Name-Atharva Sanjay shevate

Roll no-2

Srn-202201727

div-E(E1)

Sub-Ds cie 3 report

SCHOOL MANAGEMENT

INTRODUCTION

- **The provided application is a School Management System implemented in C. Its primary purpose is to manage student data within a school. Users can perform various operations to maintain and access information about students, including adding new students, finding students by roll number or first name, searching for students in specific courses, counting the total number of students, deleting students, updating student information, and displaying the details of all registered students. This system is designed to provide a simple command-line interface for**

schools or educational institutions to efficiently handle and organize student records.

Features

1. Add Student:
    1. Description: This feature allows users to add a new student to the system.
    2. Functionality: Users can input the student's roll number, first name, and course. The system then stores this information in its database of students, increasing the student count.
2. Find Student by Roll Number:
    1. Description: Users can search for a specific student using their roll number.
    2. Functionality: The system searches its database for the provided roll number and displays the student's information if found.
3. Find Student by First Name:
    1. Description: This feature enables users to search for students by their first name.
    2. Functionality: The system performs a search based on the first name and displays

information about all students with matching first names.

4. Find Students in a Course:
    1. Description: Users can find all students registered in a particular course.
    2. Functionality: The system searches for students enrolled in the specified course and lists their information.
5. Count of Students:
    1. Description: This feature provides the total count of students in the system.
    2. Functionality: The system calculates and displays the current number of registered students.
6. Delete Student:
    1. Description: Users can delete a student's record from the system.
    2. Functionality: The system prompts users for a roll number, and if the student with that roll number exists, it is removed from the database.

