INSTITUTE OF TECHNOLOGY AND MANAGEMENT
SKILLS UNIVERSITY,
KHARGHAR, NAVI MUMBAI

C++ PROGRAMMING LAB

Prepared by:

Name of Student: _Shikha singh_____

Roll No: _25_____

Batch: 2023-27

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Exp. No	List of Experiment
	Write a program to find the roots of a quadratic equation.
	Write a program to calculate the power of a number using a loop.
	Write a program to check if a given string, is a palindrome.
	Write a program that simulates a simple ATM machine, allowing users to check their balance, deposit, or withdraw money using a switch statement.
	Write a program that finds the largest among three numbers using nested if-else statements
	Write a program that determines the grade of a student based on their marks of 5 subjects using if-else-if ladder.
	Write a program to find the sum of digits of a number until it becomes a single-digit number.
	Write a program to print a Pascal's triangle using nested loops.
	Write a program to calculate the sum of series 1/1! + 2/2! + 3/3! + ... + N/N! using nested loops.
0	Write a program to create an array of strings and display them in alphabetical order.
1	Write a program that checks if an array is sorted in ascending order.
2	Write a program to calculate the sum of elements in each row of a matrix.
3	Write a program to generate all possible permutations of a string.
4	Create a C++ program to print the following pattern: ***** * * * * * * *****
5	Write a C++ program to display the following pattern: 1 232

	34543 4567654 34543 232

6	Write a program to creating an inventory management system for a small store. The system should use object-oriented principles in C++. Your program should have the following features:
	• Create a **Product** class that represents a product in the inventory. Each **Product** object should have the following attributes:
	  • Product ID (an integer)
	  • Product Name (a string)
	  • Price (a floating-point number)
	  • Quantity in stock (an integer)
	• Implement a parameterized constructor for the **Product** class to initialize the attributes when a new product is added to the inventory.

7	Write a program to manage student records. Create a class Student with attributes such as name, roll number, and marks. Implement methods for displaying student details, adding new students, and calculating the average marks of all students in the record system.
8	Write a program that implements a basic calculator. Use a class Calculator with methods to perform addition, subtraction, multiplication, and division of two numbers. The program should allow the user to input two numbers and select an operation to perform.
9	Write a program to simulate a simple online shop. Create a class Product with attributes like name, price, and quantity in stock. Implement methods for adding products to the shopping cart, calculating the total cost, and displaying the contents of the cart.
0	Write a program to manage student grades for a classroom. Create a class Student with attributes for student name and an array to store grades. Implement methods for adding grades, calculating the average grade, and displaying the student's name and grades. Use constructors and destructors to initialize and release resources.

Name of Student: _Shikha singh_____

Roll Number: __25_____

Experiment No: 17

Title:17.Write a program to manage student records. Create a class Student with attributes such as name, roll number, and marks. Implement methods for displaying student details, adding new students, and calculating the average marks of all students in the record system.

Theory:

- **Student Class:**
  - **The Student class represents a student with attributes like name, roll number, and marks.**
  - **It has getter and setter methods for each attribute, ensuring encapsulation.**
- **StudentRecordSystem Class:**
  - **The StudentRecordSystem class manages a collection of student records using a vector.**
  - **It provides methods to add students, display all students, and calculate/display the average marks of all students.**

