CIS*2500 (Intermediate Programming) Lab #1 Part A

Important notes:

- **It is recommended that part A of the lab is completed in advance.**
- Part B of the lab will be assigned to you by your lab TA in the beginning of the lab.
- You must complete part B, and then request a TA to grade your lab submission before you leave the lab.
- You MUST get your lab submission graded in the lab – the TAs will not grade your submission after the lab is over (unless otherwise notified by the lab TA).

Concepts: structures, functions, text files, makefile, gitlab

Description: For this lab, assume the following definitions:

```
#define SIZE 25
#define NUM_EMP 10


/* definition of an employee record*/
typedef struct employee
{
  char fname[20];
  char lname[20];
  int id;
  char dependents [3][20]; // assume that employees have exactly
                           // 3 registered dependents
} Employees;
```

Part A of lab1

1. You are required to write function definitions for the following tasks in a file called lab1A.c

Function 1.

Function name: saveEmployees

Description: This function takes 3 parameters – (1) an array named arrEmployees of size NUM_EMP, where each element is of type Employees, (2) an integer variable named *c* and

(3) string named fileName that holds the name of a text file. Note that fileName gets its value from **command line** and then passed to this function.

It saves information (i.e., fname, lname, id and dependents) of the first *c* number of employee records in a text file. Returns nothing.

Pre-condition: arrEmployees is populated with *N* employee records, where *c <= N.*

Function 2.
Function name: loadEmployees

Description: This function takes 2 parameters (1) an array named arrEmployees of size NUM_EMP, where each element is of type Employees, and (2) string named fileName that holds the name of a text file. Note that this name must be accepted from **command line** and then passed to this function.

It loads all employee records that are stored in a text file to array of Employees called arrEmployees.

Returns the total number of records loaded.

Pre-condition: The given text file has exactly *N* employee records, where *N <= NUM_EMP.*

2. Test the above functions by calling them in main () written in a file called lab1Main.c

3. Create a makefile that compiles your c files and creates an executable. For example, if my c files are called lab1A.c and lab1Main.c, and makefile has the following content, then running **make** utility will create an executable file called lab1_output. It can then be run with a command-line argument that holds the filename – for example,

./lab1_output fileEmployees

----------start of makefile----------

```
lab1_output: lab1A.o lab1Main.o
    gcc lab1A.o lab1Main.o -o lab1_output

lab1A.o: lab1A.c
    gcc -Wall -std=c99 -c lab1A.c

lab1Main.o: lab1Main.c
    gcc -Wall -std=c99 -c lab1Main.c

clean:
```

```
    rm *.o   lab1_output
----------end of makefile----------
```

Submission Instructions (after you complete both parts A and B):

Submit all your files to Gitlab (Lab TAs will teach you how to submit)
- makefile
- lab1A.c
- lab1Main.c
- lab1B.c
- lab1.h (if you are using a header file)

Follow these instructions to use gitlab:

Step 1: Connect to the school server linux.socs.uoguelph.ca.

Decide where cis2500 L1 work will go and make a directory (e.g. mkdir Lab1) if you need to. cd to that directory from the terminal application.

If you wish to work on your local laptop (instead of connecting to the school server), then
- Mac users can use the terminal mode (I have tested it on my mac)
- Windows users can use powershell, WSL (windows subsystem for linux) or git bash.
- decide where cis2500 work will go and make a directory (e.g. mkdir CIS2500)
- cd to that directory from the terminal application

Step 2: From the chosen directory, now type:
git clone https://git.socs.uoguelph.ca/2500W23/<your username>/L1.git

At this point, you have a directory to work with on your local system.

Step 3: Do the work

- (remember to) cd to the new directory
- create the file that you are working on
- after first save, type: git add filename

ONLY ADD THE FILE ONCE!!! //**do not** type the following: git add .

Loop every 20-30 minutes:
git commit -am "write something here about what you just did"

Once per day:
git push // this is what stores local work back to the server.

To learn more about Gitlab, go to this link on moodle
https://moodle.socs.uoguelph.ca/course/view.php?id=169