Name: Bibek Dhungana
Lab 11

CODE:

```
/*
 FILENAME:Lab11.c
 AUTHOR: BIBEK DHUNGANA
 DATE: April 30, 2021
 SPECIFICATION: This program store student
information in the university. And, allow
student to
                check schedule, add classes and
drop classes by menu driven command line
interface.
 FOR: CS 1411-504
 */

//including all the required libraries
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>

//defining the SIZE macro
#define SIZE 50

//making a structure class
struct class {
    char name[SIZE];
    int sectionNumber;
    int numberOfCredits;
    struct class *next;
```

```
};

//typealiasing the struct class as class
typedef struct class class;

//making a structure student
struct student {
    int studentId;
    class* class;
    struct student* next;
};

//typealiasing student as student
typedef struct student student;

//function prototypes
void addStudent(student** s, student *tmp);
void addClass(student **s, int studentId, class*
tmp);
int compareClass(char givenName[], char
courseName[]);
void dropClass(student **s, int studentId, char
courseName[]);
void printStudentInfo(student* s);



int main(){
    //initializing the required variables
    int input, numberOfStudents,studentId,
sectionNumber, numberOfCredits;
    int maxNumOfClass = 6;

    //creating the empty student pointer
```

```
    student *s = NULL;

    // storing the course name
    char courseName[SIZE];

    //getting number of student information from
the user
    printf("Initial initialization of the
student in the university\n");
    printf("How many student information you
want to enter?");
    scanf("%d", &numberOfStudents);

    //getting student id from the user
    printf("Please Enter student ids:\n");
    for(int i=1; i<=numberOfStudents; i++){
        scanf("%d", &studentId);

        //dynamically allocate the memory for
each student
        student* tmp =
(student*)malloc(sizeof(student));
        tmp->studentId = studentId;
        tmp->class = NULL;
        tmp->next = NULL;
        addStudent(&s, tmp);
    }

    while(true){
        printf("Follow the instruction below or
-1 to exit\n");
        printf("1 - Add student\n");
        printf("2 - Create initial schedule\n");
        printf("3 - Add class \n");
```

```
        printf("4 - Drop class\n");
        printf("5 - Get info about classes
registered\n");

        //getting input choice from the user
        printf("Enter choice: ");
        scanf("%d", &input);



        //exist the loop
        if(input == -1){
            break;
        }

        //if input is 1
        else if(input == 1){
            printf("Enter of student id: ");
            scanf("%d", &studentId);
            // allocate nodes in the heap
            student* tmp =
(student*)malloc(sizeof(student));
            tmp->studentId = studentId;
            tmp->class = NULL;
            tmp->next = NULL;
            addStudent(&s, tmp);
            printf("The student is added
successfully\n");
            printf("-------------------------
-------------------------------------\n");
        }

        //input is 2
        else if(input == 2){
```

```
            printf("Enter student id: ");
            scanf("%d", &studentId);
            printf("Enter number of class: ");
            scanf("%d", &numberOfStudents);
            printf("Enter class name,
sectionNumber and numberOfCredits:\n");

            for(int i=1; i<=numberOfStudents;
i++){
                scanf("%s%d%d", courseName,
&sectionNumber, &numberOfCredits);

                // create node for each class
                class* tmp =
(class*)malloc(sizeof(class));
                for(int j=0; j<maxNumOfClass;
j++)
                tmp->name[j] = courseName[j];
                tmp->sectionNumber =
sectionNumber;
                tmp->numberOfCredits =
numberOfCredits;
                addClass(&s, studentId, tmp);


            }
        printf("The schedule  is built
successfully\n");
        printf("-----------------------------
-----------------------------------\n");
        }


        //if input is 3
```

```
        else if(input == 3){
            printf("Enter student id: ");
            scanf("%d", &studentId);
            printf("Enter class name,
sectionNumber and numberOfCredits:\n");
            scanf("%s%d%d", courseName,
&sectionNumber, &numberOfCredits);

            // create node for class
            class* tmp =
(class*)malloc(sizeof(class));
            for(int j=0; j<6; j++)
            tmp->name[j] = courseName[j];
            tmp->sectionNumber = sectionNumber;
            tmp->numberOfCredits =
numberOfCredits;
            addClass(&s, studentId, tmp);
            printf("The class is added
sucessfully\n");
            printf("-------------------------
-----------------------------------\n");

        }

        //if input is 4
        else if(input == 4){
            printf("Enter student id: ");
            scanf("%d", &studentId);
            printf("Enter class name to drop:
");
            scanf("%s", courseName);
            dropClass(&s, studentId,
courseName);
```

```
            printf("The classes is dropped
successfully");
            printf("-----------------------------
-----------------------------------------\n");
        }

        else if(input ==5){
            printf("-----------------------------
-----------------------------------------\n");
            printf("The total info of student
are:\n");
            printStudentInfo(s);
            printf("-----------------------------
-----------------------------------------\n");
        }

    }

return 0;
}
```

```
/*
 NAME:addStudent
 PARAMETERS: student** s,student *studentPointer
 RETURN TYPE:void
 SPECIFICATION:This function help to add student
to the student chain in the university.
*/
```

```
void addStudent(student** s, student
*studentPointer){
    // empty list
    if(*s == NULL)
    {
    *s = studentPointer;
    return;
    }

    student *last = *s;

    while (last->next != NULL)
    last = last->next;

    last->next = studentPointer;
}

/*
 NAME:addClass
 INPUT PARAMETERS:student **s,int
studentId,class* classPointer
 RETURN TYPE:void
 SPECIFICATION:This function is used to add or
register classs for the students.
*/
void addClass(student **s, int studentId, class*
classPointer){
    student *last = *s;

    // search for student first
    while(last!=NULL && last->studentId !=
studentId)
    last = last->next;
```

```
    // no student found
    if(last == NULL)
    return;

    if(last->class == NULL)
    {
    last->class = classPointer;
    return;
    }

    class* c = last->class;
    while(c->next != NULL)
    c = c->next;

    c->next = classPointer;
}


/*
 NAME:compareClass
 INPUT PARAMETERS:char givenName[],char
courseName[]
 RETURN TYPE:int
 SPECIFICATION: This function compare two
character array and check if they are equal.
*/
int compareClass(char givenName[], char
courseName[]){
    int maxNumOfClass = 6;
    for(int i =0 ; i < maxNumOfClass; i++){
        if(givenName[i]!=courseName[i])
        return 0;
    }
    return 1;
```

```
}

/*
 NAME:dropClass
 INPUT PARAMETERS:student **s,int studentId,char
courseName[]
 RETURN TYPE:void
 SPECIFICATION:This function is used to drop
classes for the given student.
*/
void dropClass(student **s, int studentId, char
courseName[]){
    if(*s == NULL){
        return;
    }


    // search for the student if avaialable
    student *last = *s;
    while(last!=NULL && last->studentId !=
studentId)
    last = last->next;

    // if student not found/ class not found
    if(last==NULL || last->class == NULL){
        return;

    }

    class* c = last->class, *prev = NULL;

    // checking if the class is matched....
    while(c->next != NULL && !compareClass(c-
>name, courseName)){
```

```
            prev = c;
            c = c->next;
            }
        if(prev == NULL){
            last->class = c->next;
        }
        else{
            prev->next = c->next;
        }

}

/*
 NAME:printStudentInfo
 INPUT PARAMETERS:student* s
 RETURN TYPE:void
 SPECIFICATION:This function print all the
information of classes register by all the
student in the university.
*/
void printStudentInfo(student* s){
    // traversing the linked list
    while(s!=NULL){
        printf("Student id: %d\n", s-
>studentId);
        class* c = s->class;
        while(c!=NULL){
            printf("%s %d %d\n", c->name, c-
>sectionNumber, c->numberOfCredits);
            c = c->next;
        }
        s = s->next;
    }
}
```

