // Start here if not initializing
    *idum=IA1*(*idum-k*IQ1)-k*IR1;
    if (*idum < 0) *idum += IM1;
```

```
    k =idum2/IQ2;
    idum2=IA2*(idum2-k*IQ2)-k*IR2;
    if (idum2 < 0) idum2 += IM2;
    j=iy/NDIV;
    iy=iv[j]-idum2;
    iv[j] = *idum;
    if (iy < 1) iy += IMM1;
    if ((temp=AM*iy) > RNMX)
      return RNMX;
    else return temp;
}
#undef IM1
#undef IM2
#undef AM
#undef IMM1
#undef IA1
#undef IA2
#undef IQ1
#undef IQ2
#undef IR1
#undef IR2
#undef NTAB
#undef NDIV
#undef EPS
#undef RNMX

#define NR_END 1
#define FREE_ARG char*
void nrerror(char error_text[])
/* Numerical Recipes standard error handler */
{
    fprintf(stderr,"Numerical Recipes run-time error...\n");
    fprintf(stderr,"%s\n",error_text);
    fprintf(stderr,"...now exiting to system...\n");
    exit(1);
}
```

```
int *ivector(long nl, long nh)
/* allocate an int vector with subscript range v[nl..nh] */
{
    int *v;

    v=(int *)malloc((size_t) ((nh-nl+1+NR_END)*sizeof(int)));
    if (!v) nrerror("allocation failure in ivector()");
    return v-nl+NR_END;
}
 void free_ivector(int *v, long nl, long nh)
/* free an int vector allocated with ivector() */
{
    free((FREE_ARG) (v+nl-NR_END));
}

//#define NRANSI
#define SWAP(a,b) temp=(a);(a)=(b);(b)=temp;
#define M 7
#define NSTACK 150

void sort(unsigned long n, float arr[])
{
    unsigned long i, ir=n,j,k,l=1;
    int jstack=0,*istack;
    float a, temp;

    istack = ivector(1,NSTACK);
    for (;;)
        { if (ir-l < M)
            {  for (j=l+1;j<=ir;j++)
                  { a=arr[j];
                    for (i=j-1;i>=1;i--)
                        { if (arr[i] <= a) break;
                          arr[i+1]=arr[i];
                        }
                    arr[i+1]=a;
                  }
```

```
    if (jstack == 0) break;
    ir=istack[jstack--];
    l=istack[jstack--];
  }
else
  { k = (l+ir) >> 1;
    SWAP(arr[k],arr[l+1])

    if (arr[l+1] > arr[ir])
       { SWAP(arr[l+1],arr[ir])
       }
    if (arr[l] > arr[ir])
      { SWAP(arr[l],arr[ir])
      }
    if (arr[l+1] > arr[l])
      { SWAP(arr[l+1],arr[l])
      }

    i=l+1;
    j=ir;
    a=arr[l];
    for (;;)
       { do i++; while (arr[i] < a);
         do j--; while (arr[j] > a);
         if (j < i) break;
         SWAP(arr[i],arr[j])
       }
    arr[l]=arr[j];
    arr[j]=a;
    jstack += 2;
    if (jstack > NSTACK)
      printf("\nNSTACK too small in sort.");
    if (ir-i+1 >= j-l)
      { istack[jstack]=ir;
        istack[jstack-1]=i;
        ir=j-1;
      }
```

```
            else
              { istack[jstack]=j-1;
                istack[jstack-1]=l;
                l=i;
              }
          }
      }
    free_ivector(istack,1,NSTACK);
}

void intSort(unsigned long n, int arr[])
{
    unsigned long i,ir=n,j,k,l=1;
    int jstack=0,*istack;
    int a, temp;

    istack = ivector(1,NSTACK);
    for (;;) {
        if (ir-l < M) {
            for (j=l+1;j<=ir;j++) {
                a=arr[j];
                for (i=j-1;i>=1;i--) {
                    if (arr[i] <= a) break;
                    arr[i+1]=arr[i];
                }
                arr[i+1]=a;
            }
            if (jstack == 0) break;
            ir=istack[jstack--];
            l=istack[jstack--];
        } else {
            k=(l+ir) >> 1;
            SWAP(arr[k],arr[l+1])
            if (arr[l+1] > arr[ir]) {
                SWAP(arr[l+1],arr[ir])
            }
            if (arr[l] > arr[ir]) {
```

```
            SWAP(arr[l],arr[ir])
        }
        if (arr[l+1] > arr[l]) {
            SWAP(arr[l+1],arr[l])
        }
        i=l+1;
        j=ir;
        a=arr[l];
        for (;;) {
            do i++; while (arr[i] < a);
            do j--; while (arr[j] > a);
            if (j < i) break;
            SWAP(arr[i],arr[j])
        }
        arr[l]=arr[j];
        arr[j]=a;
        jstack += 2;
        if (jstack > NSTACK) printf("\nNSTACK too small in sort.");
        if (ir-l+1 >= j-l) {
            istack[jstack]=ir;
            istack[jstack-1]=l;
            ir=j-1;
        } else {
            istack[jstack]=j-1;
            istack[jstack-1]=l;
            l=i;
        }
      }
    }
    free_ivector(istack,1,NSTACK);
}


#undef M
#undef NSTACK
#undef SWAP
#undef NRANSI
```

```
/* (C) Copr. 1986-92 Numerical Recipes Software "U,uJ. */

void moment(float data[], int n, float *ave, float *adev, float *sdev,float *var, float *skew, float *curt)
{
    void nrerror(char error_text[]);
    int j;
    float ep = 0.0, s, p;

    if (n <= 1)
      printf("n must be at least 2 in moment");

    s=0.0;

    for (j = 1; j <= n; j++)
        s += data[j];
    *ave = s/n;
    *adev = (*var) = (*skew) = (*curt) = 0.0;
    for (j = 1; j <= n; j++)
        { *adev += fabs(s = data[j] - (*ave));
          *var  += (p = s*s);
          *skew += (p *= s);
          *curt += (p *= s);
        }
    *adev /= n;
    *var = (*var - ep*ep/n)/(n-1);
    *sdev=sqrt(*var);
    if (*var != ep)
      {
         *skew /= (n*(*var)*(*sdev));
         *curt = (*curt)/(n*(*var)*(*var))-3.0;
      }
    //else nrerror("No skew/kurtosis when variance = 0 (in moment)");
}
/* (C) Copr. 1986-92 Numerical Recipes Software "U,uJ. */
```

```
void intMoment(int data[], int n, float *ave, float *adev, float *sdev,float *var, float *skew, float *curt)
{
    void nrerror(char error_text[]);
    int j;
    float ep = 0.0, s, p;

    if (n <= 1)
      printf("n must be at least 2 in moment");

    s=0.0;

    for (j = 1; j <= n; j++)
        s += float(data[j]);
    *ave = s/n;
    *adev = (*var) = (*skew) = (*curt) = 0.0;
    for (j = 1; j <= n; j++)
        { *adev += fabs(s = float(data[j]) - (*ave));
          *var  += (p = s*s);
          *skew += (p *= s);
          *curt += (p *= s);
        }
    *adev /= n;
    *var = (*var - ep*ep/n)/(n-1);
    *sdev=sqrt(*var);
    if (*var != ep)
      {
        *skew /= (n*(*var)*(*sdev));
        *curt = (*curt)/(n*(*var)*(*var))-3.0;
      }
    //else nrerror("No skew/kurtosis when variance = 0 (in moment)");
}
/* (C) Copr. 1986-92 Numerical Recipes Software "U,uJ. */
```

```
int printSFMatrix(char title[128])
   { int i, j;
     if ((infoFptr = fopen("FreqMtch.out\0", "at")) == NULL)
        { printf("ERROR: unable to open file freqMtch.out \n");
          exit(0);
        }
     // Print out the bootMatrix sample fraction matrix and the associated bootstrap CI's
     fprintf(infoFptr, "\n %s", title);
     fprintf(infoFptr, "\n              18-19     20-24     25-29     30-34     35-39     40-44     45-49     50-54     55-59     60-64     65-69     70-74");
     for (i = 0; i < numStudyYears; i++)
        { fprintf(infoFptr,"\n\n     Pj Case");
          for (j = 0; j < numAgeCats; j++)
             fprintf(infoFptr, " %11i", projectedCases[i][j]);

          fprintf(infoFptr,"\n     AVG CTL");
          for (j = 0; j < numAgeCats; j++)
             fprintf(infoFptr, " %11.0f", bootMatrix2[i][j].avgCtrls);

          fprintf(infoFptr, "\n     AVG CI   ");
          for (j = 0; j < numAgeCats; j++)
             fprintf(infoFptr, " (%3i, %3i) ", bootMatrix2[i][j].ctrlsLow, bootMatrix2[i][j].ctrlsHigh);

          fprintf(infoFptr,"\n %i     SF ", studyBaseYear + i);
          for (j = 0; j < numAgeCats; j++)
             fprintf(infoFptr, " %11.2f", SFmatrix[i][j]);

          fprintf(infoFptr,"\n     avg SF ");
          for (j = 0; j < numAgeCats; j++)
             fprintf(infoFptr, " %11.2f", bootMatrix2[i][j].sf);

          fprintf(infoFptr,"\n     SF CI   ");
          for (j = 0; j < numAgeCats; j++)
             fprintf(infoFptr, " (%3.2f %3.2f)", bootMatrix2[i][j].sfLow, bootMatrix2[i][j].sfHigh);
        }
      fclose(infoFptr);
```

```
    return 0;
  }
Boolean calculatePermissibleCells(int yrKC1st, int yrKCLast, int birthYr, int *rowSel, int *colSel)
  { /*  Uses pointers (rowsel, colsel) which returns the i,j index of the cell that the control to be used in
        Will calculate the permissible cells a case can be in and select one based on weighted need for a control
        Alternative method for ctrls needed - use 1:1 ctrls - # ctrls returned
    */
    const int maxAgeIndx = 12;

    Boolean sel, found;
    int i, yrKCindx, ageThen, yrsKC, yearsLT18, ageThenIndx, yrsKCcount;
    int totalCtrlsNeeded = 0;
    int numPerCells = 0;
    int permisIndx = 0;
    float tempRand;
    float p[10], cp[10]; // index starts @ 1 - 10 probablity and cumulative probability of each cell needig  control


    ageThen = yrKC1st - birthYr; // yrKC1st between 83 and 91 inclusive age between 18 - 74 when 1st in KC
    yrsKC =  (yrKCLast - yrKC1st) + 1;
    if (ageThen < 18) // know ctrl qualifies but < 18 need to correct age to 18 top find earliest cell
      { yearsLT18 = 18 - ageThen;
        ageThen += yearsLT18;
        yrsKC += yearsLT18;
        yrKC1st += yearsLT18;  // was yrsKC
      }

    ageThenIndx = -1;
    for (i = 0; i < numAgeCats; i++)
       { if (ageThen >= ageCatLow[i] && ageThen <= ageCatHigh[i])
            { ageThenIndx = i;
              break;
            }
       }

    if (ageThenIndx == -1)
      printf("\n ERROR illegal control Age Thenproc calculate PermissibleCells ");
```

```
yrKCindx = yrKC1st - 83; // 0 = 1983, 1 = 1984, ... , 8 = 1991
for (i = yrKCindx, yrsKCcount = 0; ((i < numStudyYears) && (ageThenIndx < maxAgeIndx) && (yrsKCcount < yrsKC)); i++, yrsKCcount++)
   { if ((ageThen >= ageCatLow[ageThenIndx]) && (ageThen <= ageCa tHigh[ageThenIndx]))
       { if (simMatrix[i][ageThenIndx].simCtrlsNeeded > 0)  // if the cell needs more controls or num sel - num returned > 0
           { numPerCells++;
             permisIndx++;
             permisCell[permisIndx].indxi = i;
             permisCell[permisIndx].indxj = ageThenIndx;
           }
         ageThen++;
       }
      else
       { ageThenIndx++; // have moved to the next age category
         if (ageThenIndx >= maxAgeIndx)
            break;
         if ((ageThen >= ageCatLow[ageThenIndx]) && (ageThen <= ageCatHigh[ageThenIndx]) )
           { if (simMatrix[i][ageThenIndx].simCtrlsNeeded > 0) // if the cell needs more controls
               {  numPerCells++;
                  permisIndx++;
                  permisCell[permisIndx].indxi = i;
                  permisCell[permisIndx].indxj = ageThenIndx;
               }
             ageThen++;
           }
       }
   }

if (numPerCells == 0) // no cells needing ctrlss
   return False;

if (whatToDo == freqMatch) // if doing frequency matching make a record of what the results and permissible cells are
   {  fprintf(selInfoFptr, "%i %i ", ageThen, yrsKC);
      for (i = 1; i <= numPerCells; i++)
         fprintf(selInfoFptr, " (%i, %i)", permisCell[i].indxi, permisCell[i].indxj);
   }
```

```
// Now have a list of permissible cells that this control can be used in
for ( i = 1; i <= numPerCells; i++)
   { totalCtrlsNeeded += simMatrix[permisCell[i].indxi][permisCell[i].indxj].simCtrlsNeeded;
     /* Alternativee weight method for cell  for controls needed*/
     // totalCtrlsNeeded += simMatrix[permisCell[i].indxi][permisCell[i].indxj].simCtrlsNeeded;  // where is this calculated??
     // totalCtrlsNeeded +=
   }

if (totalCtrlsNeeded == 0)
  return False;

p[0] = cp[0] = 0.0;
for ( i = 1; i <= numPerCells; i++)
   { // If use 1:1 [# num ctrls needed  - (# ctrls sel - #ctrls returned)]   // requires to run response rate procedure first
     if (freqMatchRatio == 1)
       p[i] = ((float) simMatrix[permisCell[i].indxi][permisCell[i].indxj].simCtrlsNeeded - (1)) / (float) totalCtrlsNeeded;
     else
       p[i] =  (float) simMatrix[permisCell[i].indxi][permisCell[i].indxj].simCtrlsNeeded / (float) totalCtrlsNeeded;

     cp[i] = cp[i-1] + p[i]; // s[i] is the upperbound of the weighted probability
     // check that s[numPerCells] approx. = 1
     if (numPerCells == 1 && cp[1] != 1.0)
       { cp[1] = 1.0;
      printf("\n Major FuckUP cp[1] should be 1 p[1]=%f simCtlNeed=%i totalctls=%i", p[1],
            simMatrix[permisCell[i].indxi][permisCell[i].indxj].simCtrlsNeeded, totalCtrlsNeeded);
       }
   }

sel = False;
found = False;
tempRand = rand2(&iseed);
for (i = 1; i <= numPerCells; i++)
   { if ((tempRand >= cp[i-1]) && (tempRand < cp[i]))
       { *rowSel = permisCell[i].indxi;
         *colSel = permisCell[i].indxj;
         found = True;
```

```
            if ((tempRand = rand2(&iseed)) <= SFmatrix[*rowSel][*colSel]) // now figure out if should use i.e. is tempRand <= SFmatrix[i][j] Note: for initial use
SFmatrix is set to all 1.s
                { simMatrix[permisCell[i].indxi][permisCell[i].indxj].simCtrlsNeeded--;
                  simMatrix[permisCell[i].indxi][permisCell[i].indxj].simCtrlsSel++;
                  sel = True;
                }
              break;
            }
        }

    if (found == False)
      printf("\n major prob: simualted ctrl not selected");

    return sel;
  }

int simBoot(Boolean simFlag)
  { int i, j, k, x, y;
    int spin[5], spinindx;
    float adev, ave, curt, sdev, skew, vrnce;


    #define vert 0xB3
    #define horz 0x2D
    #define rlean 0x2F
    #define llean 0x5C
    #define bksp 0x08

    // now boot strap the SF chosen to see what the distribution of cells with SF's > 1 look like
    spinindx = 1;
    spin[0] = llean;
    spin[1] = vert;
    spin[2] = rlean;
    spin[3] = horz;
    spin[4] = bksp;

    if (simFlag == False)
```

```
  printf("\nBootStrap# ");
else
  printf("\nSimulate# ");

for (i = 1; i <= 1000; i++)
   { printf("%4i  ", i);
     putch(spin[4]); putch(spin[spinindx]); spinindx++; if (spinindx == 4) spinindx = 0;

     if (simFlag == True)  // either bootstrap or simulate
       simulateSampleFraction();
     else
       bootstrapSampleFraction();

     //initializeTheMatrix;
     for (j = 0; j < numStudyYears; j++)
        { for (k = 0; k < numAgeCats; k++)
             { simMatrix[j][k].row = j;
               simMatrix[j][k].col = k;
               if (caseYrDX_AgeDXprojected[j][k] == 0)
                  caseYrDX_AgeDXprojected[j][k]++; //no zero cells for proj cases
               simMatrix[j][k].TotalCasesProj = caseYrDX_AgeDXprojected[j][k];
               simMatrix[j][k].simTargetCtrls = (caseYrDX_AgeDXprojected[j][k] * targetCtrlToCaseFMratio) + 1;
               simMatrix[j][k].simCtrlsNeeded = simMatrix[j][k].simTargetCtrls;
               simMatrix[j][k].simCtrlsSel = 0;
             }
        }
     for (j = 0; j < numEstimatedControls; j++)
        { // x, y will return the cell index selected, don't need here but need in the RDD freq match code
          calculatePermissibleCells(simBootInfo[simBootOrder[j]].yr1stKC, simBootInfo[simBootOrder[j]].yrLastKC,
                                    simBootInfo[simBootOrder[j]].birthYr, &x, &y); // run through the array
        }
     simCellPtr[i] = new simCellArray; //cellSimPtr;//simCellMatrix;
     // save the results
     for (j = 0; j < numStudyYears; j++)
        { for (k = 0; k < numAgeCats; k++) // just copy order does not matter
             { simCellPtr[i]->simCellMatrix[j][k].numAdjustedCtrls = simMatrix[j][k].simCtrlsSel;
               if (simMatrix[j][k].simCtrlsSel > 0)
```

```
                    simCellPtr[i]->simCellMatrix[j][k].adjustedSF = (float)simMatrix[j][k].simTargetCtrls / (float)simMatrix[j][k].simCtrlsSel;
                  else simCellPtr[i]->simCellMatrix[j][k].adjustedSF = 1.0;  //*/
                }
            }
          putch(spin[4]); putch(spin[4]); putch(spin[4]);
          putch(spin[4]); putch(spin[4]); putch(spin[4]);
        }

      /* Now calculate:
         a) the average number of controls expected, and the the bootstrap Confidence Interval
         b) the estimated average of the adjusted sample fraction and it's bootstrap CI
      */
      // will use the adjustedSF data structure to hold the results for each cell calculate the average and bootstrap CI
      // will need to sort the array of averages and the sample fractions

      for (i = 0; i < numStudyYears; i++)
         { for (j = 0; j < numAgeNowCats; j ++)
              { for (k = 1; k <= 1000; k++)
                   { intArray[k] = simCellPtr[k]->simCellMatrix[i][j].numAdjustedCtrls;  // insertion sort might be fastest
                     floatArray[k] = simCellPtr[k]->simCellMatrix[i][j].adjustedSF;
                   }
                if (i == 1 && j == 1 )
                  printf("");
                intSort(1000, intArray);
                sort(1000, floatArray);
                bootMatrix2[i][j].sfLow = floatArray[25];
                bootMatrix2[i][j].sfHigh = floatArray[975];
                moment(floatArray, 1000, &ave,&adev,&sdev,&vrnce,&skew,&curt);
                bootMatrix2[i][j].sf = ave;
                intMoment(intArray, 1000, &ave,&adev,&sdev,&vrnce,&skew,&curt);
                bootMatrix2[i][j].avgCtrls = ave;
                bootMatrix2[i][j].ctrlsLow = intArray[25];
                bootMatrix2[i][j].ctrlsHigh = intArray[975];
              }
         }
      return 0;
}
```

```
int calculateInitialSampleFraction() // load control info
   { // will generate bootstrap estimate of the sample fraction matrix
     // move code from the freqmatch code proc so that freq match primarilry just does the freq mtching

     int age, year;
     int controlsToDate[16];
     int i, j, k, m, n;
     int yearsLT18;
     char tempString[128];
     int ctrlYrsKCindx, tmpCtrlID, tmpCtrlWaveNum, tmpCtrlAge, tmpCtrlAgeThen, tmpCtrlBirthYr, tmpCtrlYrsKC,
        tmpCtrlAgeCat, tmpCtrlYrKC1st, tmpCtrlYrKCLast;
     Boolean simFlag;
       // modify so that the name of the wave is ctrlwave.xzz  x is either a or b and zz is the wave number a is the fisrt time for wave
       // b is the hold out recalled who agree to play later
     if ((infoFptr = fopen("ctrlwave.all\0", "rt")) == NULL)
        { printf("\n ERROR: unable to open file CTRLWAVE.ALL \n");
          exit(0);
        }
     // if get to here file opened ok
     printf("\n File CTRLWAVE.ALL opened successfully\n"); // FILE opened OK!

     for (i = 0; i < numAgeNowCats; i++)
        controlsToDate[i] = 0;  // simple
     // initialize the controlYrKC_AgeNow matrix
     for ( i = 0; i < numStudyYears; i++)
        { for ( j = 0; j < numAgeCats; j++)
             controlYrKC_AgeThen[i][j] = 0;
        }

     //ctrlIndx = numCtrlsInWave = 0; // initialize the index that used to fill up the array  of conntrol information
     while (feof(infoFptr) == 0)
        { fgets(tempString, 128, infoFptr);
          if (feof(infoFptr) != 0)
             break;
          // read in the control info and save it in control info structure
          sscanf(tempString, "%li %i %i %i %i %i",&tmpCtrlID, &tmpCtrlWaveNum, &tmpCtrlAge, &tmpCtrlBirthYr, &tmpCtrlYrKC1st, &tmpCtrlYrKCLast);
```

```
    // add count to specific age category
    if ((tmpCtrlAge < 22) || (tmpCtrlYrKC1st < 83) || (tmpCtrlYrKCLast > 91) || (tmpCtrlAge > 86))
       { if (tmpCtrlAge > 86) printf("\n ERROR illegal control Age > 86 %s", tempString);
         if (tmpCtrlAge < 22) printf("\n ERROR illegal control Age < 22 %s", tempString);
         if (tmpCtrlYrKC1st < 83) printf("\n ERROR control years in King County < 1983 years %s",tempString);
        if (tmpCtrlYrKCLast > 91) printf("\n ERROR control years in King County > 1991 years %s",tempString);
       }
     else
       { // put into simple array and control matrix
         tmpCtrlAgeThen = tmpCtrlYrKC1st - tmpCtrlBirthYr; // age between 18 - 74 when 1st in KC
         tmpCtrlYrsKC =  ((tmpCtrlYrKCLast - tmpCtrlYrKC1st) + 1);
         ctrlYrsKCindx = tmpCtrlYrKC1st - 83;
         if (tmpCtrlAgeThen < 18) // know ctrl qualifies but < 18 need to correct age to 18 top find earliest cell
            { yearsLT18 = 18 - tmpCtrlAgeThen;
              tmpCtrlAgeThen += yearsLT18;
              ctrlYrsKCindx += yearsLT18;
            }
         // fprintf(selInfoFptr, "%i %i ", tmpCtrlAgeThen, tmpCtrlYrsKC);

         for (i = 0; i < numAgeCats; i++)
             { if (tmpCtrlAgeThen >= ageCatLow[i] && tmpCtrlAgeThen <= ageCatHigh[i])
                  { controlsToDateThen[i]++;
                    tmpCtrlAgeCat = i;
                    controlYrKC_AgeThen[ctrlYrsKCindx][tmpCtrlAgeCat]++;
                    SFMatrixInfo[ctrlYrsKCindx][tmpCtrlAgeCat].ageCatNumThen[tmpCtrlAgeThen - ageCatLow[tmpCtrlAgeCat]]++;
                                    //should be 0 - 4 hairy but all we doin' is keeping trach of the number of ctrls in each cell at each age
                    break;
                  }
             }
          if (tmpCtrlAgeCat == -1)
             printf("\n ERROR illegal control Age Then %s", tempString);
         }
    }
fclose(infoFptr);        // should check for error  on file close

// calculate sample fraction from data for new wave
for (i = 0; i < numAgeCats; i++)
```

```
    sampleFraction[i] =  (float)CompletesNeeded[i] / (controlsToDate[i] / percentControlsAcertainedToDate);

for ( i = 0; i < numStudyYears; i++)
    { for (j = 0; j < numAgeCats; j++)
          { if (caseYrDX_AgeDXprojected[i][j] > 0)
              projectedCases[i][j] = caseYrDX_AgeDXprojected[i][j];// /  percentCasesAcertainedToDate;
            else projectedCases[i][j] = 1;

            if (controlYrKC_AgeThen[i][j] > 0)
              projectedControlsThen[i][j] = (controlYrKC_AgeThen[i][j] / percentControlsAcertainedToDate);
            else projectedControlsThen[i][j] = 0;

            // need to project the individual yerar cells of cases and ctrls to do the simulation
            for (k = 0; k < numYearsPerCell; k++)
                { SFMatrixInfo[i][j].ageCatNum[k] = SFMatrixInfo[i][j].ageCatNum[k] / percentControlsAcertainedToDate;
                  SFMatrixInfo[i][j].ageCatNumThen[k] = 1 + SFMatrixInfo[i][j].ageCatNumThen[k] / percentControlsAcertainedToDate;
                  // do a test to make sure sum(ageCatNum[k]) = projected ctrls
                }
          }
    }

// now we need to print out the control matrices
if ((infoFptr = fopen("FreqMtch.out\0", "at")) == NULL)
    { printf("ERROR: unable to open file freqMtch.out \n");
      exit(0);
    }

// now we need to print out the controlAgeCatThen
fprintf(infoFptr, "\n ControlsYears KC  _ Age Then matrix ");
fprintf(infoFptr, "\n       18-19 20-24 25-29 30-34 35-39 40-44 45-4 9 50-54 55-59 60-64 65-69 70-74");

for (i = 0; i < numStudyYears; i++)
    { fprintf(infoFptr, "\n %i", studyBaseYear + i);
      for (j = 0; j < numAgeCats; j++)
          fprintf(infoFptr, "  %4i", controlYrKC_AgeThen[i][j]);
    }
```

```
// now we need to print out the  controlAgeCatThen Projected
fprintf(infoFptr, "\n ControlsYears KC  _ Age Then matrix Projected ");
fprintf(infoFptr, "\n       18-19 20-24 25-29 30-34 35-39 40-44 45-49 50-54 55-59 60-64 65-69 70-74");

for (i = 0; i < numStudyYears; i++)
   { fprintf(infoFptr, "\n %i", studyBaseYear + i);
     for (j = 0; j < numAgeCats; j++)
        fprintf(infoFptr, "  %4i", projectedControlsThen[i][j]);
   }
fprintf(infoFptr, "\n  Initial sample fraction information");
fprintf(infoFptr, "\n                18-19  20-24  25-29  30-34  35-39  40-44  45-49  50-54  55-59  60-64  65-69  70-74");
for (i = 0; i < numStudyYears; i++)
   { fprintf(infoFptr, "\n\n      proj cases");
     for (j = 0; j < numAgeCats; j++)
        fprintf(infoFptr, "  %5i", projectedCases[i][j]);

     fprintf(infoFptr,"\n %i proj ctrls", studyBaseYear + i);
     for (j = 0; j < numAgeCats; j++)
        fprintf(infoFptr, "  %5i", projectedControlsThen[i][j]);

     fprintf(infoFptr,"\n      samp fract");
     for (j = 0; j < numAgeCats; j++)
        fprintf(infoFptr, " %6.2f", sampleFractionMatrix[i][j]);
   }

fclose(infoFptr);

  // Now go through the array of controls and save into linear array for simulation and bootstrapping
  numEstimatedControls = 0;
  for ( i = 0, n = 0, year = 1983; i < numStudyYears; i++, year++)
     { for (j = 0, age = 18; j < numAgeCats; j++)
          { for (k = 0; (j==0) ? k < 2 : k < numYearsPerCell; k++, age++)
               { for (m = 0; m < SFMatrixInfo[i][j].ageCatNumThen[k]; m++, n++)
                    { numEstimatedControls++;
                      simBootInfo[n].i = i;     // this info for simulating and bootstraping the sample fractions to look at distribution and find optimal one.
                      simBootInfo[n].j = j;
                      simBootInfo[n].k = k;
```

```
                    simBootInfo[n].age = age;
                    simBootInfo[n].year = year;
                    simBootInfo[n].yr1stKC = year - 1900;
                    simBootInfo[n].yrLastKC = 91;
                simBootInfo[n].birthYr = year - 1900 - age;
                    }
                }
            }
        }

/* OK Now we have the simple sample fraction
   sample fraction will be recalculated for each cell each week should be based on:
     a) Age stratified Response rate - i.e. if worse than expected or by certain age group ^ the sample fraction
     b) Need to store the sample fraction used for each group
     Now calculate the complex sample fraction, include the age adjusted response rate i.e. # in age group returend/ # in age group sent if != to 0.80
     NOTE: could further modify by years of king county residency
*/


for (i = 0; i < numStudyYears; i++)  // initialize the sample fraction matrix, take everything;
   for (j = 0; j < numAgeCats; j++)
      SFmatrix[i][j] = 1.0;

targetCtrlToCaseFMratio = 100.0;  // 100:1 ctrl:Case
simBoot(simFlag = True); // do we bootstrap or simulate?  Right Now we be simulating
// now save the results
for (i = 0; i < numStudyYears; i++)
   for (j = 0; j < numAgeCats; j++)
      bootMatrix100[i][j] = bootMatrix2[i][j];
printSFMatrix("Initial Sample Fraction info ctrls:cases = 100:1");

targetCtrlToCaseFMratio = 3.0;    //2:1 Control:Case ratio
simBoot(simFlag = True); // do twice once with 3:1 ctrl:Case then with 100:1 ctrl:case so can calculate Sample Fraction ratio
for (i = 0; i < numStudyYears; i++)
   for (j = 0; j < numAgeCats; j++)
      bootMatrix3[i][j] = bootMatrix2[i][j];
printSFMatrix("Initial Sample Fraction info ctrls:cases = 3:1");
```

```
// Ok now calculate the sampleFraction

for (i = 0; i < numStudyYears; i++)
   { for (j = 0; j < numAgeCats; j++)
        { // set up the sample fraction
          if ((bootMatrix[i][j].sf =  (ceil)(bootMatrix3[i][j].avgCtrls / ProbCompleteQuest[j]) / bootMatrix100[i][j].avgCtrls) > 1.0 )
            bootMatrix[i][j].sf = 1.0;
          if ((bootMatrix[i][j].sfLow = (ceil)(bootMatrix3[i][j].ctrlsLow / ProbCompleteQuest[j]) / bootMatrix100[i][j].ctrlsLow) > 1.0 )
            bootMatrix[i][j].sfLow = 1.0;
          if ((bootMatrix[i][j].sfHigh = (ceil)(bootMatrix3[i][j].ctrlsHigh / ProbCompleteQuest[j]) / bootMatrix100[i][j].ctrlsHigh) > 1.0 )
            bootMatrix[i][j].sfHigh = 1.0;

          bootMatrix[i][j].ctrlsLow = bootMatrix3[i][j].ctrlsLow;
          bootMatrix[i][j].ctrlsHigh = bootMatrix3[i][j].ctrlsHigh;
          bootMatrix[i][j].avgCtrls = bootMatrix3[i][j].avgCtrls;
          bootMatrix[i][j].casesEst = projectedCases[i][j];
        }
   }


// Calculate initial Sample Fraction for further simulating
if ((infoFptr = fopen("FreqMtch.out\0", "at")) == NULL)
   { printf("ERROR: unable to open file freqMtch.out \n");
     exit(0);
   }

// Print out the bootMatrix sample fraction matrix and the associated bootstrap CI's
fprintf(infoFptr, "\n Adjusted Sample Fraction matrix ");
fprintf(infoFptr, "\n                18-19     20-24     25-29     30-34     35-39     40-44     45-49     50-54      55-59     60-64     65-69     70-74");
for (i = 0; i < numStudyYears; i++)
   { fprintf(infoFptr,"\n\n     Pj Case");
     for (j = 0; j < numAgeCats; j++)
        fprintf(infoFptr, " %11i", projectedCases[i][j]);

     fprintf(infoFptr,"\n     AVG CTL");
     for (j = 0; j < numAgeCats; j++)
        fprintf(infoFptr, " %11.0f", bootMatrix[i][j].avgCtrls);
```

