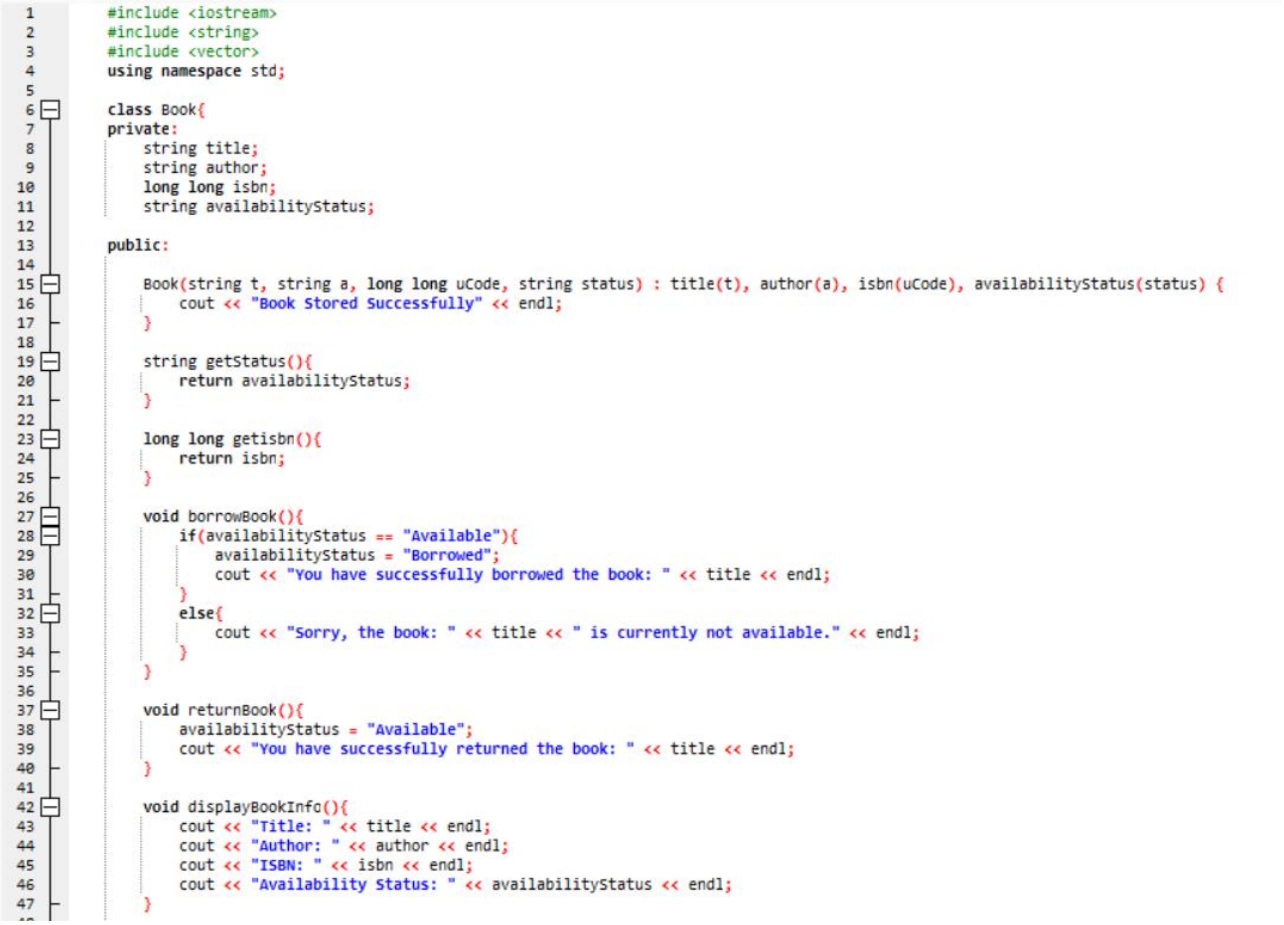
