National Institute of Technology Patna

Department of Computer Science and Engineering

MINI PROJECT

Student Enrolment System

Submitted by:

Akshat Singhal 2006059
Faisal Miyan 2006082
Abhishek Singh 2006083
Gyanu Mayank 2006089
Abhay Pratap Srivastava 2006107
Harish 2006209

Submitted to:

Dr. Somaraju Suvvari

Problem Statement:

A university with 100,000 students and 5,000 courses needs to be able to generate two types of reports. The first report lists the registration for each class, and the second report should report lists, by student, the classes that each student is registered for.
Come up with a new data structure for performing the above-mentioned tasks and implement them? Discuss about the time complexities of each task in terms of number of students (m) and number of courses (n).

TASK :-

Given a task to make a data structure that can store 100,000 students and 5,000 courses. We have to create “TWO” list that will do following work :-

- First List

  It should lists the registration for each class

- Second list

  It should list report list, by students ,the classes that each student is registered.

Implementation:-

Pre-requisites key points to understand the code that we generated :-

1. We have 2 variables “numberOfCourses” and “noOfStudents”.
2. Students will be given choice to opt for any of the course.

3.Firstly all details of students will stored , then the user will be given choice whether he wants to see list1 OR list2.

4. A variable “registered” that will help us in displaying list2.

1. Here Node will store Data for each student the course he opted. Data represent the index of the course that the student registered.

```
typedef struct node{
    int data;
    struct node *next;
}
```

2. Here Student is storing three values studentName ,classes (a node that contains the index of the course registered and next for next student), roll no(specified to student)

```
typedef struct student
{
    char *studentName;
    node *classes;
    int rollNo;
}
```

3. This is the function that will create Nodes For student,“val” here represents the index of the course that the student registered.

- TIME COMPLEXITY (worst case) = O(1), because it is simply creating a node .

```c
node *newNode(int val)
{
    node *new = (node *)malloc(sizeof(node));
    new->data = val;
    new->next = NULL;
    return new;
}
```

4. This function will take name of the student as input and will just make a new “student ” type data and will return that “student ” type data.

- TIME COMPLEXITY(worst case)=O(1), because it is just creating a new student type data.

```c
student *newStudent(char *studentName){
    char *studentName2;
    studentName2 = (char *)malloc(30);
    strcpy(studentName2, studentName);
    student *new = (student *)malloc(sizeof(student));
    new->studentName = studentName2;
    new->classes = NULL;
    return new;
}
```

5. This function will display all the courses.

- TIME COMPEXITY(worst case)=O(numberOfCourses), because it has a loop.

```c
void listCourses()
{
    printf("List of Courses open for Registration\n");
    for (int i = 0; i < numberOfCouses; i++)
    {
        printf("%d. %s\n", i + 1, courses[i]);
    }
}
```

6. This function will take the “student” type data and will assign any course that the student wants to opt.

- It will show list of all the courses that the student can take and then will ask for input from user. Student have to enter course number, and then user will have to enter 0 and press “Enter” for conforming the course .
- TIME COMPLEXITY(worst case)=O(numberOfCourses)+O(numberOfCourses)
  =2*O(numberOfCourses)
  ≈O(numberOfCourses)

```
void StartRegistration(student *curr)
{
    listCourses();
    printf("Enter numbers corresponding to your choice of course AND enter 0
and then press Enter to confirm registration\n");
    int i;
    node *classList; // head node for list of opted courses
    for (i = 0; i < numberOfCouses;)
    {
        int t;
        scanf("%d", &t);
        if (t > numberOfCouses || t < 0)
        {
            printf("Please Enter a valid choice\n");
            continue;
        }
        if (t == 0 && i == 0)
        {
            printf("You have to choose atleast one course\n");
            continue;
        }
        if (t == 0)
            break;
```

```
        node *num = newNode(t - 1);
        if (curr->classes == NULL){
            curr->classes = num;
        }
        else{
            // insert at head
            num->next = curr->classes;
            curr->classes = num;
        }
        i++;
    }
}
```

7. This function will ask user to enter his/her name, and after taking username as input it will make “student” data type and will assign the roll number to him/her. And then it will student to register for any available course. And it will store student’s data

- TIME COMPLEXITY(worst case)=O(noOfStudents*noOfCourses)

  ,because it has a loop of “noOfStudents” and for each loop its calling a function whose time complexity of “noOfCourses”.

```
void startRegistration()
{
    char studentName[30];
    for (int i = 0; i < noOfStudents; i++)
    {
        printf("\nEnter your studentName\t: ");
        scanf("%29s", studentName);
        student *student = newStudent(studentName);
        student->rollNo = i + roll_No;
        StartRegistration(student);
```

```
        list[i] = student;
        printf("Registration Successful your rollNo is %d\n", i + roll_No);
    }
    printf("\n\nRegistration Ended\n\n");
    registered = 1;
}
```

8. Here we are using our variable registered that will handle the case in which user wants to see the list2 but any student is not registered.