```
        fprintf(infoFptr, "\n      AVG CI   ");
        for (j = 0; j < numAgeCats; j++)
            fprintf(infoFptr, " (%3i, %3i) ", bootMatrix[i][j].ctrlsLow, bootMatrix[i][j].ctrlsHigh);

        fprintf(infoFptr,"\n %i avg SF ", studyBaseYear + i);
        for (j = 0; j < numAgeCats; j++)
            fprintf(infoFptr, " %11.2f", bootMatrix[i][j].sf);

        fprintf(infoFptr,"\n      SF CI   ");
        for (j = 0; j < numAgeCats; j++)
            fprintf(infoFptr, " (%3.2f %3.2f)", bootMatrix[i][j].sfLow, bootMatrix[i][j].sfHigh);
    }

fclose(infoFptr);

// create three SF matrix sflow, avg sf and highSf    i.e. 3x
if ((infoFptr = fopen("sflow.sav\0", "wt")) == NULL)
   { printf("ERROR: unable to open file sflow.sav \n");
     exit(0);
   }
//fprintf(infoFptr,"sflow\n");
for (i = 0; i < numStudyYears; i++)
    { for (j = 0; j < numAgeCats; j++)
          fprintf(infoFptr, "%f\n ", bootMatrix[i][j].sfLow);
    }
fclose (infoFptr);

if ((infoFptr = fopen("sfavg.sav\0", "wt")) == NULL)
   { printf("ERROR: unable to open file sfavg.sav \n");
     exit(0);
   }
//fprintf(infoFptr,"sfavg\n");
for (i = 0; i < numStudyYears; i++)
    { for (j = 0; j < numAgeCats; j++)
          fprintf(infoFptr, "%f\n ", bootMatrix[i][j].sf);
    }
fclose (infoFptr);
```

```
if ((infoFptr = fopen("sfhigh.sav\0", "wt")) == NULL)
   { printf("ERROR: unable to open file sfHigh.sav \n");
     exit(0);
   }
//fprintf(infoFptr,"sfhigh\n");
for (i = 0; i < numStudyYears; i++)
    { for (j = 0; j < numAgeCats; j++)
          fprintf(infoFptr, "%f\n ", bootMatrix[i][j].sfHigh);
    }

fclose (infoFptr);

// use avg SF for starters // and save to file
    for (i = 0; i < numStudyYears; i++)
    { for (j = 0; j < numAgeCats; j++)
          SFmatrix[i][j] = bootMatrix[i][j].sf;
    }

for ( i = 1; i <= 1000; i++) // clean up thr simcell info so can reuse
    delete simCellPtr[i];

sampleFractionSelected = True;

// initialize the number of controls needed in each cell
if ((infoFptr = fopen("CtrlHIST.SEL\0", "wt")) == NULL)   // set up output file
   printf("\n ERROR: Unable to open file the Wave Selected History file \n");
else
   { //fprintf(infoFptr, "\n %i  Control Selected to Date file");
     for (i = 0; i < numStudyYears; i++)
        { for (j = 0; j < numAgeCats; j++)
            { if (caseYrDX_AgeDXprojected[i][j] == 0)
                 caseYrDX_AgeDXprojected[i][j]++; //no zero cells for proj cases
              simMatrix[i][j].TotalCasesProj = caseYrDX_AgeDXprojected[i][j];
              simMatrix[i][j].simTargetCtrls = (caseYrDX_AgeDXprojected[i][j] * targetCtrlToCaseFMratio) + 1;
              simMatrix[i][j].simCtrlsNeeded = simMatrix[i][j].simTargetCtrls;
              simMatrix[i][j].simCtrlsSel = 0;
```

```
                    fprintf(infoFptr,"%i %i %i %i\n", simMatrix[i][j].TotalCasesProj, simMatrix[i][j].simTargetCtrls, simMatrix[i][j].simCtrlsNeeded,
simMatrix[i][j].simCtrlsSel);//simMatrix[i][j].simCtrlsSel);
                }
            }
          fclose(infoFptr);
        }

    return 0;
  }


int bootstrapSampleFraction()
  { int i;
    float tempRand;
    // Does esentially the same as / with some minor modifications
    // simBootInfo
    // simBootOrder
    // SFmatrix has the sample fractions
    /* Since this is a bootstrap Need to sample the data with replacement
       So generate the simBootOrder with the original data to use

    */
    for ( i = 0; i < numEstimatedControls; i++)
       { tempRand = rand2(&iseed);
         simBootOrder[i] = long(tempRand * 10000)%numEstimatedControls;
         if (simBootOrder[i] < 0 || simBootOrder[i] > numEstimatedControls)
           printf("\n Major error on rnad num in BootstrapSF");
       }

    return 0;
  }

int simulateSampleFraction()
  { int i, j, temp;
    float tempRand;

    // need to go through each position and swap with another one so lot of exchanges but we use all  of the data
```

```
    for ( i = 0; i < numEstimatedControls; i++)
        simBootOrder[i] = i;
    // now permute the order
    for (i = 0; i < numEstimatedControls; i++)
        { tempRand = rand2(&iseed);
          j = long(tempRand * 10000)%numEstimatedControls;
          temp = simBootOrder[j];  // have to use temp incase the same # is generated > 1 time which it will
          simBootOrder[j] = simBootOrder[i];
          simBootOrder[i] = temp;
        }
      return 0;
  }

void printSelectSFmenu()
    { // prints the menu for the select sample fraction method
      printf("\n SELECT Sample Fraction Method ");
      printf("\n 0: use low 95% CI value (simulate)");
      printf("\n 1: Use avg 95% CI value (simulate)");
      printf("\n 2: Use high 95% CI value (simulate)");
      printf("\n 3: use low 95% CI value (bootstrap)");
      printf("\n 4: Use avg 95% CI value (bootstrap)");
      printf("\n 5: Use high 95% CI value (bootstrap)");
      printf("\n 6: Use Per Cent (%)CI value(simulate)");
      printf("\n 7: Use Per Cent (%)CI value (bootstrap)");
      printf("\n 8: Use historical Sample Fraction Matrix Previously Used");
      printf("\n 9: Use hand modified Sample Fraction Matrix");
      printf("\n Please Enter Response: ");
    }


 int selectSampleFraction()
    { /* This Procedure needs to run after the procedure that calculates the sample fraction */
      enum sfChoises {simLowSF, simAvgSF, simHighSF, bootLowSF, bootAvgSF, bootHighSF, simPerCentCI, bootPerCentCI, historicalSF, handSF, skipSF};
      const int sfCmdNum = 10;
      int i, j;
      int toDo;
      Boolean simFlag = False;
```

```
// prompt for sample fraction file; option to list sample fraction files (type L)
printSelectSFmenu();
toDo = getValidResponse(sfCmdNum);

/* A number of options -
   a) use one of the precomputed from the bootstrap - sfavg, sflow, sfhigh or
   b) hand modify a saved SF off line and read it in
   c) In either case bootstrap/simulate to get 95% CI and see how will preform on estimated cases and controls
*/

// essentially copy sf matrix to be used to the SFmatrix

switch (toDo)
   { case simLowSF: { for (i = 0; i < numStudyYears; i++)
                        { for (j = 0; j < numAgeCats; j++)
                            SFmatrix[i][j] = bootMatrix[i][j].sfLow;
                        }
                        simFlag = True;
                      } break;
     case simAvgSF: { for (i = 0; i < numStudyYears; i++)
                        { for (j = 0; j < numAgeCats; j++)
                            SFmatrix[i][j] = bootMatrix[i][j].sf;
                        }
                        simFlag = True;
                      } break;
     case simHighSF: { for (i = 0; i < numStudyYears; i++)
                        { for (j = 0; j < numAgeCats; j++)
                            SFmatrix[i][j] = bootMatrix[i][j].sfHigh;
                        }
                        simFlag = True;
                      } break;
     case bootLowSF: { for (i = 0; i < numStudyYears; i++)
                        { for (j = 0; j < numAgeCats; j++)
                            SFmatrix[i][j] = bootMatrix[i][j].sfLow;
                        }
                          simFlag = False; // ie we bootstrap
                      } break;
```

```
    case bootAvgSF: { for (i = 0; i < numStudyYears; i++)
                        { for (j = 0; j < numAgeCats; j++)
                            SFmatrix[i][j] = bootMatrix[i][j].sf;
                        }
                          simFlag = False;
                      } break;
    case bootHighSF: { for (i = 0; i < numStudyYears; i++)
                        { for (j = 0; j < numAgeCats; j++)
                            SFmatrix[i][j] = bootMatrix[i][j].sfHigh;
                        }
                          simFlag = False;
                      } break;

    case simPerCentCI: { // will have to run
                          simFlag = True;
                       } break; // enter a percent of the bootstrap .5 is the average, .95 is the high end, 05 is the low end, or any thing between

    case bootPerCentCI: { // will have to run
                          simFlag = False;
                        } break; // enter a percent of the bootstrap .5 is the average, .95 is the high end, 05 is the low end, or any thing between

    case historicalSF: {
                       } break;
    case handSF: {
                 }; break;

    default: { return 0;
             };
  }

simBoot(simFlag); // will do 1000 simulations of case ctrl dist of ctrl:case ratio and SF

for (i = 0; i < numStudyYears; i++)
   { for (j = 0; j < numAgeCats; j++)
       { // set up the sample fraction
         if ((bootMatrix2[i][j].sf = (bootMatrix2[i][j].avgCtrls / ProbCompleteQuest[j]) / bootMatrix100[i][j].avgCtrls) > 1.0 )
           bootMatrix2[i][j].sf = 1.0;
```

```
            if ((bootMatrix2[i][j].sfLow = (bootMatrix2[i][j].ctrlsLow / ProbCompleteQuest[j]) / bootMatrix100[i][j].ctrlsLow) > 1.0 )
              bootMatrix2[i][j].sfLow = 1.0;
            if ((bootMatrix2[i][j].sfHigh = (bootMatrix2[i][j].ctrlsHigh / ProbCompleteQuest[j]) / bootMatrix100[i][j].ctrlsHigh) > 1.0 )
              bootMatrix2[i][j].sfHigh = 1.0;
          }
    }

// Print out the bootMatrix sample fraction matrix and the associated bootstrap CI's

if (simFlag == True)
   printSFMatrix("\n Simulated Sample Fraction matrix ");
else
   printSFMatrix("\n Bootstrap Sample Fraction matrix ");

for ( i = 1; i <= 1000; i++) // clean up thr simcell info so can reuse
    delete simCellPtr[i];

// initialize the number of controls needed in each cell
if ((infoFptr = fopen("CtrlHIST.SEL\0", "wt")) == NULL)   // set up output file
   printf("\n ERROR: Unable to open file the Wave Selected History file \n");
else
   { //fprintf(infoFptr, "\n %i  Control Selected to Date file");
     for (i = 0; i < numStudyYears; i++)
        { for (j = 0; j < numAgeCats; j++)
            { if (caseYrDX_AgeDXprojected[i][j] == 0)
                 caseYrDX_AgeDXprojected[i][j]++; //no zero cells for proj cases
              simMatrix[i][j].TotalCasesProj = caseYrDX_AgeDXprojected[i][j];
              simMatrix[i][j].simTargetCtrls = (caseYrDX_AgeDXprojected[i][j] * targetCtrlToCaseFMratio) + 1;
              simMatrix[i][j].simCtrlsNeeded = simMatrix[i][j].simTargetCtrls;
              simMatrix[i][j].simCtrlsSel = 0;
              fprintf(infoFptr,"%i %i %i %i\n", simMatrix[i][j].TotalCasesProj, simMatrix[i][j].simTargetCtrls,
                                                simMatrix[i][j].simCtrlsNeeded, simMatrix[i][j].simCtrlsSel);//simMatrix[i][j].simCtrlsSel);
            }
        }
     fclose(infoFptr);
   }
```

```
    sampleFractionSelected = True;  // indicator that we have selected a SF for the RDD freq match procedure

    return 0;
   }


int countControls()
   { // will count the number of controls generated to make sure what saved matches what selected
     int refYr, dead, wave, ageNow, birthYr;
     long int id;
     char month[18], tempString[256];;
     int ctrlSel[10][16];
     int refAgeCat, refYrCat, ageRefYr, ageDX, returnStatus;
     int i, j;

     // initialize ctrlSelection matrix
     for ( i = 0; i < numStudyYears; i++)
        for (j = 0; j < numAgeCats; j++)
           ctrlSel[i][j] = 0;

     // read in what I think I have
     if ((infoFptr = fopen("ctrlhist.sel\0", "rt")) == NULL)
        { printf("\n ERROR: Unable to open file the Wave Selected History file \n");
          return -1;
        }
      else
        { //fprintf(infoFptr, "\n %i  Control Selected to Date file");
          for (i = 0; i < numStudyYears; i++)
             { for (j = 0; j < numAgeCats; j++)
                  { fgets(tempString, 240,infoFptr);
                    sscanf(tempString, "%i %i %i %i",
&simMatrix[i][j].TotalCasesProj,&simMatrix[i][j].simTargetCtrls,&simMatrix[i][j].simCtrlsNeeded,&simMatrix[i][j].simCtrlsSel);//simMatrix[i][j].simCtrlsSel);
                  }
             }
          fclose(infoFptr);
        }
```

```
// read in what I actually have selected for each wave
if ((infoFptr = fopen("response.rat\0", "rt")) == NULL)
  printf("\n ERROR: Unable to open file the actual controls selected History file \n");
else
  { if ((infoFptrOut = fopen("studyID.lst\0", "wt")) == NULL)
      printf("\n ERROR: Unable to open file studyID.lst for writing\n");

    //fprintf(infoFptr, "\n %i  Control Selected to Date file");
    while (feof(infoFptr) == 0)
      { fgets(tempString, 240, infoFptr);
        if (feof(infoFptr) != 0)
          break;
        sscanf(tempString, "%i %s %i %i %li %i %i %i %i", &dead, &month, &refYr, &ageDX, &id, &returnStatus, &wave, &ageNow, &birthYr);
            // ,&yrKC1st, &yrKCLast); //simMatrix[i][j].simCtrlsSel);

        fprintf(infoFptrOut,"%li %i\n", id, returnStatus);  // print out a file of study id used to check what Amy checking off is correct

        refYrCat = refYr - 1983;
        ageRefYr = refYr - (birthYr + 1900);
        if (refYr < (birthYr + 1900))
          printf("\n major prob agerefyr < birthyear");
        if (ageDX != ageRefYr)
          printf("\n major problem Age@REfYr != calculated");

        // figure out the ref age cat
        for (i = 0; i < numAgeCats; i++)
          { if (ageRefYr >= ageCatLow[i] && ageRefYr <= ageCatHigh[i])
              { refAgeCat = i;
                break;
              }
          }
        ctrlSel[refYrCat][refAgeCat]++;
      }
    fclose(infoFptr);
    fclose(infoFptrOut);
  }
```

```
// now print out a matrix showing the controls and cases and controls I have actually selected
  if ((infoFptr = fopen("FreqMtch.out\0", "at")) == NULL)
   { printf("ERROR: unable to open file freqMtch.out \n");
      exit(0);
   }

// Print out the bootMatrix sample fraction matrix and the associated bootstrap CI's
fprintf(infoFptr, "\n Controls Selected to Date Check ");
fprintf(infoFptr, "\n                    18-19 20-24 25-29 30-34 35-39 40-44 45-49 50-54 55-59 60-64 65-69 70-74");
for (i = 0; i < numStudyYears; i++)
    { fprintf(infoFptr,"\n\n      Project Cases");
      for (j = 0; j < numAgeCats; j++)
         fprintf(infoFptr, " %5i", simMatrix[i][j].TotalCasesProj);

      fprintf(infoFptr,"\n      Target  Ctrls");
      for (j = 0; j < numAgeCats; j++)
         fprintf(infoFptr, " %5i", simMatrix[i][j].simTargetCtrls);

      fprintf(infoFptr,"\n%i  Ctrls  Needed", i + 1983);
      for (j = 0; j < numAgeCats; j++)
         fprintf(infoFptr, " %5i", simMatrix[i][j].simCtrlsNeeded);

      fprintf(infoFptr,"\n      Ctrl Selected");
      for (j = 0; j < numAgeCats; j++)
         fprintf(infoFptr, " %5i", simMatrix[i][j].simCtrlsSel);

      fprintf(infoFptr,"\n      Actual Select");
      for (j = 0; j < numAgeCats; j++)
         fprintf(infoFptr, " %5i", ctrlSel[i][j]);

    }
 fclose(infoFptr);

 // now open up the history file and correct the problem if our recort is different than the actual

if ((infoFptr = fopen("ctrlhist.sel\0", "wt")) == NULL)   // set up output file
    printf("\n ERROR: Unable to open file the Wave Selected History file \n");
```

```
    else
      { //fprintf(infoFptr, "\n %i  Control Selected to Date file");
        for (i = 0; i < numStudyYears; i++)
          { for (j = 0; j < numAgeCats; j++)
              { if ((ctrlSel[i][j] != simMatrix[i][j].simCtrlsSel)  || ( simMatrix[i][j].simCtrlsNeeded != simMatrix[i][j].simTargetCtrls - ctrlSel[i][j]))
                  { // correct the problem
                    simMatrix[i][j].simCtrlsSel = ctrlSel[i][j];
                    simMatrix[i][j].simCtrlsNeeded = simMatrix[i][j].simTargetCtrls - ctrlSel[i][j];
                  }
                fprintf(infoFptr,"%i %i %i %i\n", simMatrix[i][j].TotalCasesProj, simMatrix[i ][j].simTargetCtrls,
                        simMatrix[i][j].simCtrlsNeeded, simMatrix[i][j].simCtrlsSel);//simMatrix[i][j].simCtrlsSel);
              }
          }
        fclose(infoFptr);
      }
    return 0;
  }


int calculateResponseRate()
  { /* Will Calculate the response rate on a cell basis and print out a nice table
       can be used to modify the sample fraction matrix
    */

    int refYr, dead, wave, ageNow, birthYr, ageDX, returned;
    long int id;
    char month[18], tempString[256];;
    int refAgeCat, refYrCat, ageRefYr;
    int i, j, garbage;
    int rowTotal, denomTotal, colTotal, colDenomTotal, totalTotal;

    // initialize ctrlSelection matrix
    for ( i = 0; i < numStudyYears; i++)
       for (j = 0; j < numAgeCats; j++)
          {  responseMatrix[i][j].targetCtrlNum = 0;
             responseMatrix[i][j].numCtrlSel = 0;
             responseMatrix[i][j].actualSel = 0;
```

```
            responseMatrix[i][j].numCtrlNeed = 0;
            responseMatrix[i][j].sampleFraction = 0.0;
            responseMatrix[i][j].numCtrlReturn = 0;
            responseMatrix[i][j].responseRate = 0.0;
        }

// read in what I think I have
if ((infoFptr = fopen("ctrlhist.sel\0", "rt")) == NULL)
   { printf("\n ERROR: Unable to open file the Wave Selected History file \n");
     return -1;
   }
else
   { //fprintf(infoFptr, "\n %i  Control Selected to Date file");
     for (i = 0; i < numStudyYears; i++)
        { for (j = 0; j < numAgeCats; j++)
             { fgets(tempString, 240,infoFptr);
               sscanf(tempString, "%i %i %i %i", &garbage, &responseMatrix[i][j].targetCtrlNum,
                                                  &responseMatrix[i][j].numCtrlNeed,
                                                  &responseMatrix[i][j].numCtrlSel);
               responseMatrix[i][j].numCtrlNeed = responseMatrix[i][j].targetCtrlNum;
             }
        }
     fclose(infoFptr);
   }

// read in what I actually have
if ((infoFptr = fopen("response.rat\0", "rt")) == NULL)
   printf("\n ERROR: Unable to open file the actual controls selected History file \n");
else
   { //fprintf(infoFptr, "\n %i  Control Selected to Date file");
     while (feof(infoFptr) == 0)
        { fgets(tempString, 240, infoFptr);
          if (feof(infoFptr) != 0)
             break;
          sscanf(tempString, "%i %s %i %i %li %i %i %i %i",
                 &dead, &month, &refYr, &ageDX, &id, &returned, &wave, &ageNow, &birthYr);// ,&yrKC1st, &yrKCLast); //simMatrix[i][j].simCtrlsSel);
          refYrCat = refYr - 1983;
```

```
        ageRefYr = refYr - (birthYr + 1900);
        if (refYr < (birthYr + 1900))
           printf("\n major prob agerefyr < birthyear");
        if (ageRefYr != ageDX)
           printf("\n Major Problem calculated age at DX != ageDX\n");
        // figure out the ref age cat
        for (i = 0; i < numAgeCats; i++)
           { if (ageDX >= ageCatLow[i] && ageDX <= ageCatHigh[i])
                { refAgeCat = i;
                  break;
                }
           }
        responseMatrix[refYrCat][refAgeCat].actualSel++;
        responseMatrix[refYrCat][refAgeCat].numCtrlNeed--;  // could modify and only count if it is returned see commented code line like this below
        if (returned == 1)
           { responseMatrix[refYrCat][refAgeCat].numCtrlReturn++;
             /*  // responseMatrix[refYrCat][refAgeCat].numCtrlNeed--; */
           }
      }
    fclose(infoFptr);
  }

// read in the sample fraction info
// qyery for proper file
    if ((infoFptr = fopen("sfavg.sav\0", "rt")) == NULL)
     { printf("ERROR: unable to open file sfavg.sav \n");
       exit(0);
     }

    for (i = 0; i < numStudyYears; i++)
       for (j = 0; j < numAgeCats; j++)
            fscanf(infoFptr, "%f", &responseMatrix[i][j].sampleFraction);

    fclose (infoFptr);

  // now print out a matrix showing the controls and cases and controls I have actually selected
  if ((infoFptr = fopen("FreqMtch.out\0", "at")) == NULL)
```

```
  { printf("ERROR: unable to open file freqMtch.out \n");
    exit(0);
  }

 if ((infoFptrOut = fopen("response.out\0", "wt")) == NULL)
  { printf("ERROR: unable to open file response.out \n");
    exit(0);
  }
// Print out the bootMatrix sample fraction matrix and the associated bootstrap CI's
fprintf(infoFptr, "\n Controls Selected to Date Check ");
fprintf(infoFptr, "\n                  18-19 20-24 25-29 30-34 35-39  40-44 45-49 50-54 55-59 60-64 65-69 70-74 TOTAL");
fprintf(infoFptrOut, "\n      18-19 20-24 25-29 30-34 35-39 40-44 45-49 50-54 55-59 60-64 65-69 70-74 TOTAL");

for (i = 0; i < numStudyYears; i++)
   { fprintf(infoFptr,"\n\n     Project Cases");
     rowTotal = 0;
     for (j = 0; j < numAgeCats; j++)
        { fprintf(infoFptr, " %5i", caseYrDX_AgeDXprojected[i][j]);//responseMatrix[i][j].casesProj);
          rowTotal += caseYrDX_AgeDXprojected[i][j];
        }
     fprintf(infoFptr, " %5i", rowTotal);

     fprintf(infoFptr,"\n     Target  Ctrls");
     rowTotal = 0;
     for (j = 0; j < numAgeCats; j++)
        { fprintf(infoFptr, " %5i", responseMatrix[i][j].targetCtrlNum);
          rowTotal +=  responseMatrix[i][j].targetCtrlNum;
        }
     fprintf(infoFptr, " %5i", rowTotal);

     fprintf(infoFptr,"\n     Sample Fractn" );
     for (j = 0; j < numAgeCats; j++)
        fprintf(infoFptr, " %5.2f", responseMatrix[i][j].sampleFraction);

     fprintf(infoFptr,"\n%i  Ctrls  Needed", i + 1983);
     rowTotal = 0;
     for (j = 0; j < numAgeCats; j++)
```

```
        { fprintf(infoFptr, " %5i", responseMatrix[i][j].numCtrlNeed);
          rowTotal += responseMatrix[i][j].numCtrlNeed;
        }
      fprintf(infoFptr, " %5i", rowTotal);

      fprintf(infoFptr,"\n     Ctrl Selected");
      rowTotal = 0;
      for (j = 0; j < numAgeCats; j++)
        { fprintf(infoFptr, " %5i", responseMatrix[i][j].actualSel);
          rowTotal += responseMatrix[i][j].actualSel;
        }
      fprintf(infoFptr, " %5i", rowTotal);

      fprintf(infoFptr,"\n     Num  Returned" );
      rowTotal = 0;
      for (j = 0; j < numAgeCats; j++)
        { fprintf(infoFptr, " %5i", responseMatrix[i][j].numCtrlReturn);
          rowTotal +=  responseMatrix[i][j].numCtrlReturn;
        }
      fprintf(infoFptr, " %5i", rowTotal);

      fprintf(infoFptr,"\n     Response Rate");
      rowTotal =  denomTotal = 0;
      fprintf(infoFptrOut,"\n%i  ", i + 1983);
      for (j = 0; j < numAgeCats; j++)
        { responseMatrix[i][j].responseRate = (float) responseMatrix[i][j].numCtrlReturn / (float) responseMatrix[i][j].actualSel ;
          fprintf(infoFptr, " %5.2f", responseMatrix[i][j].responseRate );
          fprintf(infoFptrOut, " %5.2f", responseMatrix[i][j].responseRate );
          rowTotal += responseMatrix[i][j].numCtrlReturn;
          denomTotal +=  responseMatrix[i][j].actualSel;
        }
      fprintf(infoFptr, " %5.2f", (float) rowTotal / (float) denomTotal);
      fprintf(infoFptrOut, " %5.2f", (float) rowTotal / (float) denomTotal);
    }

// do column sums now
```

```
fprintf(infoFptr,"\n\n     Project Cases");
totalTotal = 0;
for (i = 0; i < numAgeCats; i++)
   { rowTotal = 0;
     for (j = 0; j < numStudyYears; j++)
        rowTotal += caseYrDX_AgeDXprojected[j][i];
     fprintf(infoFptr, " %5i", rowTotal);
     totalTotal += rowTotal;
   }
fprintf(infoFptr, " %5i", totalTotal);

totalTotal = 0;
fprintf(infoFptr,"\n     Target  Ctrls");
for (i = 0; i < numAgeCats; i++)
   { rowTotal = 0;
     for (j = 0; j < numStudyYears; j++)
        rowTotal += responseMatrix[j][i].targetCtrlNum;
     fprintf(infoFptr, " %5i", rowTotal);
     totalTotal += rowTotal;
   }
fprintf(infoFptr, " %5i", totalTotal);

totalTotal = 0;
fprintf(infoFptr,"\nTOTAL Ctrls  Needed");
for (i = 0; i < numAgeCats; i++)
   { rowTotal = 0;
     for (j = 0; j < numStudyYears; j++)
        rowTotal += responseMatrix[j][i].numCtrlNeed;
     fprintf(infoFptr, " %5i", rowTotal);
     totalTotal += rowTotal;
   }
fprintf(infoFptr, " %5i", totalTotal);

totalTotal = 0;
fprintf(infoFptr,"\n     Ctrl Selected");
for (i = 0; i < numAgeCats; i++)
   { rowTotal = 0;
```

```
      for (j = 0; j < numStudyYears; j++)
         rowTotal += responseMatrix[j][i].actualSel;
      fprintf(infoFptr, " %5i", rowTotal);
      totalTotal += rowTotal;
   }
fprintf(infoFptr, " %5i", totalTotal);

totalTotal = 0;
fprintf(infoFptr,"\n     Num  Returned" );
for (i = 0; i < numAgeCats; i++)
   { rowTotal = 0;
      for (j = 0; j < numStudyYears; j++)
         rowTotal +=  responseMatrix[j][i].numCtrlReturn;
      fprintf(infoFptr, " %5i", rowTotal);
      totalTotal += rowTotal;
   }
fprintf(infoFptr, " %5i", totalTotal);

rowTotal = denomTotal = 0;
fprintf(infoFptr,"\n     Response Rate");
fprintf(infoFptrOut, "\nTotal ");
for (i = 0; i < numAgeCats; i++)
   { colTotal = colDenomTotal = 0;
      for (j = 0; j < numStudyYears; j++)
         { colTotal += responseMatrix[j][i].numCtrlReturn;
           colDenomTotal += responseMatrix[j][i].actualSel;
         }
      fprintf(infoFptr, " %5.2f",(float) colTotal /  (float) colDenomTotal);
      fprintf(infoFptrOut, " %5.2f",(float) colTotal /  (float) colDenomTotal);
      rowTotal += colTotal;
      denomTotal += colDenomTotal;
   }

fprintf(infoFptr, " %5.2f", (float) rowTotal /  (float) denomTotal);
fprintf(infoFptrOut, " %5.2f\n", (float) rowTotal /  (float) denomTotal);

fclose(infoFptr);
```

```
    fclose(infoFptrOut);
     return 0;
    }

char * monthName(int monthNum)
    { // returns the name of the nth month
      static char *monthNameArray[] = {"January", "February", "March", "April", "May", "June", "July",
                                       "August", "September", "October", "November", "December"};
      if (monthNum < 0 || monthNum > 11)
        printf("\n ERROR ILLEGAL MONTH %i", monthNum);
      return monthNameArray[monthNum];
    }


int frequencyMatch()
    { int i, j;
      //int numCtrlsInWave; // keepd a county of the controls in the wave
      int refMonth, refYear, deadFlag, ageThenAtRefYr;
      int waveNumber, waveNumberHigh, waveNumberLow;
      float temprand;
      char tempString[128];
      int tmpCtrlID, tmpCtrlWaveNum, tmpCtrlAge, tmpCtrlBirthYr, tmpCtrlAgeCat, tmpCtrlYrKC1st, tmpCtrlYrKCLast;
      Boolean used;

      // open up the case file and count up number of cases adjust for controls by adding 4 yrs to dx date so can get right number of controls
      printf("\n ENTER WAVE Number to Process:");
      waveNumberHigh = getValidResponse(maxWaveInt);
      waveNumberLow = getValidResponse(maxWaveInt);
      waveNumber = (waveNumberHigh * 10) + waveNumberLow; // crude but will work
      printf("\n Processing wave number %i",  waveNumber);

      // assume do on a week basis , but for test run all we have for first group
      /* See if the iseed file exists, if so open it and get the seed value */
      if ((infoFptr = fopen("SAVESEED.RNG\0", "rt")) == NULL)   // set up output file
        printf("\n ERROR: Initializing seed - unable to open file the Random Number Generator saved seed file \n");
      else fscanf(infoFptr, "%li", &iseed);
      fclose(infoFptr);
```

```
    // Read in the controlsSelectedToDate Info
    if ((infoFptr = fopen("ctrlhist.sel\0", "rt")) == NULL)  // set up output file
      printf("\n ERROR: Unable to open file the Wave Selected History file \n");
    else
      { //fprintf(infoFptr, "\n %i  Control Selected to Date file");
        for (i = 0; i < numStudyYears; i++)
          { for (j = 0; j < numAgeCats; j++)
              { fgets(tempString, 128,infoFptr);
                sscanf(tempString, "%i %i %i %i",
&simMatrix[i][j].TotalCasesProj,&simMatrix[i][j].simTargetCtrls,&simMatrix[i][j].simCtrlsNeeded,&simMatrix[i][j].simCtrlsSel);//simMatrix[i][j].simCtrlsSel);
              }
          }
        fclose(infoFptr);
      }

    // check to make sure sample fraction has been selected
    if ( sampleFractionSelected != True)
      { printf("\n READING in SampleFraction Matrix FILENAME = sfavg.sav");
        // qyery for proper file
        if ((infoFptr = fopen("sfavg.sav\0", "rt")) == NULL)
         { printf("ERROR: unable to open file sfavg.sav \n");
           exit(0);
         }

        for (i = 0; i < numStudyYears; i++)
          { for (j = 0; j < numAgeCats; j++)
              fscanf(infoFptr, "%f", &SFmatrix[i][j]);
          }

        sampleFractionSelected = True;
        fclose (infoFptr);
      }

    // open output wave file so can output as we go
    if ((infoFptrOut = fopen("ctrlwave.out\0", "wt")) == NULL)  // set up output file
      { printf("ERROR: unable to open file CONTROLS.SELDAT \n");
```

```
        exit(0);
      }
    // open the wave file to be selected
    if ((infoFptr = fopen("ctrlwave.all\0", "rt")) == NULL) // modify so that the name of the wave is ctrlwave.xzz  x is either a or b and zz is the wave number a is the
fisrt time for wave b is the hold out recalled who agree to play later
      { printf("\n ERROR: unable to open file ctrlwave.all \n");
        exit(0);
      }

