CSCI 2 – Assignment 1 Part 2 (Due Date 9/7/2017 Before your lab begins)

Sorry. No debugging support is available on the day of submission! Plan in advance so that your program is debugged the day before or earlier than the submission day.

You may use the technology developed in part 1 of assignment 1 for function that would compute salary.

Study Resources For this Lab

1. All of your study materials from CS1, including Tony Gaddis textbook.
2. All your programs done in CS1.
3. Singhal chapters CS1, chapters 10, 11, 12. [See canvas website]

Main Program

As a paymaster for Badwater Brewery, you are asked to determine the gross pay of each employee based on the following information.

Table of Base Pay

Employee Description	Base Pay in Dollars	Employee Code
Factory Worker	800	S
Office Worker	1000	O
Management	1500	M

A percentage is added to the base pay depending upon as to how employee fares in the following three categories.

Category 1: Job Classification: Use the following table to add to the Base pay.

Job Classification	Percentage of base salary to add to calculate gross pay
1	5%
2	10%
3	20%

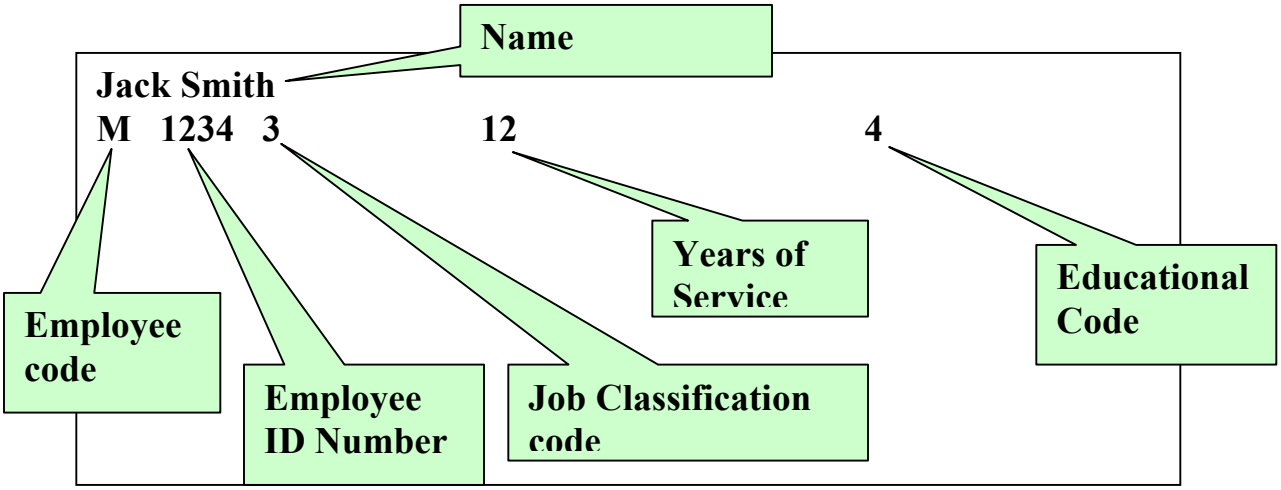
Category 2: Years of Service:

For service from 0 to 10 years, add 5% of the base pay. For above 10 years of service, add 1% for each year above 10 years. Maximum service 50 years.

<u>Category 3: Educational Adjustment:</u> Use the following table to add to the base pay.

Educational Code	Level of Education	Percentage of base pay to add to calculate the gross pay
1	High School	0%
2	Junior College	5%
3	University	12%
4	Graduate School	20%

Use the base pay and salary adjustments dictated in above three categories to calculate the employee gross pay. The employee data are stored in a ASCII file which has the structure given on next page.



Another set of employee data may look like as follows:

Joe Jones
O 1222 2 10 3

The data file may be empty or may have some invalid data. For example, the number of years of service more than 50 will be invalid data. Or job classification or Educational code may not be of the value allowed. Wrong Employee code may be present.

Your task is to design a struct to store "necessary" employee raw data and processed data. Use an array of struct to store all employee data in the file. You may

design the program to hold 30 employees (therefore 30 member array of struct). The data file may have more than 30 employee records however. In that case your program must read only 30 records and give a message that number of records exceeded the program capacity. If data file is empty, then program must print the message to inform user that data file is empty. Modularize your program with functions as much as you can.

Your program output must be of the following format.

Name	ID#	Job Type	Gross Salary
Jack Smith	1234	Management	$2205.00
……	……	…….	…….

check this number for accuracy

For employees, for which the data are bad, the program must print the message to the affect that Gross Salary could not be calculated due to bad data. The types of bad data may be:

- **Employee code in lower case letter**
- **Number of years of service exceeding 50 years**
- **Educational and job classification code not in the correct range.**

The output must be printed to a file. The output file must have processed data printed in the following order:

1. **Unsorted Employee data like the table format above.**
2. **Sorted Employee data, sorted based on ID#**
3. **Sorted Employee data, sorted based on Last Name**
4. **Sorted Employee data, sorted based on Gross Salary**

While the program is running, the user must be given a menu of choices to search the employee array based on Employee ID number (assume unique ID number for each employee), and based on Last Name. If more than one employee have the same last names, then program may print, just the first match. User must be able to search as many times as they wish and exit after the search is over. You can use any sorting algorithm you prefer. Binary search will be faster, but will require a sorted array. You may use linear search algorithm as well.

Before you show me your program test, your data thoroughly using an empty file, file with more than 30 records, and file with less than 30 records. I will provide you the sample output from one of the data file, and a Xerox copy of the data file shortly. The data file provided for the program will be namelist.txt.

Required in Program Design:
1. Since you are going to sort and/or search based on last name, ID and gross salary – it is imperative that those must be the struct data members. Design other data members as per robustness of the program.
2. Must have default and explicit constructors.
3. You may wish to decide whether you wish to use two struct or one. In case you use two struct then one struct would be Employee and other would

contain the array of struct Employee and proper operations on it. Choose array size to be 30.
4. Program must run successfully on empty file, file with more than 30 Employees, and less than 30. Keep employee array size to 30. In each case program must inform user properly. For example if file has data more than the capacity of the array then user must be informed that file has excess data and only 30 employees would be processed.
Suggestions for member and non-member function Design:
1. Provide get and set member functions for all necessary private data members.
2. Provide member function to calculate gross salary and required error correction.
3. Provide a struct member function that prints the employee data either to a file or to the standard output. The output is properly formatted. If employee gross salary was set to zero then function prints relevant message that there were calculation error due to bad file data.
4. Provide functions for sorting and searching based on different criteria.

Data Files are available on canvas. Look for file names beginning namelist. Additional information will be provided as needed.