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: 20

Title: 20. 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.

Theory:

Class Definition (Student):

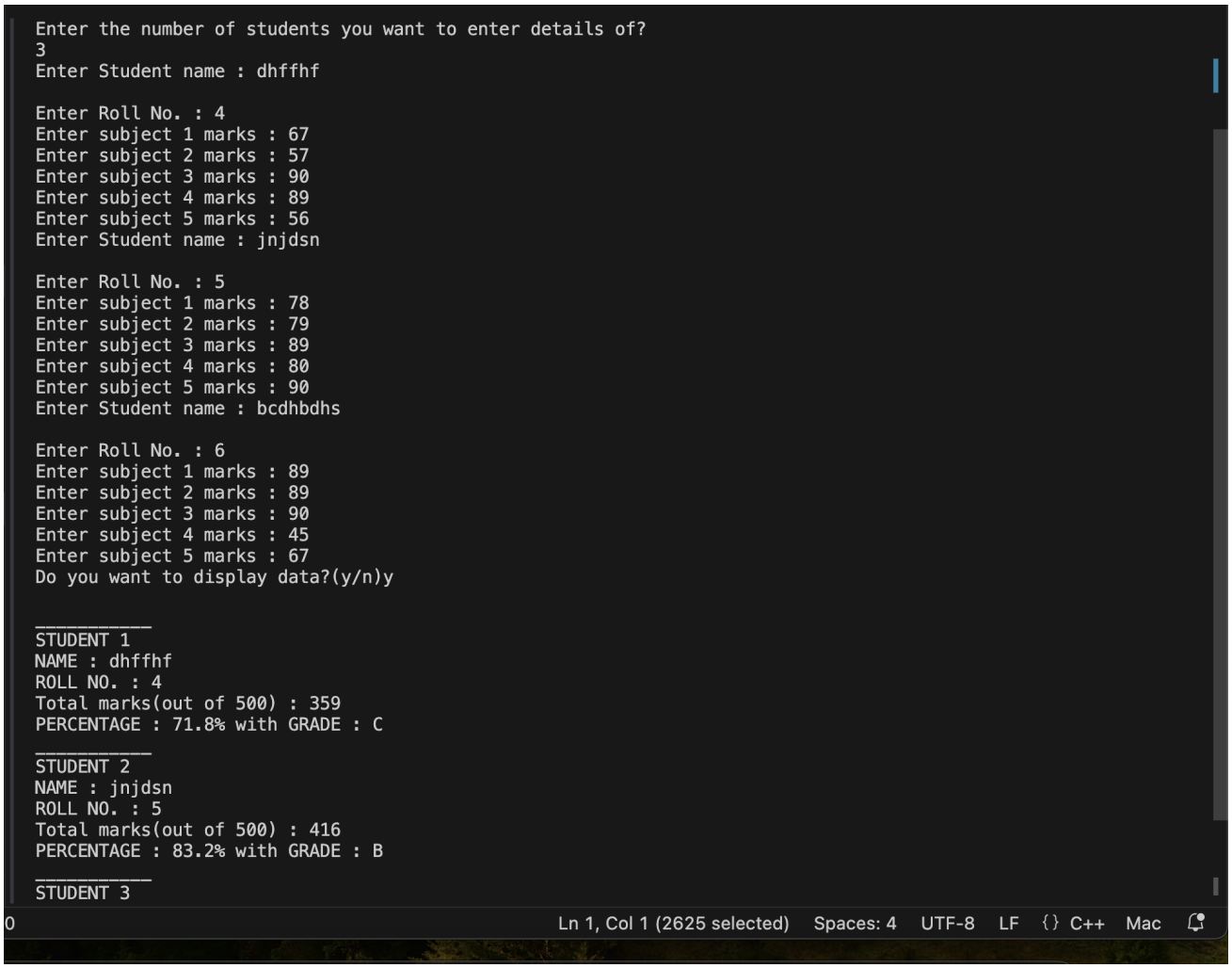
- **The Student class is defined to encapsulate information about a student, such as name, roll number, marks, and grade.**
- **Private data members (roll_no, name, grade, marks, average) are used for storing student details.**
- **Constructor (Student):**
- **The class includes a default constructor that initializes the data members to default values (e.g., average = 0, marks = 0).**
- **Member Functions (getinfo, displayinfo):**
- **getinfo(): Takes input from the user for the student's name, roll number, and marks for five subjects. It calculates the average marks and checks for valid input.**
- **displayinfo(): Displays the student's name, roll number, total marks, percentage, and corresponding grade based on the average marks.**
- **Destructor (~Student):**
- **The class includes a destructor (empty in this case) which is called when an object of the class goes out of scope.**
- **Main Function (main):**
- **The main function initializes the number of students (n) and creates an array of Student objects (stud).**
- **It then uses a loop to call the getinfo method for each student to input their details.**
- **The program prompts the user if they want to display the data and, if yes, calls the displayinfo method for each student.**
- **Input Validation (rerun Label):**
- **A label (rerun) is used for input validation, allowing the user to retry entering marks if an invalid value is provided.**
- **Output Delay (sleep Function):**
- **The sleep function is used to introduce a delay of 2 seconds before the program terminates, providing a short pause after displaying student information.**

