2/10/2025

Object Oriented Programming

Lab 4

Muhammad Arham.Usman.
FAST NUCES

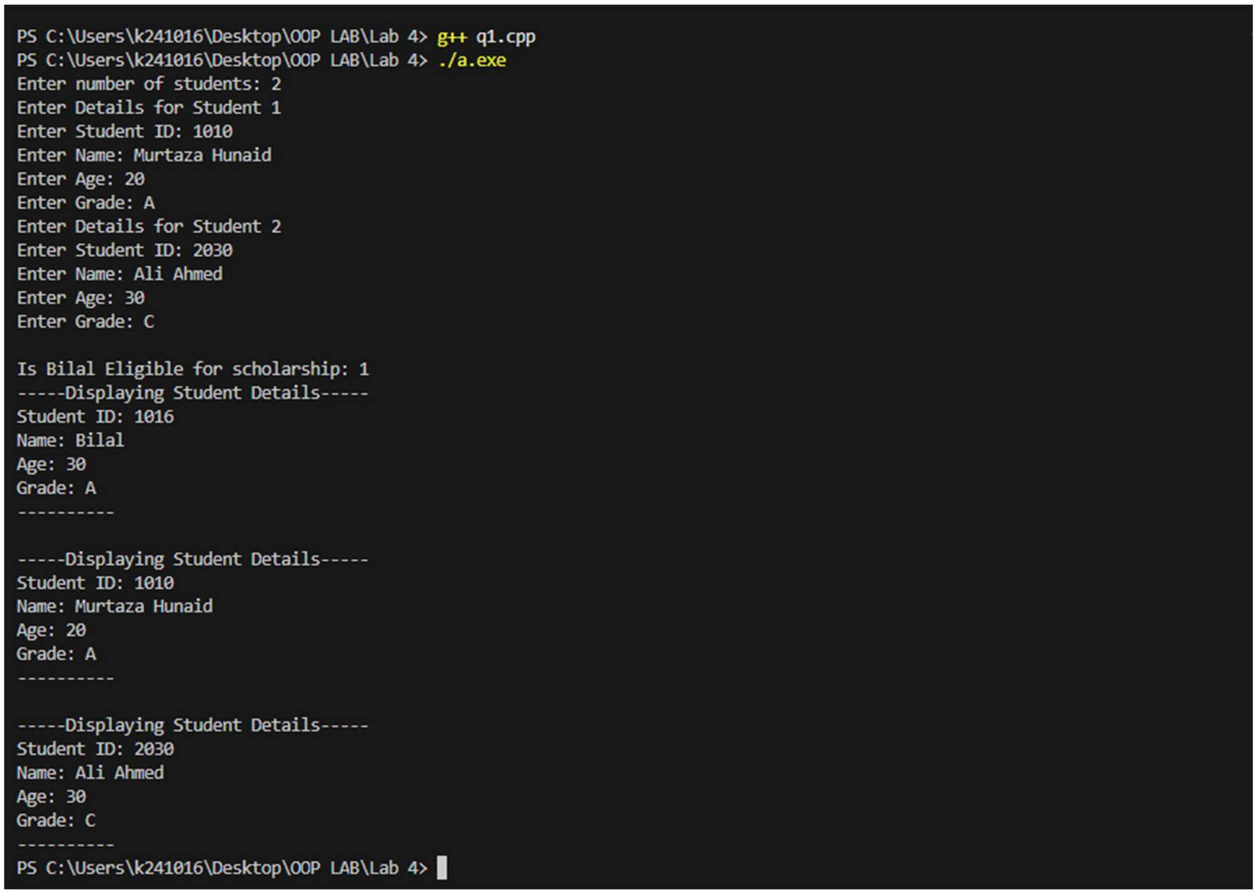
Question 1:

Source Code:

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

class Student{
    int studentID;
    string name;
    int age;
    char grade;

    public:
    Student(int studentID=0, string name="", int age=0, char grade='N'){
        this->studentID=studentID;
        this->name=name;
        this->age=age;
        this->grade=grade;
    }

    void setID(int student_id){
        studentID=student_id;
    }
    void setName(string Name){
        name=Name;
    }
    void setAge(int Age){
        age=Age;
    }
    void setGrade(char Grade){
        grade=Grade;
    }
    int getID(){
        return studentID;
    }
    string getName(){
        return name;
    }
    int getAge(){
        return age;
    }
    char getGrade(){
```

```cpp
        return grade;
    }

    void promoteStudent(){
        if (grade>'A' && grade<='D'){
            grade-=1;
        }
        else if (grade=='F'){
            grade='D';
        }
    }

    bool isEligibleForScholarship(){
        if (grade=='A') return true;
        return false;
    }

    void displayDetails(){
        cout<<"\n-----Displaying Student Details-----\n";
        cout<<"Student ID: "<<studentID;
        cout<<"\nName: "<<name;
        cout<<"\nAge: "<<age;
        cout<<"\nGrade: "<<grade;
        cout<<"\n----------\n";
    }
};

int main(){
    int n, temp_i;
    string temp_s;
    char temp_c;

    //Getting no of student
    cout<<"Enter number of students: ";
    cin>>n;