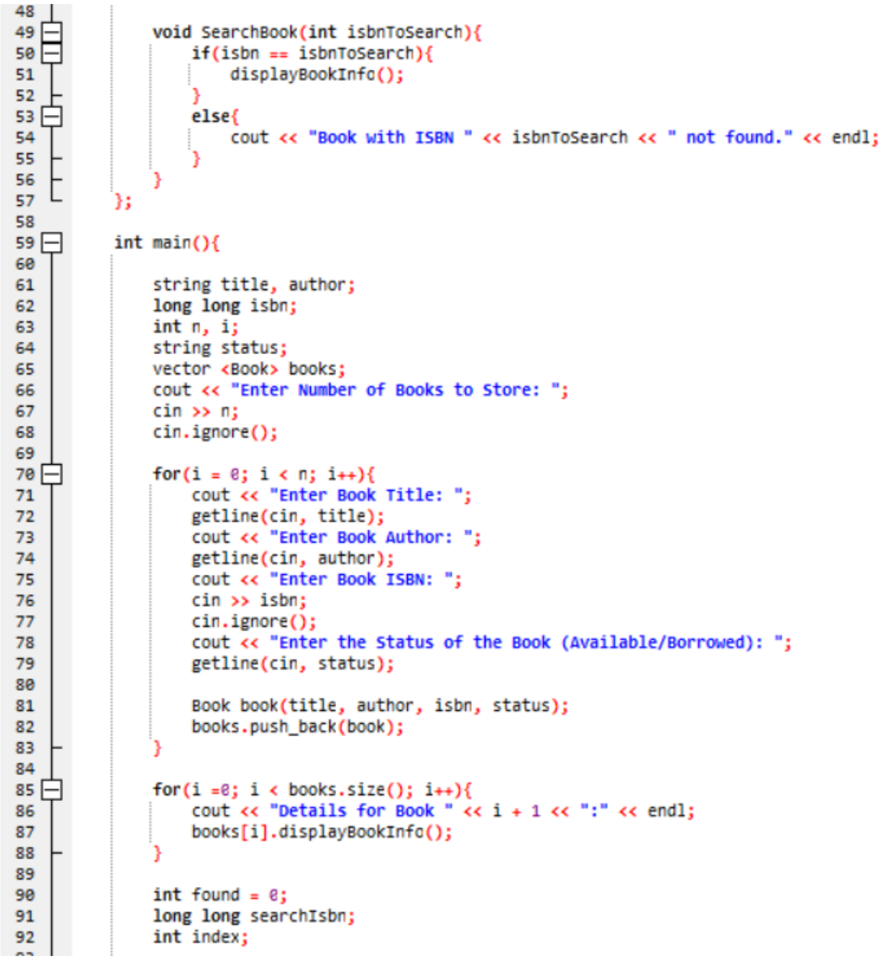
Lab 02 Tasks

