PROBLEM

Create a Student-Course registration system, where students can register into one of the course which provides by the institute.

- Institute provides five courses for this academic year. Course name and Course id are given below.

COURSE_NAME	COURSE_ID
Discrete Maths	CS001
Communication	CS005
Programming	CS2003
Logic Design	CS2019
Statistics	CS2308

- When a student comes for course registration show all the courses that provides by the institute along with course_id.
- Student can enrol to one of the given course by giving his/her name along with opted course_id.
- For each Student, registration record contains student name with course_id.
- Registration records should keep in a way such that students who enrolled to the same course arranged together

Student Name: A	Student Name: B	Student Name: C
Course_id: CS001	Course_id: CS001	Course_id: CS2003

- If a new student D registers to the course Discrete Maths, then that student record has to place near to the students who opted the same course.
- Display function should display all registration details like Student Name, enrolled course name with its course id

A	B	C
Discrete Maths	Discrete Maths	Programming
CS001	CS001	CS2003

- Students are also able to delete his/her current registration to a particular course.

NOTE: Use appropriate data structure for the implementation.

SOLUTION

Note: The code tested under 32-bit IDE (Code::Blocks)

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>
struct node
{
   char name[30];
   char cid[10];
   char cname[30];
   int order;
   struct node *link;
};
void insertDetial();
void displayDetail();
void arrangeDetail();
void deleteDetail();
struct node* createNode();
struct node *start=NULL;
char courseName[5][20]={"Discrete Maths", "Communication" ,"Programming",
"Logic Design","Statistics"};
char courseId[5][10]={"CS001","CS005","CS2003","CS2019","CS2308"};
void main()
{
   int choice;
   while(1)
   {
      system("cls");
      printf("Press\n1. Enter Student Detail\n2. Display Record\n3. Delete
      Record\n");
      printf("4. Exit\n");
      printf("Enter Your Choice: ");
      scanf("%d",&choice);
      switch(choice)
      {
      case 1:
         insertDetail();
         break;
      case 2:
         displayDetail();
         break;
      case 3:
         deleteDetail();
```

```
            break;
        case 4:
            exit(0);
        default:
            printf("Invalid Choice\n");
            printf("Press Any Key to Re-Enter");
        }
        getch();
    }
}
struct node* createNode()
{
    struct node *n;
    n=(struct node*)malloc(sizeof(struct node));
    return(n);
}
void insertDetail()
{
    struct node *n,*temp;
    int i,cmp;
    n=createNode();
    fflush(stdin);
    printf("Enter Student Name: ");
    gets(n->name);
    printf("Enter one of the following course id\n");
    printf("Course Name\tCourse ID\n");
    printf("----------\t----------\n");
    for(i=0;i<5;i++)
    {
        printf("%s\t%s\n",courseName[i],courseId[i]);
    }
    printf("Enter Course Id: ");
    gets(n->cid);
    for(i=0;i<5;i++)
    {
        cmp=stricmp(n->cid,courseId[i]);
        if(cmp==0)
        {
            strcpy(n->cname,courseName[i]);
            n->order=i+1;
        }
    }
```

```
    n->link=NULL;
    if(start==NULL)
        start=n;
    else
    {
        temp=start;
        while(temp->link!=NULL)
            temp=temp->link;
        temp->link=n;
    }
    printf("Data Successfully Write");
}
void displayDetail()
{
    struct node *display;
    display=start;
    if(display==NULL)
        printf("No Record Found");
    else
    {
        arrangeDetail();
        printf("Student Name\tCourse Name\tCourse ID\n");
        printf("------------\t----------\t----------\n");
        while(display!=NULL)
        {
            printf("%s\t\t%s\t\t%s\n",display->name,display->cname,display-
>cid);
            display=display->link;
        }
    }
}
void arrangeDetail()
{
    struct node *first,*second,*temp;
    temp=(struct node*)malloc(sizeof(struct node));
    first=start;
    while(first!=NULL)
    {
        second=first->link;
        while(second!=NULL)
        {
            if(first->order>second->order)
```

```
            {
                strcpy(temp->cname,first->cname);
                strcpy(temp->cid,first->cid);
                strcpy(temp->name,first->name);
                temp->order=first->order;
                strcpy(first->cname,second->cname);
                strcpy(first->cid,second->cid);
                strcpy(first->name,second->name);
                first->order=second->order;
                strcpy(second->cname,temp->cname);
                strcpy(second->cid,temp->cid);
                strcpy(second->name,temp->name);
                second->order=temp->order;
            }
            second=second->link;
        }
        first=first->link;
    }
}
void deleteDetail()
{
    struct node *temp,*release;
    int cmp,count=0;
    char str[30];
    fflush(stdin);
    printf("Enter Name of Student Which You Want to Delete: ");
    gets(str);
    if(start==NULL)
    {
        printf("No Record Found\n");
        printf("Deletion Not Possible");
    }
    else if(start->link==NULL)
    {
        cmp=stricmp(str,start->name);
        if(cmp==0)
        {
            release=start;
            free(release);
            printf("Record Successfully Deleted");
            start=NULL;
        }
```

```
        else
            printf("No Record Found");
    }
    else
    {
        temp=start;
        if(stricmp(str,temp->name)==0)
        {
            release=temp;
            printf("Record Successfully Deleted");
            start=start->link;
            free(release);
        }
        else
        {
            while(temp!=NULL)
            {
                cmp=stricmp(str,temp->link->name);
                if(cmp==0)
                {
                    release=temp->link;
                    temp->link=temp->link->link;
                    printf("Record Successfully Deleted");
                    count++;
                    free(release);
                }
                temp=temp->link;
            }
            if(count==0)
                printf("No Record Found");
        }
    }
}
```