Code:

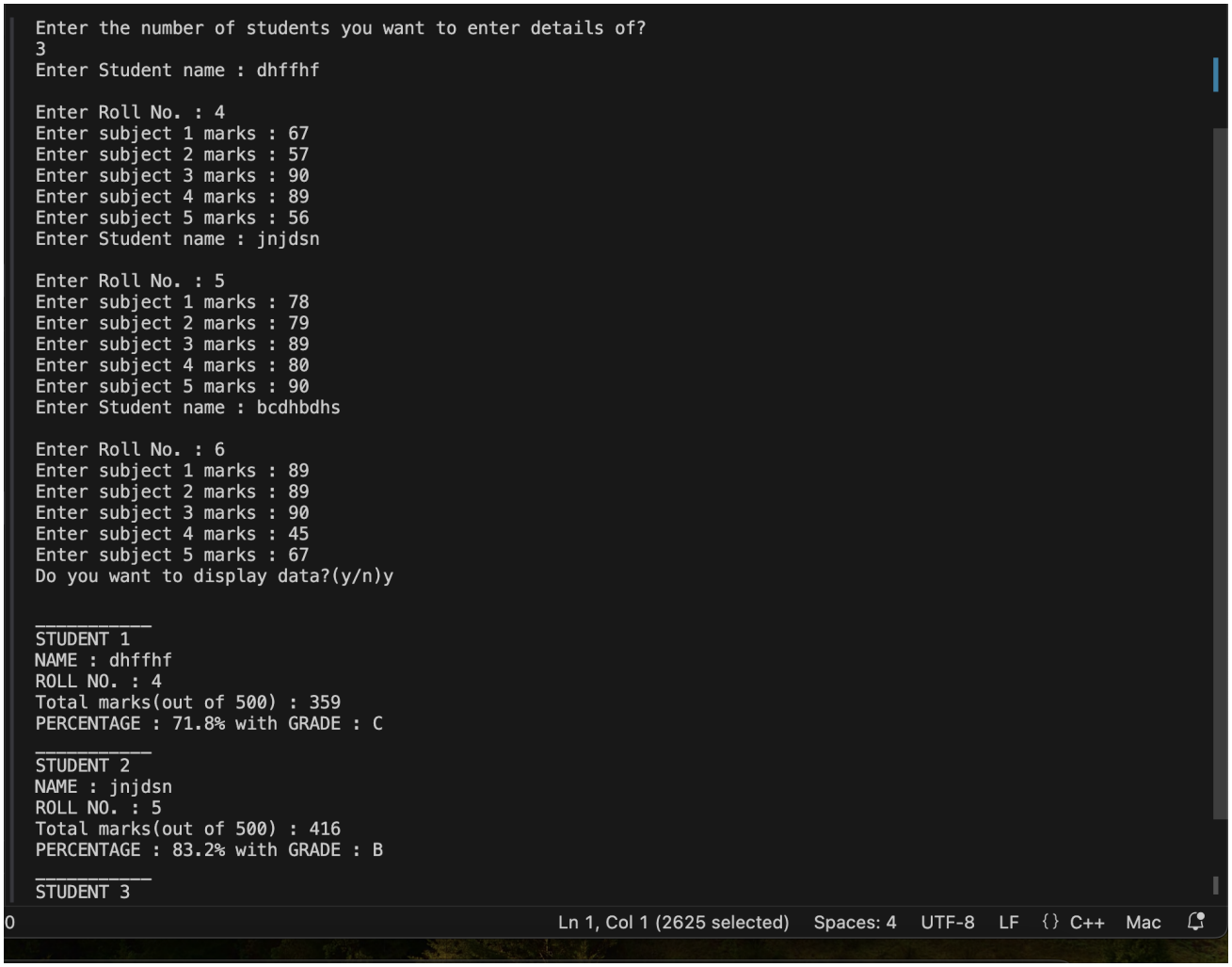
```
#include <iostream>
#include <unistd.h>
using namespace std;
class Student
{
   private:
      int roll_no ;
      string name , grade;
      float marks , average;
   public:
      Student(){
         average = 0;
         marks = 0;
         name = "";
         grade = "";
         roll_no = 0;
      };
      void getinfo()
      {
         cout<<"Enter Student name : ";
         cin>>name;
         cout<<"\nEnter Roll No. : ";
         cin>>roll_no;
         for(int i = 1 ; i <= 5 ; i++)
         {
            rerun:
            cout<<"Enter subject "<<i<<" marks : ";
            cin>>marks;
            if(marks > 100 || marks < 0)
            {
               cout<<"marks should not exceed 100 and Should not be negative :) \n";
               goto rerun;
            }
            average += marks;
         }
      }
      void displayinfo()
      {
         cout<<"NAME : "<<name<<"\n";
         cout<<"ROLL NO. : "<<roll_no<<"\n";
         cout<<"Total marks(out of 500) : "<<average<<"\n";
         average /= 5.00;
         if(average >= 85 && average < 95)
         {
            cout<<"PERCENTAGE : "<<average<<"% with GRADE : A";
         }
         else if(average >= 95)
         {
            cout<<"PERCENTAGE : "<<average<<"% with GRADE : A+";
```