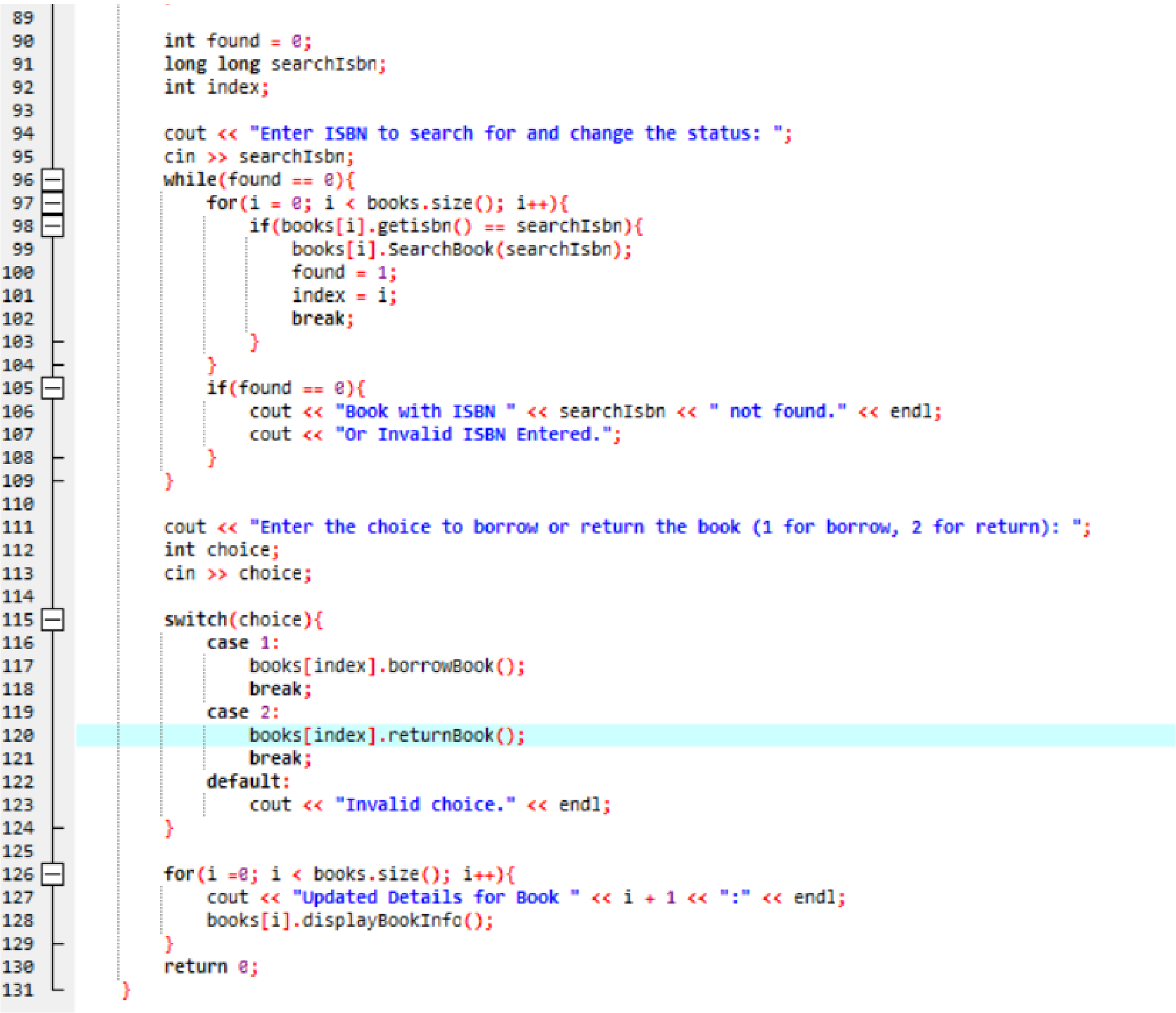
Q1



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

class Book{
private:
    string title;
    string author;
    long long isbn;
    string availabilityStatus;

public:

    Book(string t, string a, long long uCode, string status) : title(t), author(a), isbn(uCode), availabilityStatus(status) {
        cout << "Book Stored Successfully" << endl;
    }

    string getStatus(){
        return availabilityStatus;
    }

    long long getisbn(){
        return isbn;
    }

    void borrowBook(){
        if(availabilityStatus == "Available"){
            availabilityStatus = "Borrowed";
            cout << "You have successfully borrowed the book: " << title << endl;
        }
        else{
            cout << "Sorry, the book: " << title << " is currently not available." << endl;
        }
    }

    void returnBook(){
        availabilityStatus = "Available";
        cout << "You have successfully returned the book: " << title << endl;
    }