CODE

```
#include <stdio.h>
#include <string.h>

#define MAX_STUDENTS 100

struct Student {
    int rollNumber;
    char firstName[50];
    char course[50];
};

struct Student stack[MAX_STUDENTS];
int stackTop = -1; // Initialize stack top

struct Student queue[MAX_STUDENTS];
int queueFront = 0; // Initialize queue front
int queueRear = 0; // Initialize queue rear

void clearInputBuffer() {
```

```
    int c;
    while ((c = getchar()) != '\n' && c != EOF);
}

void push(struct Student student) {
    if (stackTop < MAX_STUDENTS - 1) {
        stack[++stackTop] = student;
        printf("\n\033[1;32mStudent added to the stack
successfully!\033[0m\n");
    } else {
        printf("\n\033[1;31mError: Stack is
full.\033[0m\n");
    }
}

void enqueue(struct Student student) {
    if (queueRear < MAX_STUDENTS) {
        queue[queueRear++] = student;
    } else {
```

```
        printf("\n\033[1;31mError: Queue is
full.\033[0m\n");
    }
}

void findStudentByRollNumber(int rollNumber) {
    for (int i = 0; i < stackTop + 1; i++) {
        if (stack[i].rollNumber == rollNumber) {
            printf("\n\033[1;34mStudent found in the
stack:\033[0m\n");
            printf("Roll Number: %d, Name: %s, Course:
%s\n", stack[i].rollNumber, stack[i].firstName,
stack[i].course);
            return;
        }
    }
    printf("\n\033[1;31mStudent not found in the
stack.\033[0m\n");
}

void findStudentByName(const char *name) {
```

```
    for (int i = 0; i < stackTop + 1; i++) {
        if (strcmp(stack[i].firstName, name) == 0) {
            printf("\n\033[1;34mStudent found in the stack:\033[0m\n");
            printf("Roll Number: %d, Name: %s, Course: %s\n", stack[i].rollNumber, stack[i].firstName, stack[i].course);
            return;
        }
    }
    printf("\n\033[1;31mStudent not found in the stack.\033[0m\n");
}

void findStudentByCourse(const char *course) {
    for (int i = 0; i < stackTop + 1; i++) {
        if (strcmp(stack[i].course, course) == 0) {
            printf("\n\033[1;34mStudent found in the stack:\033[0m\n");
```

```
            printf("Roll Number: %d, Name: %s, Course:
%s\n", stack[i].rollNumber, stack[i].firstName,
stack[i].course);
            return;
        }
    }
    printf("\n\033[1;31mStudent not found in the
stack.\033[0m\n");
}

int countStudents() {
    return stackTop + 1;
}

void updateStudent(int rollNumber, struct Student
updatedStudent) {
    for (int i = 0; i < stackTop + 1; i++) {
        if (stack[i].rollNumber == rollNumber) {
            stack[i] = updatedStudent;
            printf("\n\033[1;32mStudent information
updated successfully!\033[0m\n");
```

```
            return;
        }
    }
    printf("\n\033[1;31mStudent not found in the stack.\033[0m\n");
}

void deleteStudent(int rollNumber) {
    for (int i = 0; i < stackTop + 1; i++) {
        if (stack[i].rollNumber == rollNumber) {
            for (int j = i; j < stackTop; j++) {
                stack[j] = stack[j + 1];
            }
            stackTop--;
            printf("\n\033[1;32mStudent deleted from the stack successfully!\033[0m\n");
            return;
        }
    }
```

```
    printf("\n\033[1;31mStudent not found in the stack.\033[0m\n");
}

void displayMenu() {
    printf("\n\033[1;34mSchool Management System\033[0m\n"); // Blue header
    printf("\033[1;33m1. Add Student\n");
    printf("2. Find Student by Roll Number\n");
    printf("3. Find Student by Name\n");
    printf("4. Find Student by Course\n");
    printf("5. Count All Students\n");
    printf("6. Update Student\n");
    printf("7. Delete Student\n");
    printf("8. Exit\033[0m\n"); // Yellow menu items
}

int main() {
    int choice;
```

```
    while (1) {
        displayMenu();
        printf("\n\033[1;36mEnter your choice: \033[0m");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                if (stackTop < MAX_STUDENTS - 1) {
                    struct Student newStudent;
                    printf("\n\033[1;36mEnter Student Details:\033[0m\n");
                    printf("Roll Number: ");
                    scanf("%d", &newStudent.rollNumber);
                    clearInputBuffer();
                    printf("First Name: ");
                    fgets(newStudent.firstName, sizeof(newStudent.firstName), stdin);

newStudent.firstName[strcspn(newStudent.firstName, "\n")] = '\0'; // Remove newline
```

```
            printf("Course: ");
            fgets(newStudent.course,
sizeof(newStudent.course), stdin);

newStudent.course[strcspn(newStudent.course,
"\n")] = '\0'; // Remove newline
            push(newStudent);
        } else {
            printf("\n\033[1;31mError: Stack is
full.\033[0m\n");
        }
        break;
    case 2:
        printf("\n\033[1;36mEnter Roll Number to
find: \033[0m");
        int rollNumberToFind;
        scanf("%d", &rollNumberToFind);

findStudentByRollNumber(rollNumberToFind);
        break;
    case 3:
```

```
            printf("\n\033[1;36mEnter Name to find: \033[0m");
            char nameToFind[50];
            clearInputBuffer();
            fgets(nameToFind, sizeof(nameToFind), stdin);
            nameToFind[strcspn(nameToFind, "\n")] = '\0'; // Remove newline
            findStudentByName(nameToFind);
            break;
        case 4:
            printf("\n\033[1;36mEnter Course to find: \033[0m");
            char courseToFind[50];
            clearInputBuffer();
            fgets(courseToFind, sizeof(courseToFind), stdin);
            courseToFind[strcspn(courseToFind, "\n")] = '\0'; // Remove newline
            findStudentByCourse(courseToFind);
            break;
```