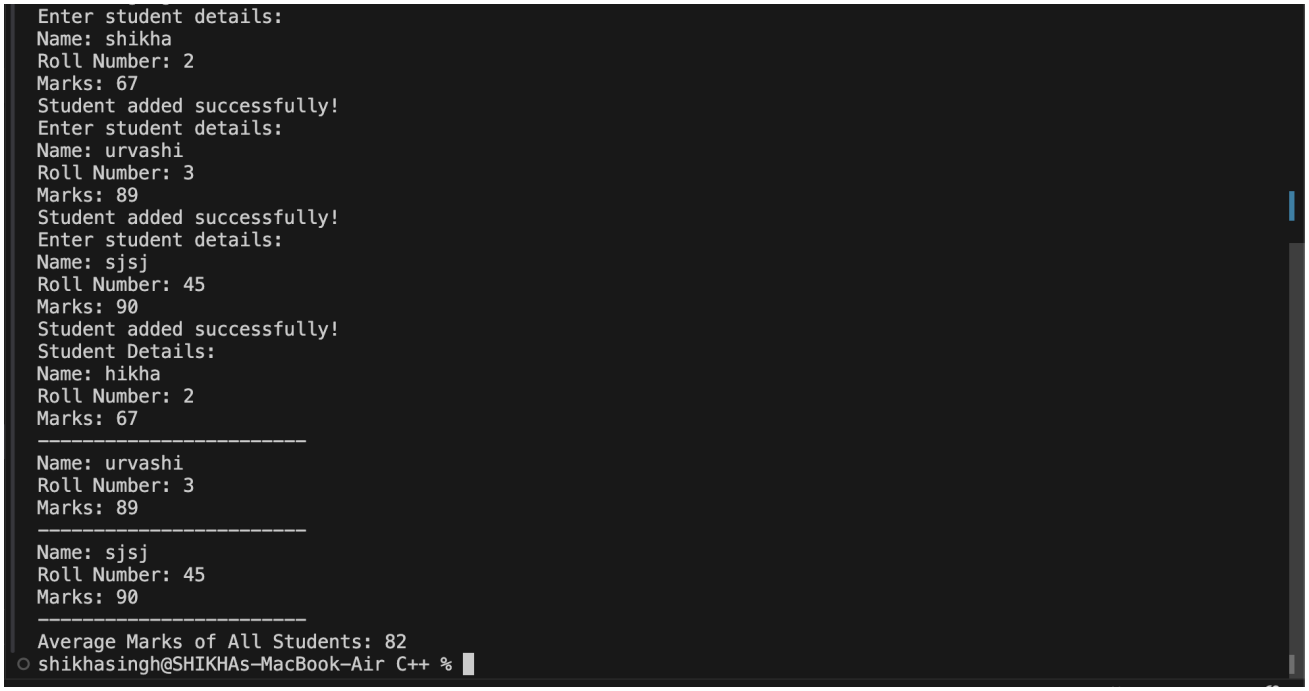
Code:

```
#include <iostream>
using namespace std;
class Student {
private:
   string name;
   int rollNumber;
   float marks;
public:
   Student(const string& studentName, int studentRollNumber, float studentMarks)
      : name(studentName), rollNumber(studentRollNumber), marks(studentMarks) {}
   void displayStudentDetails() const {
      cout << "Name: " << name << endl;
      cout << "Roll Number: " << rollNumber << endl;
```