    //using default constructor
    Student* class_1=new Student[n];
    for (int i=0; i<n; i++){
        cout<<"Enter Details for Student "<<i+1<<endl;
        cout<<"Enter Student ID: ";
        cin>>temp_i;
        class_1[i].setID(temp_i);
        cout<<"Enter Name: ";
        cin.ignore();
        getline(cin,temp_s);
        class_1[i].setName(temp_s);
        cout<<"Enter Age: ";
        cin>>temp_i;
        class_1[i].setAge(temp_i);
        cout<<"Enter Grade: ";
        cin.ignore();
        cin>>temp_c;
        cin.ignore();
        class_1[i].setGrade(temp_c);

    }

    //using parameterised constructor
    Student new_student(1016, "Bilal", 30, 'B');

    new_student.promoteStudent();
    cout<<"\nIs "<<new_student.getName()<<" Eligible for scholarship: 
"<<new_student.isEligibleForScholarship();

    new_student.displayDetails();
    for (int i=0; i<n; i++){
        class_1[i].displayDetails();
    }

    delete[] class_1;
}
```

Output:



```
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> g++ q1.cpp
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> ./a.exe
Enter number of students: 2
Enter Details for Student 1
Enter Student ID: 1010
Enter Name: Murtaza Hunaid
Enter Age: 20
Enter Grade: A
Enter Details for Student 2
Enter Student ID: 2030
Enter Name: Ali Ahmed
Enter Age: 30
Enter Grade: C

Is Bilal Eligible for scholarship: 1
-----Displaying Student Details-----
Student ID: 1016
Name: Bilal
Age: 30
Grade: A
----------

-----Displaying Student Details-----
Student ID: 1010
Name: Murtaza Hunaid
Age: 20
Grade: A
----------