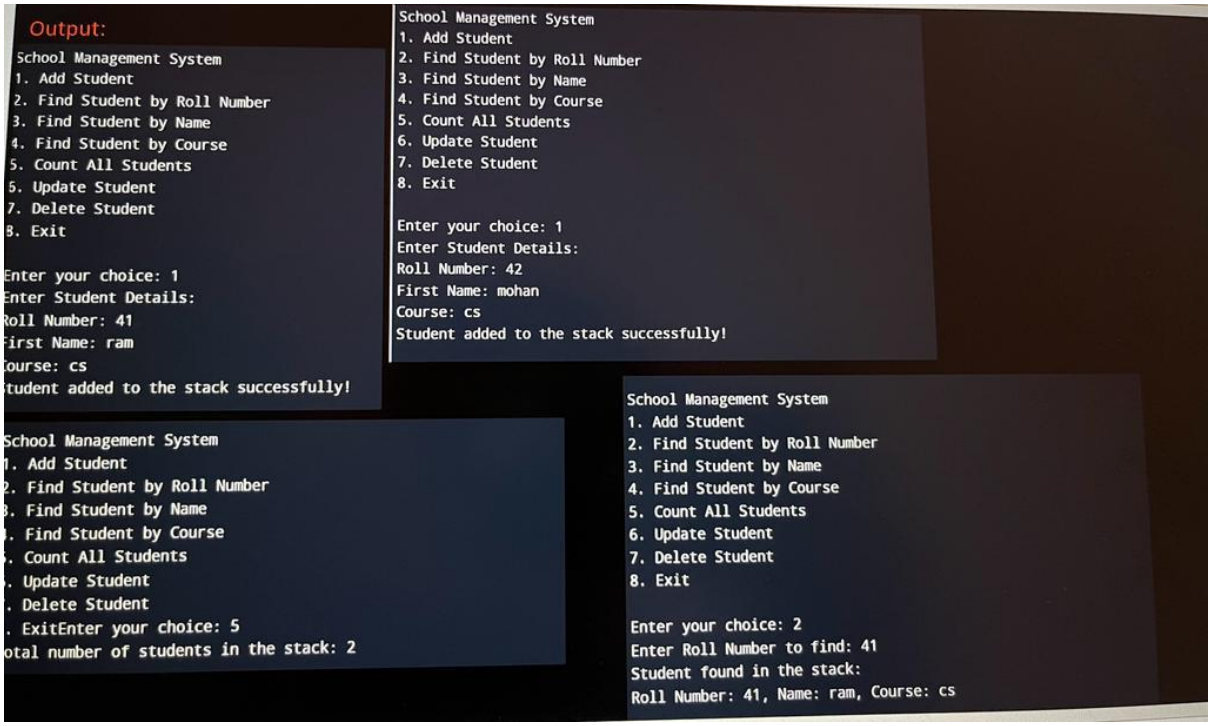
```
        case 5:
            printf("\n\033[1;32mTotal number of
students in the stack: %d\033[0m\n",
countStudents());
            break;
        case 6:
            printf("\n\033[1;36mEnter Roll Number to
update: \033[0m");
            int rollNumberToUpdate;
            scanf("%d", &rollNumberToUpdate);
            struct Student updatedStudent;
            printf("\n\033[1;36mEnter Updated Student
Details:\033[0m\n");
            printf("Roll Number: ");
            scanf("%d", &updatedStudent.rollNumber);
            clearInputBuffer();
            printf("First Name: ");
            fgets(updatedStudent.firstName,
sizeof(updatedStudent.firstName), stdin);
```

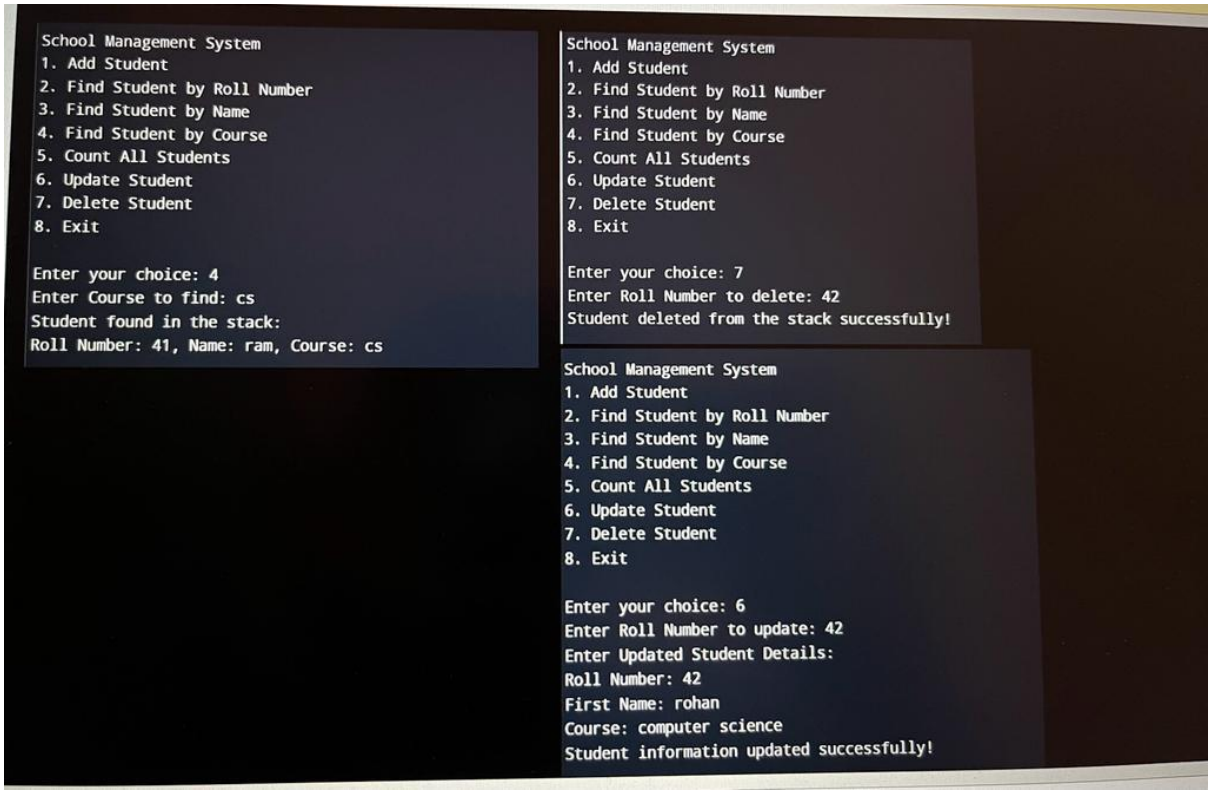
```
updatedStudent.firstName[strcspn(updatedStudent.firstName, "\n")] = '\0'; // Remove newline
            printf("Course: ");
            fgets(updatedStudent.course, sizeof(updatedStudent.course), stdin);

updatedStudent.course[strcspn(updatedStudent.course, "\n")] = '\0'; // Remove newline
            updateStudent(rollNumberToUpdate, updatedStudent);
            break;
        case 7:
            printf("\n\033[1;36mEnter Roll Number to delete: \033[0m");
            int rollNumberToDelete;
            scanf("%d", &rollNumberToDelete);
            deleteStudent(rollNumberToDelete);
            break;
        case 8:
            return 0;
```

```
            default:
                printf("\n\033[1;31mInvalid choice. Please
try again.\033[0m\n");
        }
    }

    return 0;
}
```