    // open the wave history informationfile
    if ((selInfoFptr = fopen("waveinfo.his\0", "at")) == NULL) // modify so that the name of the wave is ctrlwave.xzz  x is either a or b and zz is the wave number a
is the fisrt time for wave b is the hold out recalled who agree to play later
      { printf("\n ERROR: unable to open file waveinfo.his \n");
        exit(0);
      }
    fprintf(selInfoFptr,"CtrlID Wave# CtrlAge YrsKC AgeThenIndx YrsKCThenIndx () deadflag used, refmonth, reefYr, ageCatIndx, Seed %d\n", iseed);

    // open the file to list the not selected controls
    if ((notSelInfoFptr = fopen("ctrlnot.sel\0", "wt")) == NULL) // modify so that the name of the wave is ctrlwave.xzz  x is either a or b and zz is the wave number a
is the fisrt time for wave b is the hold out recalled who agree to play later
      { printf("\n ERROR: unable to open file ctrlNot.sel \n");
        exit(0);
      }
    fprintf(selInfoFptr,"CtrlID Wave# CtrlAge YrsKC AgeThenIndx YrsKCThenIndx () deadflag used, refmonth, reefYr, ageCatIndx, Seed %d\n", iseed);
    // read into the ctrl data structure
    // needs to be redone calculate permissible cells will return the cell that the control can be used in, and the sample fraction
    //numCtrlsInWave = 0; // initialize the index that used to fill up the array of conntrol information
    while (feof(infoFptr) == 0)
      { fgets(tempString, 128, infoFptr);
        if (feof(infoFptr) != 0)
          break;
        // read in the control info and save it in control info structure
        sscanf(tempString, "%li %i %i %i %i %i %i %i",&tmpCtrlID, &tmpCtrlWaveNum, &tmpCtrlAge, &tmpCtrlBirthYr, &tmpCtrlYrKC1st, &tmpCtrlYrKCLast);

        if (tmpCtrlWaveNum != waveNumber)
          continue;
```

```
    fprintf(selInfoFptr, "\n%i %i %i %i %i", tmpCtrlID, tmpCtrlWaveNum, tmpCtrlAge, tmpCtrlYrKC1st, tmpCtrlYrKCLast);

    if ((tmpCtrlAge < 22) || (tmpCtrlYrKC1st < 83) || (tmpCtrlYrKCLast > 91) || (tmpCtrlAge > 86))
      { if (tmpCtrlAge > 86) printf("\n ERROR illegal control Age > 86 %s", tempString);
        if (tmpCtrlAge < 22) printf("\n ERROR illegal control Age < 22 %s", tempString);
        if (tmpCtrlYrKC1st < 83) printf("\n ERROR control years in King County < 1983 years %s",tempString);
        if (tmpCtrlYrKCLast < 83) printf("\n ERROR control years in King County > 1991 years %s",tempString);
      }
    else
      { /* Key Routine - actually does the frequency matching
           Note: Both the ctrls and cases use their age as of now and ctrls are assigned to yrs btween 83 - 91 */
        used = calculatePermissibleCells(tmpCtrlYrKC1st, tmpCtrlYrKCLast, tmpCtrlBirthYr, &refYear, &tmpCtrlAgeCat); // x is year, y is age cat
        if (used == True) // SF to use Note: A lot of the cells at upper age have a SF = 1.0
          { temprand = rand2(&iseed);
            // then we include this potential as a control so now we assign a ref month and ref year to the person for this age category
            refMonth = long(temprand * 10000)%12; // note months go 0 - 11
            refYear +=  1983;
            ageThenAtRefYr = refYear - (tmpCtrlBirthYr + 1900);
            temprand = rand2(&iseed);
            if (temprand <= caseAgeSpecificMortalityRate[tmpCtrlAgeCat]) // make suri I have read this is.
              deadFlag = True; // dead
            else deadFlag = False;
            fprintf(infoFptrOut, " %i\t%s\t%i\t%i\t%s", deadFlag, monthName(refMonth), refYear, ageThenAtRefYr, tempString);
            fprintf(selInfoFptr,  " %i USED %s %i %i %i", deadFlag, monthName(refMonth), refYear, tmpCtrlAgeCat, ageThenAtRefYr);
          }
        else { fprintf(notSelInfoFptr, "%s", tempString);
               fprintf(selInfoFptr, "   NOT_USED");
             }
      }
  }
fclose(selInfoFptr);
fclose(infoFptr);     // close the wave input file
fclose(infoFptrOut);  // close the wave output file
fclose(notSelInfoFptr);

// save the seed for the next run, make a file of RNs to make sure not saving the wrong ones?
if ((infoFptr = fopen("SAVESEED.RNG\0", "wt")) == NULL)   // set up output file
```

```
    printf("\n ERROR: Unable to open file the Random Number Generator saved seed file \n");
  else
    { fprintf(infoFptr,"%li  Saved Seed for the RNG", iseed);
      fclose(infoFptr);
    }
  // also save a history of the iseeds used for each run
  if ((infoFptr = fopen("SEEDHIST.RNG\0", "at")) == NULL)   // set up output file
    printf("\n ERROR: Unable to open file - Random Number Generator History seed file \n");
  else
    { fprintf(infoFptr,"\n Wave# %li  Saved Seed: %i for the RNG", waveNumber, iseed);
      fclose(infoFptr);
    }
  // should also save the sample fraction for this wave since will need it in 2 months for the stragglers
  // ******** ALSO save the rddinfo ctrl file  ********** Number selected so far so can read in for next wave
  if ((infoFptr = fopen("SPFRLHIST.RNG\0", "at")) == NULL)   // set up output file
    printf("\n ERROR: Unable to open file the Wave Sample Fraction History file \n");
  else
    { fprintf(infoFptr, "\n %i ", waveNumber);
      for (i = 0; i < numStudyYears; i++)
        { for (j = 0; j <= numAgeCats; j++)
            fprintf(infoFptr,"%f\n", SFmatrix[i][j]);
        }
      fclose(infoFptr);
    }

  //Save the ctrl selected matrix info;


  if ((infoFptr = fopen("CTRLHIST.SEL\0", "wt")) == NULL)   // set up output file
    printf("\n ERROR: Unable to open file the Wave Selected History file \n");
  else
    { //fprintf(infoFptr, "\n %i  Control Selected to Date file");
      for (i = 0; i < numStudyYears; i++)
        { for (j = 0; j < numAgeCats; j++)
            fprintf(infoFptr,"%i %i %i %i\n", simMatrix[i][j].TotalCasesProj, simMatrix[i][j].simTargetCtrls, simMatrix[i][j].simCtrlsNeeded,
simMatrix[i][j].simCtrlsSel);//simMatrix[i][j].simCtrlsRespond);
        }
```

```
    fclose(infoFptr);
  }

if ((infoFptr = fopen("Ctrlweek.his\0", "at")) == NULL)   // set up output file
  printf("\n ERROR: Unable to open file the Wave Selected History file \n");
else
  { //fprintf(infoFptr, "\n %i  Control Selected to Date file");
    fprintf(infoFptr, "\n   18-19 20-24 25-29 30-34 35-39 40-44 45-49 50-54 55-59 60-64 65-69 70-74");
    fprintf(infoFptr, "\n WaveNumber %i\n", waveNumber);
    for (i = 0; i < numStudyYears; i++)
      { fprintf(infoFptr, "\n%i",1983 + i);
        fprintf(infoFptr, "\nCaseProjected ");
        for (j = 0; j < numAgeCats; j++)
          fprintf(infoFptr, "%3i ", simMatrix[i][j].TotalCasesProj);
        fprintf(infoFptr, "\nTarget Ctrls  ");
        for (j = 0; j < numAgeCats; j++)
          fprintf(infoFptr, "%3i ", simMatrix[i][j].simTargetCtrls);
        fprintf(infoFptr, "\nCtrls Needed  ");
        for (j = 0; j < numAgeCats; j++)
          fprintf(infoFptr, "%3i ", simMatrix[i][j].simCtrlsNeeded);
        fprintf(infoFptr, "\nCtrls Selecte ");
        for (j = 0; j < numAgeCats; j++)
          fprintf(infoFptr, "%3i ", simMatrix[i][j].simCtrlsSel);

        /*fprintf(infoFptr, "\nResponse Rate ");
        for (j = 0; j < numAgeCats; j++)
          fprintf(infoFptr, "%3i ", simMatrix[i][j].simCtrlsResponse);
    */
      }
    fclose(infoFptr);
  }

return 0;
}

/* Stuff that still needs to be done
   ==> Make sure tracking the response rate */
```

```
/* clean.h file for headers
   Prog in (from SPSS/DX)
   variable  Case/Control */
//Input Variabless     NR SPSS/DE name     purpose spss def'n                    xform output variables category definition
int ageAtRef;     /*   1    AGE@REF       -> age at reference date               */ int ageRefCat;       //  AGE CAT 0=18-19, 20<=1<25...70<=11<75
int ageDiabDX;    /*   2    AGEDX         -> age at DX of diabetes               */ int ageDiabCat;      //  AGE CAT if dx < 18 use lowest cat
int ageOophect;   /*   3    AGEOVECT      -> age at 2nd(if both) oopherectomy    */ int ageOophCat;      //  AGE CAT if < 18 at ooph use lowest cat
int agePause;     /*   4    AGEPAUSE      -> age at memapause                    */ int agePauseCat;     //  AGE CAT if < 18 at Pause use lowest cat
int age1stBsurg;  /*   5    AGESURG       -> any breast surgery                  */ int ageBScat;        //  AGE CAT if < 18 at 1st Burrg use lowest
int anyBsurg;     /*   6    BSURGERY      -> any surgery to breast, 1=No, 2=Yes  */ int BScat;           //  BIN CAT 0=NO, 1=YES
int BcancerDX;    /*   7    CANCER        -> DX of breast cancer   1=No, 2=Yes   */ int BCcat;           //  BIN CAT 0=NO, 1=YES
int caseNum;      /*   8    CASE#         -> original case number in SPSS/DE     */ //  SAME OUTPUT  //  NOT USED ORDINAL Case ID
int hairColor[5];  /*  9-13 COLOR1-5      -> hair dye color trying to achieve    */                      //  NOT Used in Analysis at Present
int caseFlag;     /*   14      CONTROL     -> ctrl = 2, case = 1                  */ int caseCtrl;        //  BIN CAT 0=Control 1=Case
char BCyearDX[6]; /*   15   DATEC         -> date of first breast cancer DX      */ int ageBCDXcat ;     //  AGE CAT AGE if < 18 use lowest cat
char dateSPP[6];/*   16   DATESPP     -> date other surgery related to BI's      */                      //  Not used
char dateSPP2[6];  //   17   DATESPP2
char dateSPP3[6];  //   18   DATESPP3
char dateBsurg[6];  /*   19   DATESURG  -> date first breast surgery any kind   */                      //  NOT USED Will use ageCat
char dateCTDSX[6];/*   20   DATESX       -> date of first SX of CTDs            */ int ageCTDsxCat;     //  AGE CAT AGE   of first CTD SX
int diabDX;       /*   21      DIABETES    -> before ref date if diabetes DX by MD */ int diabCat;       //  BIN CAT 0=NO, 1=YES DX of Diabetes
int job1Dur;      /*   22      DUR1        -> job1 duration derived   < DX date   */                      //  NOT USED start - end months
int job2Dur;      /*   23      DUR2        -> job2                                */
int job3Dur3;     /*   24      DUR3        -> job3                                */
float ICDcodeDX;/*   25      DXCODE      -> ICD-9 code for main disease       */ int DXcat;  // NOM,CAT //0=Ctrl,1=SLE,2=PSS,3=SS,4=PM/DM,5=MCTD
float ICDcodeDX2;/*  26      DXCODE2     -> ICD-9 code 2nd CTD DX of interest  */ int DX2cat;          //  NOM CAT ICD 2nd DX
int dyeBrand[5];  /* 27-31 DYEBRAND1-5  -> Hair dye Brand                      */                      //  NOT USED
int dyeUseMin[5];/*32-36  DYEEXPO1-5     -> Hair dye exposure in minutes       */ int dyeExpMin;       //  ORD CAT exposure in minutes
int dyeFirst[5];/*37-41  DYEFIRST1-5   -> Hair dye age first used < REF date  */ int logDyeExpMin;  //  ORD CAT log(hair dye exp minutes)
int dyeLast[5];   /*  42-46     DYELAST1-5 -> Hair dye age last used  < ref date  */                      //  NOT output   Array dyeLast[5]
int hairDyeUse;   /*   47      DYES        -> ever use hair dye 1=NO, 2=Yes     */ int dyeCat;          //  BIN CAT 0=NO, 1=YES hair dye use
int ectopic;      /*   48      ECTOPIC     -> number of ectopic pregnancys       */ // SAME OUTPUT  //  ORD CAT num ectopic preg
```

```
int educatn;       /*  49     EDUCAT      -> higest level of education in YEARS      */ int eduCat;       // ORD CAT years of education,
int job1End;       //  50     END1        -> year job1 end should be char array                          // NOT USED AT PRESENT
int job2End;       //  51     END2
int job3End;       //  52     END3
int estrogenUse;   /*  53     ESTROGEN    -> Ever use estrogen, etc. 1=No, 2=Yes */ int estrogenCat;  // BIN CAT 0=No, 1=YES esestrogen use
int height;        /*  54     HEIGHT   -> maximum adult height                 */ //  SAME OUTPUT //  INT CAT height inches weight -> BMI
int implantUse;    /*  55     IMPLANT     -> Ever use breast implants 1=No, 2=Yes */ int BIcat;          // BIN CAT 0=No, 1=Yes Breast Implants ,
int reasonBI;      /*  56     IMREASON    -> coded version of responses           */ int BIreasonCat; // NOM CAT Reason for BI
int otherTypeBI;   //  57     IMSPP       -> other type of not listed BI material*/                      //  NOT USED   include code in IMTYPE
int implantType;   /*  58     IMTYPE      -> listed types of BI add IMSPP to cat  */ int BItypeCat;    // NOM CAT breast implant types
int income;        /* 59      INCOME      -> income categories                    */ int incomeCat;    // ORD CAT 0:<15k, 1:<30k, 2:<45k,3:>45k
int indAbort;      //  60     INDABORT    -> number of induced aborytions         SAME OUTPUT */ // ORD CAT Number induced abortion
int indus1;        //  61     INDUST1     -> industry code                        */                     //      NOT USED Look at induatrial risk
int indus2;        //  62     INDUST2
int indus3;        //  63     INDUST3
int indusRef;      /*  64     INDUSTREF   -> industry worked in at reference date*/ int industCat;      // NOM CAT induatry at REF
int Job1;      //  65      JOB1       -> Job code i for industry i                                  //  NOT USED
int Job2;      //  66      JOB2
int Job3;      //  67      JOB3
int jobRef;        /*  68     JOBREF      -> Job at reference Date in RefInd      */ int jobCat;       //  NOM CAT house work 0:housewife, 1:Otherwise
int nowKC;         /*  69     KCNOW       -> Current KC resident                  */ int nowKCcat;    //  BIN CAT 0=NOT KCNOW, 1=KCNOW
int thenKC;        /*  70     KCTHEN      -> KC resident at ref date                */ int thenKCcat;// BIN CAT 0=NOT KCTHEN 1 = KCTHEN
int lengthOC;      /*  71     LENGTHOC    -> duration(months) OC use <= RefDate   */ //  SAME OUTPUT     // INT CAT Months OF OC use <= RefDate
int liveBirths;    /*  72     LIVEONE     -> Number of live births                */ //  SAME OUTPUT     // INT CAT Number of Live births
int marital;       /*  73     MARITAL     -> marital status at refDate            */ int maritalCat;   //  BIN CAT 0=Single, 1=Married
int ageMenarche;/*  74        MENARCHE    -> Age ant menarche                     */ int menarchCat;  //  INT CAT age begins
int menses;        /*  75     MENSES      -> menstrual period status at REFDATE */ int mensesCat;   //  BIN CAT 0=no period, 1=regular period,
int numMisCar;     /*  76     MISCARRY    -> number of miscarriages               */ //  SAME OUTPUT     // ORD CAT Number of miscarriages
int multBirth;     /*  77     MULTI       -> Number of multiple births            */ //  SAME OUTPUT     // ORD CAT NUMBER OF MULTIPLE BIRTS
int natPause;      /*  78     NATPAUSE    -> Pperiod stop naturally 1=NO 2=Yes    */ int natPauseCat; //  BIN CAT 0=NO,1=Yes
int numDyePerYr[5];/* 79-83 NUMDYE1-5   -> NumberTimes DYE hair per year line1  */                     //      NOT OUTPUT USED calhair dye exp
int OCuse;         /*  84     OCUSE       -> AT REFDATE best describes OC use     */ int OCcat;        //  ORD CAT 0=Never, 1=Current Use, 2=Past Use
int otherBirth;    /*  85     OTHER       -> Other Birth Outcome                  */ // SAME OUTPUT      // Nominal other than listed birth outcomes
int otherEdu;      /*  86     OTHEREDU    -> Other higest level of of education  */                     //  ORD CAT Higest ever of education
int otherBSproc;   /*  87     OTHERPRO    -> OTHER PROcedure for breast surgery  */                     //  ORD CAT incorporate w/ PROCEDURE breast surg
int otherBSpurp;   /*  88     OTHERPURP-> OTHER PURPose  Breast surg any type    */                     //      incorporate w/ PURPOSE any Breast surgery
```

```
int otherMenpause;/*   89   OTHERWHY     -> OTHER reason not List menses stop */                    //      incorporate w/ WHYPAUSE reasons listed
int oophectomy;    /*  90   OVECT        -> Oophectomery b4 REFDATE 1=NO, 2=YES   */ int oophectCat;  // ORDCAT
=NO,1=1,2=2,3=1l2,5=Partial,6=DK
int phone;         /*  91   PHONE        -> Phone in home currently              */ int phoneCat;    // BIN CAT Eliminate CASES w/out phone
char dateFirstBI[6];/*  92   PLACED       -> Date BI first placed                 */ int BIageCat;    // AGE CAT age BI ist placted DUR of BI <=REF
int problemsBSBI;  /*  93   PROBLEMS     -> cat problems with BI's               */ int BSBIprobCat; // ORD CAT 0=NO, 1=LEAKING, 2=OTHER PROB
int procedureBSBI;/*   94   PROCEDUR     -> How Breast size shape modified       */ int procBSBIcat; // ORD  CAT  0=SBI,  1=SBI,  2=Siliconedirect  into
breast, 4, other, X=DK
int BSpurpose;     /*  95   PURPOSE      -> purpose of any 1st breast surg       */ int purpBScat;// ORD CAT 0=increase, 1=DE, 2=Shape, 3=1 remove,
4=2 remove,  5=other reasons7=DK,
int race;          /*  96     RACE       -> race                                 */ int raceCat;     // ORD,CAT,race
0=W,1=B,2=NatAm,4=Asian,5=Hispanic,6=Other,7=Refuse, DK
char refDate[6];   /*  97     REFDATE    -> reference date DX I freq match date  */ int refYearCat;  // ORD CAT 0=83,... 8=91
int specBSprob;    /*  98     SPECPROB   -> Complications of Breast modification*/ //int BSBIprobsNOS;  // ORD CAT probs after BS for BI
int raceOther;     //  99     SPPRACE    -> Other Race                                    //      Incorporate int raceCat
int otherBIreason; //  100    SPPREAS    -> SPPREAS other reason for BI placed          //      Incorporate into reasonBI (IMREASON)
int jobStart1;     //  101    START1     -> year job1 start                             //      NOT USED
int jobStart2;     //  102    START2
int jobStart3;     //  103    START3
int stillBirth;    /*  104    STILL      -> Number of Still Births               */ //  SAME OUTPUT    // ORD CAT Num still births
int studyID;     /*  105    STUDYID    -> Study ID                             */ //  SAME OUTPUT    // ORD CAT STUDY ID
int otherBIsurg;   /*  106    SURGELSE   -> Other surgery for BI's No=1, Yes=2   */ int postBIsurgCat;// BIN CAT 0= NO, 1=YESint
int surgPurBI2;    //  107    SURGPUR3   -> Purpose Surg BI related after 1st    */                    //      Seperate Analysis NOT USED
int surgPurBI3;    //  108    SURGPUR2   -> Purpose Surg BI related after 1st    */
int surgPurBI1;    //  109    SURPURP    -> Purpose Surg BI related after 1st    */
int initCTDSX;     //  110    SXSPP      -> Specific 1st CTD SX                  */                    //      NOT USED Seperate analysis
int timePill;      //  111    TIMEPILL   -> Number months other RX for diabetes                       //      NOT USED at the moment
int timeShot;      /*  112    TIMESHOT   -> # months insulin used b4 refDate     */ int insulinExpCat;// ORD CAT #months used insulin < RefDate
int diabRX;        /*  113    TX         -> Treatment for diabetes               */ int insulinCat;    // CAT 1=Insulin 0=Other or not used
int weight;        /*  114    WEIGHT     -> Weight in Lbs.                       */ //  SAME OUTPUT // INT CAT weight, USE BMI for cats
int whatOtherBirthOut;/*115   WHAT       -> Other birth outcome not listed       */ //  SAME OUTPUT //  NOM CAT Incorporate into CAT of G, P, Ab, etc
int whyPause;      /*  116    WHYPAUSE   -> if not NATPAUSe reason               */ int whyPauseCat;//NOM CAT whyMpause incorporate otherWhy
int eduYrsComp;    /*  117    YRCOMP     -> Years of Edu Compleated at level     */ int eduYrsCat;//  INT CAT number years of education


/* DERIVED OUTPUT VARIABLES */
```

```
int ageCat6;            // ORD CAT 6 age cate 0=18-29 1=30-39 2=40-49 3=50-59 4=60-69 5=70+
int refYear3;           // ORD CAT 3 ref years 0=83-85 1=85-87 2=88-91 3 years in each
int lnWeight;           // ORD CAT ln weight
int logWeight;          // ORD CAT log weight
int weightCat;          // ORD CAT 0 <= 110, 1 <= 125, 2 <= 135, 3 <= 150, 4 <= 175, 5 175+
int BMI;                // ORD CAT body mass index
int birthYear;          // ORD INT CAT 2 digits
int gravid;             // ORD INT CAT total num preganancies
int expBIb4RefDate;     // ORD INT CAT refYear - yearFirstBI;
int anyImplantCat;      // BIN CAT any type implant
int refMonth;           // ORD CAT reference date month
int refYear;            // ORD CAT reference date year
int workOutSideHome;    // BIN CAT 1 = outside 0 = houselife
int BIleak;             // BIN CAT 0=NO LEAK 1= BI leak prob
int BIactualProbCode;   //NOM CAT actual BI prob list
int BIprocCat;          // ORD CAT Actual BI type
int BIdateLTrefDate;    // BIN CAT flag for Breast implantation date (month, year) <= refDate (monthm year)
int binRace;            // BIN CAT 0=white, 1=other
int raceCat3;           // NOM CAT 0=white, 1=black 2=other
int hysterectomyCat;    // BIN CAT 0=NO, 1=YES hysterectomy <= refdate
int polyCCflag;         // flag for Polychotomous Logistic Regression PLR
int binGravid;          // BIN CAT 0=No Pregnancys 1=YES pregnamt
int binKids;            // BIN CAT 0 = NO LIVE births 1 = YES live Births
int binOCuse;           // BIN CAT 0=NoOCUSE 1=OCuseever
int binMarit;           // BIN CAT ever married 0=single 1=all other
int binOoph;            // BIN CAT oophectomy
int binMenses;          // BIN CAT menses status at REF 0=Still having, 1=Stopped
int logLenOC;           // log10 of length of OC use
int BIduration;         // ORD CAT months of BI exposure <= refdate
int BIlogDur;           // ORD CAT LOG months of BI exposure <= refdate
int BIlnDur;            // ORD CAT LN months of BI exposure <= refdate
int BSnoBI;             // BIN CAT Breast Surgery other than for Breast Implants
int DXDvalid;           // BIN CAT case had DXD and not just DXP
int corrBinMens;        // BIN CAT corrected menstrual category from menses likely age at menapause

// CORE CTRL VARS
int coreAgeBCA;         // core age at Breast Cancer DX
```

```
int coreREFMONTH;       // core ref month year we will use
int coreREFYEAR;        // core refyear we will use and not compute from string as in case and CTRLS
int coreBIRTHMM;
int coreBiRTHYR;
int coreHTFT;
int coreHTIN;
int coreGravid;
int coreSNHYST;
int coreHRMATREF;

// Make arry of incidence of Diseases x Year
int KCpopulation8391[14][12];

int whiteKCpop8391[14][12];
int blackKCpop8391[14][12];
int otherKCpop8391[14][12];
int wbiaho90KCpop[14][12];

int ageDXyrDXReturnCases[14][12];
int ageDXyrDXReturnCtrls[14][12];
int ageDXyrDXCoreCtrls[14][12];

int CTRLageDXyrDX[14][12];
int COREageDXyrDX[14][12];
int SLEageDXyrDX[14][12];
int PSSageDXyrDX[14][12];
int SSageDXyrDX[14][12];
int MCTDageDXyrDX[14][12];
int PMDMageDXyrDX[14][12];
int ALLageDXyrDX[14][12];
int BlctrlAgeDXyrDX[14][12];
int BlcoreAgeDXyrDX[14][12];
int BlcaseAgeDXyrDX[14][12];
int allIDcasesAgeDXyrDX[14][12];// all diseases found
float caseResponseRate[14][12];
int raceDiseaseTable[14][12]; // disease race matrix for incidence paper need to know breakdown of race and disease
// document output variable order and which ones
```

```
// also tables for incidence paper

struct caseRecord { int cStudyID;
                    int returnedQuest;
                    int validDXD;
                    int cNmbr;
                    int cRace;
                    int MDid;
                    int deadCase;
                    char cBirthDate[6];
                    int cBirthYr;
                    int cBirthMonth;
                    int DX1;
                    int cYrDX1;
                    int ageDX1;
                    int cAgeDX1cat;
                    int cYrDX1cat;
                    char DXD1[6];
                    char DXP1[6];
                    int DX2;
                    int ageDX2;
                    int yrDX2;
                    int caseDX2cat;
                    int DXD2;
                    int DXP2;
                    char FirstRheum[6];
                    char FirstMd[6];
                    char dateSXstart[6];
                    int NumCriteria;
                    int cDiab;
                    int cInsulin;
                    int cAgeOnsetDiab;
                    int cYrOnsetDiab;
                    int steroidDiab;
                    char DateLastVisit[6];
                  };
struct caseRecord caseInfo[600];
```

```
/* RFCTD DATA CLEAN AND Variable TRANSFORM PROGRAM - r*/

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>

#include <C:\Research\Rfctdata\Casectrl\rfctdataClean\clean.h> // has definitions for the data arrays var in names, var ou names, range checks for input data
//#include <C:\Research\Rfctdata\Casectrl\testclean\clean.h> // Ohio changeshas definitions for the data arrays var in names, var ou names, range checks for
input data

double pow(double, double);

enum boolean {True = 1, False = 0};
enum prob {noProblem = 0, problem = 1};
enum openClose {dclose = 0, dopen = 1};

int const wideFlag = 1;


    int generateIncidenceTables();
    int getIncidenceData();
    int readKCpopulationData();
    int readDiseaseData();
    int getcaseData(int casesReadSoFar);
    int addToDiseaseMatrix(int ageYrmatrix[14][12], int incidDisArr[14][12]);
    int calculateIncidenceRates();
    int calculateAgeSpecificRaceIncidenceRates();
    int readRaceDataKC90();
    int openDiseaseFile(char fileName[90]);
    int initializeMatrix(int mat[14][12]);
    int calculateWideRaceAgeDiseaseSpecificIncidenceMatrix(int wideType, int diseaseType, int ageDXyrDXMat[14][12]);
    int calculateRaceDiseaseIncidenceMatrix(int wideFlag, int raceIndex, int diseaseType, int ageyrincidMat[14][12]);
    int calculateDiseaseIncidenceMatrix(int wideFlag, int diseaseType, int disMat[14][12]);
    int processCOREBreastSurgeryToImplant();
    int UpdateTables(int ageRefCAt, int refDateCat, int caseCtrl, int coreCtrl, int caseDiseaseDataIndx, int studyID, int caseNum);
    int outputActualCaseCtrl();
```

```
int initActualCaseCtrls();
int prtAgeDXyrDXMat(int *ageYrMatPtr, char Title[80]);
int readCaseDiseaseData();
int readCaseDiseaseRecord(FILE *fptr, char fname[80]);
int processCaseDiseaseRecord(int caseLinesRead);
int processCaseDiseaseRecord(int caseNum);
int findCaseIndex(int ID);
int readKCpopRaceData(int matrix[14][11], FILE *fptr, char tableName[80]);
int MMYY_MMDDYY(char my[6], char mdy[9]);
int printCaseResponseRateTable();
int printRowCols(int row, int labelrow, int mat[14][12]);
int printDiseaseTableLabel(int diseaseC);
int prtAgeRaceAvgMat(int ageYrMat[14][12], int diseaseType, char Title[80]);
float printRowColsFloat(int row, int labelrow, float matrix[14][11]);
int printTableLabel(int raceC, int diseaseC);
int printPopMatrix(long popMatrix[14][12]);
int printIncidenceMatrix(int wideFlag, int raceCat, int diseaseCat, float incidMat[14][12]);
int printAgeRaceIncidenceMatrix(int wideFlag, int diseaseCat, float incidMat[14][12]);
int printRaceDiseaseInfo();


int cleanRecords();
int checkControls();
int checkCases();
int checkCoreControls();
int outputRecordToFile(FILE *oFile, int coreFlag);
int outputRecord(int casectrl, int core);
int getNextDataRecord(int, int, FILE *);
int convertToken(int, int, char[100]);
int processCaseControlRecord(int caseCtrlFlag, int coreFlag);
int ICDtoInteger(float ICDcode, int studyID, int caseNum);
int ProcessDXcodes(float ICDcodeDX, float ICDcodeDX2, int studyID, int caseNum);
int checkAge(int ageAtRef, int refMonth, int refYear, int cBirYr, int cBirMonth, int studyID, int caseNum);
int calculateAgeCatAtReferenceDate(int, int, int);
int calculateAgeCategory(int, int, char[], int);
int calculateWideAgeCategory(int age, char VarName[15], int studyID, int caseNum);
```

```
int calculateRefYearCat(char refDate[], int coreflag, int, int);
int calculateBMI(int weight, int height, int studyID, int caseNum);
int makeBinaryCat(int inVar, int studyID, int caseNum); // actually 0=No, 1=Yes, 9=DK or missing
int calculateHairDyeTime(int hairDyeCat, int studyID, int caseNum);
int checkDiabetesVars(int studyID, int caseNum);
int checkBinaryCat(int binValue, char varName[], int studyID, int caseNum);
int processMenarch(int ageMenarche, int studyID, int caseNum);
int processPregnancyHx(int coreFlag, int studyID, int caseNum);   // will also process parity, abortions, etc;
int processMenses(int menses, int coreFlag, int studyID, int caseNum);
int processOveries(int oophectomy, int studyID, int caseNum);
int checkBinaryCat(int estrogenUse, char[], int studyID, int caseNum);
int processOCuse(int OCuse, int coreFlag, int studyID, int caseNum);
int crossCheckMenstrualVars(int studyID, int caseNum);
int processBreastCancer(int BcanDX, char yearBCDX[6], int yrBorn, int coreFlag, int studyID, int caseNum);
int processBreastSurgery(int anyBsurg, int coreFlag, int studyID, int caseNum);
int processBreastImplants(int implantUse, int studyID, int caseNum);
int crossCheckBreastCats(int studyID, int caseNum);   // check to make sure the implant related info is consistent
int processCOREbreastSurgeryToImplantVars();
int checkHairDyeUse(int hairDyeUse, int studyID, int caseNum);
int calculateHairDyeTime(int hairDyeCat, int studyID, int caseNum);
int ProcessMaritalStatus(int marital, int studyID, int caseNum);
int checkRace(int race, int raceOther, int caseDiseaseDataIndx, int caseFlag, int coreFlag, int studyID, int caseNum);
int checkEducation(int educatn, int otherEdu, int yrsComp, int coreFlag, int studyID, int caseNum); // two eduCat is level, yearsEdu is Ordered
int checkIncome(int income, int studyID, int caseNum);
boolean preProcessCoreControlRecord();
int calculateAgeCatFromYearDateString(char date[6], int birthYear, char varName[15], int studyID, int caseNum);
int calculateAgeFromDateString(char date[6], int birthYr, char varName[15], int studyID, int caseNum);
int ProcessWorkHistory(int jobRef, int studyID, int caseNum);


int numCasesOk = 0;
int numCases = 0, numSLEcases = 0, numPSScases = 0, numSScases = 0,
   numMCTDcases = 0, numPMDMcases = 0, numCtrls = 0, numCores = 0;
int numLinesRead = 0;
int caseLinesRead = 0;

boolean recordProblem = False; // Global reset for each record
```

```
const int numVars = 117; // 117 variables in the case and control data files; 130 in the CORE control file
const int numCOREvars = 127;