-----Displaying Student Details-----
Student ID: 2030
Name: Ali Ahmed
Age: 30
Grade: C
----------
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4>
```

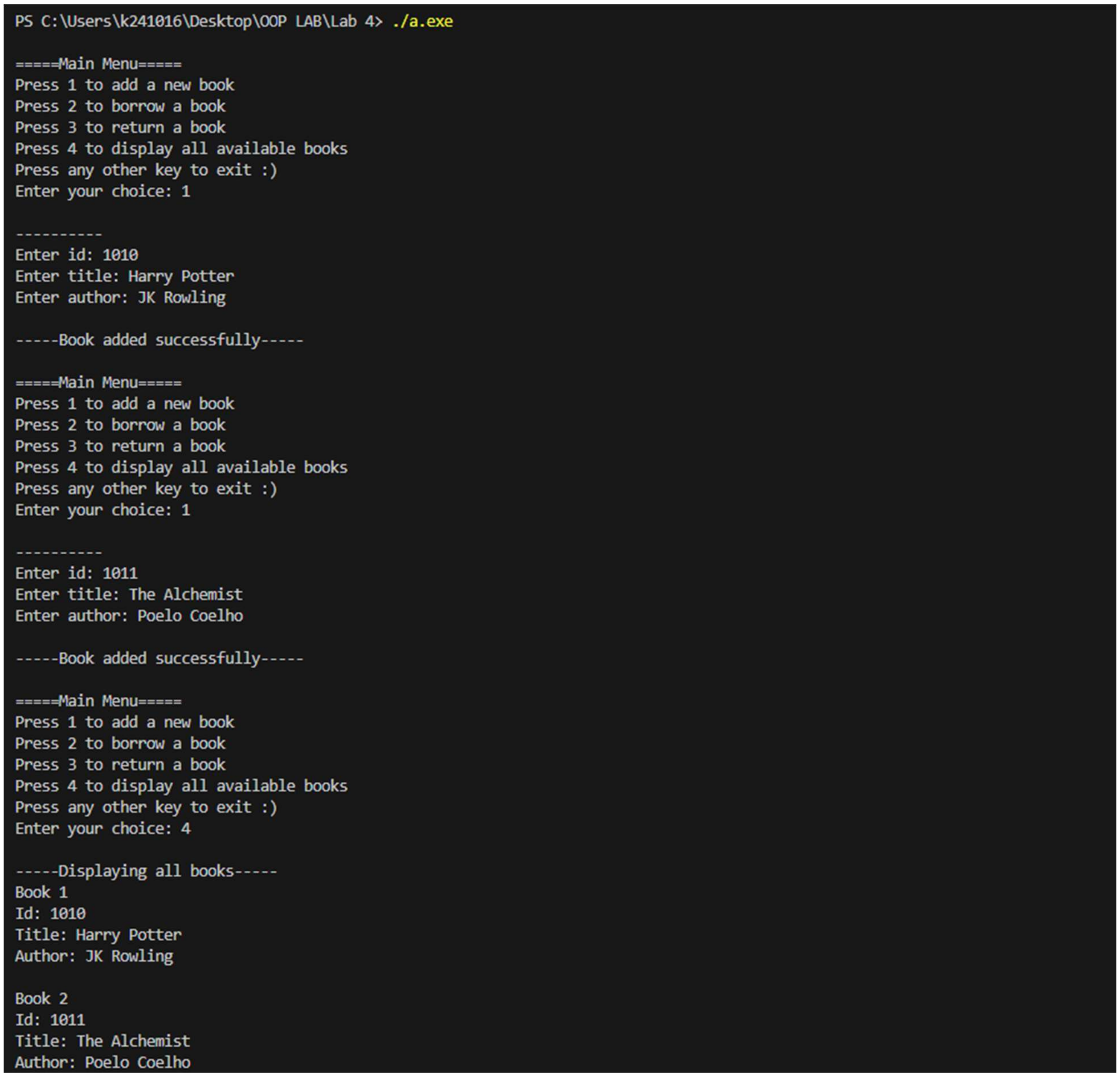
Question 2:

Source Code:

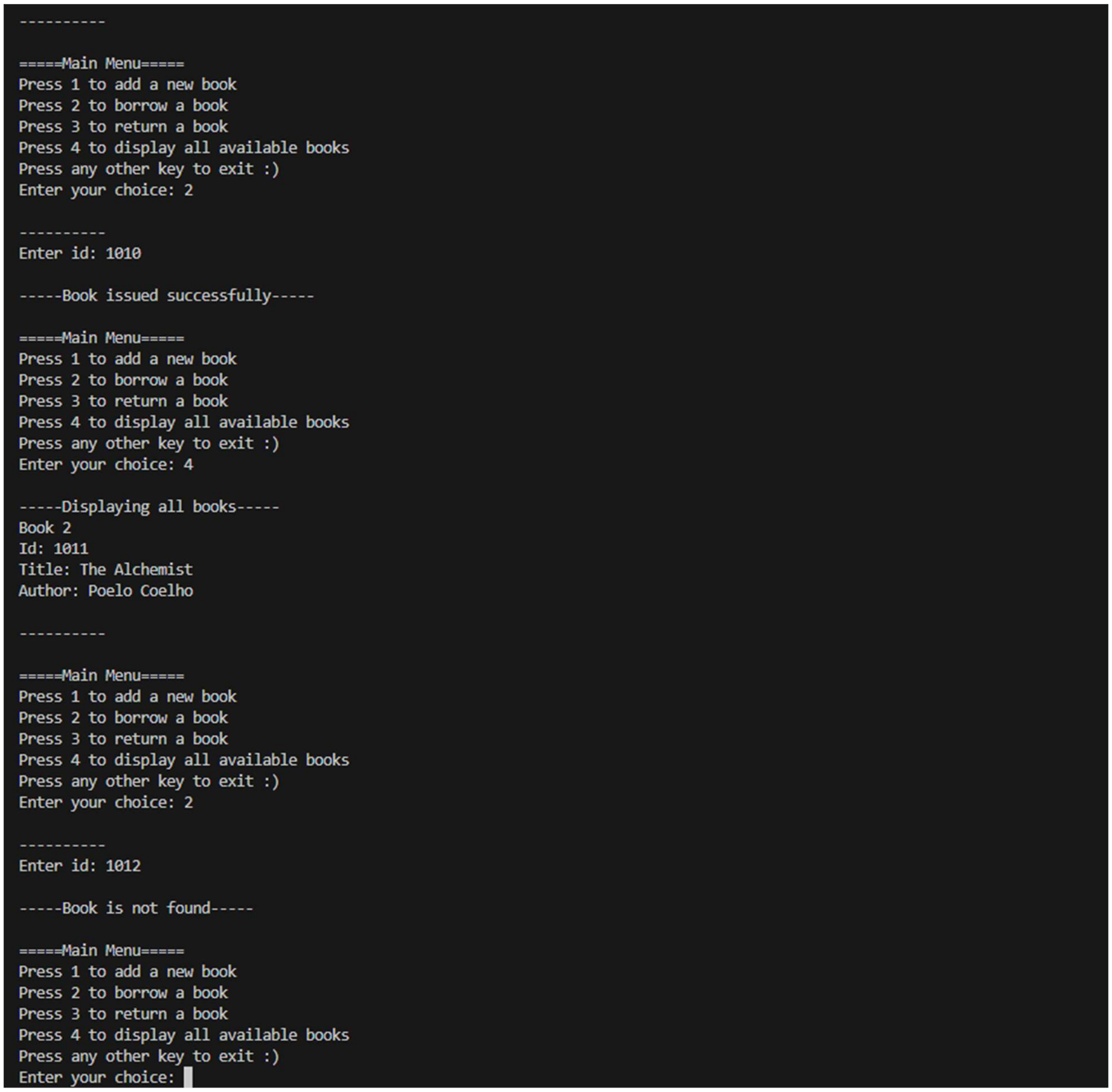
```cpp
/*
A library system allows users to borrow and return books. The system should:
1. Add new books to the collection.
2. Borrow books (check availability).
3. Return books.
4. Display all available books.
Requirements:
- Implement a Book class with attributes: ID, title, author, availability.
- Implement a Library class with:
1. Method to add a book (with unique ID).
2. Method to borrow a book (updates availability).
3. Method to return a book (marks it available).
4. Method to display all available books.
- Store book records dynamically using pointers and DMA.
- Use constructor overloading to initialize books with or without availability
status.
*/

