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

xp. 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: 16

Title:

16.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.

Theory: **Class and Object:**

- **In C++, a class is a user-defined data type that encapsulates data members and member functions. Objects are instances of classes.**
- **The Product class in this program represents a blueprint for creating product objects in the inventory.**
- **Private Members:**
- **The attributes (productId, productName, price, and quantityInStock) are declared as private members of the Product class. This encapsulation ensures that these attributes are accessible only within the class, adhering to the principle of data hiding.**
- **Parameterized Constructor:**
- **The Product class has a parameterized constructor, which is a special member function that initializes the attributes when a new product is added to the inventory. This constructor allows for the creation of objects with specific initial values.**
- **Getter Methods:**

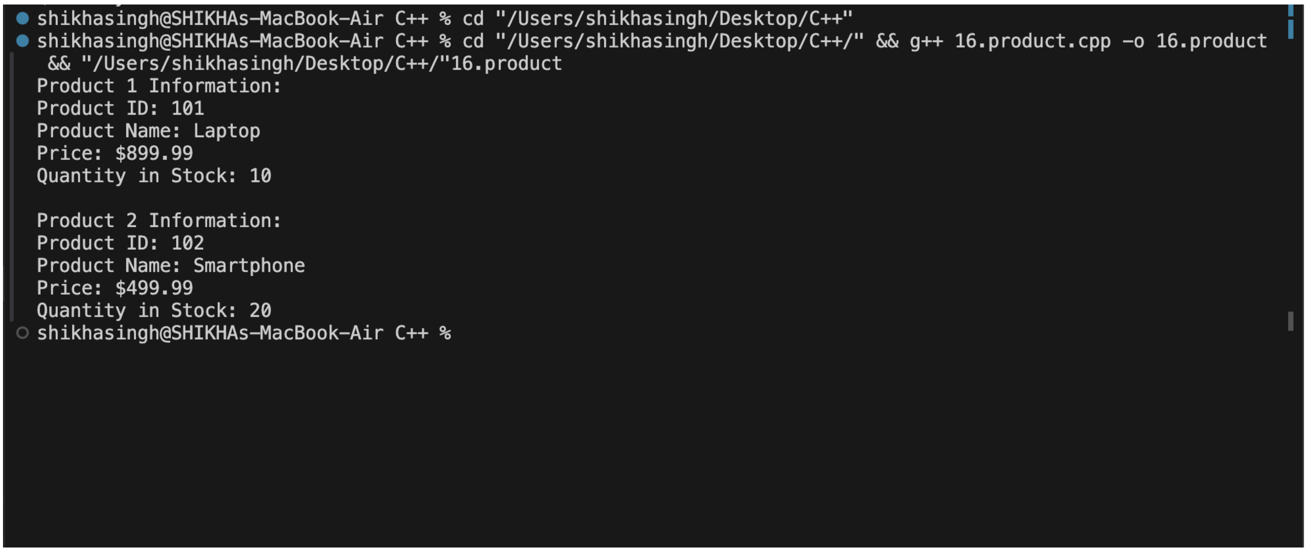
  - **Getter methods (getProductId, getProductName, getPrice, getQuantityInStock) are provided to access the private attributes of the Product class. These methods allow controlled access to the class's internal state.**
- **Main Function:**
  - **The main function demonstrates the creation of two Product objects, initializing them with specific values using the parameterized constructor. It also displays the information of each product using the displayProductInfo method.**

Code:

```
#include <iostream>

#include <string>
using namespace std;
class Product {
private:
  int productId;
  string productName;
  float price;
  int quantityInStock;
public:
  Product(int id, const string& name, float pr, int quantity)
    : productId(id), productName(name), price(pr), quantityInStock(quantity) {}
  void displayProductInfo() const {
    cout << "Product ID: " << productId << endl;
    cout << "Product Name: " << productName << endl;
    cout << "Price: $" << price << endl;
    cout << "Quantity in Stock: " << quantityInStock << endl;
  }
  int getProductId() const {
    return productId;
  }
  const string& getProductName() const {
    return productName;
  }
  float getPrice() const {
    return price;
  }
  int getQuantityInStock() const {
    return quantityInStock;
  }
};
int main() {

  Product product1(101, "Laptop", 899.99, 10);
  Product product2(102, "Smartphone", 499.99, 20);

  cout << "Product 1 Information:" << endl;
  product1.displayProductInfo();
```