FILE *inFile, *errFile, *debFile, *KCpopFile, *caseFptr, *outFile, *bugFile, *coreFile, *allDisFptr, *missCore, *cccFile;


int main() {

   printf("\nRFCTD DATA CLEAN PROGRAM 9/7/96");
   // open the error file
   if ((errFile = fopen("errFile.lst", "w" )) == NULL ) {
      printf( "The file 'errFile.lst' was not opened\n" );
      exit(-1);
   }
   else
      printf( "The file 'errorFile.lst' was opened\n" );

   fprintf(errFile, "\nRFCTD DATA CLEAN PROGRAM");

   // open the debug file
   if ((debFile = fopen("debFile.lst", "w" )) == NULL ) {
      printf( "The file 'debFile.lst' was not opened\n" );
      exit(-1);
   }
   else
      printf( "The file 'debFile.lst' was opened\n" );

   fprintf(debFile, "\nRFCTD DATA CLEAN PROGRAM");

   // open the bug file
   if ((bugFile = fopen("bugFile.lst", "w" )) == NULL ) {
      printf( "The file 'bugFile.lst' was not opened\n" );
      exit(-1);
   }
   else
      printf( "The file 'bugFile.lst' was opened\n" );
```

```
    fprintf(bugFile, "\nThe file 'bugFile.lst' was opened\n" );


// open the output file
if ((outFile = fopen("C:/RESEARCH/RFCTDATA/ANALYSIS/rfctd.raw", "w" )) == NULL ) {
    printf( "The file 'rfctd.raw' was not opened\n" );
    exit(-1);
}
else
    printf( "The file 'rfctd.raw' was opened\n" );


// open the output file for cases and CORE controls
if ((coreFile = fopen("C:/RESEARCH/RFCTDATA/ANALYSIS/rfctdCor.raw", "w" )) == NULL ) {
    printf( "The file 'rfctdCor.raw' was not opened\n" );
    exit(-1);
}
else
    printf( "The file 'rfctdCor.raw' was opened\n" );

// open the output file for Cases, Controls and CORE controls
if ((cccFile = fopen("C:/RESEARCH/RFCTDATA/ANALYSIS/rfctdCCC.raw", "w" )) == NULL ) {
    printf( "The file 'rfctdCCC.raw' was not opened\n" );
    exit(-1);
}
else
    printf( "The file 'rfctdCCC.raw' was opened\n" );

// open the output file for cases and CORE controls
if ((missCore = fopen("C:/RESEARCH/RFCTDATA/ANALYSIS/missCore.lst", "w" )) == NULL ) {
    printf( "The file 'missCore.lst' was not opened\n" );
    exit(-1);
}
else
    printf( "The file 'missCore.lst' was opened\n" );
```

```
/*
    // put in the variable header in
    fprintf(outFile,"studyID,caseNum,caseCtrl,polyCCflag,DXcat,DXDvalid,ageAtRef,ageRefCat,refYearCat,nowKCcat,thenKCcat,phoneCat,");
    fprintf(outFile,"height,weight,lnWeight,logWeight,weicgtCat,BMI,eduYrsCat,eduCat,jobCat,industCat,maritalCat,binMarital,incomeCat,
                raceCat,raceCat3,binRace,");
    fprintf(outFile,"diabCat,ageDiabCat,insulinCat,insulinExpCat,");
    fprintf(outFile,"dyeCat,dyeExpMin,logDyeExpMin,");
    fprintf(outFile,"menarchCat,mensesCat,binMenses,corrBinMens,OCcat,binOCuse,lengthOC,logLenOC,estrogenCat,agePause,agePauseCat,
                natPause,hysterectomyCat,OophectCat,binOophC,ageOophCat,whyPauseCat,");
    fprintf(outFile,"gravid,numMisCar,ectopic,indAbort,stillBirth,liveBirths,multBirth,otherBirth,binGravid,binKids,");
    fprintf(outFile,"BCcat,ageBCDXcat,BScat,ageBScat,purpBScat,procBSBIcat,BSBIprobCat,BSnoBI,");
    fprintf(outFile,"BIcat,BIduration,BIlogDur,BIlnDurn,BIageCat,BIreasonCat,BItypeCat,postBIsurgCat,BIleak\n");
*/
    getIncidenceData();
    cleanRecords();

    fclose(outFile);
    fclose(coreFile);
    fclose(cccFile);

    printRaceDiseaseInfo();

    generateIncidenceTables();

    fclose(errFile);
    fclose(debFile);
    fclose(missCore);
    fclose(bugFile);

    return 0;
}

int getIncidenceData() {
    // Will generate the incidence Data
    int prob = 0;

    prob = readKCpopulationData();
```

```
    prob = readCaseDiseaseData(); //

    return 0;
}

int cleanRecords() {
    // Change this a bit to do options CLEAN, or which files to out put and merge for analysis etc

    initActualCaseCtrls();
    checkCases();
    checkControls();
    checkCoreControls();
    outputActualCaseCtrl();

    return 0;
}


int generateIncidenceTables() {
    //

    calculateIncidenceRates(); // will also print the incidence data calculate race data from the controls
   calculateAgeSpecificRaceIncidenceRates();
    return 0;
}

int checkControls()  {

    int inResult = 0; // result of reading the case raw data file from SPSS/DE
    int cleanFlag = 0; // -1 = abort, 0 = ok
    const int ctrlFlag = 0; // caseFlag = 1
    const int coreFlag = 0; // not a core control
    int numControlsOK = 0;
    int outResult = 0;
    int ctrlLinesRead = 0;

    printf("\nCleaning Controls");
```

```
    caseCtrl = 0;  // set for out put of controls
    polyCCflag = 0;

    if ((inFile = fopen("C:/research/rfctdata/casectrl/ctrldata.asc", "r" )) == NULL ) {
        printf( "The file 'ctrldata.asc' was not opened\n" );
        exit(-1);
    } else
        printf( "The file 'ctrldata.asc' was opened\n" );

    while (inResult == 0) {
        //if(eof(inFile)) if return 0, -1 means abort, 1 is eof
        // get next record
        inResult = 0; // initalize before call
        inResult = getNextDataRecord(numVars, ctrlFlag, inFile);      // have mneumonic names of variables now check each value
        // process it by transforming values and flaging problems
        if (inResult == 0) {
            // ok good record read now process the variables so can use them in data analysis
            ctrlLinesRead++;
            cleanFlag = processCaseControlRecord(ctrlFlag, coreFlag); // ugly but true we are using global vars
            outResult = fprintf(bugFile,"\n%i %i", studyID, caseNum);
            // if no problems then put in data processong file otherwise in problem record file
            if (cleanFlag == 0) {
                outResult = outputRecord(ctrlFlag, coreFlag);
                numControlsOK++;
            }
        }
        else if (inResult != -1)  //i.e. problem other than end-of-file
            printf("\nERROR READING CONTROL FILE NR %i", ctrlLinesRead);
    }
    printf("\nCompleted cleaning Controls Num read = %i NumOutput = %i", ctrlLinesRead,  numControlsOK);
    return 0;
}
```

```
int checkCases() {
   int inResult = 0; // result of reading the case raw data file from SPSS/DE
   int outResult = 0;
   int cleanFlag = 0; // -1 = abort, 0 = ok
   const caseFlag = 1;
   const int coreFlag = 0; // not core control
   int caseLinesRead = 0;

   int numCasesOK = 0;  // keeps track of the number of cases successfully processed

   printf("\nCleaning Cases");

  caseCtrl = 1;    // set the flag so it can be output

   if ((inFile = fopen("C:/research/rfctdata/casectrl/casedata.asc", "r" )) == NULL ) {
      printf( "The file 'casedata.asc' was not opened\n" );
      exit(-1);
   } else
      printf( "The file 'casedata.asc' was opened\n" );

   while (inResult == 0) {
      //!feof(inFile)) if return 0, -1 means abort, 1 is eof
      // get next record
      inResult = 0; // initalize before call
      inResult = getNextDataRecord(numVars, caseFlag, inFile);     // have mneumonic names of variables now check each value
      // process it by transforming values and flaging problems
      if (inResult == 0) {
         // ok good record read now process the variables so can use them in data analysis
         caseLinesRead++;
         cleanFlag = processCaseControlRecord(caseFlag, coreFlag); // ugly but true we are using global vars
         //outResult = fprintf(bugFile,"\n%i %i", studyID, caseNum);
         // if no problems then put in data processong file otherwise in problem record file
         if (cleanFlag == 0) {
            outResult = outputRecord(caseFlag, coreFlag);
            numCasesOK++;
         }
      }
```

```
        else if (inResult != -1)  // i.e. problem other thaaan end-of-file
            fprintf(errFile, "\nERROR problems reading Case record studyID %i CaseNum %i NR %i", studyID, caseNum);
    }
    printf("\nCompleted cleaning Cases Cases read = %i Cases output = %i",   caseLinesRead, numCasesOK);
    return 0;
}

boolean preProcessCoreControlRecord() {
    // some minor differences need to be fixed
    // Some we can fix here, others will vfix in the variable code
    nowKC = 9;    // Don't care


    return True; //ok
}

int checkCoreControls() {
    // vars have same order missing only Q's I added about breast implants directly

    const int ctrlFlag = 0;
    const int coreFlag = 1;

    int inResult = 0;;
    int outResult = 0;
    int numCtrlsOK = 0;
    boolean preProcessNoProb = True;
    int cleanFlag = 1;

    printf("\nCleaning Core controls");

    if ((inFile = fopen("C:/research/rfctdata/analysis/core/coredata.asc", "r" )) == NULL ) {
        printf( "The file 'corectrl.asc' was not opened\n" );
        exit(-1);
    } else
        printf( "The file 'coredata.asc' was opened\n" );

    while (inResult == 0) {
```

```
    //!feof(inFile)) if return 0, -1 means abort, 1 is eof
    // get next record
    inResult = 0; // initalize before call
    inResult = getNextDataRecord(numCOREvars, ctrlFlag, inFile);     // have mneumonic names of variables now check each value
    // process it by transforming values and flaging problems
    if (inResult == 0) {
       // first pre process some odities in the core dataset
       preProcessNoProb = preProcessCoreControlRecord();
       // ok good record read now process the variables so can use them in data analysis
       if (preProcessNoProb == True)
          cleanFlag = processCaseControlRecord(caseFlag, coreFlag); // ugly but true we are using global vars
       // process additional stuff for this data set

    }
    // if no problems then put in data processong file otherwise in problem record file
    if (cleanFlag == 0) {
       outResult = outputRecord(caseFlag, coreFlag);
       numCtrlsOK++;
    } else
       printf("\nBad RECORD studyID %i", studyID);
  }
  printf("\nCompleted cleaning CORE controls Num controls output = %i", numCtrlsOK);

  return 0;
}


int getNextDataRecord(int ntoks, int caseCtrlFlag, FILE *filein ) {
  int NR = 0; // number of tokens read
  // Janets ctrls have some extra tokens
  char inString[4096];
  char saveString[4096];
  char seps[] = ",";
  char *token;
  char tokenCopy[100]; // long because some are comments though we coded them - check w/ Milda

  // use variable list in clean.h
```

```
    // printf( "%s\n\nTokens:\n", string );
    /* Establish string and get the first token: */
    fgets(inString,4097, filein);
    numLinesRead++;
    strcpy(saveString, inString); // save a copy for debug
    if (feof(filein)) {
        // make sure we read and process all records
        printf("We have reached end-of-file\n");
        fclose(filein); /* close the file */
       return -1;
    }
    token = strtok(inString, seps );
    NR++;
    while (token != NULL) { // check of end of file
        /* While there are tokens in "string" convert the token to integer or whatever and save into mneumonic variables */
        strcpy(tokenCopy, token);
        convertToken(NR, caseCtrlFlag, tokenCopy);
       /* Get next token: */
        token = strtok( NULL, seps );
        if (token != NULL)
            NR++; // count after know not at end of line i.e. last token read is not valid because eol not set until last read
    }
    /* OK should have read the right number of tokens if not flag an error */
    if (NR != ntoks) {
        fprintf(errFile, "\n ERROR wrong number of tokens read %i should be %i Num Lines = %i", NR, ntoks, numLinesRead);
        fprintf(errFile, "\n %s", saveString);
        return problem; // set flag to not use record
    } else
        return noProblem; //no problems = 0 for this case record
}


int convertToken(int NR, int caseCtrlFlag, char tokenCopy[100] ) { // add core ctrl flag
    // Will use a big case statment to toss tokens from the proper variable will handle the flaky case/ ctrl(s) fifferent field number
    // NR is number read
    // additional cases > 117 are for the CORE control Group
    int result = 0;
```

```
switch (NR) {
  // long but good just do conversion, process case record will do range check and transformation
  //                                          SPSS/DE var name   SPSS codeing
  case  1: { ageAtRef = atoi(tokenCopy); } break;          // AGE@REF  age at reference date (DX date for cases, randomly assigned for ctrls)
  case  2: { ageDiabDX = atoi(tokenCopy); } break;         // AGEDX   age at DX of diabetes
  case  3: { ageOophect = atoi(tokenCopy); } break;        // AGEOVECT    AGE AT 2ND OOPHECTEOMY IF > 1
  case  4: { agePause = atoi(tokenCopy); } break;          // AGEPAUSEAGE AT MEMOPAUSE
  case  5: { age1stBsurg = atoi(tokenCopy); } break;       // AGESURG      age first of any breast surgery
  case  6: { anyBsurg = atoi(tokenCopy); } break;          // BSURGERY    cat any breast surgery
  case  7: { BcancerDX = atoi(tokenCopy);} break;       // CANCERever had breast cancer
  case  8: { caseNum = atoi(tokenCopy);} break;            // CASE#
  case  9: case 10: case 11: case 12: case 13:{ } break; // COLOR* not using HAIR DYE COLOR desired
  case 14: { caseFlag = atoi(tokenCopy); } break;          // CONTROL 0 = Control 1 = CASE
  case 15: { strcpy(BCyearDX, tokenCopy); } break;         // DATEC   date of first breast cancer DX
  case 16: case 17: case 18:  {} break;                 // DATESPP,2,3 date of other surgery related to BI's
  case 19: { strcpy(dateBsurg, tokenCopy); } break;        // DATESURG date of 1st breast surgery of any kind
  case 20: { strcpy(dateCTDSX, tokenCopy); } break;    // DATESX date of 1st SX of CTDs
  case 21: { diabDX = atoi(tokenCopy); } break;         // DIABETES
  case 22: case 23: case 24: {} break;                     // DUR1,2,3    job* duration
  case 25: { ICDcodeDX = (float)atof(tokenCopy); } break;  // ICD-9 code
  case 26: { ICDcodeDX2 = (float)atof(tokenCopy); } break;    // ICD-9 second code if multiple CTDs
  case 27: case 28: case 29: case 30:{} break;             // dyeBrand 2-5 is ist in matrix but output last by brain damageg SPSS/DE
  case 31: {} break;                                       // DYEBRAND 1
  case 32: case 33: case 34: case 35: case 36:
     { dyeUseMin[NR - 32] = atoi(tokenCopy);               // dyeExposure in minutes 1 - 5 ok no name alphabetic prob as below
     } break;
  case 37: case  38: case 39: case 40:
     {  dyeFirst[NR - 36] = atoi(tokenCopy);               // hair dye age first used dyefirst is listed last so fix dyefirs2 -> dyeFirst[2]
     } break;
  case 41: { dyeFirst[0] = atoi(tokenCopy); } break;    // output after the other first 4 entrys
  case  42: case 43: case 44: case 45: case 46:
     { dyeLast[NR - 42] = atoi(tokenCopy);                 // hair dye age last used
     } break;
  case  47: { hairDyeUse = atoi(tokenCopy); } break;       // DYES dye used 0 = NO, 1 = YES
  case  48: { ectopic = atoi(tokenCopy); } break;       // ECTOPIC pregnancy cat
  case  49: { educatn = atoi(tokenCopy); } break;          // EDUCAT higest level of education years
  case  50: case 51: case 52: { } break;                // END1,2,3 end date of job
```

```
case 53: { estrogenUse = atoi(tokenCopy); } break;    // ESTROGEN
case 54: { height = atoi(tokenCopy); } break;           // HEIGHT in inches
case 55: { implantUse = atoi(tokenCopy); } break;       // IMPLANT
case 56: { reasonBI = atoi(tokenCopy); } break;         // IMREASON
case 57: { otherTypeBI = atoi(tokenCopy); } break;      // IMSPP
case 58: { implantType = atoi(tokenCopy); } break;      // IMTYPE
case 59: { income = atoi(tokenCopy); } break;          // INCOME
case 60: { indAbort = atoi(tokenCopy); } break;         // INDABORT
case 61: case 62: case 63: { } break;                  // INDUST1,2,3   NOT USED
case 64: { indusRef = atoi(tokenCopy); } break;         // industry at ref date
case 65: case 66: case 67: { } break;                  // JOB1,2,3 other main job codes NOT USED at present
case 68: { jobRef = atoi(tokenCopy); } break;           // JOBREF job at reference date 0 = housewife, 1 = outside
case 69: { nowKC = atoi(tokenCopy); } break;           // KCNOW
case 70: { thenKC = atoi(tokenCopy); } break;           // KCTHEN
case 71: { lengthOC = atoi(tokenCopy);} break;          // LENGTHOC length of oral contraceptive before ref date in months
case 72: { liveBirths = atoi(tokenCopy); } break;       // LIVEONE
case 73: { marital = atoi(tokenCopy);} break;           // MARITAL status at reference date
case 74: { ageMenarche = atoi(tokenCopy); } break;      // MENARCHE
case 75: { menses = atoi(tokenCopy); } break;           // MENSES
case 76: { numMisCar = atoi(tokenCopy); } break;        // MISCARRY
case 77: { multBirth = atoi(tokenCopy); } break;        // MULTI
case 78: { natPause = atoi(tokenCopy); } break;         // NATPAUSE
case 79: case 80: case 81: case 82: case 83:
    { numDyePerYr[NR - 79] = atoi(tokenCopy);     // NUMDYE
    } break;
case 84: { OCuse = atoi(tokenCopy); } break;           // OCUSE
case 85: { otherBirth = atoi(tokenCopy); } break;      // OTHER
case 86: { otherEdu = atoi(tokenCopy); } break;         // OTHEREDU
case 87: { otherBSproc = atoi(tokenCopy); } break;     // OTHERPRO any other procedure for breast surgery
case 88: { otherBSpurp = atoi(tokenCopy); } break;     // OTHERPURP any other
case 89: { otherMenpause = atoi(tokenCopy); } break; // OTHERWHY other reason for menapause to stop not listed
case 90: { oophectomy = atoi(tokenCopy); } break;      // OVECT
case 91: { phone = atoi(tokenCopy); } break;           // PHONE
case 92: { strcpy(dateFirstBI, tokenCopy); } break;    // PLACED date ist breast implant placed
case 93: { problemsBSBI = atoi(tokenCopy); } break;    // PROBLEMS problems with BI
case 94: { procedureBSBI = atoi(tokenCopy); } break; // PROCEDUR   how breast size or shape modified
case 95: { BSpurpose = atoi(tokenCopy); } break;       // PURPOSE reason for ist breast implant surgery
```

```
    case  96: { race = atoi(tokenCopy); } break;                  // RACE
    case  97: { strcpy(refDate,tokenCopy); } break;                  // REFDATE
    case  98: { specBSprob = atoi(tokenCopy); } break;               // Complications of breast modifications
    case  99: { raceOther = atoi(tokenCopy); } break;             // SPPRACE
    case 100: { otherBIreason = atoi(tokenCopy); } break;            // SPPREAS other reasons to get BI's other than listed
    case 101: case 102: case 103: { } break;                      // START1,2,3 job start dates
    case 104: { stillBirth = atoi(tokenCopy); } break;               // STILL
    case 105: { studyID = atoi(tokenCopy); } break;                  // STUDYID
    case 106: { otherBIsurg = atoi(tokenCopy); } break;              // SURGELSE   CAT
    case 107: { surgPurBI2 = atoi(tokenCopy); } break;               // SURGPUR2
    case 108: { surgPurBI3 = atoi(tokenCopy); } break;               // SURGPUR3
    case 109: { surgPurBI1 = atoi(tokenCopy); } break;               // SURPURP
    case 110: { initCTDSX = atoi(tokenCopy); } break;                // SXSPP
    case 111: { timePill = atoi(tokenCopy); } break;                 // TIMEPILL
    case 112: { timeShot = atoi(tokenCopy); } break;                 // TIMESHOT
    case 113: { diabRX = atoi(tokenCopy); } break;                   // TX    treatment for diabetes
    case 114: { weight = atoi(tokenCopy); } break;                // Weight
    case 115: { whatOtherBirthOut = atoi(tokenCopy); } break;     // WHAT
    case 116: { whyPause = atoi(tokenCopy);} break;                  // WHYPAUSE  reason other than listed for menapause
    case 117: { eduYrsComp = atoi(tokenCopy); } break;               // YRCOMP
    // next cases are are for the core controls should check for core controls
    case 118: { coreAgeBCA = atoi(tokenCopy);} break;
    case 119: { coreREFMONTH = atoi(tokenCopy); } break;
    case 120: { coreREFYEAR = atoi(tokenCopy);} break;
    case 121: {coreBIRTHMM = atoi(tokenCopy);} break;
    case 122: {coreBIRTHYR = atoi(tokenCopy);} break;
    case 123: {coreHTFT = atoi(tokenCopy);} break;
    case 124: {coreHTIN = atoi(tokenCopy);} break;
    case 125: {coreGravid = atoi(tokenCopy); } break;
    case 126: {coreSNHYST = atoi(tokenCopy);} break;
    case 127: { coreHRMATREF = atoi(tokenCopy);  } break;
    default: printf("\n MAJOR problem with convert token NR=", NR);
  }
  return 0;
}
```

```
int processCaseControlRecord(int caseCtrlFlag, int coreFlag) {
   // will process each variable in record to make sure is in valid range and then transform it if need be
   // use common set of error messages in checking routine if there is a problen will not output record
   const int Case = 1;
   int chkAge= 0;    // check reference age is same as reference date (DXDIDXP) - birth date
   int bprobs = 0;
   int caseDiseaseDataIndx;

   // RefDate generate refYearCat
   refYearCat = calculateRefYearCat(refDate, coreFlag, studyID, caseNum);  // will also assign refMonth and refYear actual year not CATs

   // CHECK age at REF is corectly encoded
   if (caseCtrlFlag == 1) { // only for cases
      if ( -1 == (caseDiseaseDataIndx = findCaseIndex(studyID)))
         fprintf(debFile, "\nERROR!!!!!!! CASE ID %i returned Questionneer but NOT disease database", studyID); // indicates that the case i
      else {
         chkAge = checkAge(ageAtRef, refMonth, refYear, caseInfo[caseDiseaseDataIndx].cBirthYr,
                           caseInfo[caseDiseaseDataIndx].cBirthMonth, studyID, caseNum);
         if (chkAge != -1 && chkAge != ageAtRef) {
            fprintf(debFile, "\nERROR ageAtRef %i != %i CalculatedAge@DX (refdate %i/%i - BirthDateref %i/%i) studyID %i CaseNum %i", ageAtRef, chkAge,
                  refMonth, refYear, caseInfo[caseDiseaseDataIndx].cBirthMonth, caseInfo[caseDiseaseDataIndx].cBirthYr, studyID, caseNum);
            ageAtRef = chkAge;    // CORRECT the AGE AT REFERENCE
         }
         // CHECK if VALID DXD i.e. if has DXD and not DXP will analize both ways DXD + DXP and just DXD
         DXDvalid = caseInfo[caseDiseaseDataIndx].validDXD;
      }
   }
   else
      DXDvalid = 1; // for controls

   // age@REF
   ageRefCat = calculateAgeCatAtReferenceDate(ageAtRef, studyID, caseNum);    // calculate ageCat (age at reference date) will do for a number of vars

   // Derive Birth Year = refDate - ageAtRef; // only need to do for CORE controls and RDD controls
   if (coreFlag == 1)
      birthYear = coreBIRTHYR; // don't need to calculate for core ctrls
   else if (caseCtrlFlag != Case) // Dont need to calculate for cases get from Chart Abstraction Form
```

```
    birthYear = (refYear - ageAtRef); // this assumes 2 digit year

// check DX codes
if (caseCtrlFlag == Case) // set DX codes
    DXcat = ProcessDXcodes(ICDcodeDX, ICDcodeDX2, studyID, caseNum);
else {// for controls no DX codes
    DXcat = -1;
    DX2cat = -1;
}

// Diabetes related variables
diabCat = checkDiabetesVars(studyID, caseNum);  // set other vars globally

// weight and height check; convert to BMI int
BMI = calculateBMI(weight, height, studyID, caseNum);

// check phone and King County
phoneCat = checkBinaryCat(phone, "Phone\0", studyID, caseNum);
if (phoneCat == 0)
    fprintf(errFile, "\nCHECK phone CAT = %i studyID %i caseNum %i", phoneCat, studyID, caseNum);
nowKCcat = checkBinaryCat(nowKC, "nowKC\0", studyID, caseNum);
thenKCcat =   checkBinaryCat(thenKC, "thenKC\0", studyID, caseNum);
if (thenKCcat == 0)
    fprintf(errFile,"\nERROR: Not in KC studyID %i caseNum %i", studyID, caseNum);

// reproductive history
menarchCat = processMenarch(ageMenarche, studyID, caseNum); //
gravid = processPregnancyHx(coreFlag, studyID, caseNum);    // will also process parity, abortions, etc;
oophectCat = processOveries(oophectomy, studyID, caseNum);  // This procedure must proceed process menses
mensesCat = processMenses(menses, coreFlag, studyID, caseNum);
estrogenCat = checkBinaryCat(estrogenUse, "Estrogen\n", studyID, caseNum);
OCcat = processOCuse(OCuse, coreFlag, studyID, caseNum);
crossCheckMenstrualVars(studyID, caseNum);

// Breast related
BCcat = processBreastCancer(BcancerDX, BCyearDX, birthYear, coreFlag, studyID, caseNum);
BScat = processBreastSurgery(anyBsurg, coreFlag, studyID, caseNum);
```

```
    BIcat = processBreastImplants(implantUse, studyID, caseNum);
    if (coreFlag == 1)
       processCOREbreastSurgeryToImplantVars(); // generate BI vars
    bprobs = crossCheckBreastCats(studyID, caseNum);   // check to make sure the implant related info is consistent
    // set var for Breast Surgery for other than Implants
    if (BScat == 1 && BIcat == 0)
       BSnoBI = 1;
    else if (BScat == -1 || BIcat == -1)
       BSnoBI = -1;
    else
       BSnoBI = 0;   // BI+ women case/ctrl = 0 not missing so can put the other in the model as well

    // hairdye if Hair dye used will return exposure in minutes
    dyeCat = checkBinaryCat(hairDyeUse, "HairDyeUse\0", studyID, caseNum);
    dyeExpMin = calculateHairDyeTime(dyeCat, studyID, caseNum); // also calculates a logDyeExpMin = ln(dyeExpMin) cat

    // personal demographics
    maritalCat = ProcessMaritalStatus(marital, studyID, caseNum);
    jobCat = ProcessWorkHistory(jobRef, studyID, caseNum);    // Job just cat of 0 = HouseWife 1 = outside of home work, refine later
    raceCat = checkRace(race, raceOther, caseDiseaseDataIndx, caseCtrlFlag, coreFlag, studyID, caseNum);
    eduCat = checkEducation(educatn, otherEdu, eduYrsComp, coreFlag, studyID, caseNum);   // two eduCat is level, yearsEdu is Ordered
    incomeCat = checkIncome(income, studyID, caseNum);

    // Update the various tables for disease matrix, all disease matrix, control matrix
    UpdateTables(ageRefCat, refYearCat, caseCtrlFlag, coreFlag, caseDiseaseDataIndx, studyID, caseNum);

    return recordProblem;
}


int checkBinaryCat(int binValue, char varName[15], int studyID, int caseNum) {
    // transform and check YES NO value
    // will return -1 if error, else binValue -1
    // input 2=YES 1=NO 9=DK or Missing

    binValue --;
    if (binValue == 0 || binValue == 1)
```

```
        return binValue;  // 0 = NO, 1 = YES
    else if (binValue == 8)
        return -1; // missing
    else {
        fprintf(errFile, "\n ERROR %s %i not valid code studyID %i, case# %i", (binValue +1), va rName, studyID, caseNum);
        return -1;    // illegal value
    }
}

int calculateWideAgeCategory(int age, char VarName[15], int studyID, int caseNum) {
    // Will calculate wide age categories 0=18-29 1=30-39 2=40-49 3=50-59 4=60-69 5=70+
    int wideAgeCat = -1;

    if (age > 0 && age < 30)
        wideAgeCat = 0;
    else if (age >= 30 && age < 40)
        wideAgeCat = 1;
    else if (age >= 40 && age < 50)
        wideAgeCat = 2;
    else if (age >= 50 && age < 60)
        wideAgeCat = 3;
    else if (age >= 60 && age < 70)
        wideAgeCat = 4;
    else if (age >= 70 && age < 75)
        wideAgeCat = 5;
    else {
        fprintf(errFile, "\nERROR age %i for %s out of WIDE age study limits studyID %i CaseNum %i", age, VarName, studyID, caseNum);
    }
    return wideAgeCat;
}

int calculateAgeCategory (int age, char VarName[15], int studyID, int caseNum) {
    // will take an age and convert it into a age category
    int ageCat = -1;

    /*if (age > 0 && age < 20) // check age for study seperatly since this is used for other var age cats
        ageCat = 0;*/
```

```
    if (age > 0 && age < 25)
        ageCat = 0;
    else if (age >= 25 && age < 30)
        ageCat = 1;
    else if (age >= 30 && age < 35)
        ageCat = 2;
    else if (age >= 35 && age < 40)
        ageCat = 3;
    else if (age >= 40 && age < 45)
        ageCat = 4;
    else if (age >= 45 && age < 50)
        ageCat = 5;
    else if (age >= 50 && age < 55)
        ageCat = 6;
    else if (age >= 55 && age < 60)
        ageCat = 7;
    else if (age >= 60 && age < 65)
        ageCat = 8;
    else if (age >= 65 && age < 70)
        ageCat = 9;
    else if (age >= 70 && age < 75)
        ageCat = 10;
    else {
        fprintf(errFile, "\nERROR age %i for %s out of study limits studyID %i CaseNum %i", age, VarName, studyID, caseNum);
    }
    return ageCat;
}


int calculateAgeFromDateString(char date[6], int birthYr, char varName[15], int studyID, int caseNum) {
    // Returns the calculated age in years from birth year
    int age, result, month, year;

    result = sscanf(date, "%d/%d", &month, &year); // pull month year out of date string
    if (birthYr > 1900)
        birthYr = birthYr - 1900;
```

```
    age = year - birthYr;

    if (age <= 0) {
        fprintf(errFile, "\nERROR age %i for %s out of study limits studyID %i CaseNum %i", age, varName, studyID, caseNum);
        return -1;
    } else
        return age;
}


int calculateAgeCatFromYearDateString(char date[6], int birthYr, char varName[15], int studyID, int caseNum) {
    // will take a date string and calculate the age category given the string and the birthyear

    int ageCat = 0;
    int age;

    age = calculateAgeFromDateString(date, birthYr, varName, studyID, caseNum); // first have to convert date from string to Month, Year
    if (age < 0)
        return -1;
    else
        ageCat = calculateAgeCategory(age, varName, studyID, caseNum);
    return ageCat;
}


 int checkAge(int ageAtRef, int refMonth, int refYear, int cBirYr, int cBirMonth, int studyID, int caseNum) {
    // Will check if age at reference is same as ref date - birthYear
    // can only do for cases don't have birth year for controls it is calculated so all controls will be ok
    int calculatedAgeAtRef = -1;
    if ((cBirYr < 10 ||  cBirYr > 74) || cBirYr == 99) {
        fprintf(debFile, "\nERROR - checkAge Proc BirthYear %i out of Study Bounds studyID %i caseNum %i", cBirYr, studyID, caseNum);
        birthYear = (refYear - ageAtRef);
    } else { // ok good birth year check age
        if (cBirYr != 99)
            birthYear = cBirYr;  // get off of the Chart abs. form
        else
            cBirYr = birthYear = (refYear - ageAtRef); // missing from CAF
```