#include <iostream>
using namespace std;

class Book{
    string ID, title, author, availability;
    public:
    Book(){
        this->author="";
        this->availability="";
        this->ID="";
        this->title="";
    }
    Book(string id, string Title, string Author, string Availability){
        this->author=Author;
        this->availability=Availability;
        this->ID=id;
        this->title=Title;
    }
    string get_ID(){
        return ID;
    }
    string get_title(){
        return title;
    }
    string get_author(){
        return author;
    }
    string get_availability(){
        return availability;
    }
    void set_ID(string ID){
        this->ID = ID;
    }
    void set_title(string title){
        this->title = title;
    }
    void set_author(string author){
        this->author = author;
    }
    void set_availability(string availability){
        this->availability = availability;
    }
    void displayDetails(){
        cout<<"Id: "<<ID;
        cout<<"\nTitle: "<<title;
        cout<<"\nAuthor: "<<author<<endl;
    }
};

class Library{
    Book *books;
    int noOfBooks;
    public:
        Library(Book* Books=nullptr, int NoOfBooks=0){
            noOfBooks=NoOfBooks;
            if (noOfBooks==0) books=nullptr;
            else{
                books=new Book[noOfBooks];
                for (int i=0; i<noOfBooks; i++){
                    books[i]=Books[i];
                }
            }
```

```cpp
        }

        ~Library(){
            delete[] books;
        }

        void AddBook(Book book){
            Book *new_stock=new Book[++noOfBooks];
            int i, found=0;
            for (i=0; i<noOfBooks-1; i++){
                if (books[i].get_ID()==book.get_ID()){
                    found=1; break;
                }
                new_stock[i]=books[i];
            }
            if (found==0){
                new_stock[i]=book;
                delete[] books;
                books=new_stock;
                cout<<"\n-----Book added successfully-----\n";
            }
            else{
                delete[] new_stock;
                noOfBooks--;
                cout<<"\n-----Error! Book must have a unique ID-----\n";
            }
        }

        void BorrowBook(string id){
            for (int i=0; i<noOfBooks; i++){
                if (id==books[i].get_ID()){
                    if (books[i].get_availability()=="y"){
                        books[i].set_availability("n");
                        cout<<"\n-----Book issued successfully-----\n";
                    }
                    else{
                        cout<<"\n-----Book is not available-----\n";
                    }
                    return;
                }
            }
            cout<<"\n-----Book is not found-----\n";
        }

        void ReturnBook(string id){
            for (int i=0; i<noOfBooks; i++){
                if (id==books[i].get_ID()){
                    books[i].set_availability("y");
                    cout<<"\n-----Book returned sucessfully-----\n";
                    return;
                }
            }
            cout<<"\n-----Book is not found-----\n";
        }

        void DisplayAll(){
            cout<<"\n-----Displaying all books-----\n";
            for (int i=0; i<noOfBooks; i++){
                if (books[i].get_availability()=="y"){
                    cout<<"Book "<<i+1<<endl;
                    books[i].displayDetails();
                    cout<<endl;
                }
            }
            cout<<"----------\n";
        }
};

