**Unit8.hw (This HW is woth 200 points!)**

NOTE WELL: It was annouced in class that the lowest HW will be dropped. If a student scored 100 in all the previous HWs and didn’t submit this HW, then only half of this HW will be dropped. Consequently, that student will not end up with 100 as the overall HW average. On Webcourses, you will see two scores associated with this HW: Unit7.hw.1 and Unit7.hw.2. Say a student scored 150 points out of 200 for this hw, then Unit7.hw.1 = 75 and Unit7.hw.2 = 75 for that student. Submission is done in Unit7.hw.1

Create the structure ***employee*** with ***int id, char name[50], float salary*** and ***int numberOfDependents*** as fields. The employee id is the sum of the ascii codes of the uppercase of the letters of the name, including white spaces in the name. For example, if the name is **AaBc xYz** then:

**id = ascii of A + ascii of A + ascii of B + ascii of C+ 32 + ascii of of X+ ascii of Y + ascii of Z**

**= 562**

The goal of this HW is to create a ***special*** ***linked list*** that you may call ***company*** to store the employees. We seek to implement a list that can be pictured as:

***company***

***Ed Renu***

***Reda Ali***

***Rim Oz***

***Kim Oz***

***Aidan Jones***

***Nadia Jones***

***Naadi Jones***

Let ***emp1*** and ***emp2*** and ***emp3*** be three different ***employee*** variables to add to ***company*** in this order, and assume that ***Aidan Jones***, ***Nadia Jones*** and ***Naadi Jones*** are the names of those employees. Those employees have the same employee id, and in this collision case, we first add Aidan to the list then Nadia and then Naadi are to be added to a new linked list as seen in the picture above.

Consequently, each node of ***company*** should contain two pointers that you may call ***next*** and ***below*** . The dots in the picture above denote the ***NULL*** pointer.

Your job is to write the code that allows to:

1. Print all the employees
2. Add a new employee
3. Search an employee by name. Your code prints **Found/Not Found** to the screen
4. Find the highest net salary

C o d e S k e l e t o n

#include <stdio.h>

#include <stdlib.h>

#include<string.h>

struct employee {

int id;

char name[50];

float salary;

int numberOfDependents;

//---------------------

struct employee \* next;

struct employee \* below;

};

//----------------------- FUNCTION PROTOTYPES ---------------------------

//0.

int getId ( char \* name);

/\*

returns the sum of the ascii codes of the letters of name.

\*/

//1.

void printEmployee ( struct employee );

/\*

prints an employee in the format [id,name,net salary], where:

Net salary = salary\*0.91 + (numberOfDependent \*0.01\*salary)

Examples: **[562,AaBc xYz,1253.56] [562,bAac ZyX,6253.06]**

\*/

//2.

void printAllEmployees ( struct employee \* list);

/\*

prints all employees in the list. Loops through list and calls

printEmployee for each employee in list

\*/

//3.

struct employee \* addEmployee ( struct employee \* list , struct employee e);

/\*

adds e to list.

\*/

//4.

int searchEmployee ( struct employee \* list , char \* name);

/\*

returns 1 if name is found in list. Otherwise, it returns 0

\*/

//5.

float highestNetSalary ( struct employee \* list);

/\*

returns the highest net salary.

\*/

//----------------------- FUNCTION DEFINITIONS ---------------------------

//0.

int getId ( char \* name ){

printf ("COMING UP\n");

return 0;

}

//1.

void printEmployee ( struct employee ){

printf ("COMING UP\n");

}

//2.

void printAllEmployees ( struct employee \* list){

printf ("COMING UP\n");

}

//3.

struct employee \* addEmployee ( struct employee \* list , struct employee e){

printf ("COMING UP\n");

return list;

}

//4.

int searchEmployee ( struct employee \* list , char \* name){

printf ("COMING UP\n");

return 0;

}

//5.

float highestNetSalary ( struct employee \* list){

printf ("COMING UP\n");

return 0.0;

}

**//DO NOT CHANGE ANYTHING IN main**

int main() {

struct employee \* company = NULL , tempEmployee;

int tempInt = 0;

char tempName [50] = "Allal Kamch" ;

char dummy;

printf ("Enter 1 to add a new employee, 0 to stop: ");

scanf ("%d", &tempInt);

while ( tempInt == 1 ){

printf("Enter Full Name: ");

scanf("%c",&dummy);

fgets(tempEmployee.name, 50, stdin);

tempEmployee.id = getId ( tempEmployee.name );

printf ("Enter salary-numberOfDependents: ");

scanf ("%f-%d", &tempEmployee.salary,

&tempEmployee.numberOfDependents);

company = addEmployee ( company , tempEmployee );

printf ("Enter 1 to add a new employee, 0 to stop: ");

scanf ("%d", &tempInt);

}

printAllEmployees ( company ) ;

printf ("Highest Net Salary = %.2f\n", highestNetSalary(company) );

tempInt = searchEmployee ( company , tempName) ;

if ( tempInt == 1 ) printf ("%s is found!\n", tempName);

else printf ("%s is NOT found!\n", tempName);

return 0;

}