- It will display the courses that are opted by all the registered courses.
- TIME COMPLEXITY(worst case)=O(noOfStudents*noOfCourses),

  because it also has inner loop that display all the courses registered by the student.

```
void List2()
{
    if(!registered) {
        printf("Please Register and get your roll_Number first\n");
    }
    printf("\n\nList of Courses for each Student\n\n");
    for (int i = 0; i < noOfStudents; i++)
    {
        student *curr = list[i];
        printf("studentName = %s\nRoll_No = %d\nOpted Course List\n", curr-
>studentName,curr->rollNo);
        node *temp = curr->classes;
        int i = 1;
        while (temp != NULL)
        {
            printf("%d.%s\n", i++,courses[temp->data]);
            temp = temp->next;
        }
    }
}
```

9. This function will ask user to enter his/her choice.

- If user entered 1 ,then it will show list1 , i.e., list of opted course by a student by taking his/her roll no as input.
- If user entered 2, then it will call “list2()” function.
- If user entered 3,then it will end the program.
- TIME COMPLEXITY(worst case)=O(noOfStudents*noOfCourses), ,because it will call “list2()” function .

```
void userChoice() {
    int c;
    printf("\nEnter Your Choice \n");
    while (1) {
        printf("1. List opted Course List for a Student.\n");
        printf("2. List All Students with opted Courses.\n");
        printf("3. Quit\n");
        scanf("%d", &c);
        if (c == 1) {
            int rn;
            printf("Enter Your Roll Number\n");
            scanf("%d", &rn);
            if (rn >= 1 && rn <= 1 +noOfStudents ) {
                int idx = rn - 1;
                student *curr = list[idx];
                printf("studentName = %s\n", curr->studentName);
                node *temp = curr->classes;
                // printf("subject = %s %d\n",curr->studentName,temp->data);
                while (temp != NULL)
                {
                    printf("%s\n", courses[temp->data]);
                    // printf("%d\n",temp->data);
                    temp = temp->next;
                }
            }
            else {
                printf("Invalid Roll Number\n");
            }
        }
```

```
        else if(c == 2) {
            List2();
        }
        else if (c == 3) {
                return;
            }
        else {
            printf("Invalid Response\n");
            printf("Try Again\n");
        }
    }
```

Output Screenshots :

```
PS D:\Asss> gcc main.c
PS D:\Asss> .\a.exe
List of Courses open for Registration
1. maths
2. english
3. hindi
4. arts
5. Social Science
6. Humanities

Enter your studentName  : Rajesh
List of Courses open for Registration
1. maths
2. english
3. hindi
4. arts
5. Social Science
6. Humanities
Enter numbers corresponding to your choice of course AND enter 0 and then press Enter to confirm registration
1
0
Registration Successful your rollNo is 1

Enter your studentName  :  Kamlesh
List of Courses open for Registration
1. maths
2. english
3. hindi
4. arts
5. Social Science
6. Humanities
Enter numbers corresponding to your choice of course AND enter 0 and then press Enter to confirm registration
3
0
Registration Successful your rollNo is 2
```

```
Registration Ended



List of Courses for each Student

studentName = Rajesh
Roll_No = 1
Opted Course List
1.maths
Enter Your Roll Number
2
studentName = Kamlesh
hindi
1. List opted Course List for a Student.
2. List All Students with opted Courses.
3. Quit
2


List of Courses for each Student

studentName = Rajesh
Roll_No = 1
Opted Course List
1.maths
studentName = Kamlesh
Roll_No = 2
Opted Course List
1.hindi
1. List opted Course List for a Student.
2. List All Students with opted Courses.
3. Quit
3
Course registration ended
PS D:\Asss> ^S
```