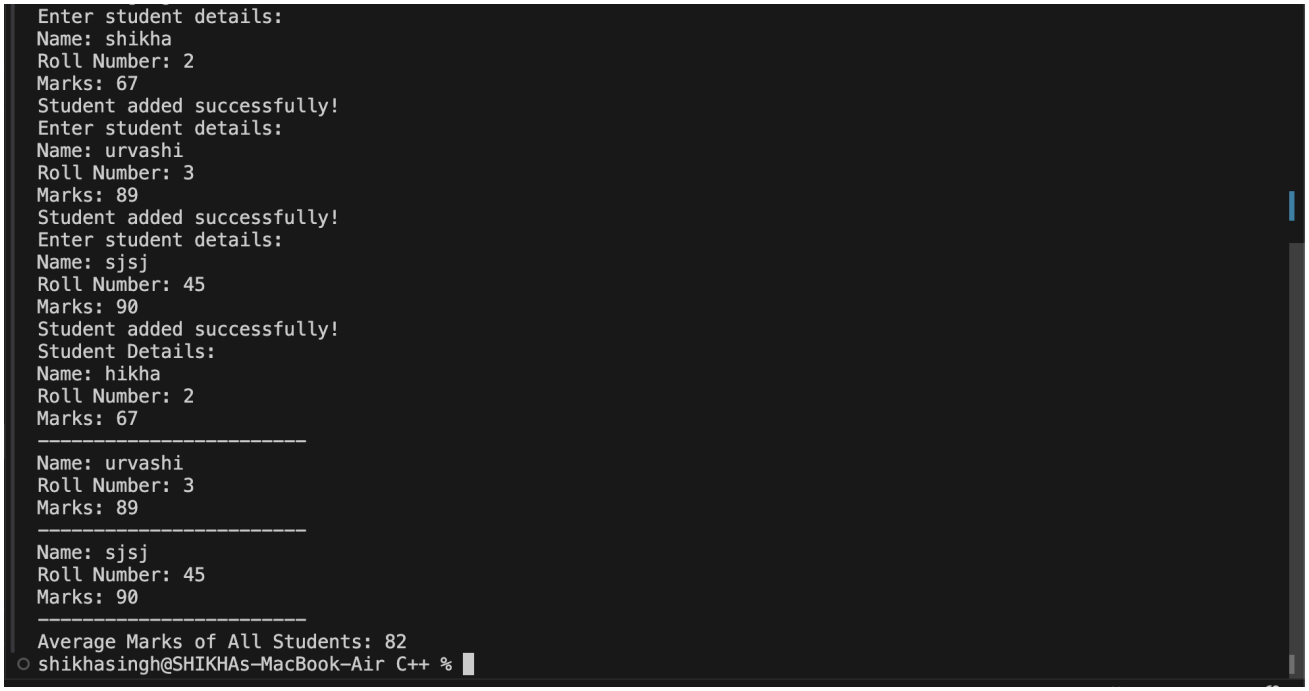
```
        cout << "Marks: " << marks << endl;
        cout << "------------------------\n";
    }
    const string& getName() const { return name; }
    void setName(const string& studentName) { name = studentName; }
    int getRollNumber() const { return rollNumber; }
    void setRollNumber(int studentRollNumber) { rollNumber = studentRollNumber; }
    float getMarks() const { return marks; }
    void setMarks(float studentMarks) { marks = studentMarks; }
};
class StudentRecordSystem {
private:
    vector<Student> studentRecords;
public:
    void addStudent() {
        string name;
        int rollNumber;
        float marks;
        cout << "Enter student details:\n";
        cout << "Name: ";
        cin.ignore();
        getline(cin, name);
        cout << "Roll Number: ";
        cin >> rollNumber;
        cout << "Marks: ";
        cin >> marks;
        studentRecords.push_back(Student(name, rollNumber, marks));
        cout << "Student added successfully!\n";
    }
    void displayAllStudents() const {
        if (studentRecords.empty()) {
            cout << "No student records available.\n";
        } else {
            cout << "Student Details:\n";
            for (const auto& student : studentRecords) {
                student.displayStudentDetails();
            }
        }
    }
    void calculateAndDisplayAverageMarks() const {
        if (studentRecords.empty()) {
            cout << "No student records available.\n";
        } else {
            float totalMarks = 0;
            for (const auto& student : studentRecords) {
                totalMarks += student.getMarks();
            }
            float averageMarks = totalMarks / studentRecords.size();
            cout << "Average Marks of All Students: " << averageMarks << std::endl;
        }
    }
};
```

```
int main() {
    StudentRecordSystem recordSystem;
    recordSystem.addStudent();
    recordSystem.addStudent();
    recordSystem.addStudent();
    recordSystem.displayAllStudents();
    recordSystem.calculateAndDisplayAverageMarks();
    return 0;
}
```

Output: (screenshot)



Test Case: Any two (screenshot)



Conclusion:

- **The code creates a simple system for managing student records.**
- **The Student class encapsulates student details and provides methods for accessing and modifying them.**
- **The StudentRecordSystem class handles the collection of student records and provides functionality for adding students, displaying all students, and calculating the average marks.**
- **The program demonstrates the usage of classes, vectors, and basic I/O operations.**