int main(){
    string choice, id, title, author;
    Library lib1;
    Book b1;
    while (1){
        cout<<"\n=====Main Menu=====\n";
        cout<<"Press 1 to add a new book\nPress 2 to borrow a book\nPress 3 to
return a book\nPress 4 to display all available books\nPress any other key to
exit :)\n";
        cout<<"Enter your choice: ";
        cin>>choice;
        if (choice=="1"){
            cout<<"\n----------\n";
            cout<<"Enter id: ";
            cin>>id;
            cout<<"Enter title: ";
            cin.ignore();
            getline(cin, title);
            cout<<"Enter author: ";
            getline(cin, author);
            lib1.AddBook(Book(id, title, author, "y"));
        }
        else if (choice=="2"){
            cout<<"\n----------\n";
            cout<<"Enter id: ";
            cin>>id;
            lib1.BorrowBook(id);
        }
        else if (choice=="3"){
            cout<<"\n----------\n";
            cout<<"Enter id: ";
            cin>>id;
            lib1.ReturnBook(id);
        }
        else if (choice=="4"){
            lib1.DisplayAll();
        }
        else{
            cout<<"\n=====Thank You for using our service=====";
            break;
        }
    }
}
```

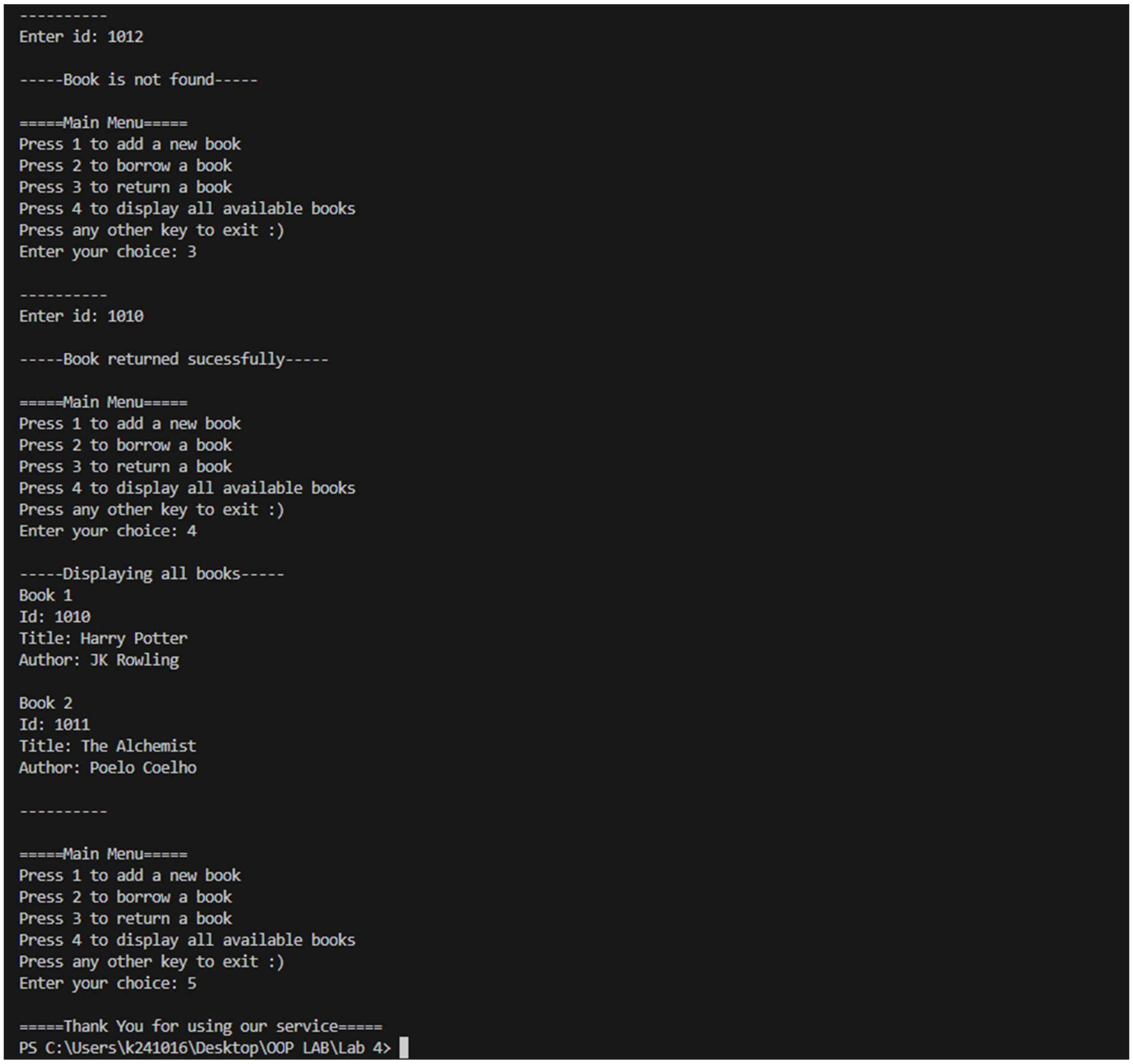
Output:



```
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> ./a.exe

=====Main Menu=====
Press 1 to add a new book
Press 2 to borrow a book
Press 3 to return a book
Press 4 to display all available books
Press any other key to exit :)
Enter your choice: 1

----------
Enter id: 1010
Enter title: Harry Potter
Enter author: JK Rowling

-----Book added successfully-----

=====Main Menu=====
Press 1 to add a new book
Press 2 to borrow a book
Press 3 to return a book
Press 4 to display all available books
Press any other key to exit :)
Enter your choice: 1

----------
Enter id: 1011
Enter title: The Alchemist
Enter author: Poelo Coelho

-----Book added successfully-----

=====Main Menu=====
Press 1 to add a new book
Press 2 to borrow a book
Press 3 to return a book
Press 4 to display all available books
Press any other key to exit :)
Enter your choice: 4

-----Displaying all books-----
Book 1
Id: 1010
Title: Harry Potter
Author: JK Rowling

Book 2
Id: 1011
Title: The Alchemist
Author: Poelo Coelho
```



```
----------

=====Main Menu=====
Press 1 to add a new book
Press 2 to borrow a book
Press 3 to return a book
Press 4 to display all available books
Press any other key to exit :)
Enter your choice: 2

----------
Enter id: 1010

-----Book issued successfully-----

=====Main Menu=====
Press 1 to add a new book
Press 2 to borrow a book
Press 3 to return a book
Press 4 to display all available books
Press any other key to exit :)
Enter your choice: 4

-----Displaying all books-----
Book 2
Id: 1011
Title: The Alchemist
Author: Poelo Coelho

----------

=====Main Menu=====
Press 1 to add a new book
Press 2 to borrow a book
Press 3 to return a book
Press 4 to display all available books
Press any other key to exit :)
Enter your choice: 2

----------
Enter id: 1012

-----Book is not found-----

=====Main Menu=====
Press 1 to add a new book
Press 2 to borrow a book
Press 3 to return a book
Press 4 to display all available books
Press any other key to exit :)
Enter your choice: 
```