Source Code:

```
#include
<stdio.h>
            #include <stdlib.h>
            #include <string.h>
            #define numberOfCouses 6     // currently only 6 courses for making program
            practical (for question 5,000)
            #define noOfStudents 2     // currently only 2 students for making program
            practical (for question 100,000)
            int roll_No = 1; //base roll no (you can change it)
            int registered = 0; //will be used as if students are registered (in simple
            language it will be used as boolean)
            // List of Available Courses(list - 1)(list can be expanded by changing
            value of courses)
            char *courses[numberOfCouses] = {"maths", "english", "hindi", "arts",
            "Social Science", "Humanities"};
            typedef struct node{
                int data;
                struct node *next;
            } node;
            typedef struct student
            {
                char *studentName;
                node *classes;
                int rollNo;
            } student;
            node *newNode(int val)
            {
                node *new = (node *)malloc(sizeof(node));
                new->data = val;
                new->next = NULL;
                return new;
            }
            student *list[noOfStudents + 1];
            student *newStudent(char *studentName){
                char *studentName2;
                studentName2 = (char *)malloc(30);
                strcpy(studentName2, studentName);
                student *new = (student *)malloc(sizeof(student));
                new->studentName = studentName2;
                new->classes = NULL;
```

```
    return new;
}
void listCourses()
{
    printf("List of Courses open for Registration\n");
    for (int i = 0; i < numberOfCouses; i++)
    {
        printf("%d. %s\n", i + 1, courses[i]);
    }
}
//function for starting the Registration
void StartRegistration(student *curr)
{
    listCourses();
    printf("Enter numbers corresponding to your choice of course AND enter 0
and then press Enter to confirm registration\n");
    int i;
    node *classList;
    for (i = 0; i < numberOfCouses;)
    {
        int t;
        scanf("%d", &t);
        if (t > numberOfCouses || t < 0)
        {
            printf("Please Enter a valid choice\n");
            continue;
        }
        if (t == 0 && i == 0)
        {
            printf("You have to choose atleast one course\n");
            continue;
        }
        if (t == 0)
            break;
        node *num = newNode(t - 1);
        if (curr->classes == NULL){
            curr->classes = num;
        }
        else{
            num->next = curr->classes;
            curr->classes = num;
        }
        i++;
    }
}
void startRegistration()
```

```
{
    char studentName[30];
    for (int i = 0; i < noOfStudents; i++)
    {
        printf("\nEnter your studentName\t: ");
        scanf("%29s", studentName);
        student *student = newStudent(studentName);
        student->rollNo = i + roll_No;
        StartRegistration(student);
        list[i] = student;
        printf("Registration Successful your rollNo is %d\n", i + roll_No);
    }
    printf("\n\nRegistration Ended\n\n");
    registered = 1;
}
void List2()
{
    if(!registered) {
        printf("Please Register and get your roll_Number first\n");
    }
    printf("\n\nList of Courses for each Student\n\n");
    for (int i = 0; i < noOfStudents; i++)
    {
        student *curr = list[i];
        printf("studentName = %s\nRoll_No = %d\nOpted Course List\n", curr-
>studentName,curr->rollNo);
        node *temp = curr->classes;
        int i = 1;
        while (temp != NULL)
        {
            printf("%d.%s\n", i++,courses[temp->data]);
            temp = temp->next;
        }
    }
}
void userChoice() {
    int c;
    printf("\nEnter Your Choice \n");
    while (1) {
        printf("1. List opted Course List for a Student.\n");
        printf("2. List All Students with opted Courses.\n");
        printf("3. Quit\n");
        scanf("%d", &c);
        if (c == 1) {
            int rn;
            printf("Enter Your Roll Number\n");
```

```
            scanf("%d", &rn);
            if (rn >= 1 && rn <= 1 +noOfStudents ) {
                int idx = rn - 1;
                student *curr = list[idx];
                printf("studentName = %s\n", curr->studentName);
                node *temp = curr->classes;
                while (temp != NULL)
                {
                    printf("%s\n", courses[temp->data]);
                    temp = temp->next;
                }
            }
            else {
                printf("Invalid Roll Number\n");
            }
        }
        else if(c == 2) {
            List2();
        }
        else if (c == 3) {
                return;
            }
        else {
            printf("Invalid Response\n");
            printf("Try Again\n");
        }
    }
}
void main()
{
    listCourses();
    startRegistration();
    List2();
    userChoice();
    printf("Course registration ended\n");
}
```

GitHub Link:

https://github.com/abhi070902/MiniProject

Please visit here for the source code.

THANK YOU

National Institute of Technology Patna
Patna, Bihar (800005), India
Phone: +91-0612-237 1715 / 237 2715
FAX : +91-0612-2670631 , 0612-2660480