Object Oriented Programming Lab

Assignment 7

Submitted by:

Navdeep Singh

30th September 2025

Roll No: 24124073
Group: 3
Branch: Information Technology
Year: 2nd Year

Q1. Bank System with BankAccount and Auditor Classes

Problem Statement:

Build a system for a bank. There are two classes involved:

- **BankAccount** – This class stores private information such as the account holder's name and account balance.
- **Auditor** – This class represents an external auditor. It needs to check the balance of different accounts for auditing purposes, but should not be a member of the `BankAccount` class.

Write a C++ program that:

1. Defines a class `BankAccount` with **private** data members: `accountHolder` and `balance`.
2. Defines a class `Auditor` with a member function called `auditAccount()` that checks the balance of a `BankAccount`.
3. Declares the Auditor's member function `auditAccount()` as a **friend** inside the `BankAccount` class.
4. Supports multiple bank accounts with meaningful account names and balances.
5. Includes clear and well-formatted output to simulate the auditing process.

Code

```cpp
#include <bits/stdc++.h>
using namespace std;

// Forward declaration
class Auditor;

class BankAccount {
private:
    string accountHolder;
    double balance;

public:
    BankAccount(string name, double bal) {
        accountHolder = name;
        balance = bal;
    }

    friend void Auditor::auditAccount(BankAccount &acc);
};

class Auditor {
public:
    void auditAccount(BankAccount &acc) {
        cout << "Auditing account of " << acc.accountHolder << endl;
        cout << "Current Balance: $" << acc.balance << endl << endl;
    }
};
```

```

int main() {
    BankAccount acc1("Navdeep Singh", 5000);
    BankAccount acc2("Rishi Kumar", 7500);
    BankAccount acc3("Anita Sharma", 12000);

    auditor.auditAccount(acc1);
    auditor.auditAccount(acc2);
    auditor.auditAccount(acc3);

    return 0;
}
```

Sample Output

```
Auditing account of Navdeep Singh
Current Balance: $5000

Auditing account of Rishi Kumar
Current Balance: $7500

Auditing account of Anita Sharma
Current Balance: $12000
```

Q2. Product Class with Operator Overloading

Problem Statement:

Write a C++ program that:

- Defines a class `Product` with members for `name`, `price per unit`, and `quantity`.
- Overloads the `+` operator to add two `Product` objects only if their `name` matches.
- Displays the result in a user-friendly format.
- If the products do not match, print a message indicating they cannot be added.

Code

```cpp
#include <iostream>
#include <string>
using namespace std;

class Product {
private:
    string name;
    double price;
    int quantity;

public:
    Product(string n, double p, int q) {
        name = n;
        price = p;
        quantity = q;
    }

    // Overload + operator
    Product operator+(const Product &p) {
        if (name == p.name) {
            return Product(name, price, quantity + p.quantity);
        } else {
            cout << "Products cannot be added as names do not match!"
                << endl;
            return Product("", 0, 0); // Return empty product
        }
    }

    void display() {
        if (name != "")
            cout << "Product: " << name << ", Price per unit: $" <<
                price
                  << ", Quantity: " << quantity << endl;
    }
};

int main() {
    Product p1("Laptop", 1000, 5);
    Product p2("Laptop", 1000, 3);
    Product p3("Phone", 500, 2);

    cout << "Adding two matching products:\n";
    Product p4 = p1 + p2;
    p4.display();

    cout << "\nTrying to add two different products:\n";
    Product p5 = p1 + p3;
    p5.display(); // Will show nothing as addition failed

    return 0;
}
```

Sample Output

```
Adding two matching products:
Product: Laptop, Price per unit: $1000, Quantity: 8

Trying to add two different products:
Products cannot be added as names do not match!
```