```
----------
Enter id: 1012

-----Book is not found-----

=====Main Menu=====
Press 1 to add a new book
Press 2 to borrow a book
Press 3 to return a book
Press 4 to display all available books
Press any other key to exit :)
Enter your choice: 3

----------
Enter id: 1010

-----Book returned sucessfully-----

=====Main Menu=====
Press 1 to add a new book
Press 2 to borrow a book
Press 3 to return a book
Press 4 to display all available books
Press any other key to exit :)
Enter your choice: 4

-----Displaying all books-----
Book 1
Id: 1010
Title: Harry Potter
Author: JK Rowling

Book 2
Id: 1011
Title: The Alchemist
Author: Poelo Coelho

----------

=====Main Menu=====
Press 1 to add a new book
Press 2 to borrow a book
Press 3 to return a book
Press 4 to display all available books
Press any other key to exit :)
Enter your choice: 5

=====Thank You for using our service=====
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4>
```

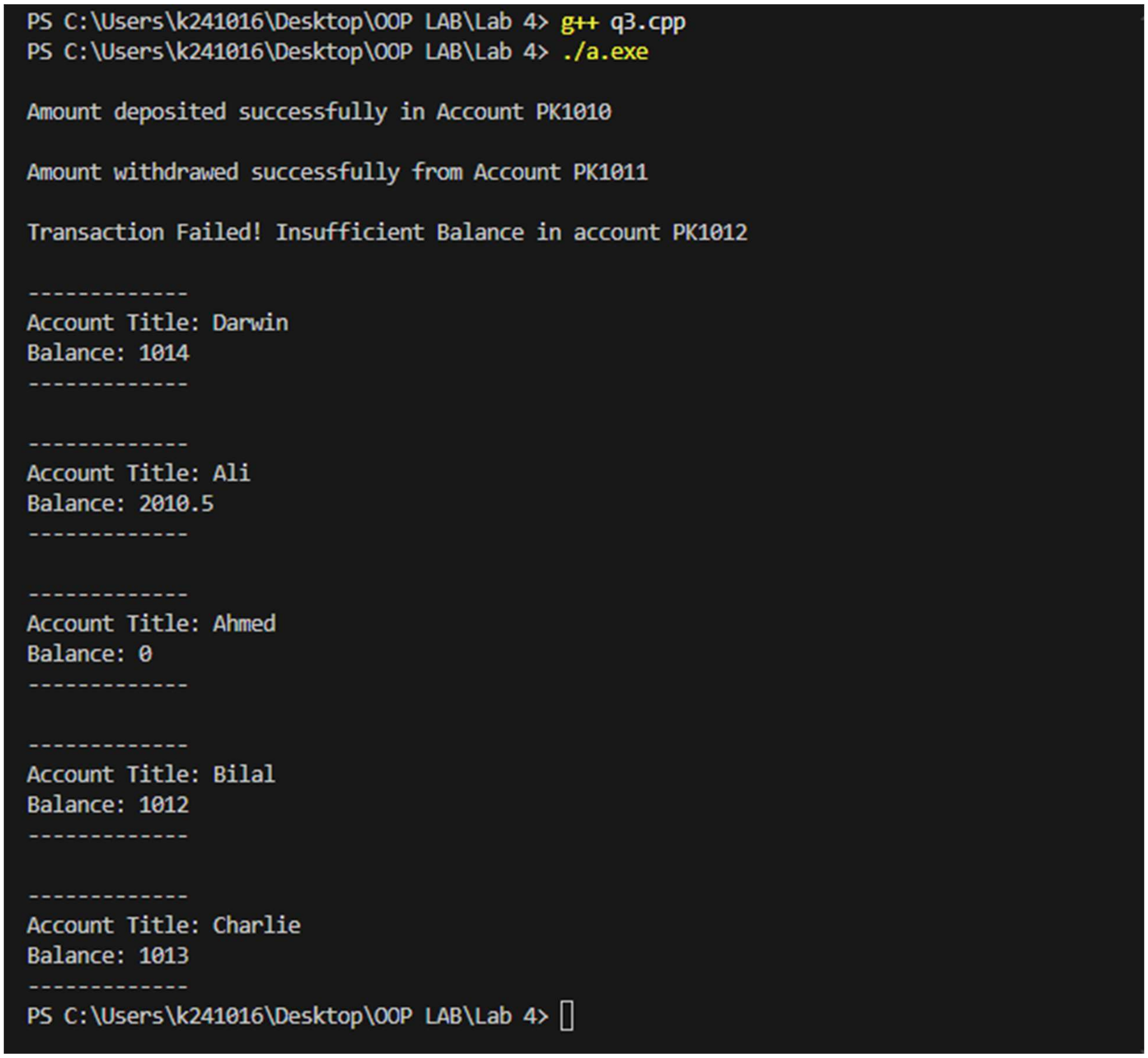
Question 3:

Source Code:

```
/*
You are building a bank account management system. Create a class `Account` to
manage bank
accounts.
Requirements:
1. Attributes:
- `accountNumber` (string)
- `accountHolderName` (string)
- `balance` (double)
2. Define a default constructor that initializes `balance` to `0.0`.
3. Define a parameterized constructor to initialize all attributes.
4. Add methods:
- `deposit(double amount)`: Adds the amount to the balance.
- `withdraw(double amount)`: Deducts the amount from the balance (if sufficient
funds are
available).
- `checkBalance()`: Displays the current balance.
5. Create a few `Account` objects and test the methods.
*/