```
        calculatedAgeAtRef = refYear - cBirYr;
        if (refMonth < cBirMonth)
            calculatedAgeAtRef -= 1;
        if (ageAtRef != calculatedAgeAtRef)
            fprintf(errFile, "\nERROR CHECK AGE age@REF %i != Calculated AGE AT REF %i studyID %i caseNum %i",
                    ageAtRef,calculatedAgeAtRef, studyID, caseNum);
    }
    return calculatedAgeAtRef;
}

int calculateAgeCatAtReferenceDate(int age, int studyID, int caseNum) {
    // check that  18 <= age at ref >= 74
    int refCat = 0;

    if (age >=18 && age <=74) {
        refCat = calculateAgeCategory(age, "ageAtRef\0", studyID, caseNum);
        ageCat6 = calculateWideAgeCategory(age, "ageWideCatAtRef\0", studyID, caseNum); // side effect
        return refCat;
    }
    else {
        fprintf(errFile, "\nERROR Age %i out of study limits studyID %i CaseNum %i", age, studyID, caseNum);
        refCat = -1;
        return refCat;
    }
}

int calculateRefYearCat(char refDate[], int coreFlag, int studyID, int caseNum) {
    // will calculate study reference year
    int yearCat = -1;
    int result = 0;

    // first have to cobvert date from string to int refMonth, int refYear
    if (coreFlag ==1) { // CORE controls don't have string refdate
        refMonth = coreREFMONTH;
        refYear = coreREFYEAR;
    }
    else result = sscanf(refDate, "%d/%d", &refMonth, &refYear);
```

```
    yearCat = refYear - 83;

    if (yearCat < 0 || yearCat > 8) {
        fprintf(errFile, "\nERROR refYear %i out of study limits core %i studyID %i CaseNum %i", refYear, coreFlag, studyID, caseNum);
        //recordProblem = True; // will not print out to analysis file
        refYear3 = -1; // sideeffect
        return -1;
    } else {
        if (yearCat >= 0 && yearCat <= 2)
            refYear3 = 0;
        else if (yearCat >= 3 && yearCat <= 5)
            refYear3 = 1;
        else if (yearCat >= 6 && yearCat <= 8)
            refYear3 = 2;

        return yearCat;
    }
}

int ICDtoInteger(float ICDcode, int studyID, int caseNum) {
    if (ICDcode == (float)710.000)
        return 0;
    else if (ICDcode == (float)710.100)
        return 1;
    else if (ICDcode == (float)710.200)
        return 2;
    else if (ICDcode == (float)710.300 || ICDcode == (float)710.400)
        return 3;
    else if (ICDcode == (float)710.900)
        return 4;
    else {
        fprintf(errFile, "\nERROR illegal ICD code %f.1 studyID %i caseNum %i", ICDcode, studyID, caseNum);
        return -1;
    }
}
```

```
int ProcessDXcodes(float ICDcodeDX, float ICDcodeDX2, int studyID, int caseNum) {
   // will turn in float 710.x codes into int x code 0 -> 4 range
   // Will also polychotomous logistic regression case control flag

   int intDXcode;

   if (ICDcodeDX < (float)710.000 || ICDcodeDX > (float)710.900) {
      fprintf(errFile, "\nERROR illegal ICD-9 codeDX %i studyID %i caseNum %i", ICDcodeDX, studyID, caseNum);
      return -1;
   } else {
      intDXcode = ICDtoInteger(ICDcodeDX, studyID, caseNum);
      polyCCflag = intDXcode + 1; // set polychotomous case control flag
   }

   if (ICDcodeDX2 > 710.9)
      DX2cat = -1;
   else {
      if (ICDcodeDX2 < (float)710.000 || (float)ICDcodeDX2 > 710.900) {
         fprintf(errFile, "\nERROR illegal ICD-9 codeDX2 %i studyID %i caseNum %i", ICDcodeDX2, studyID, caseNum);
         DX2cat = -1;
         return intDXcode;
      } else
         DX2cat = ICDtoInteger(ICDcodeDX2, studyID, caseNum);    // side-effect
   }

   return intDXcode;
}


int calculateBMI(int Weight, int Height, int studyID, int caseNum) {
   // weight and height check; convert to BMI int
   double tmpBMI;

   if ((Height < 48 || Height > 77) || (Weight < 75 || Weight > 300)) {
      fprintf(errFile, "\nPosssible Error height %i  or weight %i out of range studyID %i CaseNum %i", Height, Weight, studyID, caseNum);
      BMI = -1;
```

```
        if (Height == 99 || Height == 999)
            height = -1;// side effect set for output record
        if (Weight == 999)
            weight = logWeight = lnWeight = weightCat = -1; // side effect set for output record
        return -1;
    }
    else {
        // calculate BMI
        logWeight = (int) log10(weight);
        lnWeight = (int) log(weight);

        if (weight <= 110)    // set up weight cat
            weightCat = 0;
        else if (weight <= 125)
            weightCat = 1;
        else if (weight <= 135)
            weightCat = 2;
        else if (weight <= 150)
            weightCat = 3;
        else if (weight <= 175)
            weightCat = 4;
        else
            weightCat = 5;

      tmpBMI = (weight / 2.204622622) / pow(((height * 2.54) / 100),2); // should be in kg & m
        BMI = (int) tmpBMI;
        if (BMI  < 12 || BMI > 55)
            fprintf(errFile, "\nPosssible Error BMI %f.2 out of range height %i weight %i studyID %i c aseNum %i", BMI, height, weight, studyID, caseNum);
        return BMI;
    }
}


int calculateHairDyeTime(int hairDyeCat, int studyID, int caseNum) {
    // comes up with minutes of hair dye

    int i, numYrs = 0;
```

```
int cumDyeExpMin;

logDyeExpMin = cumDyeExpMin = 0; //  init

if (hairDyeCat == 0 || hairDyeCat == -1){
   // check for descreptency between hair dye use flag flag and first row of matrix vars -1 is blank
   if (dyeUseMin[1] != -1 || numDyePerYr[1] != -1){
      fprintf(errFile,"\nError Hair dye use (DYES) is 0 or DK, but NON-ZERO matrix valves dyeExpo %i numDyePerYr %i studyID %i CaseNum %i",
            (hairDyeUse + 1), studyID, caseNum);
      logDyeExpMin = -1;
      return -1;   // problem
   }
}

if (hairDyeCat == 0)
      return 0;// 0 exposure flag and record match
else if (hairDyeCat == -1) {
   logDyeExpMin = -1; // missing so set to reflect missing on associated variables
   return -1;
} else { // have a positive hair dye use
   for (i = 0; i < 5; i++) {
      if (dyeUseMin[i] != -1 && numDyePerYr[i] != -1 && dyeFirst[i] != -1 && dyeLast[i] != -1 &&
          dyeUseMin[i] != 99 && numDyePerYr[i] != 99 && dyeFirst[i] != 99 && dyeLast[i] != 99) {
         // calculate hair dye exposure
         numYrs = (dyeLast[i] - dyeFirst[i]) + 1;
         cumDyeExpMin += numYrs * numDyePerYr[i] * dyeUseMin[i];
      }
   }
   // convert to log base 10 minutes
   logDyeExpMin = (int)(log10(cumDyeExpMin)); // side-effect
   if (logDyeExpMin == 0)
      logDyeExpMin = 1;  // ofter they used it but lot of unknown values so we assign this exposure to
                         // lowest exposure category so will not  be same category as women that never use hair dye
   return cumDyeExpMin;
}
}
```

```
int checkDiabetesVars(int studyID, int caseNum) {
   // if diabetes DX'ed <= refdate set other related vars

   ageDiabCat = -1;    // initalize
   insulinExpCat = 0;

   diabCat = checkBinaryCat(diabDX, "DiabDX\0", studyID, caseNum);
   if (diabCat == 0 || diabCat == -1) {
      insulinCat = 0; // not use insulin and not diabetic
      return diabCat;   // no dx of diabetes or DK or Missing assume no assign assoc vars 0 so set other relates diab vals to 0
   }
   else if (ageDiabDX > ageAtRef) {
      // diabetic, so check if age of diabetes DX <= age at refdate, if so assign associated variables
      diabCat = 0;
      insulinCat = 0;
      return diabCat;
   } else { // age diabDX <= refDate
      ageDiabCat = calculateAgeCategory(ageDiabDX, "ageDiabDX\0", studyID, caseNum);
      if (diabRX == 4  || diabRX == 5) { // shots or shots and pills
         insulinCat = 1; // insulin use want silicone exposure
         if (timeShot != 999)
            insulinExpCat = (int)log(timeShot);
         else
            insulinExpCat = 1; // some insulin exposure but can not quantify but > 0

         fprintf(errFile, "\nInsulin use diabetic studyID %i caseNum %i age %i ageDiabCat %i insulinCat %i insulinExpCat %i",
               studyID, caseNum, ageDiabDX, ageDiabCat, insulinCat, insulinExpCat);
      } else
         insulinCat = 0;
   }
   return diabCat;
}


int processMenarch(int firstPeriod, int studyID, int caseNum) {
   // How were you when you had first menstrual period?
   int mCat;
```

```
    mCat = firstPeriod; // - 9; // youngest was 9 FOR NOW JUST RETURN NUMBER ENTERED
    if (mCat == 99)
        mCat = -1;
    if ((mCat < 8 || mCat > 20) && mCat != -1) {
        fprintf(errFile, "\n CHECK age Menarche %i possible out of range, studyID %i, caseNum %i", firstPeriod, studyID, caseNum);
    }
    return mCat;
}


int processPregnancyHx(int coreFlag,int studyID, int caseNum) {
    // will also process parity, abortions, etc;
    // will generate Gravid
    int gravid = 0;
    // assume that the numbers are correct, just check for < 0 and > 10 of any one category of birth outcomes

    if (numMisCar == 99)
        numMisCar = 0;
    else if (numMisCar > 0)
        gravid += numMisCar;

    if (ectopic == 99)
        ectopic = 0;
    else if (ectopic > 0 )
        gravid += ectopic;

    if (indAbort == 99)
        indAbort = 0;
    else if (indAbort > 0)
        gravid += indAbort;

    if (stillBirth == 99)
        stillBirth = 0;
    else if (stillBirth > 0 )
        gravid += stillBirth;
```

```
    if (liveBirths == 99)
        liveBirths = 0;
    else if (liveBirths > 0)
        gravid += liveBirths;

    if (multBirth == 99)
        multBirth = 0;
    else if (multBirth > 0 )
        gravid += multBirth;

    if (otherBirth == 9)
        otherBirth = 0;
    else if (otherBirth > 0) {
        gravid += otherBirth;
        if (coreFlag == 1)
            fprintf(missCore, "\n OtherBirth CORE studyID %i", studyID);
    }

    if (gravid > 12)
        fprintf(errFile, "\n CHECK Gravid %i > 12 pregnancies, StudyID %i, case# %i", gravid, studyID, caseNum);

    if (gravid > 0)
        binGravid = 1;
    else
        binGravid = 0;

    if (liveBirths > 0)
        binKids = 1;
    else
        binKids = 0;

    if (coreFlag == 1 && gravid != coreGravid)
        fprintf(errFile, "\nCOREctrl ERROR gravid %i != coreGravid %i, StudyID %i, case# %i", gravid, coreGravid, studyID, caseNum);

    return gravid;
}
```

```
int processMenses(int menses, int coreFlag, int studyID, int caseNum) {
    // menstrual period at refDate
    // defaulty values
    hysterectomyCat = 0;
    agePauseCat = -1;
    agePauseCat = natPauseCat = -1; // initalize
    whyPauseCat = -1;
    corrBinMens = 1; // stopped == 1
    binMenses = 1;


    menses--;  // 0=Having reg per, 1=begun to stop, 2=stop perm, 3=preglnursing, 7->5=never started
    if (menses == 0 || menses == 3) {
        // still having periods
        if (ageAtRef <= 55) {
            agePause = -1; //set to missing since still having periods
            corrBinMens = binMenses = 0;
            whyPauseCat = 0; // 0=periods still 1=stopnat 2=hyster 3=radationRX 4=hormoneRX 5 = other
        } else { // age at ref is > 55, very likely to had had menapause, OCs could still induce monthly bleeding stopped is default
            if (agePause == 99)
                agePause = -1;
            if (whyPause == 0)
                whyPauseCat = 1; // set to natural menapause Stopped
            else
                whyPauseCat = whyPause;
            if (whyPause == 9)
                whyPauseCat = -1;
            fprintf(errFile, "\n CHECK! ageAtRef > 55 and menses = 0, highly unlikely!, studyID: %i, caseNum %i", studyID, caseNum);
        }
    } else if (menses == 7) { // never started
        agePause = -1;
        menses = 4; // recode never started
        binMenses = 1;
        whyPauseCat = 5; // start with other reason and assign specific reason if exists i.e. never start will be included in other
    } else if (menses == 2 || menses == 1) {
        // either begun to stop or stop permenatly check if stop date <= refdate
```

```
if (agePause <= ageAtRef && agePause > 12 )
   agePauseCat = calculateAgeCategory(agePause, "agePause\0", studyID, caseNum); // get age category for menapause
else agePauseCat = -1; // missing so check age at ref and other vars

if (agePause < 12 || agePause == -1 || agePause == 99)
   agePause = -1;

natPauseCat = checkBinaryCat(natPause, "NatPause\0", studyID, caseNum);

if (ageAtRef > 55 || agePauseCat != -1) { // missing age at Pause
   binMenses = 1;
   if (natPauseCat == 1)  // yes stopped naturally
      whyPauseCat = 1;       // reflect it here as well
   else if (natPauseCat == 0) { // not stop naturally
      if (whyPause == 2 ) //&& agePause <= ageAtRef)
         hysterectomyCat = 1;
      whyPauseCat = whyPause;  // copy other reasons
      if (whyPauseCat == 9)
         whyPauseCat = -1;  // somehow slipped through
      if (whyPauseCat == 8) { // other reason
         if (otherMenpause ==16)
            menses = 4; // never started
         if (otherMenpause == 17 || otherMenpause == 25 || binOoph == 1)
            whyPauseCat = 6; // oophectomy
         whyPauseCat = 5; //otherMenpause reason;
         if (coreFlag == 1)
            fprintf(missCore, "\n OtherPauseWhy CORE studyID %i", studyID);
      } else if (whyPause == 9 && menses == 8)
         whyPauseCat = -1;  // missing
   }
} else { // age pause > ref date so still having periods at ref date or missing age at period stop date
   if (natPauseCat != 1) {
      menses = 0;
      whyPauseCat = 0;
      natPauseCat = 0;
      binMenses = 0;
      corrBinMens = 0;
```

```
        }
      }
   } else if (menses == 8) { // MISSING OR SOMEONE FILLED in box but not check menses status
            agePause = -1; // missing
            menses = -1;
            binMenses = -1;
            corrBinMens = -1;
   } else
      fprintf(errFile, "\n ERROR Menses illegal value %i, StudyID %i, case# %i", (menses + 1), studyID, caseNum);

   if (whyPauseCat == -1 && binOoph == 1)
      whyPauseCat = 6;

   if (agePause == 99)
      agePause = -1;

   return menses;
}

int processOveries(int oophectomy, int studyID, int caseNum) {
   // before reference date and one or more overies been removed
   // 0=NO, 1=YES-one, 2=YES-two, 3=YES-1or2, 4=Yes-Partial, 5 = Don't know
   // recodeOph NO = 0, 1, 4  YES = 2, MISS = 3, 5

   ageOophCat = -1;

   oophectomy--;

   if (oophectomy == 8)
      oophectomy = -1; // donk know -> missing


   if (oophectomy >= 1 && oophectomy <=4) {
      if (ageOophect <= ageAtRef  && ageOophect > 0 && ageOophect !=99)
         ageOophCat = calculateAgeCategory(ageOophect, "ageOophect\n", studyID, caseNum); // side effect
      else {
         oophectomy = 0; // age at oophectomy > refdate
```

```
        fprintf(errFile, "\n NOTE: Age at Oophectomy ageOvect %i > ageatRef %i, StudyID %i, case# %i", ageOophect, ageAtRef, studyID, caseNum);
      }
   } else if (oophectomy == 98)
      oophectomy = -1;    // missing
   else if (oophectomy != 0){ // if  oophectomy == 0 => has overies else illegal value
      fprintf(errFile, "\n ERROR Oophectomy illegal value %i, StudyID %i, case# %i", (oophectomy +1), studyID, caseNum);
      oophectomy = -1;
   }

   if (oophectomy == 0 || oophectomy == 1 || oophectomy == 4)
      binOoph = 0;  // binary CAT
   else if (oophectomy == 2 || oophectomy == 3)
      binOoph = 1;
   else if (oophectomy = -1)
      binOoph = -1;

   return oophectomy;
}


int crossCheckMenstrualVars(int studyID, int caseNum) {
   // Will check variables related to menstrual period
   // corrMens
   int corMen = 0;

   if (mensesCat == 0 && ageAtRef > 55 && corrBinMens == 0) { // unlikely to still have menstrual period at age > 50
      corMen = 1; // problem to
   }

   return 0;
}


int processOCuse(int OCuse, int coreFlag, int studyID, int caseNum) {
   // At refDate what best describes use of OCs
   int maxLengthOCuse = 0;
   if (coreFlag == 1)
```

```
{  // Change to 0=NEVER USE, 1=PastUse, 2= CURRENT USE
   switch (OCuse) {
      case  3: { OCuse =  1; // past used OCs
                 binOCuse = 1;
               } break;
      case  2: { OCuse =  2; // current OC use
                 binOCuse = 1;
               } break;
      case  1: { OCuse =  0;  // Never used
                 binOCuse = 0;
               } break;
      case  9: { OCuse = -1;  // missing
                 binOCuse = -1;
               } break;
      default:  { // problem illegal OCuse value
                 OCuse = -1;
                 binOCuse = -1;
                 fprintf(errFile, "\n ERROR OCuse illegal value %i, StudyID %i, case# %i", OCuse, studyID, caseNum);
                 return -1;
               }
   }
} else {   // 'NEW Controls'Change to 0=NEVER USE, 1=PastUse, 2= CURRENT USE
      switch (OCuse) {
         case  3: { OCuse =  0; // never used OCs
                    binOCuse = 0;
                  } break;
         case  2: { OCuse =  1; // past OC use
                    binOCuse = 1;
                  } break;
         case  1: { OCuse =  2;  // current use
                    binOCuse = 1;
                  } break;
         case  9: { OCuse = -1;  // missing
                    binOCuse = -1;
                  } break;
         default:  { // problem illegal OCuse value
                    OCuse = -1;
```

```
                    binOCuse = -1;
                    fprintf(errFile, "\n ERROR OCuse illegal value %i, StudyID %i, case# %i", OCuse, studyID, caseNum);
                    return -1;
                   }
            }
      }

      if (OCuse == 0 || OCuse == -1) {
         lengthOC = 0;
         logLenOC = 0;
      }
      else if (lengthOC == 999) {
         lengthOC = -1;    // missing but used
         logLenOC = -1;
      }
      else {
         maxLengthOCuse = (ageAtRef - ageMenarche + 1) * 12;
         if ((ageMenarche != 99 && lengthOC != 999) && (lengthOC > maxLengthOCuse)) {// Do we want to just calculate years, do we care about the increase in
each month risk?
            fprintf(errFile, "\nCHECK lenthcOCuse %i months > ageAtRef %i - ageMenarche %i * 12 = %i", lengthOC, ageAtRef, ageMenarche, maxLengthOCuse);
            lengthOC = maxLengthOCuse; // if date are avail to check then truncate can't use for more than menarche to age at ref
         }
         logLenOC = (int)log(lengthOC);
      }
      return OCuse;
}


int processBreastCancer(int BcanDX, char mYrBCDX[6],  int yrBorn, int coreFlag, int studyID, int caseNum) {
      // Have ever had Breast Cancer
      int yrBCDX, monthBCDX;

      ageBCDXcat = -1; // initalize if no BC not sure of this value, not missing but no BC
      BCcat = checkBinaryCat(BcanDX, "BreastCancerDX\0", studyID, caseNum);
      if (BCcat == 1) {
         if (coreFlag == 1) { // has age at breast CA DX so not need to compute
            if (coreAgeBCA <= ageAtRef)
```

```
            ageBCDXcat = calculateAgeCategory (coreAgeBCA, "Breast CAncer DX\0", studyID, caseNum);
         else
            BCcat = 0;
      } else {
         if (strcmp(mYrBCDX, "99/99") != 0) {
            sscanf(mYrBCDX, "%d/%d", &monthBCDX, &yrBCDX);  // get year motht of BCDX out of string
            if (yrBCDX < refYear || (yrBCDX == refYear && monthBCDX <= refMonth))
               ageBCDXcat = calculateAgeCatFromYearDateString(mYrBCDX, yrBorn, "Breast CAncer DX\0", studyID, caseNum);
            else
               BCcat = 0; // breast cancer not before ref date
         } else {
            fprintf(errFile, "\nERROR BREAST CANCER DX but no BC DX date %s studyID %i caseNum %i", mYrBCDX, studyID, caseNum);
            BCcat = 0;
         }
      }
   }
   return BCcat;
}


int processBreastSurgery(int anyBsurg, int coreFlag, int studyID, int caseNum) {
   // if processing CORE control will need to set variables for BI question variables Do seperatly!
   // first initalize all output codes to 0;

   int result, BSyear, BSmonth;
   BIleak = -1;
   BScat = -1;
   ageBScat = -1;
   purpBScat = -1;
   procBSBIcat = 0;// 0=no BI 1=BI
   BSBIprobCat = 0;

   // ok process Breast surgery variables
   BScat = checkBinaryCat(anyBsurg, "Any Breast Surgery\0", studyID, caseNum);
   if (BScat == 1) { // yes has had breast surgery see if it is < reference date
      // double check know both age and date so check both
      BSyear = 99;
```

```
if (strcmp(dateBsurg, "99/99") == 0 && age1stBsurg != 99) { // calclate date from age
   BSmonth = 6;
   BSyear = (age1stBsurg + birthYear) + 1; // not as accurate but close
} else if (strcmp(dateBsurg, "99/99") != 0)
   result = sscanf(dateBsurg, "%d/%d", &BSmonth, &BSyear);

if (age1stBsurg == 99 && strcmp(dateBsurg,"99/99") !=0) // Need to calculate age if missing and moth yr avail or vice versa
   age1stBsurg = calculateAgeFromDateString(dateBsurg, birthYear, "age1stBsurg\0", studyID, caseNum);

if (age1stBsurg != 99 || BSyear != 99) {
   // first a check to see that age1st breast surg year == date year
   if (abs((age1stBsurg + birthYear) - BSyear) > 1)
      fprintf(errFile,
            "\n ERROR: AGE %i 1st BS year %i != Year %i  1st Bsurg Age @1st Breast Surgery DATE %i %i > REFDATE %i %i , StudyID %i, case# %i",
            age1stBsurg, age1stBsurg, BSyear, BSmonth, BSyear, refMonth, refYear, studyID, caseNum);

   if ((BSyear < refYear || (BSyear == refYear && BSmonth <= refMonth)) || (age1stBsurg <= ageAtRef)) {
      ageBScat = calculateAgeCategory(age1stBsurg, "ageBsurg\0", studyID, caseNum); // don't bother with year date use age to compare
   } else {
      BScat = 0;
      ageBScat = -1;
      fprintf(errFile,"\n NOTE:AGE / Date of first breast surg Age %i @1st Breast Surgery DATE %i %i > REFDATE %i %i , StudyID %i, case# %i",
            age1stBsurg, BSmonth, BSyear, refMonth, refYear, studyID, caseNum);
   }
} else {
   BScat = -1;
   ageBScat = -1;
   fprintf(errFile, "\n ERROR: MISSing both of date %s and age %i of First Breast Surgery DATE StudyID %i, case# %i", dateBsurg,
                  age1stBsurg, studyID, caseNum);
}

purpBScat = BSpurpose -1;  // now get BS purpose
if (purpBScat == 7) {
   purpBScat = otherBSpurp;
   if (coreFlag == 1)
      fprintf(missCore, "\n OtherPurpose for Breast Surgery CORE studyID %i", studyID);
}
```

```
        if ((procedureBSBI >= 1 && procedureBSBI <= 3) ||
            (purpBScat == 0 || purpBScat == 2) ||
            (procedureBSBI == 8 && (otherBSproc == 36 || otherBSproc == 37 || otherBSproc == 39 || otherBSproc == 44 || otherBSproc == 45)))
           procBSBIcat = 1;  // Implant
        else
           procBSBIcat = 0; // No implant

        if (procBSBIcat == 1) {
           if (problemsBSBI == 2 || (problemsBSBI == 3 && specBSprob == 23)) { // BI problem - leaking
              BSBIprobCat = 1;
              BIleak = 1;
           } else if (problemsBSBI ==3) {
              BSBIprobCat = 1;
              BIactualProbCode = specBSprob;
              BIleak = 0;
           } else
              BSBIprobCat = 0;
        }
     } else if (BScat == 0 || BScat == -1) {// no breast surgery ever so null other variables, or missing
        return BScat;
     } else {  // Error illegal value
        fprintf(errFile, "\n ERROR Breast Surgery illegal value %i, StudyID %i, case# %i", implantUse,  studyID, caseNum);
        return -1;
     }
     return BScat;
}

int processBreastImplants(int implantUse, int studyID, int caseNum) {
     // These variables have been set by Breast surgery section if it indicated breast implants
     int result = 0;
     int BImonth, BIyear, ageBI;


     BIageCat = -1;
     BIcat = -1; // implantUse 2=Yes -> 1=YES
     BIduration = 0;
```

```
BIlogDur = 0;
BIlnDur = 0;
BItypeCat = -1;
BIreasonCat = -1;
postBIsurgCat = -1;

BIcat = checkBinaryCat(implantUse, "implantUse", studyID, caseNum);
if (BIcat == 1) { // implantUse Yes = 2
    // Need to check if date of implant is <= refDate (month, year) if NOT BIcat = 0
    result = sscanf(dateFirstBI, "%d/%d", &BImonth, &BIyear);
    ageBI = calculateAgeFromDateString(dateFirstBI, birthYear, "implantUse",  studyID, caseNum);
    if (BIyear < refYear || (BIyear == refYear && BImonth < refMonth)) { //|| (ageBI <= ageAtRef))
        // last if only have age
        BIageCat = calculateAgeCategory(ageBI, "implantUse", studyID, caseNum);
        // calculate BI duration of implants <= refDate
        BIduration = ((refYear - BIyear) * 12) + 12 - BImonth + refMonth;
        BIlogDur = (int) log10(BIduration);
        BIlnDur = (int) log(BIduration);
    }
    else {
        BIcat = 0;
        fprintf(errFile, "\n NOTE: Breast Implant DATE %i %i > REFDATE %i %i StudyID %i, caseNum %i", BImonth, BIyear, refMonth, refYear, studyID,
caseNum);
    }

    fprintf(debFile, "\n BI+ StudyID %i\t CaseNum %i\t BIcat %i\t BImonth %i\t BIyear %i\t RefMonth %i\t refYear %i\t ", studyID, caseNum, BIcat, BImonth,
BIyear, refMonth, refYear);

    if (BIcat == 0)  // had a BI but after ref date - don't bother checking other records
        return BIcat;

    if (implantType == 1 || implantType == 2) { // Silicone = 1, Saline = 2,
        // either Silicone or Saline
        BItypeCat = implantType -1; // Silicone = 0, Saline = 1, Both = 2; 3 = DK
    } else if (implantType == 9) {
        if (otherTypeBI == 11 || otherTypeBI == 12 || otherTypeBI == 13) // double lumen,both saline & sil,gel
            BItypeCat = 2;
```

```
            else
                BItypeCat = -1;
        } else if (implantType == 99) {
            BItypeCat = -1; // Missing
            fprintf(errFile, "\n CHECK Breast Implant Type MISSING value %i, StudyID %i, case# %i", implantType, studyID, caseNum);
        } else  // illegal value for implant type
            fprintf(errFile, "\n ERROR Breast Implant Type illegal value %i, StudyID %i, case# %i", implantType, studyID, caseNum);

        if (reasonBI == 1)
            BIreasonCat = 0;
        if (reasonBI == 3 || reasonBI == 4)
            BIreasonCat = reasonBI -2;
        else if (reasonBI == 9 && otherBIreason == 13)
            BIreasonCat = 2; // reconstruction
        else if (reasonBI == 9 && (otherBIreason == 11 || otherBIreason == 12))
            BIreasonCat = 0; // Augmentation husband or friend wanted
        else if (reasonBI == 9)
            BIreasonCat = -1;
        else
            fprintf(errFile, "\n ERROR Breast Implant Reason illegal value %i, StudyID %i, case# %i", reasonBI, studyID, caseNum);

        postBIsurgCat = checkBinaryCat(otherBIsurg, "other BI surgery", studyID, caseNum);

    }
    else if (BIcat == 0 || BIcat == -1) {  // NO BI's or MISSING
        return BIcat;
    }
    else { // illegal value
        fprintf(errFile, "\n ERROR Breast Implants illegal value %i, StudyID %i, case# %i", implantUse, studyID, caseNum);
        return -1;
    }
    return BIcat;
}
```

```
int processCOREbreastSurgeryToImplantVars() {
   // Will generate BI related variables from breast surgery questions Same info is there
   BIageCat = -1;
   BItypeCat = -1;
   BIreasonCat = -1;
   postBIsurgCat = -1;
   BIduration = 0;
   BIlogDur = 0;
   BIlnDur = 0;

   if (procBSBIcat != 1) // have breast implant
      BIcat = 0;
   else {
      BIcat = 1;
      BIageCat = ageBScat;
      BItypeCat = procBSBIcat -1; // may need to do something different here
      BIreasonCat = purpBScat;
      postBIsurgCat = BSBIprobCat;
      fprintf(debFile, "\n CORE BI+ StudyID %i\t CaseNum %i\t BIcat %i\t BIageCat %i\t BSBIage %i\t RefMonth %i\t refYear %i\t ", studyID, caseNum,
               BIcat, BIageCat, age1stBsurg, refMonth, refYear);
      if (age1stBsurg <= ageAtRef) {
         // calculate BI duration of implants <= refDate
         BIduration = ((ageAtRef - age1stBsurg) * 12) + 6;// 6 avg months since don't know exactly
         BIlogDur = (int) log10(BIduration);
         BIlnDur = (int) log(BIduration);
      } else {
         BIcat = 0;
         fprintf(errFile, "\n NOTE: Breast Implant Age %i > REFDATE %i %i StudyID %i, caseNum %i", age1stBsurg, refMonth, refYear, studyID, caseNum);
      }
   }
   return 0;
}

int crossCheckBreastCats(int studyID, int caseNum) {
   // check to make sure the inplant related info is consistent between the two questions
   // Couple of question 16 answered NO to BI's and Yes on Q 17, will fix the Q16 to reflect Q 17
   if (BIcat == 1 || procBSBIcat ==1) {
```

```
        if (BIcat != procBSBIcat)
            fprintf(errFile, "\n Possible ERROR procBSBIcat %i != BIcat %i studyID %i caseNum %i", procBSBIcat, BIcat, studyID, caseNum);
        if (BIcat == 1 && ageBScat != BIageCat)
            fprintf(errFile, "\n Possible ERROR ageBScat %i != BIageCat %i studyID %i caseNum %i", ageBScat, BIageCat, studyID, caseNum);
//      if (purpBScat != BIreasonCat)
//          fprintf(errFile, "\n Possible ERROR purpBScat %i != BIreasonCat%i studyID %i caseNum %i", purpBScat, BIreasonCat , studyID, caseNum);
//      if (BSBIprobCat != postBIsurgCat)
//          fprintf(errFile, "\n Possible ERROR procBSBIcat %i != BIotherSurgCat %i studyID %i caseNum %i", BSBIprobCat, postBIsurgCat, studyID, caseNum);
    }
    // Also need to check BIleak, BIactualProc

    /* MAKE SURE DO CHECK THAT 1ST IMPLANT DATE < REFDATE on CASES AND CONTROLS!!!!!! */
    return 0;
}

int ProcessMaritalStatus(int marital, int studyID, int caseNum) {
    // What was marital status @ ref Date?