    void displayBookInfo(){
        cout << "Title: " << title << endl;
        cout << "Author: " << author << endl;
        cout << "ISBN: " << isbn << endl;
        cout << "Availability Status: " << availabilityStatus << endl;
    }
```

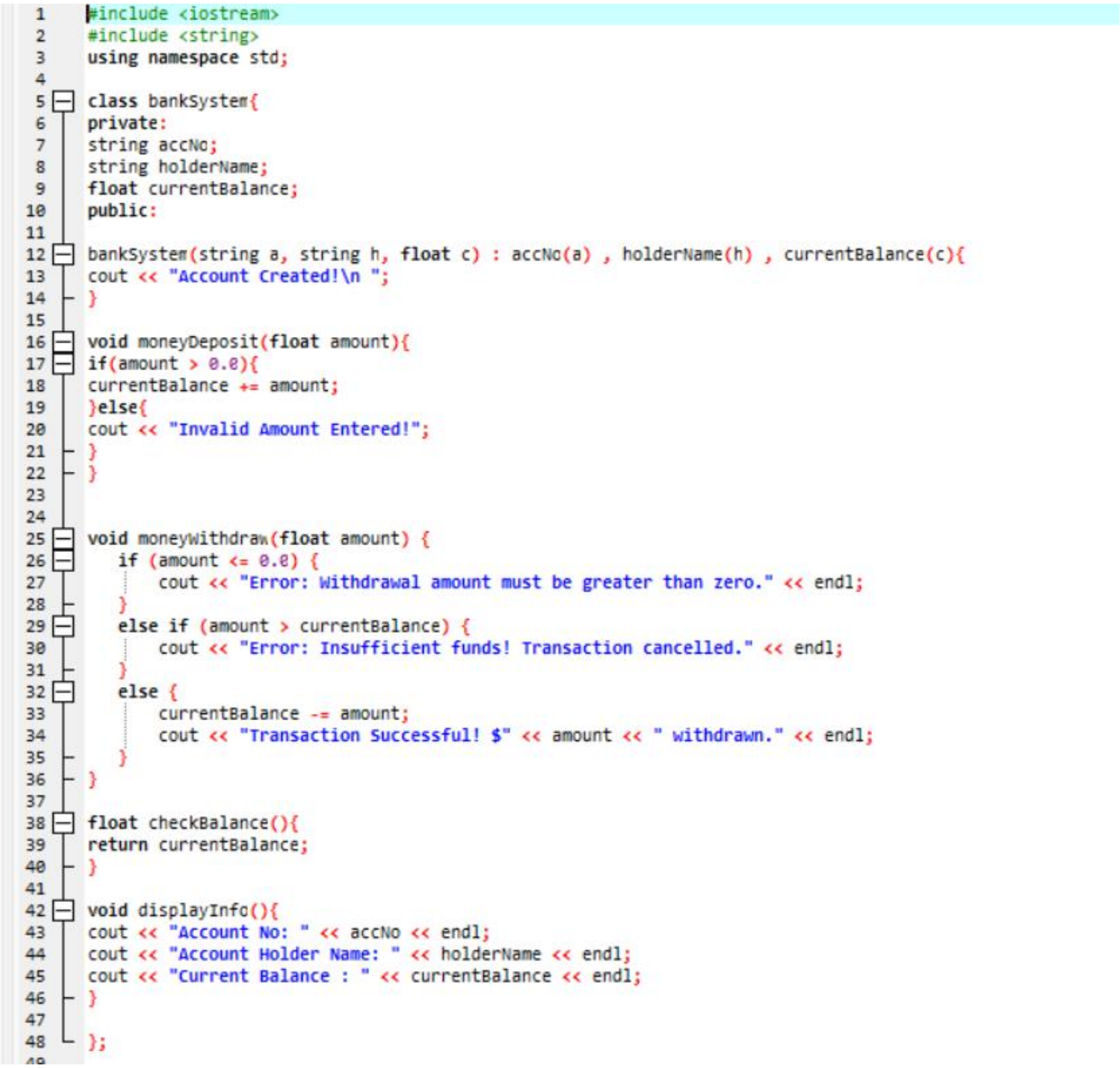


```
    void SearchBook(int isbnToSearch){
        if(isbn == isbnToSearch){
            displayBookInfo();
        }
        else{
            cout << "Book with ISBN " << isbnToSearch << " not found." << endl;
        }
    }
};

int main(){

    string title, author;
    long long isbn;
    int n, i;
    string status;
    vector <Book> books;
    cout << "Enter Number of Books to Store: ";
    cin >> n;
    cin.ignore();

    for(i = 0; i < n; i++){
        cout << "Enter Book Title: ";
        getline(cin, title);
        cout << "Enter Book Author: ";
        getline(cin, author);
        cout << "Enter Book ISBN: ";
        cin >> isbn;
        cin.ignore();
        cout << "Enter the Status of the Book (Available/Borrowed): ";
        getline(cin, status);

        Book book(title, author, isbn, status);
        books.push_back(book);
    }

    for(i =0; i < books.size(); i++){
        cout << "Details for Book " << i + 1 << ":" << endl;
        books[i].displayBookInfo();
    }

