CS 4323 Design and Implementation of Operating Systems I

Assignment 00: Full Marks 100
(Due Date: 09/02/2020, 11:59 PM CDT)

This assignment is created to make you familiar with C programming language. We will be using C to study operating system concepts and hence to be able to complete the assignment successfully, you will need to be familiar with the following C programming concepts:

- Basic data types such as int, float/double and char
- Familiarization with Arrays (Strings as array of char), Structures (Very important)
- Basic use of printf() to print text to the terminal and scanf(), fgets() to read from the keyboard
- Read and write from and to the text file
- Control structures: if-then-else, switch
- Loops: while, for and do-while
- Functions and function calls
- Pointers (Very important)
- Pointers to structures (Very important)
- Call by reference
- Dynamic memory allocation using malloc, calloc and free (Very important)
- Basic use of the #define directive
- Separate compilation of individual C files

All these assignments must be completed on CSX machine. Logging on to the CSX account and other relevant information can be easily accessed in: http://www.cs.okstate.edu/loggingon.html

Your task is to create employee database for Baltimore City. You have been given two files:

1. BaltimoreCityEmployeeSalariesFY2018.csv
2. BaltimoreCityEmployeeSalariesFY2019.csv

that contains the employee information for Baltimore city in 14 and 15 departments for the fiscal year 2018 and 2019 respectively. The file information is self-evident but is also explained below.

These two files should be taken as the input to your program and create the database for the city. You need to return a single file (saved as employeeRecord.txt) as output that should contain following columns:

FULL_NAME	JOB_TITLE	DEPT_ID	DESCR	HIRE_DT	TERMINATION_DT	ANNUAL_RT	GROSS

You need to have a record of all the employee for both fiscal year 2018 and 2019. Some new employees are hired in the 2019. Any employee detail missing in 2019 should be considered as terminated and should enter in the TERMINATION_DT column.

You should have another output file (saved as departmentBudget.txt) as output that should contain following columns:

DESCR	ANNUAL_SALARY_2018	ANNUAL_SALARY_2019	NET_CHANGE

where, DESCR is the department description (from the file).
ANNUAL_SALARY_2018 represents the summation of all salary in the particular department for the year 2018.
ANNUAL_SALARY_2019 represents the summation of all salary in the particular department for the year 2019.
NET_CHANGE is the change in the salary by comparing annual salary for 2018 and 2019 result for each department.

The header file employee_record.h should contain:

- all function declarations used in your project
- structure used in your project

In the file employee_record.c, you should implement the functions declared in employee_record.h.

The main.c file is the main file for your project and contains the function: main()

To represent each employee, you will need to create a structure (use **struct** Employee). This structure should store information like:

- full name (in the files given by NAME)
- job title (in the files given by JOBTITLE)
- dept ID (in the files given by DEPTID)
- department name (in the files given by DESCR)
- hire date (in the files given by HIRE_DT)
- annual salary (in the files given by ANNUAL_RT)
- gross salary (in the files given by Gross)

Choose an appropriate datatype for each fields of the structure.

You need to make a menu-driven program where the menus would be:

1. List of employees by department
2. Create an employee record for the city
3. Create a summary report of budget for the city

Option 1 leads to the display of all employee information for the selected department. It should consider entries from both files. Upon choosing 1, give a menu-driven option to pull the record of employee for that department. This output needs to be displayed on the screen.

Option 2 creates employeeRecord.txt.

Option 3 creates departmentBudget.txt.

Program requirement:

- You need to use dynamic array for each employee record.
- The dynamically allocated memory should be freed.

The grading will be as follows:

- Use of dynamic array correctly for storing employee record. **[15 Points]**
- Freeing the dynamic memory used by the array. **[5 Points]**
- Correctly displaying results for option 1. Should work properly for each department **[40 Points]**
- Correctly displaying results for option 2 **[20 Points]**
- Correctly displaying results for option 3 **[20 Points]**

Failure to compile the program correctly will deduct 20 points automatically.

- Your program will be evaluate based on the correctness
- Points will be deducted for any bad programming practices like:
    - not using proper variable name,
    - not using comments
- Points will be deducted if the instruction is not followed properly.

Submission Guidelines:

- You need to submit three files:
  - employee_record.h
  - employee _record.c
  - main.c

- In the main.c file, use the following header information:
  - Author Name:
  - Email: <Use only official email address>
  - Date:
  - Program Description:

- Use comments throughout your program.

- Each function should be properly commented:
  - Mention each argument type, purpose
  - Function description
  - Return type of the function

- Failure to follow standard programming practices will lead to points deduction, even if the program is running correctly. Some of the common places where you could lose points are:
  - Program not compiling successfully: -20 points
  - No comments on code (excluding function): -5 points
  - No comments on function: -5 points
  - Not writing meaningful identifiers: -5 points
  - Failure to submit files as specified: -10 poimts