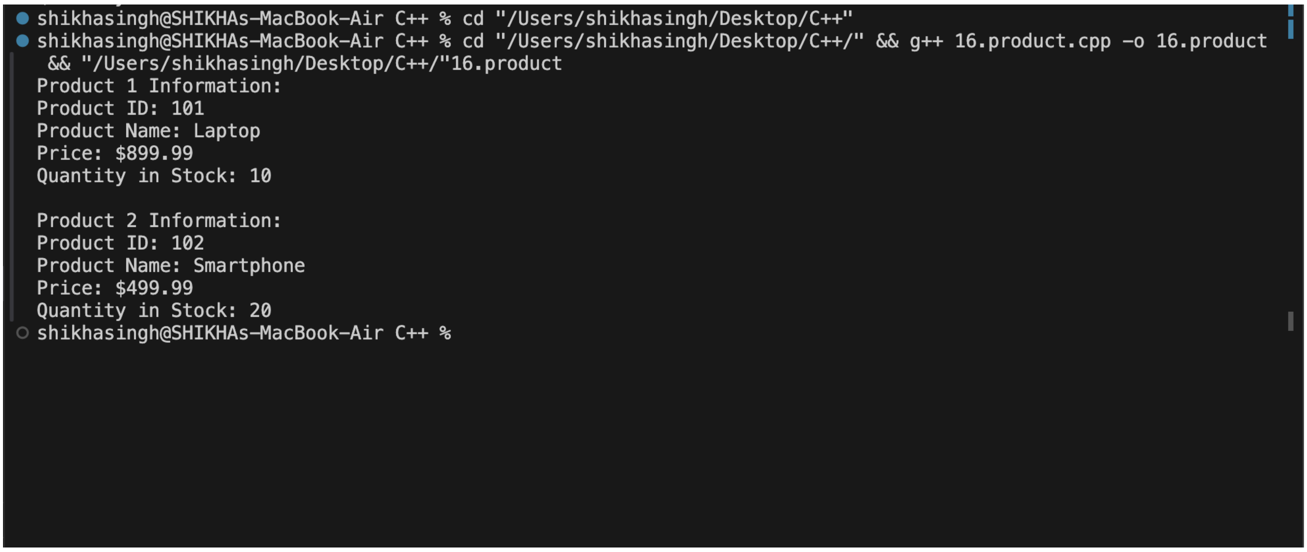
```
    cout << "\nProduct 2 Information:" << endl;
    product2.displayProductInfo();
    return 0;
}
```

Output: (screenshot)



```
● shikhasingh@SHIKHAs-MacBook-Air C++ % cd "/Users/shikhasingh/Desktop/C++"
● shikhasingh@SHIKHAs-MacBook-Air C++ % cd "/Users/shikhasingh/Desktop/C++/" && g++ 16.product.cpp -o 16.product
   && "/Users/shikhasingh/Desktop/C++/"16.product
  Product 1 Information:
  Product ID: 101
  Product Name: Laptop
  Price: $899.99
  Quantity in Stock: 10

  Product 2 Information:
  Product ID: 102
  Product Name: Smartphone
  Price: $499.99
  Quantity in Stock: 20
○ shikhasingh@SHIKHAs-MacBook-Air C++ %
```

Test Case: Any two (screenshot)



```
shikhasingh@SHIKHAs-MacBook-Air C++ % cd "/Users/shikhasingh/Desktop/C++"
shikhasingh@SHIKHAs-MacBook-Air C++ % cd "/Users/shikhasingh/Desktop/C++/" && g++ 16.product.cpp -o 16.product
 && "/Users/shikhasingh/Desktop/C++/"16.product
Product 1 Information:
Product ID: 101
Product Name: Laptop
Price: $899.99
Quantity in Stock: 10

Product 2 Information:
Product ID: 102
Product Name: Smartphone
Price: $499.99
Quantity in Stock: 20
shikhasingh@SHIKHAs-MacBook-Air C++ %
```

Conclusion:

The program showcases the implementation of an inventory management system using object-oriented principles in C++.

- **Encapsulation is achieved by declaring attributes as private members, limiting direct access to these attributes from outside the class.**
- **The parameterized constructor enables the initialization of object attributes during object creation, ensuring that objects are in a valid state.**
- **Getter methods provide controlled access to the private attributes, promoting information hiding and encapsulation.**
- **The program is a basic example illustrating how classes, objects, constructors, and encapsulation can be used to model and manage inventory information in a simple system.**