    int found = 0;
    long long searchIsbn;
    int index;
```

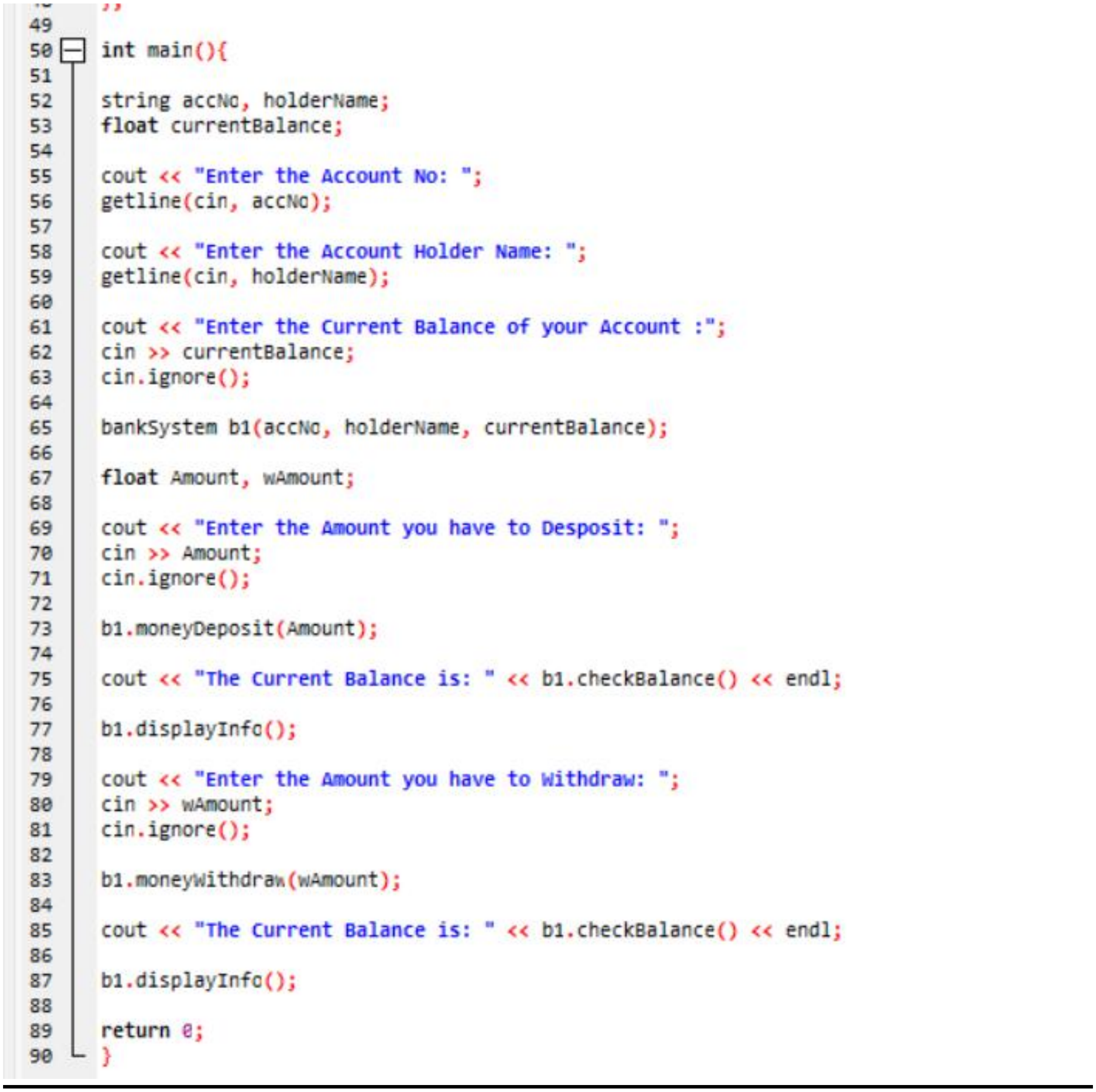


```
    int found = 0;
    long long searchIsbn;
    int index;

    cout << "Enter ISBN to search for and change the status: ";
    cin >> searchIsbn;
    while(found == 0){
        for(i = 0; i < books.size(); i++){
            if(books[i].getisbn() == searchIsbn){
                books[i].SearchBook(searchIsbn);
                found = 1;
                index = i;
                break;
            }
        }
        if(found == 0){
            cout << "Book with ISBN " << searchIsbn << " not found." << endl;
            cout << "Or Invalid ISBN Entered.";
        }
    }

    cout << "Enter the choice to borrow or return the book (1 for borrow, 2 for return): ";
    int choice;
    cin >> choice;

    switch(choice){
        case 1:
            books[index].borrowBook();
            break;
        case 2:
            books[index].returnBook();
            break;
        default:
            cout << "Invalid choice." << endl;
    }