    if (marital >= 1 && marital <= 6) {
        binMarit = 1;   // binary marital status ever married
        if (marital == 1)
            binMarit = 0;
        return (marital -1);
    }
    else if (marital == 9) {
        binMarit = -1;
        return -1; // missing
        fprintf(errFile, "\n CHECK: Marital Status MISSING value %i, StudyID %i, case# %i", marital, studyID, caseNum);

    }
    else {
        // illegal value for marital status
        fprintf(errFile, "\n ERROR Marital Status illegal value %i, StudyID %i, case# %i", marital, studyID, caseNum);
        binMarit = -1;
        return -1;
    }
}
```

```
int ProcessWorkHistory(int jobRef, int studyID, int caseNum) {
   // 0 = out side home job, 1 = housewife
   int notHouseWife;
   if (jobRef == 998  || jobRef == 997) // house wife or retired
      notHouseWife = 0;
   else
      notHouseWife = 1; // i.e. works outside of home

   industCat = indusRef;  // side effect
   return notHouseWife;
}

int checkRace(int race, int raceOther, int caseDiseaseDataIndx, int caseFlag, int coreFlag, int studyID, int caseNum) {
   // will process race cat and other race cat and make just 1 variable

   race--;
   // Find chart abstraction form race data and compare and update raceDiseaseTable if different
   if (caseFlag == 1 && ((race + 1) != caseInfo[caseDiseaseDataIndx].cRace)) {
      // if chart abstraction race does not match questionneer race, use chart category as definite over chart abs form
      fprintf(errFile, "\n ERROR Chart Abs. Form Race %i different than quest race  %i StudyID %i, case# %i",
            race, caseInfo[caseDiseaseDataIndx].cRace, studyID, caseNum);
      if ((race + 1) != 99)
        caseInfo[caseDiseaseDataIndx].cRace = race + 1;
      else
         if (caseInfo[caseDiseaseDataIndx].cRace >0 && caseInfo[caseDiseaseDataIndx].cRace < 6)
           race = caseInfo[caseDiseaseDataIndx].cRace - 1;
   }

   if (race == 0) {
      binRace = 0;  // white
      raceCat3 = 0;
   } else if (race == 1) {
      binRace = 1;  // every one else
      raceCat3 = 1; // Black
   } else if (race == 8 || race == 6) {
      binRace = -1;
      raceCat3 = -1;
```

```
    } else {
        binRace = 1;
        raceCat3 = 2; // Other
    }

    if (race >= 0 && race <=4)
        return race;
    else if (race == 6) // refused
        return -1;
    else if (race == 7) { // other
        if (coreFlag == 1) { // just for core ctrls already map -> asian
            fprintf(missCore, "\n Other RACE CORE studyID %i", studyID);
            return 5;
        } else if (raceOther == 14 || raceOther == 11 || raceOther ==13) { // see if can merge other races into known cat
                if (caseFlag == 1)
                    caseInfo[caseDiseaseDataIndx].cRace = 4;
                return 3;
            }
        else
            return 5;
    }
    else if (race == 8) { // Don't Know
        binRace = raceCat3 = -1;
        return -1;
    }
    else if (race == 98) { // left blank
        binRace = raceCat3 = -1;
        return -1;
    }
    else {
        fprintf(errFile, "\n ERROR Race illegal value %i, StudyID %i, case# %i", (race + 1), studyID, caseNum);
        binRace = raceCat3 = -1;
        return -1;
    }
    return 0;
}
```

```
int checkEducation(int educatHigest, int otherEd, int yrsComp, int coreFlag, int studyID, int caseNum) {
   /*  Generate two variables -
       eduCat is the higest categort of education RETURNED
       eduYrsCat is the total number of years of education SIDE EFFECT
   */

   educatHigest--;
   if (yrsComp == 99)
      yrsComp = 1; // attended at least one year

   if (educatHigest == 0) { // Grade school
      if (yrsComp >= 0 && yrsComp <= 9)
         eduYrsCat = yrsComp;
      else // something fully
         fprintf(errFile, "\n CHECK Grade School EDU Yrs Comp out of Bound %i, StudyID %i, case# %i", yrsComp, studyID, caseNum);
      return educatHigest;
   }
   else if (educatHigest == 1) { // high school
      eduYrsCat = 8;
      if (yrsComp >= 0 && yrsComp <= 4)
         eduYrsCat += yrsComp;
      else if (yrsComp >= 8 && yrsComp <= 15)
         eduYrsCat = yrsComp;
      else
         fprintf(errFile, "\n CHECK High School Edu Yrs Comp out of Bound %i, StudyID %i, case# %i", yrsComp, studyID, caseNum);
      return educatHigest;
   }
   else if (educatHigest == 2) { // tech school
      eduYrsCat = 12;
      if (yrsComp >= 0 && yrsComp <= 5)
         eduYrsCat += yrsComp;
      else if (yrsComp >= 12 && yrsComp <= 16)
         eduYrsCat = yrsComp;
      else
         fprintf(errFile, "\n CHECK Technical School Edu Yrs Comp out of Bound %i, StudyID %i, case# %i", yrsComp, studyID, caseNum);
      return educatHigest;
   }
```

```
else if (educatHigest == 3) { // College
   eduYrsCat = 12;
   if (yrsComp >= 0 && yrsComp <= 6)
      eduYrsCat += yrsComp;
   else if (yrsComp >= 12 && yrsComp <= 16)
      eduYrsCat = yrsComp;
   else
      fprintf(errFile, "\n CHECK College Edu Yrs Comp out of Bound %i, StudyID %i, case# %i", yrsComp, studyID, caseNum);
   return educatHigest;
}
else if (educatHigest == 4) { //Grad school
   eduYrsCat = 16;
   if (yrsComp >= 0 && yrsComp <= 7) // possib MED school
      eduYrsCat += yrsComp;
   else if (yrsComp >= 16 && yrsComp <= 23)
      eduYrsCat = yrsComp;
   else
      fprintf(errFile, "\n CHECK Graduate School Edu Yrs Comp out of Bound %i, StudyID %i, case# %i", yrsComp, studyID, caseNum);
   return educatHigest;
}
else if (educatHigest == 7) {
   // other education - lot of categories to check
   //First have to get baisc start value of Yrs set right
   if (coreFlag == 1)
      fprintf(missCore, "\n Other EDUCATION CORE studyID %i", studyID);

   switch (otherEdu) {
      case -1:
         { educatHigest = -1; eduYrsCat = -1; } break;// missing
      case 11: case 21: case 23: case 24:
         { educatHigest = 2; eduYrsCat = 14; } break;   // Beauty, certificate NOSReality Art School
      case 16:
         { educatHigest = 3; eduYrsCat = 18; } break; // Business school
      case 15: case 17: case 18:
         { educatHigest = 2; eduYrsCat = 14; } break; // secretarial, vocational nurseing school
      case 20:
         { educatHigest = 2; eduYrsCat = 14; } break;   // certificate unspecified art school interior design
```

```
            case 19: case 22:
               { educatHigest = 3; eduYrsCat = 16; } break;   // College night school, bible school
            case 12:
               { educatHigest = 1; eduYrsCat = 12; } break;   // GED
            case 14:
               { educatHigest = 1; eduYrsCat = 10; } break;   // Jr. Hi secondary
            default:
               { fprintf(errFile, "\n CHECK Graduate School Edu Yrs Comp out of Bound %i, StudyID %i, case# %i", yrsComp, studyID, caseNum);
                 educatHigest = -1; eduYrsCat = -1;
               }
         }
         return educatHigest;
      }
      else if (educatHigest == 8) {
         // Don't know, Hard to believe unless surrogate doing quest., even then unlikely
         fprintf(errFile, "\n CHECK Does not Know Higest level %i, StudyID %i, case# %i", yrsComp, studyID, caseNum);
         eduYrsCat = -1;
         return -1;
      }
      else if (educatHigest == 99 || educatHigest == 98) {
         fprintf(errFile, "\n CHECK Educationest Graduate School Edu Yrs Comp out of Bound %i, StudyID %i, case# %i", yrsComp, studyID, caseNum);
         eduYrsCat = -1;
         return -1;
      }
      else {
         // illegal value
         fprintf(errFile, "\n ERROR Education illegal value %i, StudyID %i, case# %i", (educatHigest + 1), studyID, caseNum);
         return -1;
      }
      return 0;// should never get here
}


int checkIncome(int income, int studyID, int caseNum) {
      income--;
      if (income >= 0 && income <= 3)
          return income;
```

```
    else if (income == 8)
        return -1; // DK or missing
    else {
        fprintf(errFile, "\n ERROR Income illegal value %i, StudyID %i, case# %i", (income + 1), studyID, caseNum);
        //recordProblem = True;
        return -1;
    }
}


int outputRecordToFile(FILE *oFile, int coreFlag) {
    // will put the information to specified file
    fprintf(oFile,"%i %i %i %i %i %i %i %i %i %i %i %i %i %i %i ",
            coreFlag, studyID, caseNum, caseCtrl, polyCCflag, DXcat, DXDvalid, ageAtRef, ageRefCat, ageCat6, refYearCat,
            refYear3, nowKCcat, thenKCcat, phoneCat);
    fprintf(oFile,"%i %i %i %i %i %i %i %i %i %i %i %i %i %i %i %i ",
            height, weight, lnWeight, logWeight, weightCat, BMI, eduYrsCat, eduCat, jobCat, industCat, maritalCat, binMarit,
            incomeCat, raceCat, raceCat3, binRace);
    fprintf(oFile,"%i %i %i %i ", diabCat, ageDiabCat, insulinCat, insulinExpCat);
    fprintf(oFile,"%i %i %i ", dyeCat, dyeExpMin, logDyeExpMin);
    fprintf(oFile,"%i %i %i %i %i %i %i %i %i %i %i %i %i %i %i %i %i ",
           menarchCat, mensesCat, binMenses, corrBinMens, OCcat, binOCuse, lengthOC, logLenOC, estrogenCat,
           agePause, agePauseCat, natPauseCat, hysterectomyCat, oophectCat, binOoph, ageOophCat, whyPauseCat);
    fprintf(oFile,"%i %i %i %i %i %i %i %i %i %i ", gravid, numMisCar, ectopic, indAbort, stillBirth, liveBirths,  multBirth, otherBirth, binGravid, binKids);
    fprintf(oFile,"%i %i %i %i %i %i %i %i %i ", BCcat, ageBCDXcat, BScat, ageBScat, purpBScat, procBSBIcat, BSBIprobCat, BSnoBI);
    fprintf(oFile,"%i %i %i %i %i %i %i %i %i\n", BIcat, BIduration, BIlogDur, BIlnDur, BIageCat, BIreasonCat, BItypeCat, postBIsurgCat, BIleak);
    return 0;
}

int outputRecord(int isCaseRec, int isCoreRec) {
    // will out put the cleaned and transformed data record
    // Try to put this out in a more sensible format than what it was read in as

    if (isCaseRec == 1 || isCoreRec == 0) // output cases and new controls to file for analysis
        outputRecordToFile(outFile, isCoreRec);

    if (isCaseRec == 1 || isCoreRec == 1) {
```

```
      // output cases and Core controls to file for analysis except not use year 91 since no controls and also other years in age cats
      if (refYearCat == 8 || (refYearCat == 0 && (ageRefCat > 3)) ||(refYearCat == 1 && (ageRefCat > 4)) ||
          (refYearCat == 2 && (ageRefCat > 8)) || (refYearCat == 3 && (ageRefCat > 8))) {
          //NULL BODY Don't output record
      } else
          outputRecordToFile(coreFile, isCoreRec);
   }

   // Out put everything (Cases, Controls, CoreControls) to one file
   if (isCaseRec == 1)
      outputRecordToFile(cccFile, 2);// not CORE is case
   else if (isCaseRec == 0 && isCoreRec == 0)
      outputRecordToFile(cccFile, 1);// not CORE is other control
   else
      outputRecordToFile(cccFile, 0);// is CORE control

   return 0;
}


int initActualCaseCtrls() {
   // initalizes the case disease and control matricies for the actual ageDx year DX distribution
   // can reuse tables for incidence data as well
   int i,j;

   for (i = 0; i < 14; i++)
      for (j = 0; j < 12; j++) {
         COREageDXyrDX[i][j] = 0;
         CTRLageDXyrDX[i][j] = 0;
         SLEageDXyrDX[i][j] = 0;
         PSSageDXyrDX[i][j] = 0;
         SSageDXyrDX[i][j] = 0;
         MCTDageDXyrDX[i][j] = 0;
         PMDMageDXyrDX[i][j] = 0;
         ALLageDXyrDX[i][j] = 0;
         BlctrlAgeDXyrDX[i][j] = 0;
         BlcoreAgeDXyrDX[i][j] = 0;
```

```
            BlcaseAgeDXyrDX[i][j] = 0;
            ALLageDXyrDX[i][j] = 0;
            //raceDiseaseTable[i][j] = 0;
        }
    }
    return 0;
}


int UpdateTables(int ageRefCat, int refDateCat, int caseCtrl, int coreCtrl, int caseDiseaseDataIndx, int studyID, int caseNum) {

    if (caseCtrl == 1) { //CASE
        numCases++;
        ALLageDXyrDX[ageRefCat][refDateCat]++;    // keeps track of returned cases
        if (Blcat == 1)
            BlcaseAgeDXyrDX[BlageCat][refDateCat]++;

        if (ICDcodeDX == (float)710.000) {
            SLEageDXyrDX[ageRefCat][refDateCat]++;
            numSLEcases++;
        } else if (ICDcodeDX == (float)710.100) {
            PSSageDXyrDX[ageRefCat][refDateCat]++;
            numPSScases++;
        } else if (ICDcodeDX == (float)710.200) {
            SSageDXyrDX[ageRefCat][refDateCat]++;
            numSScases++;
        } else if (ICDcodeDX == (float)710.300 || ICDcodeDX == (float)710.400) {
            PMDMageDXyrDX[ageRefCat][refDateCat]++;
            numPMDMcases++;
        } else if (ICDcodeDX == (float)710.900) {
            MCTDageDXyrDX[ageRefCat][refDateCat]++;
            numMCTDcases++;
        } else {
            fprintf(errFile, "\nError Illegal ICD-9 code %f.1, studyID %i, caseNum %i", ICDcodeDX, studyID, caseNum);
            fprintf(debFile, "\nError Illegal ICD-9 code %f.1, studyID %i, caseNum %i", ICDcodeDX, studyID, caseNum);
        }
    } else if (coreCtrl == 1) { // core control
        COREageDXyrDX[ageRefCat][refDateCat]++;
```

```
        numCores++;
        if (BIcat == 1)
            BIcoreAgeDXyrDX[BIageCat][refDateCat]++;

    } else { // new control group
        CTRLageDXyrDX[ageRefCat][refDateCat]++;
        numCtrls++;
        if (BIcat == 1)
            BIctrlAgeDXyrDX[BIageCat][refDateCat]++;
    }
    return 0;
}

float printRowColsFloat(int row, int labelRow, float matrix[14][12]) {
    // will print columns from row from matrix
    int col;
    float rowSum = (float) 0.0;

    if (labelRow == 0)
        fprintf(debFile,"\t\t");
    for (col = 0; col <= 8; col++) {
//      if (row == 0)
//          matrix[11][col] = (float)0.0;    // initialize so can accumulate sum for col totals in 11throw

        if (row == 11)
            fprintf(debFile,"%1.2f\t", matrix[row][col]);
        else {
            fprintf(debFile,"%1.2f\t", matrix[row][col]);
            //matrix[11][col] += matrix[row][col]; // column sum
        }
        rowSum += matrix[row][col];
    }
    return rowSum;
}
```

```
int printRowCols(int row, int labelRow, int matrix[14][12]) {
    // will print columns from row from matrix
    int col, rowSum = 0;

    if (labelRow == 0)
        fprintf(debFile,"\t\t");
    for (col = 0; col <= 8; col++) {
    //  if (row == 0)
    //      matrix[11][col] = 0;   // initialize so can accumulate sum for col totals in 11th row

        if (row == 11)
            fprintf(debFile,"%i\t\t", matrix[row][col]);
        else {
            fprintf(debFile,"%i\t\t", matrix[row][col]);
//          matrix[11][col] += matrix[row][col]; // column sum
        }
        rowSum += matrix[row][col];
    }
    return rowSum;
}

int printCaseResponseRateTable() {
    // Will Print response rate simply first
    int row, rowSum1, rowSum2;
    float fRowSum;
    int labelRow;

    fprintf(debFile, "\n %s", "Response Rate by YrDX age DX\0");
    fprintf(debFile, "\nAGE\t\t1983   1984 1985 1986 1987 1988 1989 1990 1991 Total");

    for (row = 0; row <= 11; row++) {
        rowSum1 = 0; rowSum2 = 0; fRowSum = (float)0.0;
        if (row == 0)
            fprintf(debFile, "\n%i-%i\t", 18, 24);
        else if (row == 11)
            fprintf(debFile, "\nTotal\t");
        else
```

```
            fprintf(debFile, "\n%i-%i\t", 20 + ((row) * 5), 24 + ((row) * 5));

        labelRow = 1; // to tell col print when to put in tabs
        rowSum1 = printRowCols(row, labelRow, allIDcasesAgeDXyrDX);     // cases identified
        fprintf(debFile,"%i\t\n", rowSum1);
        labelRow = 0;
        rowSum2 = printRowCols(row, labelRow, ALLageDXyrDX);            // cases returned quest
        fprintf(debFile,"%i\t\n", rowSum2);
        fRowSum = printRowColsFloat(row, labelRow, caseResponseRate); // % response rate
        fprintf(debFile,"%1.2f\t\n", (float) rowSum2/ (float) rowSum1);
    }
    return 0;
}


int prtAgeRaceAvgMat(int ageYrMat[14][12], int diseaseType, char Title[80]) {
    int row, col, rowSum;

    fprintf(debFile, "\n %s", Title);
    printDiseaseTableLabel(diseaseType);
    fprintf(debFile, "\nAGE\t\twhite\tblack\tAmInd\tAsian\tHispanic\tOther\tTotal");
    for (row = 0; row <= 11; row++) {
        rowSum = 0;
        if (row ==0)
            fprintf(debFile, "\n%i-%i\t", 18, 24);
        else if (row == 11)
            fprintf(debFile, "\n%Total\t");
        else
            fprintf(debFile, "\n%i-%i\t", 20 + ((row) * 5), 24 + ((row) * 5));
        for (col = 0; col <= 6; col++) {
            if (row == 0)
                ageYrMat[11][col] = 0;
            if (row == 11)
                fprintf(debFile,"%i\t\t", ageYrMat[row][col]);
            else {
                fprintf(debFile,"%i\t\t", ageYrMat[row][col]);
                ageYrMat[11][col] += ageYrMat[row][col]; // column sum
```

```
            }
            rowSum = rowSum + ageYrMat[row][col];
        }
        fprintf(debFile,"%i\t", rowSum);
    }
    return 0;
}


int prtAgeDXyrDXMat(int ageYrMat[14][12], char Title[80]) {
    int row, col, rowSum;

    fprintf(debFile, "\n %s", Title);
    fprintf(debFile, "\nAGE\t\t1983  1984 1985 1986 1987 1988 1989 1990 1991 Total");
    for (row = 0; row <= 11; row++) {
        rowSum = 0;
        if (row ==0)
            fprintf(debFile, "\n%i-%i\t", 18, 24);
        else if (row == 11)
            fprintf(debFile, "\n%Total\t");
        else
            fprintf(debFile, "\n%i-%i\t", 20 + ((row) * 5), 24 + ((row) * 5));
        for (col = 0; col <= 8; col++) {
            if (row == 0)
                ageYrMat[11][col] = 0;
            if (row == 11)
                fprintf(debFile,"%i\t\t", ageYrMat[row][col]);
            else {
                fprintf(debFile,"%i\t\t", ageYrMat[row][col]);
                ageYrMat[11][col] += ageYrMat[row][col]; // column sum
            }
            rowSum = rowSum + ageYrMat[row][col];
        }
        fprintf(debFile,"%i\t", rowSum);
    }
    return 0;
}
```

```
int outputActualCaseCtrl() {
   // Simply output the data have read in in a more organized and intelligible fashion
   int i,j;

   prtAgeDXyrDXMat(ALLageDXyrDX, "\n\t\tRFCTD All CASES that Returned Questionneer ");

   prtAgeDXyrDXMat(SLEageDXyrDX, "\n\t\tSLE CASES ");

   prtAgeDXyrDXMat(PSSageDXyrDX, "\n\t\tScleroderma Cases ");

   prtAgeDXyrDXMat(SSageDXyrDX, "\n\t\tSjogrens Syndrome Cases");

   prtAgeDXyrDXMat(PMDMageDXyrDX, "\n\t\tPolymyositis/ Dermatomyositis");

   prtAgeDXyrDXMat(MCTDageDXyrDX, "\n\t\tMCTD cases all ");

   prtAgeDXyrDXMat(CTRLageDXyrDX, "\n\t\tControls ");

   prtAgeDXyrDXMat(COREageDXyrDX, "\n\t\tCore Controls ");

   prtAgeDXyrDXMat(BIcoreAgeDXyrDX, "\n\t\tBI+ Core Controls ");

   prtAgeDXyrDXMat(BIctrlAgeDXyrDX, "\n\t\tBI+ Controls ");

   prtAgeDXyrDXMat(BIcaseAgeDXyrDX, "\n\t\tBI+ CASES ");

   prtAgeDXyrDXMat(allIDcasesAgeDXyrDX, "\n\t\tALL CASES ");

   // need to calculate case response rate
   for (i = 0; i <= 11; i++)
      for (j = 0; j <= 9; j++) {
         if (allIDcasesAgeDXyrDX[i][j] > 0 && ALLageDXyrDX[i][j] > 0)
            caseResponseRate[i][j] = (float) ALLageDXyrDX[i][j] / (float) allIDcasesAgeDXyrDX[i][j];
         else
            caseResponseRate[i][j] = (float) 0.00;
```

```
        }
    printCaseResponseRateTable();


    fprintf(debFile, "\nSummary of calls to count cases and controls, if this agrees with tables then the calls not  made");
    fprintf(debFile, "\nnumCases=%i numSLEcases=%i numPSScases=%i numSScases=%i numMCTDcases=%i numPMDMcases=%i
            numCtrls=%i numCores=%i", numCases, numSLEcases, numPSScases, numSScases, numMCTDcases, numPMDMcases, numCtrls, numCores);
    return 0;
}


int readKCpopRaceData(int KCpop[14][12], FILE *KCpopFile, char tableName[80]) {
    // Will read in incidence matrix info and print it out to debug file
    char inString[512];
    char inString2[512];

    char seps[] = ",";
    char *token, *token2;
    int NR = 0; // number of tokens read
    int row, col, rowSum;
    int linesRead = 0;

    for (row = 0; row <= 11; row++) {
        fgets(inString, 511, KCpopFile);
        if (row == 0)
            fgets(inString2, 511, KCpopFile);   // need to combine 18-19 and 20-24

        linesRead ++;
        if (feof(KCpopFile)) {
            // make sure we read and process all records
            printf("\nWe have reached end-of-file %s", KCpopFile);
            fclose(KCpopFile); /* close the file */
            return -1;
        }

        NR++;
        for (col = 0; col <= 9; col++) {
```

```
        if (col == 0)
            token = strtok(inString, seps );
        else
            token = strtok( NULL, seps );
        KCpop[row][col] = atoi(token); // excell dumped out backwards
        NR++;
        if (token != NULL)
            NR++; // count after know not at end of line i.e. last token read is not valid because eol not set until last read
    }
    /* OK should have read the right number of tokens if not flag an error */
}

for (col = 0; col <= 9; col++) { // adjust 1st row 19-24
    if (col == 0)
        token2 = strtok(inString2, seps );
        else
            token2 = strtok( NULL, seps );
        KCpop[0][col] += atoi(token2);
    }


fprintf(debFile,"\n %s", tableName);
 fprintf(debFile, "\nAGE\t\t1983  1984 1985 1986 1987 1988 1989 1990 1991 Average ");
 for (row = 0; row <= 11; row++) {
    rowSum = 0;
    if (row ==0)
        fprintf(debFile, "\n%i-%i\t", 18, 24);
    else if (row == 11)
        fprintf(debFile, "\n%Total  ");
    else
        fprintf(debFile, "\n%i-%i\t", 25 + ((row-1) * 5), 29 + ((row-1) * 5));
    for (col = 0; col <= 9; col++) {
        if (row == 11)
            fprintf(debFile,"%i  ", KCpop[row][col]);
        else
            fprintf(debFile,"%i\t", KCpop[row][col]);
//        if (col != 9)
```

```
//              rowSum += KCpop[row][col];
        }
//      fprintf(debFile,"%i\t", rowSum/9);
    }

    return 0;
}

int readRaceDataKC90(int KCpop[14][12]) {
    // Will read in population all races 1990 data into wbiahoKC90 matrix and print it out to debug file
    char inString[256];
    char seps[] = ",";
    char *token;
    int NR = 0; // number of lines, tokens read
    int row, col, rowSum;
    int linesRead = 0;

    if ((KCpopFile = fopen("C:/research/rfctdata/casectrl/incidence/wbiaho90.csv", "r" )) == NULL ) {
        printf( "Error the file 'wbiaho90.csv' was not opened\n" );
        exit(-1);
    }
    else
        printf( "The file 'wbiaho90.csv' was opened\n" );

    for (row = 0; row <= 11; row++) {
        fgets(inString, 511, KCpopFile);
        // already combined 18-19 and 20-24
        NR = 0; // reinitialize
        linesRead ++;
        if (feof(KCpopFile)) {
            // make sure we read and process all records
            printf("\nWe have reached end-of-file %s", KCpopFile);
            fclose(KCpopFile); /* close the file */
        }

        for (col = 0; col <= 5; col++) {
            if (col == 0)
```

```
            token = strtok(inString, seps );
        else
            token = strtok( NULL, seps );
        KCpop[row][col] = atoi(token); // excell dumped out backwards  ?

        if (token != NULL)
            NR++; // count after know not at end of line i.e. last token read is not valid because eol not set until last read
      }
      /* OK should have read the right number of tokens if not flag an error */
   }

  fprintf(debFile,"\n %s", "All races 1990");
   fprintf(debFile, "\nAGE\t\t White Black AmInd Asian Hispanic Other ");
   for (row = 0; row <= 11; row++) {
      rowSum = 0;
      if (row ==0)
         fprintf(debFile, "\n%i-%i\t", 18, 24);
      else if (row == 11)
         fprintf(debFile, "\n%Total  ");
      else
         fprintf(debFile, "\n%i-%i\t", 25 + ((row-1) * 5), 29 + ((row-1) * 5));
      for (col = 0; col <= 5; col++) {
         if (row == 11)
            fprintf(debFile,"%i  ", KCpop[row][col]);
         else
            fprintf(debFile,"%i\t", KCpop[row][col]);
//       if (col != 9)
//          rowSum += KCpop[row][col];
      }
//    fprintf(debFile,"%i\t", rowSum/9);
   }

   return 0;
}

int readKCpopulationData() {
   // Will readint the King County Data to generate incidence data
```

```
int inResult = 0;
int i, j;

readRaceDataKC90(wbiaho90KCpop);

if ((KCpopFile = fopen("C:/research/rfctdata/casectrl/incidence/KC8391.CSV", "r" )) == NULL ) {
    printf( "Error the file 'kc8391.csv' was not opened\n" );
    exit(-1);
}
else
    printf( "The file 'kc8391.csv' was opened\n" );

readKCpopRaceData(KCpopulation8391,  KCpopFile, "KC total POP");
fclose (KCpopFile);

if ((KCpopFile = fopen("C:/research/rfctdata/casectrl/incidence/whiteKC.CSV", "r" )) == NULL ) {
    printf( "Error the file 'whiteKC.CSV' was not opened\n" );
    exit(-1);
}
else
    printf( "The file 'whiteKC.CSV' was opened\n" );


for (i = 0; i < 14; i++)    // initialize all cases found matrix
    for (j = 0; j < 12; j++)
        allIDcasesAgeDXyrDX[i][j] = 0;

readKCpopRaceData(whiteKCpop8391,  KCpopFile, "KC White Pop");
fclose (KCpopFile);

if ((KCpopFile = fopen("C:/research/rfctdata/casectrl/incidence/blackKC.CSV", "r" )) == NULL ) {
    printf( "Error the file 'blackKC.csv' was not opened\n" );
    exit(-1);
}
else
    printf( "The file 'blackKC.csv' was opened\n" );
```

```
   readKCpopRaceData(blackKCpop8391,  KCpopFile, "KC black Pop");
   fclose (KCpopFile);

   if ((KCpopFile = fopen("C:/research/rfctdata/casectrl/incidence/otherKC.CSV", "r" )) == NULL ) {
      printf( "Error the file 'otherKC.csv' was not opened\n" );
      exit(-1);
   }
   else
      printf( "The file 'otherKC.csv' was opened\n" );

   readKCpopRaceData(otherKCpop8391,  KCpopFile, "KC other Pop");
   fclose (KCpopFile);


   return 0; // no problem
}


int getCaseDiseaseData(FILE *cFptr, char diseaseType[30]) {
   // Will read in data about case disease

   int inResult = 0; // result of reading the case raw data file from SPSS/DE
   int outResult = 0;
   int cleanFlag = 0; // -1 = abort, 0 = ok
   const caseFlag = 1;
   const int coreFlag = 0; // not core control
   int casesReadIn = 0;
   int numCasesOK = 0;  // keeps track of the number of cases successfully processed

   printf("\nReading CASE disease information");

  //caseCtrl = 1;  // set the flag so it can be output

   while (inResult >= 0) {
      //!feof(inFile)) if return 0, -1 means abort, 1 is eof
      // get next record
      // initalize before call
```

```
        inResult = readCaseDiseaseRecord(cFptr, diseaseType);      // have mneumonic names of variables now check each value
        // process it by transforming values and flaging problems
        // ok good record read now process the variables so can use them in data analysis

        if (inResult > 0) {
            caseLinesRead++;
            casesReadIn++;
            cleanFlag = processCaseDiseaseRecord(caseLinesRead);
        }
        else if (inResult != -1)
            fprintf(errFile, "\nERROR CASE NUM != problems reading Case record studyID %i CaseNum %i NR %i", studyID, caseNum);
        }

    if (inResult == -1)    // i.e. problem other thaaan end-of-file
        printf("\nCompleted Reading CASE DISEASE %s DATA read = %i Total = %i ",diseaseType, casesReadIn, caseLinesRead);

    return casesReadIn;;
}


int MMYY_MMDDYY(char my[6], char mdy[9]) {
    // will return a month year string from month day year string

    my[0] = mdy[0];
    my[1] = mdy[1];
    my[2] = '/';
    my[3] = mdy[6];
    my[4] = mdy[7];
    my[5] = '\0';
    return 0;
}


int readCaseDiseaseRecord(FILE * Fptr, char fname[80]) {
    // WE read in the case information and put into array
    // will return number of cases read in file pointer global caseFptr;
    // caseNum is number of cases read in so far
```

```
int inResult = 0;
int NR;
int linesRead = 0;
char inString[512];
char seps[] = ",";
char mmyy[6];
char *token;
int numDiseaseFields = 18; // number of tokens read

fgets(inString, 511, Fptr);
linesRead ++;
if (feof(Fptr)) {
   // make sure we read and process all records
   printf("\nWe have reached end-of-file %s", fname);
   fclose(KCpopFile); /* close the file */
   return -1;
}

caseNum++;  // increment number of cases read so far
for (NR = 1; NR <= numDiseaseFields; NR++) {
   if (NR == 1)
      token = strtok(inString, seps );
   else
      token = strtok( NULL, seps );

   if (token != NULL) { // count after know not at end of line i.e. last token read is not valid because eol not set until last read
      // just decrement pointer
      switch (NR)   {
         case  1: { fprintf(allDisFptr, "\n%i ",caseInfo[caseNum].cStudyID = atoi(token));} break;
         case  2: { fprintf(allDisFptr, "%i ", caseInfo[caseNum].cNmbr = atoi(token));} break;
         case  3: { fprintf(allDisFptr, "%i ", caseInfo[caseNum].MDid = atoi(token));} break;
         case  4: { fprintf(allDisFptr, "%i ", caseInfo[caseNum].deadCase = atoi(token));} break;
         case  5: { fprintf(allDisFptr, "%i ", caseInfo[caseNum].DX1 = atoi(token));} break;
         case  6: { fprintf(allDisFptr, "%i ", caseInfo[caseNum].ageDX1 = atoi(token));} break;
         case  7: { MMYY_MMDDYY(mmyy, token); strcpy(caseInfo[caseNum].cBirthDate, mmyy);//cAgeDX1cat
                    fprintf(allDisFptr, "%s ", caseInfo[caseNum].cBirthDate);} break;
         case  8: { fprintf(allDisFptr, "%i ", caseInfo[caseNum].cRace = atoi(token)); } break;
```