barobar ahe ka

output


Output:
School Management System
1. Add Student
2. Find Student by Roll Number
3. Find Student by Name
4. Find Student by Course
5. Count All Students
6. Update Student
7. Delete Student
8. Exit
Enter your choice: 1
Enter Student Details:
Roll Number: 41
First Name: ram
Course: cs
Student added to the stack successfully!
School Management System
1. Add Student
2. Find Student by Roll Number
3. Find Student by Name
4. Find Student by Course
5. Count All Students
6. Update Student
7. Delete Student
8. Exit
Enter your choice: 1
Enter Student Details:
Roll Number: 42
First Name: mohan
Course: cs
Student added to the stack successfully!
School Management System
1. Add Student
2. Find Student by Roll Number
3. Find Student by Name
4. Find Student by Course
5. Count All Students
6. Update Student
7. Delete Student
8. Exit
Enter your choice: 5
Total number of students in the stack: 2
School Management System
1. Add Student
2. Find Student by Roll Number
3. Find Student by Name
4. Find Student by Course
5. Count All Students
6. Update Student
7. Delete Student
8. Exit
Enter your choice: 2
Enter Roll Number to find: 41
Student found in the stack:
Roll Number: 41, Name: ram, Course: cs


School Management System
1. Add Student
2. Find Student by Roll Number
3. Find Student by Name
4. Find Student by Course
5. Count All Students
6. Update Student
7. Delete Student
8. Exit
Enter your choice: 4
Enter Course to find: cs
Student found in the stack:
Roll Number: 41, Name: ram, Course: cs
School Management System
1. Add Student
2. Find Student by Roll Number
3. Find Student by Name
4. Find Student by Course
5. Count All Students
6. Update Student
7. Delete Student
8. Exit
Enter your choice: 7
Enter Roll Number to delete: 42
Student deleted from the stack successfully!
School Management System
1. Add Student
2. Find Student by Roll Number
3. Find Student by Name
4. Find Student by Course
5. Count All Students
6. Update Student
7. Delete Student
8. Exit
Enter your choice: 6
Enter Roll Number to update: 42
Enter Updated Student Details:
Roll Number: 42
First Name: rohan
Course: computer science
Student information updated successfully!

Result Analysis

1. **Efficient Student Record Management: Schools can efficiently manage their student records, making it easy to add, find, update, or delete student information as needed.**
2. **Quick Access to Information: Staff and administrators can swiftly access student data, enabling them to provide better support to students and make informed decisions.**
3. **Reduced Administrative Workload: The automation of data management tasks reduces the administrative workload, streamlining the school's operations.**

conclusion

school management software is an essential tool for educational institutions. It helps to streamline administrative tasks, improve communication, and enhance overall efficiency. By using school management software, schools can improve student

outcomes and provide a better education for their students.