    for(i =0; i < books.size(); i++){
        cout << "Updated Details for Book " << i + 1 << ":" << endl;
        books[i].displayBookInfo();
    }
    return 0;
}
```

Q2:

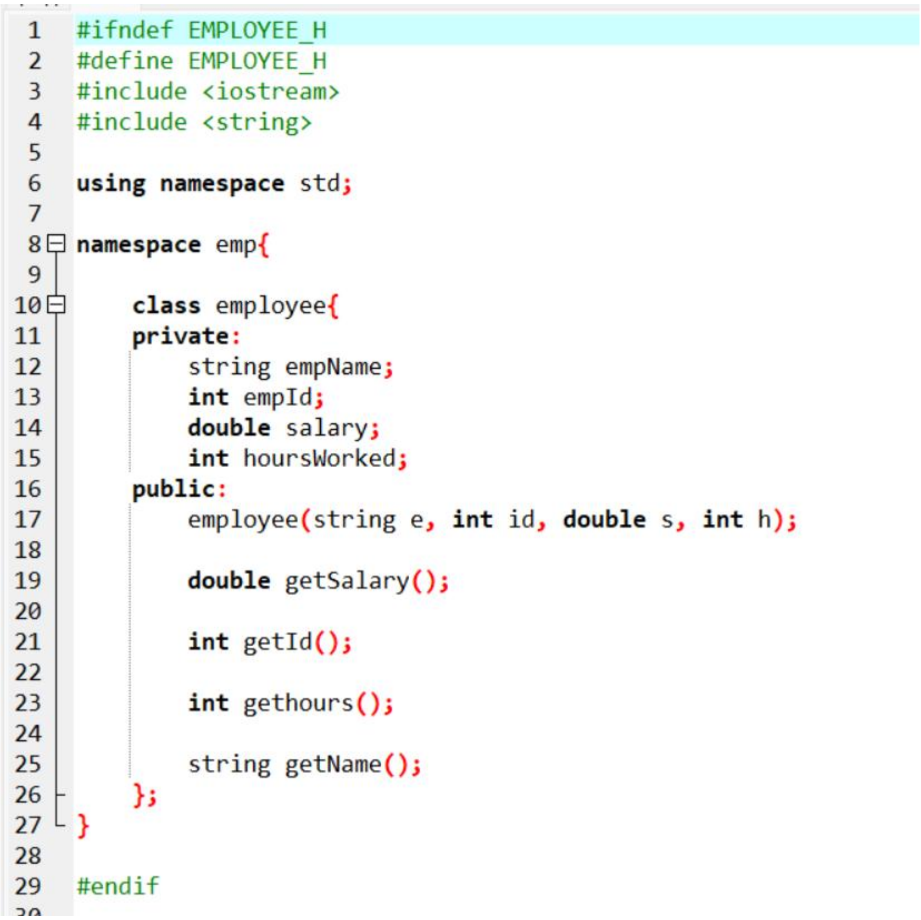


```
#include <iostream>
#include <string>
using namespace std;

class bankSystem{
private:
string accNo;
string holderName;
float currentBalance;
public:

bankSystem(string a, string h, float c) : accNo(a) , holderName(h) , currentBalance(c){
cout << "Account Created!\n ";
}

void moneyDeposit(float amount){
if(amount > 0.0){
currentBalance += amount;
}else{
cout << "Invalid Amount Entered!";
}
}


void moneyWithdraw(float amount) {
    if (amount <= 0.0) {
        cout << "Error: Withdrawal amount must be greater than zero." << endl;
    }
    else if (amount > currentBalance) {
        cout << "Error: Insufficient funds! Transaction cancelled." << endl;
    }
    else {
        currentBalance -= amount;
        cout << "Transaction Successful! $" << amount << " withdrawn." << endl;
    }
}

float checkBalance(){
return currentBalance;
}

void displayInfo(){
cout << "Account No: " << accNo << endl;
cout << "Account Holder Name: " << holderName << endl;
cout << "Current Balance : " << currentBalance << endl;
}