```
            case  9: { MMYY_MMDDYY(mmyy, token); strcpy(caseInfo[caseNum].DXD1, mmyy);          //DXD1
                       fprintf(allDisFptr, "%s ", caseInfo[caseNum].DXD1);} break;
            case 10: { MMYY_MMDDYY(mmyy, token); strcpy(caseInfo[caseNum].DXP1, mmyy);
                       fprintf(allDisFptr, "%s ", caseInfo[caseNum].DXP1);} break;             //DXP1
            case 11: { MMYY_MMDDYY(mmyy, token); strcpy(caseInfo[caseNum].FirstRheum, mmyy);
                       fprintf(allDisFptr, "%s ", caseInfo[caseNum].FirstRheum);} break;     //cYrDX1cat
            case 12: { MMYY_MMDDYY(mmyy, token); strcpy(caseInfo[caseNum].FirstMd, mmyy);
                       fprintf(allDisFptr, "%s ", caseInfo[caseNum].FirstMd);} break;
            case 13: { MMYY_MMDDYY(mmyy, token); strcpy(caseInfo[caseNum].dateSXstart, mmyy);
                       fprintf(allDisFptr, "%s ", caseInfo[caseNum].dateSXstart);} break;
            case 14: { MMYY_MMDDYY(mmyy, token); strcpy(caseInfo[caseNum].DateLastVisit, mmyy);
                       fprintf(allDisFptr, "%s ",caseInfo[caseNum].DateLastVisit);} break;
            case 15: { fprintf(allDisFptr, "%i ", caseInfo[caseNum].cDiab = atoi(token)); } break;
            case 16: { fprintf(allDisFptr, "%i ", caseInfo[caseNum].cInsulin = atoi(token)); } break;
            case 17: { fprintf(allDisFptr, "%i ", caseInfo[caseNum].cAgeOnsetDiab = atoi(token)); } break;
            case 18: { fprintf(allDisFptr, "%i ", caseInfo[caseNum].cYrOnsetDiab = atoi(token)); } break;
            case 19: { fprintf(allDisFptr, "%i", caseInfo[caseNum].steroidDiab = atoi(token)); } break;
            default: { fprintf(debFile, "\nERROR more case disinfo than slots"); } break;
          }
        }
      }
    return caseNum;
}


int processCaseDiseaseRecord(int caseNum) {
    // Actually do transformations here similar transformations as in questionnaire make categoreisetc.
    caseInfo[caseNum].returnedQuest = 0;// initialize return quest to 0;
    caseInfo[caseNum].cAgeDX1cat= calculateAgeCatAtReferenceDate(caseInfo[caseNum].ageDX1, caseInfo[caseNum].cStudyID, caseInfo[caseNum].cNmbr);
    if (caseInfo[caseNum].ageDX1 < 18 || caseInfo[caseNum].ageDX1 > 74)
        fprintf(debFile,"\nERROR MISSING AGE AT DX INFO IN DISEASE DATABASE STUDYID %i DISEASE %i DATE %s",
                                                caseInfo[caseNum].cStudyID, caseInfo[caseNum].DX1, caseInfo[caseNum].DXD1);

    if (strcmp(caseInfo[caseNum].cBirthDate,"99/99/99") != 0) {
        caseInfo[caseNum].cBirthYr = atoi(strrchr(caseInfo[caseNum].cBirthDate, '/')+1);
        caseInfo[caseNum].cBirthMonth = atoi(caseInfo[caseNum].cBirthDate);
    }
```

```
else
   fprintf(debFile,"\nERROR MISSING BIRTH YEAR INFO IN DISEASE DATABASE STUDYID %i DISEASE %i DATE %s",
            caseInfo[caseNum].cStudyID, caseInfo[caseNum].DX1, caseInfo[caseNum].DXD1);

caseInfo[caseNum].cYrDX1 = atoi(strrchr(caseInfo[caseNum].DXD1, '/')+1);
caseInfo[caseNum].cYrDX1cat = caseInfo[caseNum].cYrDX1 - 83;// 83 is base reference year, 99 if missing
if (caseInfo[caseNum].cYrDX1cat > 8) { // no DXD try DXP,
   caseInfo[caseNum].validDXD = 0; // Set invalid (FALSE) so can analyize only definite CTD Dx as an option
   fprintf(debFile,"\nERROR MISSING DXD REF YEAR INFO IN DISEASE DATABASE STUDYID %i DISEASE %i DATE %s",
                        caseInfo[caseNum].cStudyID, caseInfo[caseNum].DX1, caseInfo[caseNum].DXD1);
  caseInfo[caseNum].cYrDX1cat = atoi(strrchr(caseInfo[caseNum].DXP1, '/')+1) - 83;   // 83 is base reference year
  if (caseInfo[caseNum].cYrDX1cat > 8) { // NO DXD DXP, TRY AGE DX - BIRTH YEAR
      fprintf(debFile,"\nERROR MISSING BOTH DXD & DXP REF YEAR INFO IN DISEASE DATABASE STUDYID %i DISEASE %i DATE %s",
                      caseInfo[caseNum].cStudyID, caseInfo[caseNum].DX1, caseInfo[caseNum].DXD1);
      if (caseInfo[caseNum].cBirthYr != 99) {
         caseInfo[caseNum].cYrDX1cat = (caseInfo[caseNum].ageDX1 + caseInfo[caseNum].cBirthYr) - 83;
         if (caseInfo[caseNum].cYrDX1cat > 8 || caseInfo[caseNum].cYrDX1cat < 0)
            fprintf(debFile,"\nERROR MISSING BOTH DXD & DXP REF YEAR, AND BIRTH YEAR INFO IS WRONG %i IN DISEASE DATABASE STUDYID
            %i DISEASE %i DATE %s", caseInfo[caseNum].cBirthYr, caseInfo[caseNum].cStudyID, caseInfo[caseNum].DX1, caseInfo[caseNum].DXD1);
      }
      else fprintf(debFile,"\nERROR MISSING DXD & DXP REF YEAR & BIRTH YEAR INFO %i IN DISEASE DATABASE STUDYID %i DISEASE %i
            DATE %s", caseInfo[caseNum].cBirthYr, caseInfo[caseNum].cStudyID, caseInfo[caseNum].DX1, caseInfo[caseNum].DXD1);
   }
} else
   caseInfo[caseNum].validDXD = 1; // Set invalid (FALSE) so can analyize only definite CTD Dx as an option
caseInfo[caseNum].DX2 = -1;
allIDcasesAgeDXyrDX[caseInfo[caseNum].cAgeDX1cat][caseInfo[caseNum].cYrDX1cat]++; // matrix of all cases identified
// need to map DX1 code to same used
switch (caseInfo[caseNum].DX1) {
case 1:
   { caseInfo[caseNum].DX1 = 1; } break;   // PSS not really need to change
case 2:
   { caseInfo[caseNum].DX1 = 0; } break;   // SLE 710.0 cade same way quest db is.
case 3:
   { caseInfo[caseNum].DX1 = 2; } break;   // SS 710.2 cade same way quest db is.
case 6:
   { caseInfo[caseNum].DX1 = 3; } break;   // DM 710.0 PM 710.4.
```

```
    case 4:
       { caseInfo[caseNum].DX1 = 4; } break;   // SLE 710.0 cade same way quest db is.
    default:
       { fprintf(debFile, "\nERROR ILLEGAL Disease code %i studyID = %i", caseInfo[caseNum].DX1, caseInfo[caseNum].cStudyID);} break;
    // UCTD should not be in the database.
    }


    // also need to process race data  Now done in read in the case quest data
    switch (caseInfo[caseNum].cRace) {
    case 1: { //caseInfo[caseNum].cRace = 0;
              //raceDiseaseTable[caseInfo[caseNum].DX1][caseInfo[caseNum].cRace] += 1;
            } break; // white
    case 2: { //caseInfo[caseNum].cRace = 1;
              //raceDiseaseTable[caseInfo[caseNum].DX1][caseInfo[caseNum].cRace] += 1;
            } break; // black
    case 3: case 4: case 5: case 7: case 8: case 9:
            { //raceDiseaseTable[caseInfo[caseNum].DX1][caseInfo[caseNum].cRace] += 1;
              //caseInfo[caseNum].cRace = 2;
            } break; // all others
    default:
       {fprintf(debFile,"\nERROR PROCESSING INCIDENCE ILLEGAL RACE value %i studyID %i",caseInfo[caseNum].cRace,caseInfo[caseNum].cStudyID); }
    }
    return 0;
}


int readCaseDiseaseData() {
    // reads in the 5 disease files SLE, PSS, SS, MCTD, PMDM
    // data goes into an array of each case
    // will make a case table with important info on each case will also be used by clean data to generate statistical analysis data
    int totalNumCases = 0;

    FILE *caseFptr;

    if ((caseFptr = fopen("C:/research/rfctdata/casectrl/incidence/allCases.asc", "r" )) == NULL ) {
       printf( "\nError the file %s was not opened\n", "C:/research/rfctdata/casectrl/incidence/allCases.asc" );
```

```
        exit(-1);
    }
    else
        printf( "\nThe file %s was opened", "C:/research/rfctdata/casectrl/incidence/allCases.asc" );

    if ((allDisFptr = fopen("C:/research/rfctdata/casectrl/incidence/allCases.lst", "w" )) == NULL ) {// list file for cases
        printf( "\nError the file %s was not opened\n", "allCases.lst\0");
        exit(-1);
    }
    else
        printf( "\nThe file %s\n was opened\n", "allCases.lst\0");

    // Read in file that is concatenation of all diseases
    totalNumCases = getCaseDiseaseData(caseFptr, "SLE\0");
    fclose(caseFptr);

    printf("\n Total Number of DISEASE CASES read in = %i", totalNumCases);
    fclose(allDisFptr);

        return 0;
}

int printRaceDiseaseInfo() {
    // will print out the disease race table
    int i; // case index

    for (i=0; i< caseLinesRead; i++) {
      raceDiseaseTable[caseInfo[i].DX1][caseInfo[i].cRace] += 1; // update disease race table

        // also need to process race data
        switch (caseInfo[i].cRace) {
            case 1: { caseInfo[i].cRace = 0;
                    } break; // white
            case 2: { caseInfo[i].cRace = 1;
                    } break; // black
            case 3: { caseInfo[i].cRace = 2;
                    } break; // Am. Ind  / eskimo
```

```
            case 4: { caseInfo[i].cRace = 3;
                    } break; // asian
            case 5: { caseInfo[i].cRace = 4;
                    } break; // hispanic
            case 7: case 8: case 9:
                    { caseInfo[i].cRace = 5;
                    } break; // all others
            default:
                    {fprintf(debFile,"\nERROR PROCESSING INCIDENCE ILLEGAL RACE value %i studyID
%i",caseInfo[caseNum].cRace,caseInfo[caseNum].cStudyID); }
        }
    }

    return 0;
}


int findCaseIndex(int ID) {
    // This used to match returned case info with chart abstraction information
    // Will return index to case number that matched ID, if no match will return -1
    // nothing fancy because don't have it sorted, might be better but should not be too slow with such a small number of cases
    int index;
    int found = -1;

    for (index = 1; index <= caseLinesRead; index++) {
        if (caseInfo[index].cStudyID == ID) {
            found = index;
            caseInfo[index].returnedQuest = 1;
            break;
        }
    }
    if (found == -1)
        fprintf(debFile, "\nERROR can not find studyID = %i in findCAseIndex", ID);

    return found;
}
```

```
int calculateAgeSpecificRaceIncidenceRates() {
   // will calculate with avg. KC pop'n with data from Census for white,black,amind asian,hispanic,other
   // will calculate wide age limits
   int const numRaces = 6;
   int const numDiseases = 5;
   int numCases = caseLinesRead;
   int disType, caseIndx;
   int ageDXyrDXMatrix[14][12]; // Will be used 25 times for calculation of incidence rates by Disease x Race
   int incidenceDiseaseMatrix[14][12];
   /* One for each disease, wblaho90 is the coressponding denominator pop'n data
         W  B  AI A  H  O
    18-29
    30-39
    40-49
    50-59
    60-69
    70-74
  */

   for (disType = 0; disType < numDiseases; disType++) {
      initializeMatrix(incidenceDiseaseMatrix);// initialize the Incidence Disease Matrix
      initializeMatrix(ageDXyrDXMatrix);  // ageDXyrDXMatrix reused 5 times for each disease
      for (caseIndx = 1; caseIndx <= caseLinesRead; caseIndx++) {
         if (caseInfo[caseIndx].DX1 == disType) {
               ageDXyrDXMatrix[caseInfo[caseIndx].cAgeDX1cat][caseInfo[caseIndx].cRace]++;
               ageDXyrDXMatrix[caseInfo[caseIndx].cAgeDX1cat][6]++; // row sum # cases across races
               ageDXyrDXMatrix[11][caseInfo[caseIndx].cRace]++;    // col sum # cases within race
               ageDXyrDXMatrix[11][6]++;                           // grand total of incident cases for this race disease
         }
      }

      calculateWideRaceAgeDiseaseSpecificIncidenceMatrix(wideFlag, disType, ageDXyrDXMatrix);

   }

   calculateDiseaseIncidenceMatrix(wideFlag, disType, incidenceDiseaseMatrix);
```

```
    return 0;
}

int calculateIncidenceRates() {
    // Will run throught the case table 15 times to make lots of incidence tables

    int const numRaces = 6;
    int const numDiseases = 5;
    int numCases = caseLinesRead;
    int raceIndx, disType, caseIndx;
    int ageDXyrDXMatrix[14][12]; // Will be used 25 times for calculation of incidence rates by Disease x Race
    int incidenceDiseaseMatrix[14][12];

    for (disType = 0; disType < numDiseases; disType++) {
        initializeMatrix(incidenceDiseaseMatrix);// initialize the Incidence Disease Matrix
        for (raceIndx = 0; raceIndx < numRaces; raceIndx++) {
            initializeMatrix(ageDXyrDXMatrix);  // ageDXyrDXMatrix reused 15 times for race disease combos
            for (caseIndx = 1; caseIndx <= caseLinesRead; caseIndx++) {
                if (caseInfo[caseIndx].cRace == raceIndx && (caseInfo[caseIndx].DX1 == disType )) { //|| caseInfo[caseIndx].DX2 == disType)) {
                    ageDXyrDXMatrix[caseInfo[caseIndx].cAgeDX1cat][caseInfo[caseIndx].cYrDX1cat]++; // increment count in cell
                    ageDXyrDXMatrix[caseInfo[caseIndx].cAgeDX1cat][9]++; // row sum
                    ageDXyrDXMatrix[11][caseInfo[caseIndx].cYrDX1cat]++;  // col sum
                    ageDXyrDXMatrix[11][9]++;                           // grand total of incident cases for this race disease
                }
            }
            calculateRaceDiseaseIncidenceMatrix(0, raceIndx,disType, ageDXyrDXMatrix);
            calculateRaceDiseaseIncidenceMatrix(wideFlag, raceIndx, disType, ageDXyrDXMatrix);
            addToDiseaseMatrix(ageDXyrDXMatrix, incidenceDiseaseMatrix);
        }
        calculateDiseaseIncidenceMatrix(0,disType, incidenceDiseaseMatrix);
        calculateDiseaseIncidenceMatrix(wideFlag, disType, incidenceDiseaseMatrix);
    }
    return 0;
}

int initializeMatrix(int mat[14][12]) {
    int i, j;
```

```
    for (i = 0; i <= 12; i++)
        for (j = 0; j <= 11; j++)
            mat[i][j] = 0;
    return 0;
}


int printDiseaseTableLabel(int diseaseC) {
    // will print label

    switch (diseaseC) {
    case 0:
        fprintf(debFile, "\n\n\t\t DISEASE = SLE\t\t"); break;
    case 1:
        fprintf(debFile, "\n\n\t\t DISEASE = PSS\t\t"); break;
    case 2:
        fprintf(debFile, "\n\n\t\t DISEASE = SS\t\t"); break;
    case 3:
        fprintf(debFile, "\n\n\t\t DISEASE = PMDM\t\t"); break;
    case 4:
        fprintf(debFile, "\n\n\t\t DISEASE = MCTD\t\t"); break;
    }

    return 0;
}


int printTableLabel(int raceC, int diseaseC) {
    // will print label

    switch (diseaseC) {
    case 0:
        fprintf(debFile, "\n\n\t\t DISEASE = SLE\t\t"); break;
    case 1:
        fprintf(debFile, "\n\n\t\t DISEASE = PSS\t\t"); break;
    case 2:
        fprintf(debFile, "\n\n\t\t DISEASE = SS\t\t"); break;
```

```
    case 3:
        fprintf(debFile, "\n\n\t\t DISEASE = PMDM\t\t"); break;
    case 4:
        fprintf(debFile, "\n\n\t\t DISEASE = MCTD\t\t"); break;
    }
    switch (raceC) {
    case 0:
        fprintf(debFile, "\t\t RACE = White"); break;
    case 1:
        fprintf(debFile, "\t\t RACE = Black"); break;
    case 2:
        fprintf(debFile, "\t\t RACE = Am. Ind / Eskimo"); break;
    case 3:
        fprintf(debFile, "\t\t RACE = Asian"); break;
    case 4:
        fprintf(debFile, "\t\t RACE = Hispanic"); break;
    case 5:
        fprintf(debFile, "\t\t RACE = Other"); break;
    }
    return 0;
}

int printPopMatrix(long popMatrix[14][12]) {
    // will print wide population matrix used to calculate incidence rates
    int row, col, rowSum;
    int wideFlag = 1; // just print the wide pop'n person years
    fprintf(debFile, "\nwide pop'n person years");
    fprintf(debFile, "\nAGE\t\twhite\tblack\tAmInd\tAsian  Hispanic  Othere");

    if (wideFlag == 0) {
        for (row = 0; row <= 11; row++) {
            rowSum = 0;
            if (row ==0)
                fprintf(debFile, "\n%i-%i\t", 18, 24);
            else if (row == 11)
                fprintf(debFile, "\n%Total\t");
            else
```

```
            fprintf(debFile, "\n%i-%i\t", 25 + ((row-1) * 5), 29 + ((row-1) * 5));
        for (col = 0; col <= 6; col++) {
            if (col == 6)
                fprintf(debFile,"%i\t", popMatrix[row][col]);
            else
                fprintf(debFile,"%i\t", popMatrix[row][col]);
        //  rowSum += KCpopulation8391[row][col];
        }
    }
  } else {
    for (row = 0; row <= 6; row++) {
        rowSum = 0;
        if (row ==0)
            fprintf(debFile, "\n%i-%i\t", 18, 29);
        else if (row == 5)
            fprintf(debFile, "\n%i-%i\t", 70, 74);
        else if (row == 6)
            fprintf(debFile, "\n%Total\t");
        else
            fprintf(debFile, "\n%i-%i\t", 30 + ((row-1) * 10), 39 + ((row-1) * 10));
        for (col = 0; col <= 6; col++) {
            if (col == 6)
                fprintf(debFile,"%i\t", popMatrix[row][col]);
            else
                fprintf(debFile,"%i\t", popMatrix[row][col]);
     //     rowSum += KCpopulation8391[row][col];
        }
    }

  }
  return 0;
}

int printAgeRaceIncidenceMatrix(int wideFlag, int diseaseCat, float incidMat[14][12]) {
  // willprint out the race and diasese header if race == -1 then just print out the disease matrix
  int row, col, rowSum;
  printDiseaseTableLabel(diseaseCat);
```

```
fprintf(debFile, "\nAGE\t\twhite\tblack\tAmInd\tAsian  Hispanic  Other\tAverage");

if (wideFlag == 0) {
    for (row = 0; row <= 11; row++) {
        rowSum = 0;
        if (row ==0)
            fprintf(debFile, "\n%i-%i\t", 18, 24);
        else if (row == 11)
            fprintf(debFile, "\n%Total\t");
        else
            fprintf(debFile, "\n%i-%i\t", 25 + ((row-1) * 5), 29 + ((row-1) * 5));
        for (col = 0; col <= 6; col++) {
            if (col == 6)
                fprintf(debFile,"%.3f\t", incidMat[row][col]);
            else
                fprintf(debFile,"%.3f\t", incidMat[row][col]);
            rowSum += KCpopulation8391[row][col];
        }
    }
} else {
    for (row = 0; row <= 6; row++) {
        rowSum = 0;
        if (row ==0)
            fprintf(debFile, "\n%i-%i\t", 18, 29);
        else if (row == 5)
            fprintf(debFile, "\n%i-%i\t", 70, 74);
        else if (row == 6)
            fprintf(debFile, "\n%Total\t");
        else
            fprintf(debFile, "\n%i-%i\t", 30 + ((row-1) * 10), 39 + ((row-1) * 10));
        for (col = 0; col <= 6; col++) {
            if (col == 9)
                fprintf(debFile,"%.3f\t", incidMat[row][col]);
            else
                fprintf(debFile,"%.3f\t", incidMat[row][col]);
            rowSum += KCpopulation8391[row][col];
        }
```

```
        }

    }
    return 0;
}

int printIncidenceMatrix(int wideFlag, int raceCat, int diseaseCat, float incidMat[14][12]) {
    // willprint out the race and diasese header if race == -1 then just print out the disease matrix
    int row, col, rowSum;
    printTableLabel(raceCat, diseaseCat);
    fprintf(debFile, "\nAGE\t\t1983   1984 1985 1986 1987 1988 1989 1990 1991 Average");

    if (wideFlag == 0) {
        for (row = 0; row <= 11; row++) {
            rowSum = 0;
            if (row ==0)
                fprintf(debFile, "\n%i-%i\t", 18, 24);
            else if (row == 11)
                fprintf(debFile, "\n%Total\t");
            else
                fprintf(debFile, "\n%i-%i\t", 25 + ((row-1) * 5), 29 + ((row-1) * 5));
            for (col = 0; col <= 9; col++) {
                if (col == 9)
                    fprintf(debFile,"%.3f\t", (incidMat[row][col] / 9));
                else
                    fprintf(debFile,"%.3f\t", incidMat[row][col]);
                rowSum += KCpopulation8391[row][col];
            }
        }
    } else {
        for (row = 0; row <= 6; row++) {
            rowSum = 0;
            if (row ==0)
                fprintf(debFile, "\n%i-%i\t", 18, 29);
            else if (row == 5)
                fprintf(debFile, "\n%i-%i\t", 70, 74);
            else if (row == 6)
```

```
            fprintf(debFile, "\n%Total\t");
        else
            fprintf(debFile, "\n%i-%i\t", 30 + ((row-1) * 10), 39 + ((row-1) * 10));
        for (col = 0; col <= 9; col++) {
            if (col == 9)
                fprintf(debFile,"%.3f\t", (incidMat[row][col] / 9));
            else
                fprintf(debFile,"%.3f\t", incidMat[row][col]);
            rowSum += KCpopulation8391[row][col];
        }
    }

  }
  return 0;
}


int calculateRaceDiseaseIncidenceMatrix(int wideType, int raceIndex, int diseaseType, int ageDXyrDXMat[14][12]) {
  // will also print out the calculated disease matrix
  float incidenceMatrix[14][12];
  int i, j, w;
  // 6/24/97 modify to use row sume *0.95 / race catnum
  if (wideType == 0) {
    for (i = 0; i <= 11; i++)
      for (j = 0; j <= 9; j++) {
        switch (raceIndex) {
          case 0: // white
            incidenceMatrix[i][j] = ((float) ageDXyrDXMat[i][j] * 100000) / (float) whiteKCpop8391[i][j]; break;
          case 1: // black
            incidenceMatrix[i][j] = ((float) ageDXyrDXMat[i][j] * 100000) / (float) blackKCpop8391[i][j]; break;
          case 2: // other
            incidenceMatrix[i][j] = ((float) ageDXyrDXMat[i][j] * 100000) / (float) otherKCpop8391[i][j]; break;
        }
      }
  } else { // wide type i.e. age 18-29, 30-39,...
    for (i = 0, w = 0; i <= 12; i+=2, w++) {
      if (i == 12)  // ( 0, 2, 4, 6, 8) = 2 years at a time , 10=70-74, 11=TOTAL row
```

```
            i = 11;
        for (j = 0; j <= 9; j++) {
            switch (raceIndex) {
                case 0: // white
                    if (i == 10 || i == 11)
                        incidenceMatrix[w][j] = ((float) ageDXyrDXMat[i][j] * 100000) / (float) whiteKCpop8391[i][j];
                    else
                        incidenceMatrix[w][j] = (((float) ageDXyrDXMat[i][j] + (float) ageDXyrDXMat[i+1][j]) * 100000)  /
                                                ((float) whiteKCpop8391[i][j] + (float) whiteKCpop8391[i+1][j]);
                    break;
                case 1: // black
                    if (i == 10 || i == 11)
                        incidenceMatrix[w][j] = ((float) ageDXyrDXMat[i][j] * 100000) / (float) blackKCpop8391[i][j];
                    else
                        incidenceMatrix[w][j] = (((float) ageDXyrDXMat[i][j] + (float) ageDXyrDXMat[i+1][j]) * 100000) /
                                                ((float) blackKCpop8391[i][j] + (float) blackKCpop8391[i+1][j]);
                    break;
                case 2: // other
                    if (i ==10 || i == 11)
                        incidenceMatrix[w][j] = ((float) ageDXyrDXMat[i][j] * 100000) / (float) otherKCpop8391[i][j];
                    else
                        incidenceMatrix[w][j] = (((float) ageDXyrDXMat[i][j] + (float) ageDXyrDXMat[i+1][j]) * 100000) /
                                                ((float) otherKCpop8391[i][j] + (float) otherKCpop8391[i+1][j]);
                    break;
            }
        }
    }
}
prtAgeDXyrDXMat(ageDXyrDXMat, "INCIDENCE MATRIX\n");
printIncidenceMatrix(wideType, raceIndex, diseaseType, incidenceMatrix); // now print the matrix out

return 0;
}
```

```
int calculateWideRaceAgeDiseaseSpecificIncidenceMatrix(int wideType, int diseaseType, int ageDXyrDXMat[14][12]) {
   // will also print out the calculated disease matrix
   float incidenceMatrix[14][12];
   long popMatrix[14][12];
   int i, j, w; //wide pop'n person years;
   // 6/24/97 modify to use row sume *0.95 / race catnum
   if (wideType == 0) {
      for (i = 0; i <= 11; i++) {
         for (j = 0; j <= 6; j++)
            incidenceMatrix[i][j] = ((float) ageDXyrDXMat[i][j] * 100000) / (float) wbiaho90KCpop[i][j]; break;
      }
   } else
      { // wide type i.e. age 18-29, 30-39,...
        // print title
        for (i = 0, w = 0; i <= 12; i+=2, w++) {
           if (i == 12)  // ( 0, 2, 4, 6, 8) = 2 years at a time , 10=70-74, 11=TOTAL row
              i = 11;
           for (j = 0; j <= 6; j++) {
              if (i == 10 || i == 11) {
                 incidenceMatrix[w][j] = ((float) ageDXyrDXMat[i][j] * 100000) / (float) (9 * wbiaho90KCpop[i][j]) ;
                 popMatrix[w][j] = 9 * wbiaho90KCpop[i][j];
              } else {
                 incidenceMatrix[w][j] = (((float) ageDXyrDXMat[i][j] + (float) ageDXyrDXMat[i+1][j]) * 100000) / (float) (9 * (wbiaho90KCpop[i][j] +
wbiaho90KCpop[i+1][j]));
                 popMatrix[w][j] = 9 * (wbiaho90KCpop[i][j] + wbiaho90KCpop[i+1][j]);
              }
           }
        }
      }

   printPopMatrix(popMatrix);
   prtAgeRaceAvgMat(ageDXyrDXMat, diseaseType, "Race / AGE Number of Cases  Disease =");
   printAgeRaceIncidenceMatrix(wideFlag, diseaseType, incidenceMatrix);

   return 0;
}
```

```
int calculateDiseaseIncidenceMatrix(int wideFlag, int diseaseType, int incidenceDisMatrix[14][12]) {
   // will also print out the calculated disease matrix
   float incidenceMatrix[14][12];
   int raceIndex = -1;
   int i, j, w;

   if (wideFlag == 0) {  // wide flag => age cats 18-29, 30-39, 40-49, 50-59, 60-69, 70+
      for (i = 0; i <= 11; i++)
         for (j = 0; j <= 6; j++)
            incidenceMatrix[i][j] = ((float) incidenceDisMatrix[i][j] * 100000) / (float) KCpopulation8391[i][j];
   } else {  // compressed age cats i.e. 18-29, 30-39,...,60-69,70+
      for (i = 0, w = 0; i <= 12; i+=2, w++) {
         if (i == 12)  // ( 0, 2, 4, 6, 8) = 2 years at a time , 10=70-74, 11=TOTAL row
            i = 11;
         for (j = 0; j <= 11; j++) {
            if (i == 10 || i == 11)
               incidenceMatrix[w][j] = ((float) incidenceDisMatrix[i][j] * 100000) / (float) KCpopulation8391[i][j];
            else
               incidenceMatrix[w][j] = (((float) incidenceDisMatrix[i][j] + (float)  incidenceDisMatrix[i+1][j]) * 100000) / ((float) KCpopulation8391[i][j] + (float)
KCpopulation8391[i+1][j]);
         }
      }
   }
   prtAgeDXyrDXMat(incidenceDisMatrix, "INCIDENCE MATRIX\n");
   printIncidenceMatrix(wideFlag, -1, diseaseType, incidenceMatrix);