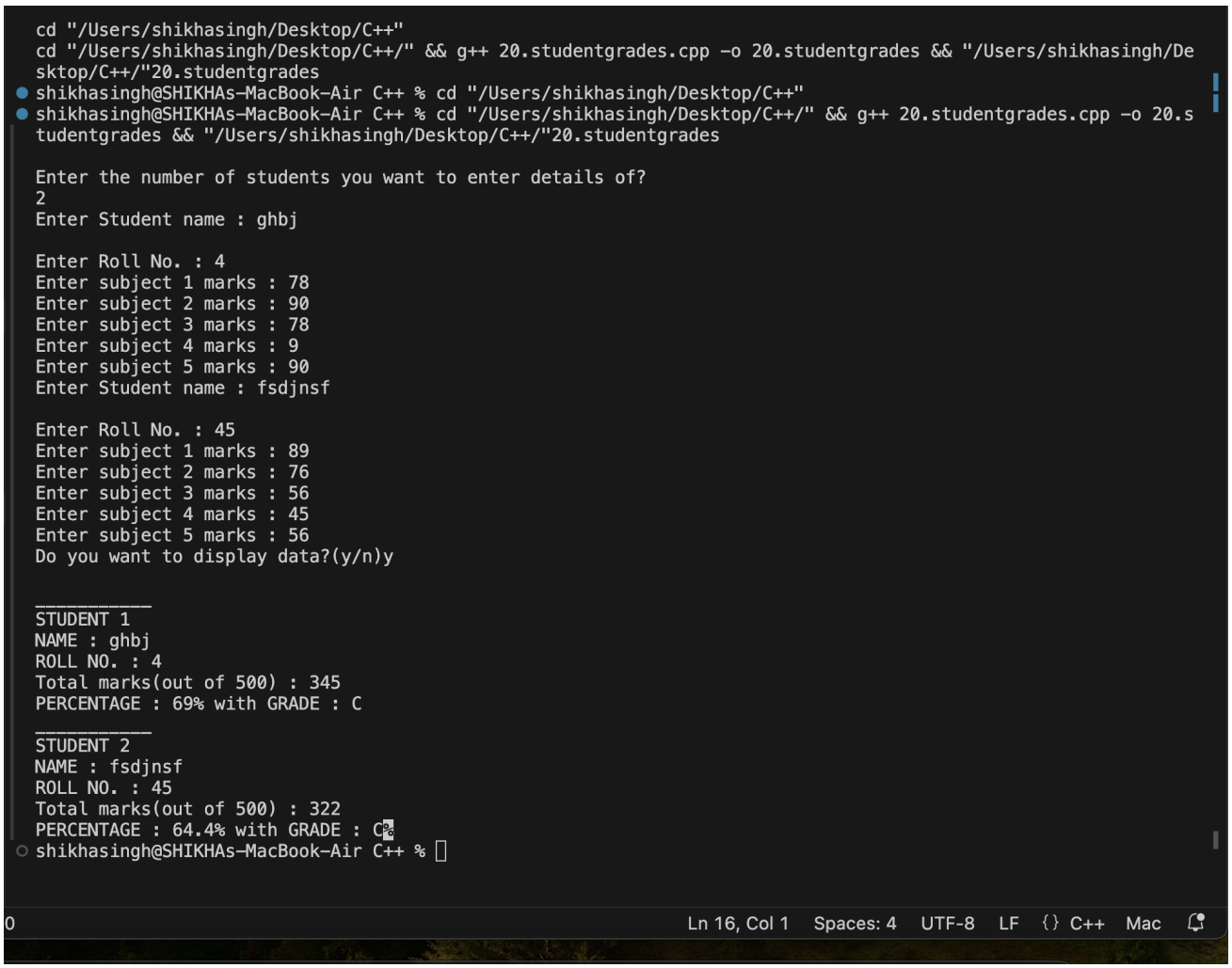
```
        }
        else if(average >= 75 && average < 85)
        {
           cout<<"PERCENTAGE : "<<average<<"% with GRADE : B";
        }
        else if(average < 75 && average >= 60)
        {
           cout<<"PERCENTAGE : "<<average<<"% with GRADE : C";
        }
        else if(average < 60 && average > 33)
        {
           cout<<"PERCENTAGE : "<<average<<"% with GRADE : D";
        }
        else
        {
           cout<<"PERCENTAGE : "<<average<<"% and failed class with GRADE : F";
        }
        sleep(2);
    }

    ~Student(){

    }
};
int main()
{
   int n;
   char y;
   cout<<"\nEnter the number of students you want to enter details of? \n";
   cin>>n;
   Student stud[n];
   for(int i = 0 ; i<n ; i++)
   {
      stud[i].getinfo();
   }
   cout<<"Do you want to display data?(y/n)";
   cin>>y;
   if(toupper(y) == 'Y')
   {
      for(int i = 0 ; i<n ; i++)
      {
         cout<<"\n___________\n";
         cout<<"STUDENT "<<i+1<<"\n";
         stud[i].displayinfo();

      }
   }
   else
   {
      return 0;
   }
}
```

Output: (screenshot)



Test Case: Any two (screenshot)





Conclusion:

- **The program is designed to manage student information for multiple students, allowing the user to input details such as name, roll number, and marks.**
- **Input validation is implemented to ensure that marks are within the valid range (0 to 100).**
- **The program calculates and displays the total marks, percentage, and corresponding grade for each student based on their average marks.**
- **The use of a class (Student) encapsulates related functionalities and data, promoting code organization and readability.**
- **The program concludes by displaying the information of each student and providing a short delay before exiting.**