#include <iostream>
using namespace std;

class Account{
    string accountNumber;
    string accountHolderName;
    double balance;
    public:
    Account(string Account_Number="", string Holder_Name="", double Balance=0.0){
        accountNumber=Account_Number;
        accountHolderName=Holder_Name;
        balance=Balance;
    }

    void deposit(double amount){
        balance+=amount;
        cout<<"\nAmount deposited successfully in Account "<<
accountNumber<<endl;
    }
```

```cpp
    void withdraw(double amount){
        if (amount<=balance){
            balance-=amount;
            cout<<"\nAmount withdrawed successfully from Account "<<
accountNumber<<endl;
        }
        else{
            cout<<"\nTransaction Failed! Insufficient Balance in account "<<
accountNumber<<endl;
        }
    }
    void checkBalance(){
        cout<<"\n-------------";
        cout<<"\nAccount Title: "<<accountHolderName;
        cout<<"\nBalance: "<<balance;
        cout<<"\n-------------\n";
    }
};

int main(){
    Account *a= new Account[5];
    a[0]=Account("PK1014", "Darwin", 1014);
    a[1]=Account("PK1010", "Ali", 1010);
    a[2]=Account("PK1011", "Ahmed", 1011);
    a[3]=Account("PK1012", "Bilal", 1012);
    a[4]=Account("PK1013", "Charlie", 1013);
    a[1].deposit(1000.50);
    a[2].withdraw(1011);
    a[3].withdraw(1014);
    for (int i=0; i<5; i++){
        a[i].checkBalance();
    }
    delete[] a;
}
```

Output:



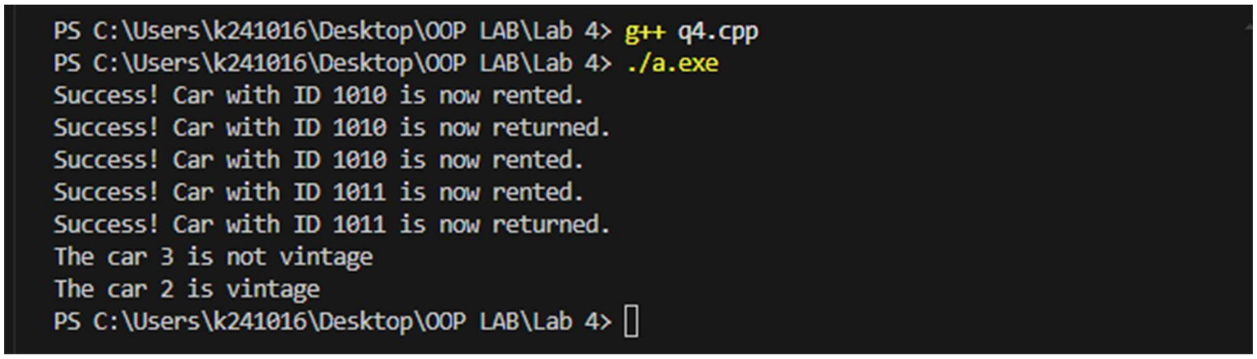
```
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> g++ q3.cpp
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> ./a.exe

Amount deposited successfully in Account PK1010

Amount withdrawed successfully from Account PK1011

Transaction Failed! Insufficient Balance in account PK1012

-------------
Account Title: Darwin
Balance: 1014
-------------

-------------
Account Title: Ali
Balance: 2010.5
-------------

-------------
Account Title: Ahmed
Balance: 0
-------------

-------------
Account Title: Bilal
Balance: 1012
-------------

-------------
Account Title: Charlie
Balance: 1013
-------------
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> 
```

Question 4:

Source Code:

```cpp
/*
You are building a car rental system. Create a class `Car` to manage cars
available for rent.
Requirements:
1. Attributes:
- `carID` (int)
- `model` (string)
- `year` (int)
- `isRented` (bool)
2. Define a default constructor that initializes `isRented` to `false`.
3. Define a parameterized constructor to initialize all attributes.
4. Add methods:

- `rentCar()`: Marks the car as rented.
- `returnCar()`: Marks the car as available.
- `isVintage()`: Returns `true` if the car's year is before 2000.