   return 0;
}
```

```
int addToDiseaseMatrix(int ageYrDXmat[14][12], int incidDisMat[14][12]) {
    int i, j;
    for (i = 0; i <= 11; i++)
        for (j = 0; j <= 9; j++)
            incidDisMat[i][j] += ageYrDXmat[i][j];
    return 0;
}
```

APPENDIX D: INFORMED CONSENT FORM

University of Washington
CONSENT FORM

Name of Study: Risk Factors for Connective Tissue Diseases Study

Investigators:

Mr. William Teel, Ph.D. Candidate, Department of Epidemiology, University of Washington. Phone: 616-3112
Dr. Thomas Koepsell, Chairman, Department of Epidemiology, University of Washington. Phone: 543-8830
Dr. Janet R. Daling, Professor, Department of Epidemiology, University of Washington. Phone: 467-4630
Dr. Carin Dugowson, Associate Professor, Department of Medicine, University of Washington. Phone: 467-2881
Dr. Bruce Psaty, Associate Professor, Department of Epidemiology, University of Washington.
Dr. Richard Kronmal, Professor, Department of Biostatistics, University of Washington.

Purpose and Benefits:

The purpose of this study is to investigate the possible association between certain connective tissue diseases (Scleroderma, Systemic Lupus, Mixed Connective Tissue Disease, Sjogren's Syndrome, Polymyositis / Dermatomyositis) and various possible risk factors, such as silicone breast implants, diabetes, or hair dye. We hope this study will contribute to better knowledge about women's health, and to better understanding of the risk factors for connective tissue diseases.

Procedures:

If you agree to take part in this study, you will be asked to fill out a short questionnaire about your health history that is also included in the package of material you have received with this form. The questionnaire should take less than 20 minutes of your time to fill out. It will include questions about your health habits, pregnancy history (including whether any pregnancies have ended in miscarriage or abortion), diabetes, silicone breast implants, and also such things as your education and occupation. You are free to skip any questions you do not wish to answer. We will also consult the medical records maintained by your rheumatologist to learn some details of your connective tissue disease history. We will look for information about when your symptoms started, what they were, and for the results of relevant lab tests, so that we can get an overall picture of your illness. We will look at your medical records once, and we will destroy any identifying information at the end of the study (within one year).

Risk, Stress, and Discomfort:

Completing the questionnaire involves no physical risk or discomfort. You may feel some questions are personal, and may feel discomfort or embarrassment. You are free not to answer any question you do not wish to answer. You are free not to participate and to withdraw from the study at any time without penalty and without loss of benefits to which you may otherwise be entitled.

Other Information:

1. This project is part of a Ph.D. dissertation.
2. The only people who will have access to study data about you will be the investigators named above. None of the participants will be identified in any study reports.
3. To protect the privacy of participants, all identifying information (such as names and addresses) will be destroyed at the end of the study. The rest of the study data will be retained indefinitely for research use.
4. You may ask any questions you wish at any time. If you live outside the Seattle area, you may call the investigators collect at one of the above telephone numbers.
5. All information will be kept confidential as provided by law.

B/N Steel

Signature of Investigator

July 25, 1997

Date

Subject' s Statement:

The study described above has been understood by me. I voluntarily consent to participate in this research and agree to allow members of the research staff to review my medical records for connective tissue disease history if necessary. I understand that any questions I may have about the research or about subjects' rights will be answered by one of the investigators listed above. A copy of this Consent Form is provided to me for my records.

Signature of Subject

Date

Copies to: Subject
Investigator's File

APPENDIX E CHART ABSTRACTION FORM

Abstraction Date (/ /) STUDY ID: _____

Chart Abstraction Form
Risk Factors for Connective Tissue Diseases Study

Primary Sjogren's Syndrome: (Sjogren's + if DX on chart and any 3 criteria)

SX Start:___/___/___ **1st MD Visit for SX:**___/___/___ **1st Rheum :**___/___/___ **DXD:**___/___/___ **DXP:**___/___/___

Clinical Findings	Date	Present	Source	Result	Start SX	Date
Ocular symptoms:	(/ /)	(+ I –)	(hx_____)		_____	(/ /)
Oral symptoms:	(/ /)	(+ I –)	(hx_____)		_____	(/ /)
Salivary gland biopsy:	(/ /)	(+ I –)	(N I L)			
Rose Bengal Test	(/ /)	(+ I –)	(N I L)	(Positive if score ≥ 4)		
Schirmer Test:	(/ /)	(+ I –)	(N I L)	L____ mm R____ mm _____ minutes		
Salivary Scintigraphy I Parotid Sialography I Unstimulated Salivary Flow	(/ /)	(+ I –)	(N I L)			
AntiNuclear Antibody:	(/ /)	(+ I –)	(N I L)	Titer 1:____ (___)	(Spec Homo Rim Nucl Cent)	
Rheumatoid Factor:	(/ /)	(+ I –)	(N I L)	Titer 1:____ (___)		
SS-A(Ro):	(/ /)	(+ I –)	(N I L)	Titer 1:____ (___)		
SS-B(La):	(/ /)	(+ I –)	(N I L)	Titer 1:____ (___)		

Systemic Lupus Erythematosus: (SLE+ if MD DXD I DXP & ≥ 3 criteria present serially or simultaneously, during any period of observation)

SX Start:___/___/___ **1st MD Visit for SX:**___/___/___ **1st Rheum :**___/___/___ **DXD:**___/___/___ **DXP:**___/___/___

Clinical Findings	Date	Present	Source	Result	Start SX	Date
Malar Rash	(/ /)	(+ I –)	(hx_____)			
Discoid Rash	(/ /)	(+ I –)	(hx_____)		Arthralgia	(/ /)
Photosensitivity	(/ /)	(+ I –)	(hx_____)		Myalgia	(/ /)
Oral ulcers	(/ /)	(+ I –)	(hx_____)		Raynaud's	(/ /)
Arthritis	(/ /)	(+ I –)	(hx_____)		Fatigue	(/ /)
Serositis					_____	(/ /)
(a) Pleuritis	(/ /)	(+ I –)	(hx_____)		_____	(/ /)
(b) Pericarditis	(/ /)	(+ I –)	(hx_____)			
Neurologic Disorder						
(a) Seizures	(/ /)	(+ I –)	(hx_____)			
(b) Psychosis	(/ /)	(+ I –)	(hx_____)			
Renal Disorder						
(a) Proteinuria	(/ /)	(+ I –)	(N I L)	Count:_____ (___)		
(b) Cellular casts	(/ /)	(+ I –)	(N I L)	Count:_____		
Hematologic Disorder						
(a) Hemolytic anemia	(/ /)	(+ I –)	(N I L)	Count:_____ Test:_____		
(b) Leukopenia	(/ /)	(+ I –)	(N I L)	Count:_____ (/ /) Count:_____		
(c) Lymphopenia	(/ /)	(+ I –)	(N I L)	Count:_____ (/ /) Count:_____		
(d) Thrombocytopenia	(/ /)	(+ I –)	(N I L)	Count:_____ (/ /) Count:_____		
Immunologic Disorder						
(a) Positive LE cell	(/ /)	(+ I –)	(N I L)			
(b) Anti-dsDNA:	(/ /)	(+ I –)	(N I L)	Titer:1:_____ (___)		
(c) Anti-Sm:	(/ /)	(+ I –)	(N I L)	Titer:1:_____ (___)		
(d) False + syphilis test	(/ /)	(+ I –)	(N I L)	(/ /) (VDRL, RPR, FTA-ABS, MHA-TP, HATTS)		
Antinuclear Antibody	(/ /)	(+ I –)	(N I L)	Titer 1:____ (___) (Spec Homo Rim Nucl Cent)		

Lab Tests	Date	Present	Source	Results
DRUG INDUCED	(/ /)	(+ I –)	(N I L)	drug: _____
ESR	(/ /)	(+ I –)	(N I L)	mm/hr: _____
anti-ssDNA	(/ /)	(+ I –)	(N I L)	Titer 1: _____ (___)
SS-A(Ro)	(/ /)	(+ I –)	(N I L)	Titer 1: _____ (___)
SS-B(La)	(/ /)	(+ I –)	(N I L)	Titer 1: _____ (___)
anti-RNP	(/ /)	(+ I –)	(N I L)	Titer 1: _____ (___)
Lupus anticoagulant Test	(/ /)	(+ I –)	(N I L)	PTT1: _____ (___) PTT1:1: _____ secs.
Russel Viper Venom Test	(/ /)	(+ I –)	(N I L)	Titer 1: _____ (___)
Antcardiolipin Antibodies	(/ /)	(+ I –)	(N I L)	Titer 1: _____ (___)
Antiphospholipid Antibodies	(/ /)	(+ I –)	(N I L)	Titer 1: _____ (___)

Birth Date:_____ **Race:**_____ **Date Last Seen:**_____ **Other MD** ___ **MD Permission to Contact Patient?** (Y I N)

LAST NAME: _____ **FIRST:** _____ **MI:** ___ **MS:** _____ **SSN:** ___-___-___

ADDRESS: _____ STUDY ID: _____

Next of Kin: _____

Diabetes (Y I N I NR) AGE of ONSET:_____ Date of ONSET: _____ Insulin Dependent (Y I N)

Scleroderma: (Systemic Sclerosis) (SD* if the one major or two or more minor criteria are present)

SX Start:___/___/___ **1st MD Visit for SX:**___/___/___ **1st Rheum :**___/___/___ **DXD:**___/___/___ **DXP:**___/___/___

Clinical Findings	Date	Present	Source	Result	Start SX	Date
A. **Major Criterion**						
Proximal scleroderma:	(/ /)	(+ I –)	(hx_____)		_____	(/ /)
B. **Minor Criteria**					_____	(/ /)
Sclerodactyly:	(/ /)	(+ I –)	(hx_____)			
Digital pitting, scars, loss of substance - finger pads	(/ /)	(+ I –)	(hx_____)			
Bibasilar pulmonary fibrosis	(/ /)	(+ I –)	(hx_____)			
C. **Lab tests:** (Note: this is not part of criteria)						
AntiNuclear Antibodies	(/ /)	(+ I –)	(N I L) Titer 1:____	(___)	(Spec, Homo, Rim, Nucl Cent)	
Anti SCL-70	(/ /)	(+ I –)	(N I L) Titer: 1:_____	(___)		
Anti-Centromere Antibody	(/ /)	(+ I –)	(N I L) Titer: 1:_____	(___)		

D. **Localized Scleroderma (CREST)** (DX and 3 criteria)

Clinical Findings	Date	Present	Start SX	Date
Calcinosis:	(/ /)	(+ I –) (hx_____)	_____	(/ /)
Raynaud's Syndrome:	(/ /)	(+ I –) (hx_____)	_____	(/ /)
Esophageal dysfunction:	(/ /)	(+ I –) (hx_____)		
Sclerodactyly:	(/ /)	(+ I –) (hx_____)		
Telangiectasia:	(/ /)	(+ I –) (hx_____)		

Mixed Connective Tissue Disease (MCTD+ if positive serology and 3 or more clinical findings)

SX Start:___/___/___ **1st MD Visit for SX:**___/___/___ **1st Rheum :**___/___/___ **DXD:**___/___/___ **DXP:**___/___/___

Clinical Findings	Date	Present	Source	Start SX	Date
A. **Serologic:** Anti-RNP	(/ /)	(+ I –) (N I L) + if > 1:600	Titer: 1:____ (___)		
B. **Clinical Criteria**				_____	(/ /)
Hand Edema	(/ /)	(+ I –) (hx_____)		_____	(/ /)
Synovitis	(/ /)	(+ I –) (hx_____)	number of joints: ____ at any one time		
Myositis					
biopsy	(/ /)	(+ I –)			
CK (creatine kinase)	(/ /)	(+ I –) (N I L)	elevation: CK Level: _____ u/L		
Raynaud's Syndrome	(/ /)	(+ I –) (hx_____)			
Acrosclerosis	(/ /)	(+ I –) (hx_____)			

Undifferentiated Connective Tissue Disease (UCTD+ if all of following are present and notated on patient chart.)

SX Start:___/___/___ **1st MD Visit for SX:**___/___/___ **1st Rheum :**___/___/___ **DXD:**___/___/___ **DXP:**___/___/___

Clinical Findings	Date	Present	START SX	Date
Diagnosis in chart	(/ /)	(+ I –)	_____	(/ /)
Raynaud's Syndrome:	(/ /)	(+ I –) (hx_____)	_____	(/ /)
Arthralgias:	(/ /)	(+ I –) (hx_____)		
Myalgias:	(/ /)	(+ I –) (hx_____)		
Hand Edema:	(/ /)	(+ I –) (hx_____)		

Polymyositis / Dermatomyositis (Polymyositis clinical DX without the rash; Dermatomyositis clinical DX with the rash)

SX Start:___/___/___ **1st MD Visit for SX:**___/___/___ **1st Rheum :**___/___/___ **DXD:**___/___/___ **DXP:**___/___/___

Clinical Findings	Date	Source	Start SX	Date
Symmetric muscle weakness	(/ /)	(+ I –) (hx_____)	_____	(/ /)
Abnormal EMG	(/ /)	(+ I –) (N I L)		
Elevated muscle enzymes	(/ /)	(+ I –) (N I L) (CPK____, SGOT____, SGPT____, LDH____, aldolase____)		
Positive muscle biopsy.	(/ /)	(+ I –) (N I L)		
Dermatologic features	(/ /)	(+ I –) (hx_____)		

OTHER INFORMATION: _____

DO NOT WRITE BELOW THIS LINE

APPENDIX F QUESTIONNAIRES

STUDY ID: 1107

Patient Questionnaire
Risk Factors For Connective Tissue Diseases Study

INSTRUCTIONS: Please answer the questions below by checking boxes or printing the requeste information. Answer all of the questions in order and do not go back and change any of your response based on later questions. If you have any questions, please call the study manager, Bill Teel, at hi University of Washington office (616-3112). When you are finished, please return this questionnaire, an one signed copy of the consent form in the self-addressed, stamped envelope provided. Thank you!

Please note that many questions ask about events prior to **September, 1991**. Please answer all suc questions using only information up to this date.

1. Do you currently have a telephone in your home? 2 YES 1 NO

2. Do you currently live in King County, Washington? 2 YES 1 NO

3. Did you live in King County, Washington in September, 1991? 2 YES 1 NO

4. What was your MARITAL STATUS in September, 1991? (Please check one of the responses)

1 Single
2 Married
6 Living as married
3 Separated
4 Divorced
5 Widowed

5. What is your maximum adult height? _____ feet _____ inches

6. About how much did you weigh one year prior to September, 1991? _____ lbs.
(Note: if pregnant at that time, record your weight just prior to the pregnancy.)

7a. When did you first experience symptoms which resulted in your being seen by Dr. Carlin?

_____/_____ (month/year)

7b. What were the **first** symptoms you experienced? _____

_____

_____

PLEASE CONTINUE ON OPPOSITE SIDE

STUDY ID: 11070

8. Before September, 1991 were you ever told by a doctor that you have diabetes ?

1 NO

2 YES

8a. How old were you when diabetes was diagnosed? _____

8b. Which of the following treatments were you receiving before September, 1991?

1 None
2 Diet
3 Pills
4 Insulin shots
5 Both pills and shots
8 Other
9 Don't know

8c. If you had been using either pills or shots, how long before September, 1991 had you been using them?

Pills: ____ months ____ years

Shots: _____ months _____ years

9. How old were you when you had your first menstrual period? (AGE in YEARS) _____

10. How many pregnancies have you had that resulted in ...

a. ... a miscarriage ? (less than 20 weeks into the pregnancy) _____
b. ... an ectopic (tubal) pregnancy? _____
c. ... an induced abortion? _____
d. ... a stillbirth? (at least 20 weeks into the pregnancy) _____
e. ... a single live birth? _____
f. ... twins, triplets, or more? (at least one born live) _____
g. ... never pregnant? _____
h. ... any other outcome? Specify:_____ _____

PLEASE CONTINUE ON THE NEXT PAGE

STUDY ID: 110

11. Which statement below best describes your menstrual periods in September of 1991? (Check one)

1 I was having my regular periods.
4 I was pregnant or nursing at that time.
8 They had never started

2 My periods had begun to stop.
3 My periods had stopped permanently.

11a. How old were you when your periods stopped? _____ years old

11b. Did they stop naturally?
2 YES 1 NO

11c. If NO, how did they stop?
(CHECK ONE ANSWER)
2 Hysterectomy
4 Hormone therapy
3 Radiation therapy
8 Other: (SPECIFY)_____

12. Before September, 1991 had one or both of your ovaries been removed? (CHECK ONE)

1 No
9 Don't Know

2 Yes, one
3 Yes, both
4 Yes, but don't know if one or two
5 Yes, partial removal

12a. How old were you at the time? ____ years old.
(IF MORE THAN 1 EPISODE, ENTER AGE AT LAST)

13. Before September, 1991, had you ever used estrogens (such as Premarin), progesterone, (such as Provera), or any female hormones other than for birth control? These could include vaginal creams, suppositories, pills, injections, or skin patches.
1 NO 2 YES

14. In September, 1991 what best describes your use of oral contraceptives or birth control pills?

3 Never Used

1 Current Use
2 Past Use

14a. If Yes, for how long had you used birth control pills prior to September, 1991.
_____ months; _____ years.

PLEASE CONTINUE ON OPPOSITE SIDE

STUDY ID: 11070

15. Have you ever had breast cancer? 1 NO 2 YES

15a. What was the **date** of your first breast cancer diagnosis? ____/____ (month / year)

16. Have you ever had surgery of any type to your breasts?

1 NO 2 YES

16a. How old were you when you first had breast surgery?_____years old.

16b. What was the date of that breast surgery? _____/_____(month / year)

16c. What was the purpose of that breast surgery? (CHECK ANY THAT APPLY)

1 INcrease breast size
2 DEcrease breast size
3 Change breast shape
4 To have one breast removed
5 To have both breasts removed
9 Don't know
8 Another reason, (PLEASE SPECIFY)_____

If the purpose was to Increase size or Change breast shape please continue with question 16d. Otherwise skip to question 17.

16d. Which procedure was used? (PLEASE CHECK ONE OF THE FOLLOWING)

1 Sealed implant containing silicone
2 An implant into which silicone was injected
3 Silicone injected directly into the breast
9 Uncertain
9 Another procedure (PLEASE SPECIFY)_____

16e. Were there any complications during or after the procedure, such as leaking? (PLEASE CHECK ONE OF THE ANSWERS BELOW)

1 NO 2 YES, there was leaking
3 YES, another problem (SPECIFY):_____

PLEASE CONTINUE ON THE NEXT PAGE

STUDY ID: 1107

17. Have you ever had breast implants?

1 NO 2 YES

17a. What was the date they were first placed? _____/_____(month/year)

17b. What type of implant were they?

1 Silicone
2 Saline
9 Other (PLEASE SPECIFY) _____

17c. What was the reason for having breast implants placed?

1 INcrease breast size
2 DEcrease breast size
3 Change breast shape
4 Breast reconstruction
9 Another reason, (PLEASE SPECIFY)_____

17d. Have you had any other surgery related to breast implants since they were first inserted?

1 NO 2 YES

	purpose	mon./yr.
If YES please specify:	_____	_____
	_____	_____
	_____	_____

18. **Before** September, 1991 had you ever used any type of hair dye? 1 NO 2 YES

18a. If Yes, please fill in the chart below answering the following questions

a. What **Brand/Type(s)** of hair coloring did you use?
b. How old were you when you **First Used** it?
c. How old were you when you **Last Used** it?
d. How many **Times per Year** did you use it?
e. **How Many Minutes** did you leave it on to "work"?
f. What **Color** were you trying to achieve?

Brand or Type	Age First Used	Age Last Used	Times used **per year**	How Long (in minutes)	Color trying to achieve

PLEASE CONTINUE ON THE OPPOSITE SIDE

STUDY ID: 11070

19. What were your three main occupations **since** you were 18 years old and **prior** to September, 1991?

	INDUSTRY (What did your company make or do?)	Job Title	Years Employed
a.	_____	_____	from: ____ to:____
b.	_____	_____	from: ____ to:____
c.	_____	_____	from: ____ to:____

20. What was your occupation during September, 1991?

Industry_____ Job Title or Position_____

21. What race do you consider yourself? (Please check one):

1 White
2 Black
3 Native American / Eskimo
4 Asian
5 Hispanic
7 Refused
8 Other _____
9 Don't Know

22. What is the highest level you attended in school? (Please check one):

1 Grade school
2 High school
3 Technical school
4 College
5 Graduate school
8 Other _____
9 Don't Know

22a. How **many** years did you complete at that level? _____ years.

23. What was your combined family income in September, 1991? (Please check one):

1 less than $15,000
2 $15,000 to $30,000
3 $30,000 to $45,000
4 more than $45,000

Thank you for completing this questionnaire.
Please return it with a copy of the signed consent form in the stamped, self-addressed envelope provided.
If you would like a journal reprint of this research study when it is published, please check this box.

STUDY ID: 901002

Control Questionnaire
Risk Factors For Connective Tissue Diseases Study

INSTRUCTIONS: Please answer the questions below by checking boxes or printing the requested information. Answer all of the questions in order and do not go back and change any of your responses based on later questions. If you have any questions, please call the study manager, Bill Teel, at his University of Washington office (616-3112). When you are finished, please return this questionnaire, and one signed copy of the consent form in the self-addressed, stamped envelope provided. Thank you!

Please note that many questions ask about events prior to **July, 1991**. Please answer all such questions using only information up to this date.

1. Do you currently have a telephone in your home? 2 YES 1 NO

2. Do you currently live in King County, Washington? 2 YES 1 NO

3. Did you live in King County, Washington in **July, 1991**? 2 YES 1 NO

4. Before **July 1991**, have you ever been diagnosed by a physician as having any of the diseases below? (check any boxes that apply)

1 Systemic Sclerosis (Scleroderma)
1 Limited Sclerosis (CREST)
2 Systemic Lupus Erythematosus
2 Drug-induced Lupus
0 None
3 Sjogren's Syndrome
4 Mixed Connective Tissue Disease
5 Undifferentiated Connective Tissue Disease
6 Polymyositis or Dermatomyositis

5. What was your MARITAL STATUS in **July, 1991**? (Please check one of the responses)

1 Single
2 Married
6 Living as married
3 Separated
4 Divorced
5 Widowed

6. What is your maximum adult height? _____ feet _____ inches

7. About how much did you weigh one year prior to **July, 1991**? _____ lbs.
(Note: if pregnant at that time, record your weight just prior to the pregnancy.

PLEASE CONTINUE ON OPPOSITE SIDE

STUDY ID: 901002

8. Before **July, 1991** were you ever told by a doctor that you have diabetes ?

1 NO

2 YES

8a. How old were you when diabetes was diagnosed? _____

8b. Which of the following treatments were you receiving before **July, 1991**?

1 None
2 Diet
3 Pills
4 Insulin shots
5 Both pills and shots
8 Other
9 Don't know

8c. If you had been using either pills or shots, how long before July, 1991 had you been using them?

Pills: _____ months _____ years

Shots: _____ months _____ years

9. How old were you when you had your first menstrual period? (AGE in YEARS) _____

10. How many pregnancies have you had that resulted in ...

a. ... a miscarriage ? (less than 20 weeks into the pregnancy) _____
b. ... an ectopic (tubal) pregnancy? _____
c. ... an induced abortion? _____
d. ... a stillbirth? (at least 20 weeks into the pregnancy) _____
e. ... a single live birth? _____
f. ... twins, triplets, or more (at least one born live) _____
g. ... never pregnant _____
h. ... any other outcome? Specify:_____ _____

PLEASE CONTINUE ON THE NEXT PAGE

STUDY ID: 90100

11. Which statement below best describes your menstrual periods in **July of 1991**? (Check one)

1 I was having my regular periods.
4 I was pregnant or nursing at that time.
8 They had never started

2 My periods had begun to stop.
3 My periods had stopped permanently.

11a. How old were you when your periods stopped? _____ years old

11b. Did they stop naturally?
2 YES 1 NO

11d. If NO, how did they stop?
(CHECK ONE ANSWER)
2 Hysterectomy
4 Hormone therapy
3 Radiation therapy
8 Other: (SPECIFY)_____

12. Before **July, 1991** had one or both of your ovaries been removed? (CHECK ONE)

1 No
9 Don't Know

2 Yes, one
3 Yes, both
4 Yes, but don't know if one or two
5 Yes, partial removal

12a. How old were you at the time? ____ years old.
(IF MORE THAN 1 EPISODE, ENTER AGE AT LAST)

13. Before **July, 1991**, had you ever used estrogens (such as Premarin), progesterone, (such as Provera), or any female hormones other than for birth control? These could include vaginal creams, suppositories, pills, injections, or skin patches.
1 NO 2 YES

14. In **July, 1991** what best describes your use of oral contraceptives or birth control pills?

3 Never Used

1 Current Use
2 Past Use

14a. If Yes, for how long had you used birth control pills prior to July, 1991.
_____ months; _____ years.

PLEASE CONTINUE ON OPPOSITE SIDE

STUDY ID: 901002

15. Have you ever had breast cancer? 1 NO 2 YES

15a. When were you first diagnosed with breast cancer? ____/____ (month / year)

16. Have you ever had surgery of any type to your breasts?

1 NO 2 YES

16a. How old were you when you first had breast surgery?_____years old.

16b. What was the date of that breast surgery? _____/_____(month / year)

16c. What was the purpose of that breast surgery?

1 INcrease breast size
2 DEcrease breast size
3 Change breast shape
4 To have one breast removed
5 To have both breasts removed
9 Don't know
8 Another reason, (PLEASE SPECIFY)_____

If the purpose was to Increase size or Change breast shape please continue with question 16d. Otherwise skip to question 17.

16d. Which procedure was used? (PLEASE CHECK ONE OF THE FOLLOWING)

1 Sealed implant containing silicone
2 An implant into which silicone was injected
3 Silicone injected directly into the breast
9 Uncertain
8 Another procedure (PLEASE SPECIFY)_____

16e. Were there any complications during or after the procedure, such as leaking? (PLEASE CHECK ONE OF THE ANSWERS BELOW)

1 NO 2 YES, there was leaking
3 YES, another problem (SPECIFY):_____

PLEASE CONTINUE ON THE NEXT PAGE

STUDY ID: 90100

17. Have you ever had breast implants?

1 NO 2 YES

17a. What was the date they were first placed? _____/_____(month/year)

17b. What type of implant were they?

1 Silicone
2 Saline
9 Other, (PLEASE SPECIFY)_____

17c. What was the reason for having breast implants placed?

1 INcrease breast size
4 Breast reconstruction
3 Change breast shape
9 Other reason, (PLEASE SPECIFY)_____

17d. Have you had any other surgery related to breast implants since they were first inserted?

1 NO 2 YES

	purpose	mon./yr.
If YES please specify:	_____	_____
	_____	_____
	_____	_____

18. Before **July, 1991** had you ever used any type of hair dye?

1 NO 2 YES

18a. If Yes, please fill in the chart below answering the following questions

a. What **Brand/Type**(s) of hair coloring did you use?
b. How old were you when you **First Used** it?
c. How old were you when you **Last Used** it?
d. How many **Times per Year** did you use it?
e. **How Many Minutes** did you leave it on to "work"?
f. What **Color** were you trying to achieve?

Brand or Type	Age First Used	Age Last Used	Times used **per year**	How Long (in minutes)	Color trying to achieve

PLEASE CONTINUE ON THE OPPOSITE SIDE

STUDY ID: 901002

19. What were your three main occupations **since** you were 18 years old and **prior** to **July, 1991**?

	INDUSTRY (What did your company make or do?)	Job Title	Years Employed
a.	_____	_____	from: ____ to: ____
b.	_____	_____	from: ____ to: ____
c.	_____	_____	from: ____ to: ____

20. What was your occupation during **July, 1991**?

Industry_____ Job Title or Position_____

21. What race do you consider yourself? (Please check one):

1 White
2 Black
3 Native American / Eskimo
4 Asian
5 Hispanic
7 Refused
8 Other _____
9 Don't Know

22. What is the highest level you attended in school? (Please check one):

1 Grade school
2 High school
3 Technical school
4 College
5 Graduate school
8 Other _____
9 Don't Know

22a. How **many** years did you complete at that level? _____ years.

23. What was your combined family income in **July, 1991**? (Please check one):

1 less than $15,000
2 $15,000 to $30,000
3 $30,000 to $45,000
4 more than $45,000

Thank you for completing this questionnaire. Please return it with a copy of the signed consent form in the stamped, self-addressed envelope provided. If you would like a journal reprint of this research study when it is published, please check this box.

VITA

William Baldwin Teel

University of Washington

1997

Education

University of Michigan, Ann Arbor, Michigan	B.S.	1977	Mathematics
University of Michigan, Ann Arbor, Michigan	M.S.	1980	Computer Science
University of Washington, Seattle, Washington	Ph.D.	1997	Epidemiology

Experience

Ph.D. Candidate, Dept. of Epidemiology, University of Washington, Seattle, Washington. 10/90 to 7/97.
Taught Computer Concepts and Skills for Epidemiologists, a two credit course winter term 1991. Interested in disease model simulation, and neuroepidemiology, Attended week-long course at University of Minnesota on micro-population disease simulation. Research assistant on cardiovascular case-control study. Currently working on a meta-analysis of aspirin migraine prophylaxis and writing a graphical simulation of a classic migraine visual prodrome with scintillating scotomata. Dissertation: A Population-Based Case-Control Study of Risk factor for Connective Tissue Diseases. Have taken all required epidemiology and biostatistics courses as well as a Design of Clinical Trials course, Health Services Research Methods, Exposure Measurement, and Infectious Disease Epidemiology. Attended Summer New England Epidemiology Institute at Tufts during summer 1990. Wrote Frequency matching program (C++, ~2400 lines of code) for controls in RFCTD study. Wrote Hytime used to prepare Egret input for analysis of multiple exposure cohort studies. (C++, ~3200 lines of code).

Staff Software Engineer, Intel Development Operation, Hillsboro, Oregon. 5/86 to 5/90.
Wrote Macrosimulator for the iWarp systolic array processor, used to verify hardware design, and develop software. Capable of simulating 20 cells in multiple topologies (Mainsail,~ 35,000 lines of code). Wrote memory allocation module for the kernel software. Wrote programmers manual for the simulator.

Senior Software Engineer, Intel Scientific Computers, Beaverton, Oregon. 1/86 to 5/86.
Responsible for benchmarking LISP on hypercube and 286. Wrote a string search demo in LISP for AAAI show. Wrote short note on results.

Manager, Architectural and Logic Design CAD, Intel Corp., Hillsboro, Oregon. 9/83 to 1/86.
Built a team of 10 CAD engineers from ground zero. Coordinated development of Architectural tools, the microcode workbench, switch-level simulation tools, and PLA workbench tools. Designed ADSLIB, a software engineering source algorithm reuse librarian tool. Rewrote part of the PGS parser generator and obtained a 2x performance improvement. Participated in review of the DOD VHDL program.

President, Sisyphean Endeavors, Lake Oswego, Oregon. 1/83 to 10/90.
Designed a 2 megabyte RAM/disk-emulation board with EDC and error logging for IEEE-696 bus. Responsible for design, simulation, prototype and component purchasing. Designed and built a passive solar super-insulated home.

Senior Engineer, Logic Design CAD, Intel Corp., Hillsboro, Oregon. 5/81 to 9/83.
Created a PLA design and implementation workbench. The PLA workbench contained a complete set of tools for physical layout generation, folding, design validation and logic design including LOGMIN, an interactive program to automate the capture, generation, minimization and translation of finite state machines and combinatorial logic to PLAs. Project leader for PLA workbench tools. Designed Applicon 860 schematic package, netlister and 870 to 860 schematic database translator. Wrote a graphical FSM capture package. Wrote a hierarchical schematic netlist to MOSSIM netlist translator. Evaluated ZYCAD and YORKTOWN logic simulation engines.

Research/Teaching Assistant, University of Michigan, Ann Arbor, Michigan. 9/80 to 5/81.
Wrote translator from CIF to format for plotter and electron beam mask machine. Designed part of a waveform generator IC. Project coordinator for VLSI design course. Responsible for final chip frame, coordinating 16 designers and interfacing with the fabrication facility. Wrote a PLA layout generator and also taught a section of a data structures course. Wrote a graphical program for conversion of layout database into Cyto-computer format.

Firmware Design Engineer, Bell Northern Research, Inc., Ann Arbor, Michigan. 7/78 to 9/80.
Designed firmware and part of hardware for intelligent floppy disk controller. Developed communication firmware for Coax/RS232 printer. Designed package to emulate IBM SNA SCS mode for printers. Redesigned SDLC, BSC and remote station adapter firmware. Maintained all physical layer communication firmware.

Systems Engineer, Vega Servo-Control, Troy, Michigan. 6/77 to 6/78.
Designed a DNC communication system utilizing an Eclipse C330 to service 32 Nova-2 computers and a Gerber drafting machine. Wrote firmware for a Z80 floppy disk controller. Wrote translator for CL-data to Gerber drafting format. Developed application and diagnostic programs for the Vega III-G machine tool controller.

PROGRAMMING LANGUAGES: C, C++, AWK, COMMON LISP, MAINSAIL, ALGOL, BCPL, PL/1, FORTRAN.
OPERATING SYSTEMS: Win/NT, UNIX, MS-DOS/WIN, NeXTSTEP, VAX/VMS, IBM/UTS/CMS, RDOS, MTS, CP/M.
STATISTICAL PACKAGES: Stata, S+, EGRET, BMDP, LISP-STAT, SAS, SPSS.
MATHEMATICAL PACKAGES: MATHEMATICA, MACSYMA, MATLAB.

PUBLICATIONS:

Teel WB, Koepsell TD, Daling J, Psaty B, Dugowson C, Kronmal R, Voigt L. 'A population-based case-control study of breast implants as a risk factor for connective tissue diseases'. Submitted.

Teel WB, Koepsell TD, Daling J, Psaty B, Dugowson C, Kronmal R, Voigt L. 'The incidence of primary Sjogren's Syndrome, systemic lupus erythematosus, systemic sclerosis / CREST syndrome, polymyositis / dermatomyositis, in King County, Washington, 1983 - 1991'. In prep.

Teel WB, Martin D, Voigt L. 'Frequency matching for population based epidemiologic studies'. In. prep.

Teel WB, Koepsell TD, Daling J, Psaty B, Dugowson C, Kronmal R, Voigt L. 'A population-based case-control study of exogenous female hormones as a risk factor for systemic lupus erythematosus. In prep.

Teel WB, Koepsell TD, Daling J, Psaty B, Dugowson C, Kronmal R, Voigt L. 'Hair Dye as a risk factor for connective tissue diseases. In prep.

Teel WB, Thwin SS, Daling J. 'Big is not necessarily better: maternal weight gain as a risk factor for cesarean section'. In Prep.

Teel WB. 'A meta-analysis of aspirin for migraine prophylaxis'. In prep.

Teel WB, Wilde D, 'A Logic Minimizer for VLSI PLA Design', Proceedings of the nineteenth design automation conference, ACM, IEEE, 1982.

Mudge TN, Teel WB, Lougheed RM, 'Cellular image processing techniques for checking vlsi circuit layouts', Conference Proceedings - Information Sciences and Systems, 1981.