OUTPUT

Find printStudentInfo

```
Initial initialization of the student in the university
How many student information you want to enter?3
Please Enter student ids:
11679703
11679704
11679705
Follow the instruction below or -1 to exit
1 - Add student
2 - Create initial schedule
3 - Add class
4 - Drop class
5 - Get info about classes registered
Enter choice: 1
Enter of student id: 11679706
The student is added successfully
------------------------------------------------------------------------
Follow the instruction below or -1 to exit
1 - Add student
2 - Create initial schedule
3 - Add class
4 - Drop class
5 - Get info about classes registered
Enter choice: 2
Enter student id: 11679703
Enter number of class: 1
Enter class name, sectionNumber and numberOfCredits:
cs1411 1 4
The schedule  is built successfully
------------------------------------------------------------------------
Follow the instruction below or -1 to exit
1 - Add student
2 - Create initial schedule
3 - Add class
4 - Drop class
5 - Get info about classes registered
Enter choice: 3
Enter student id: 11679703
Enter class name, sectionNumber and numberOfCredits:
math 2 4
The class is added sucessfully
------------------------------------------------------------------------
```

```
------------------------------------------------------------
Follow the instruction below or -1 to exit
1 - Add student
2 - Create initial schedule
3 - Add class
4 - Drop class
5 - Get info about classes registered
Enter choice: 5
------------------------------------------------------------
The total info of student are:
Student id: 11679703
cs1411 1 4
math 2 4
Student id: 11679704
Student id: 11679705
Student id: 11679706
------------------------------------------------------------
Follow the instruction below or -1 to exit
1 - Add student
2 - Create initial schedule
3 - Add class
4 - Drop class
5 - Get info about classes registered
Enter choice: 4
Enter student id: 11679703
Enter class name to drop: math
The classes is dropped successfully----------------------
Follow the instruction below or -1 to exit
1 - Add student
2 - Create initial schedule
3 - Add class
4 - Drop class
5 - Get info about classes registered
Enter choice: 5
------------------------------------------------------------
The total info of student are:
Student id: 11679703
cs1411 1 4
Student id: 11679704
Student id: 11679705
Student id: 11679706
------------------------------------------------------------
Follow the instruction below or -1 to exit
1 - Add student
2 - Create initial schedule
3 - Add class
4 - Drop class
5 - Get info about classes registered
Enter choice:
```

All Output ^