};
```

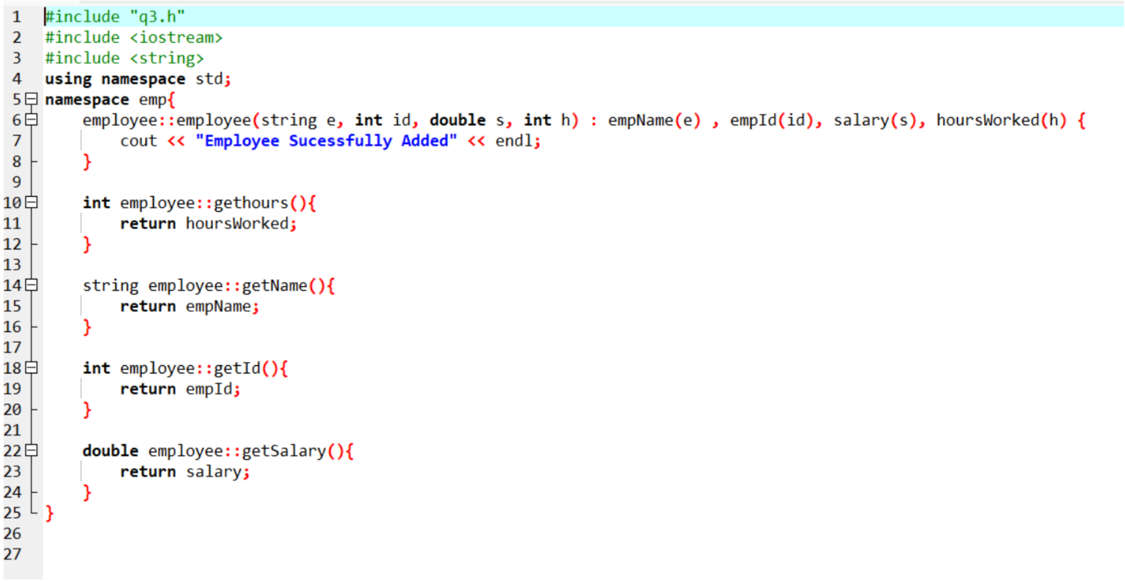


```cpp
};

int main(){

string accNo, holderName;
float currentBalance;

cout << "Enter the Account No: ";
getline(cin, accNo);

cout << "Enter the Account Holder Name: ";
getline(cin, holderName);

cout << "Enter the Current Balance of your Account :";
cin >> currentBalance;
cin.ignore();

bankSystem b1(accNo, holderName, currentBalance);

float Amount, wAmount;

cout << "Enter the Amount you have to Desposit: ";
cin >> Amount;
cin.ignore();

b1.moneyDeposit(Amount);

cout << "The Current Balance is: " << b1.checkBalance() << endl;

b1.displayInfo();

cout << "Enter the Amount you have to Withdraw: ";
cin >> wAmount;
cin.ignore();

b1.moneyWithdraw(wAmount);

cout << "The Current Balance is: " << b1.checkBalance() << endl;

b1.displayInfo();

