ECE263/264

Semester Project

**Due date: Monday Dec 16th 5:00 pm (digital upload + hard copy)**

**No Late**

**You must turn in all code and project**

The semester project will be based on the data type struct anonApplication and is defined below. Within the structure we use the following

struct anonApplication {

int appID;

int appDate;

int timeIn;

char positionApplied[30];

char university[30];

float collegeGPA;

int skillLevelProgramming; // 1 to 5 where 5 is most skilled

int skillLevelCircDesign; // 1 to 5 where 5 is most skilled

int skillLevelPrjManage; // 1 to 5 where 5 is most skilled

struct anonApplication \*nextApp;

};

You need to download prjmain.c, prj.h, prjstud.h prjstud.c, prj.c and libVS2017.lib. You will need to link a library a video on how to do it wil be posted. You will need to use vidual studio. The library libVS2017.lib was build with VS 2017 (this is available via cnc24hour we may support other version of VS)

This program will simulate the actions of a HR manager who makes final decisions on hiring. The process that this corporation has adopted is to make applications to the “decision maker” anonymous. This person is provided university information, GPA, and three skill level scores skillLevelProgramming, skillLevelCircDesign, skillLevelPrjManage. where each were determined by a lab manager in the corporation, who met the applicant and assessed their skill. These skill level scorers are 1 to 5 where 5 is the best score. The possible positions are: "systems engineer", "software engineer", "controls engineer", "engineering management". In main a variable call clockTime is maintained. The applicant list is stored as a link list. Each unit of time, the hiring manager will increment the timeIn for each applicant. Any application that is 5 or more units old should be deleted. Next the hiring manager will“hire” one person for each of the four job positions. This person must have applied for the position. The process of hiring is to output their appID and the position and then delete the node. The applicant that will be hired is the first person in the link list. So at the start of a day the first applicant in the list that has positionApplied equal to "systems engineer" will be hired as systems engineer, the same is true for the three other position. In addition, each unit of time a link list of new applicants is provided and must be insert into the HRManager link list. Whenever an insertion of a new node is made into the link list you must sort by average skill level (there is a function provided to you in the library that you can call).

Functions that we have written for you are:

struct anonApplication \*sortByAvgSkill(struct anonApplication \*head)

struct anonApplication \*makeInitialList()

void fillData(struct anonApplication \*node, int clockTime)

struct anonApplication\*checkForApp(struct anonApplication\* head, int x)

struct anonApplication\* NewApplications(int clocktime);

Functions you are to write:

struct anonApplication \*sortByUniversity(struct anonApplication \*head)

struct anonApplication \*deleteApplication (struct anonApplication \*head, struct anonApplication \*to\_be\_gone)

struct anonApplication \*processDay(struct anonApplication\* head, int \*clockTime)

void printFormatted (struct anonApplication \* head);

void analyzeApplicantList (struct anonApplication \* head);

void terminateAndWrite (struct anonApplication \* head);

# Functional descriptions of functions provided to you

**struct anonApplication \*sortByAvgSkill(struct anonApplication \*head)**

this function is written and provided in libarary sorts the link list by average skill in descending order

**struct anonApplication \*makeInitialList()** this is written in prj.c and makes an initial list of applications

**void fillData(struct anonApplication \*node, int clockTime)** this is written in prj.c, if you send this function an address of a struct anonApplication it will assign the struct random application information

**struct anonApplication\*checkForApp(struct anonApplication\* head, int x)** this is in prj.c and is used in main, if you enter an application id it returns a pointer to the struct with that id, if no such id is in link list it returns NULL

# Functional descriptions of the files you need to write

**struct anonApplication \*sortByUniversity(struct anonApplication \*head**) you are to write this function in prjstud.c (note the shell of the function is already there. This should sort the link list by university. For full credit when you swap you must move the modes around (i.e. links) not just copy data

**struct anonApplication \*deleteApplication (struct anonApplication \*head, struct anonApplication \*to\_be\_gone**). This function will delete the node to\_be\_gone from the li list. You can assume the node is in the link list. You are to write this in prjstud.c the shell of the code is already there

**struct anonApplication \*processDay(struct anonApplication\* head, int \*clockTime**) You are to increment the clock (notice you have a pointer to clocktime so you can change it in main. Each unit of time, the hiring manager will increment the timeIn for each applicant. Any application that is 5 or more units old should be deleted. Next the hiring manager will“hire” one person for each of the four job positions. This person must have applied for the position. The process of hiring is to output their appID and the position and then delete the node. The applicant that will be hired is the first person in the link list. So at the start of a day the first applicant in the list that has positionApplied equal to "systems engineer" will be hired as systems engineer, the same is true for the three other position. In addition, each unit of time a link list of new applicants is provided and must be insert into the HRManager link list. The list of new applications is found by calling struct anonApplication\* NewApplications(int clocktime); You should make a call like ptr=NewApplications(clocktime); Then insert ptr to the end of your list. Then call sortByAvgSkill like head= sortByAvgSkill(head);

**void printFormatted (struct anonApplication \* head);** here you should display all the data in an organized and formatted manner. This is in prjstud.c

**void analyzeApplicantList (struct anonApplication \* head**); this is in prjstud.c for each possible position you are to count the number of applicants, determine their average GPA, their average skill level in circuit design, their average skill level in project management and their average skill level in programming and output aall in a formatted manner

**void terminateAndWrite (struct anonApplication \* head**); this is in prjstud.c you are to write all the data in the link list to file and then free the list and then exit the program

# Individually how much each function will be worth

**sortByUniversity 13**

**deleteApplication 14**

**processDay 27**

**printFormatted 14**

**analyzeApplicantList 14**

**terminateAndWrite 14**

**To correctly build the project, linking and adding all files (4 points)**