return 0;
}
```

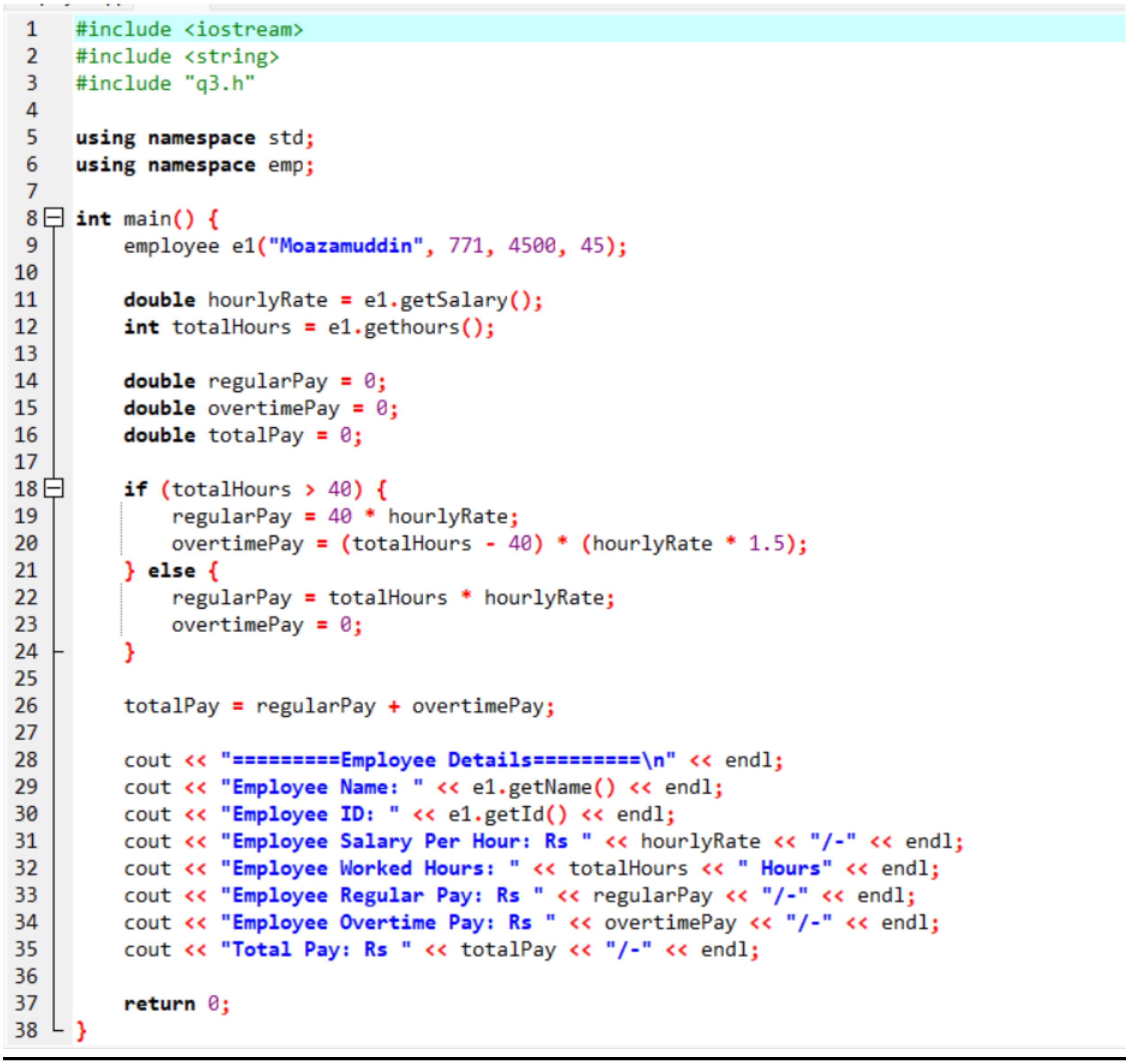
Q3:



```
#ifndef EMPLOYEE_H
#define EMPLOYEE_H
#include <iostream>
#include <string>

using namespace std;

namespace emp{

    class employee{
    private:
        string empName;
        int empId;
        double salary;
        int hoursWorked;
    public:
        employee(string e, int id, double s, int h);

        double getSalary();

        int getId();

        int gethours();

        string getName();
    };
}

#endif
```



```
#include "q3.h"
#include <iostream>
#include <string>
using namespace std;
namespace emp{
    employee::employee(string e, int id, double s, int h) : empName(e) , empId(id), salary(s), hoursWorked(h) {
        cout << "Employee Sucessfully Added" << endl;
    }

    int employee::gethours(){
        return hoursWorked;
    }

    string employee::getName(){
        return empName;
    }

    int employee::getId(){
        return empId;
    }

    double employee::getSalary(){
        return salary;
    }
}
```



```cpp
#include <iostream>
#include <string>
#include "q3.h"

using namespace std;
using namespace emp;

int main() {
    employee e1("Moazamuddin", 771, 4500, 45);

    double hourlyRate = e1.getSalary();
    int totalHours = e1.gethours();

    double regularPay = 0;
    double overtimePay = 0;
    double totalPay = 0;

    if (totalHours > 40) {
        regularPay = 40 * hourlyRate;
        overtimePay = (totalHours - 40) * (hourlyRate * 1.5);
    } else {
        regularPay = totalHours * hourlyRate;
        overtimePay = 0;
    }

    totalPay = regularPay + overtimePay;

    cout << "=========Employee Details=========\n" << endl;
    cout << "Employee Name: " << e1.getName() << endl;
    cout << "Employee ID: " << e1.getId() << endl;
    cout << "Employee Salary Per Hour: Rs " << hourlyRate << "/-" << endl;
    cout << "Employee Worked Hours: " << totalHours << " Hours" << endl;
    cout << "Employee Regular Pay: Rs " << regularPay << "/-" << endl;
    cout << "Employee Overtime Pay: Rs " << overtimePay << "/-" << endl;
    cout << "Total Pay: Rs " << totalPay << "/-" << endl;

    return 0;
}
```