5. Create a few `Car` objects and test the methods.
*/

#include <iostream>
using namespace std;

class Car {
private:
    int carID;
    string model;
    int year;
    bool isRented;

public:
    Car() {
        isRented = false;
    }

    Car(int car_id, string Model, int Year) {
        carID = car_id;
        model = Model;
        year = Year;
        isRented = false;
    }

    void rentCar() {
        if (isRented) {
            cout << "Error! Car with ID " << carID << " is already rented.\n";
        }
        else {
            isRented = true;
            cout << "Success! Car with ID " << carID << " is now rented.\n";
        }
    }

    void returnCar() {
        if (isRented) {
            isRented = false;
            cout << "Success! Car with ID " << carID << " is now returned.\n"; }
        else {
            cout << "Error! Car with ID " << carID << " is already available.\n";
}
    }

    bool isVintage() const {
        if (year < 2000) {
            return true; }
        else {
            return false; }
    }

};

int main() {
    Car car1(1010, "Bugatti Chiron", 2010);
    Car car2(1011, "Porsche 911", 1985);
    Car car3(1012, "Benz s 6200", 2018);
    car1.rentCar();
    car1.returnCar();
    car1.rentCar();
    car2.rentCar();
    car2.returnCar();
    if (car3.isVintage()) {
        cout << "The car 3 is vintage" << endl;
    }
    else {
        cout << "The car 3 is not vintage" << endl;
    }
    if (car2.isVintage()) {
        cout << "The car 2 is vintage" << endl;
    }
    else {
        cout << "The car 2 is not vintage" << endl;
    }
}
```

Output:

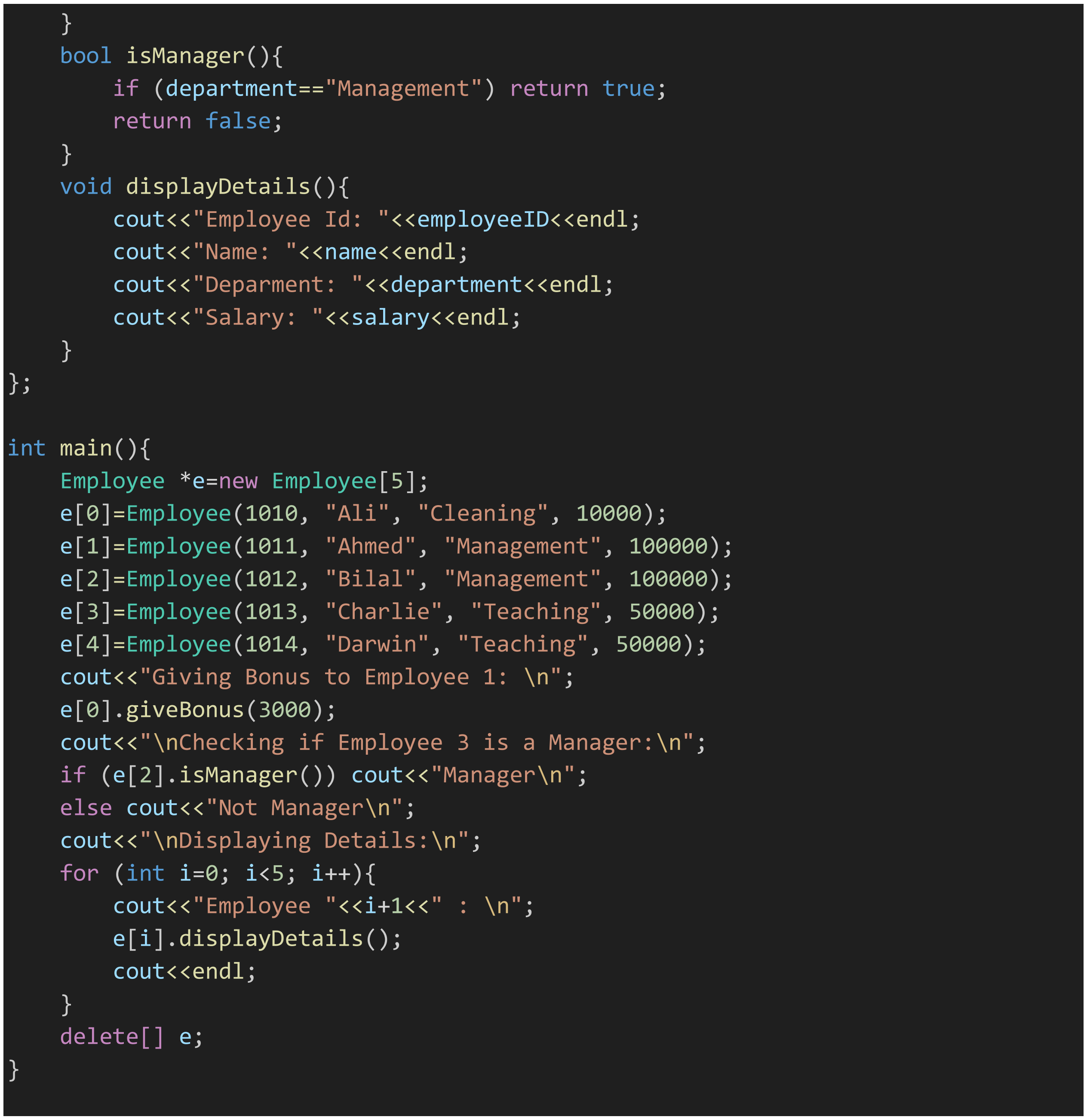


```
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> g++ q4.cpp
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> ./a.exe
Success! Car with ID 1010 is now rented.
Success! Car with ID 1010 is now returned.
Success! Car with ID 1010 is now rented.
Success! Car with ID 1011 is now rented.
Success! Car with ID 1011 is now returned.
The car 3 is not vintage
The car 2 is vintage
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4>
```

Question 5:

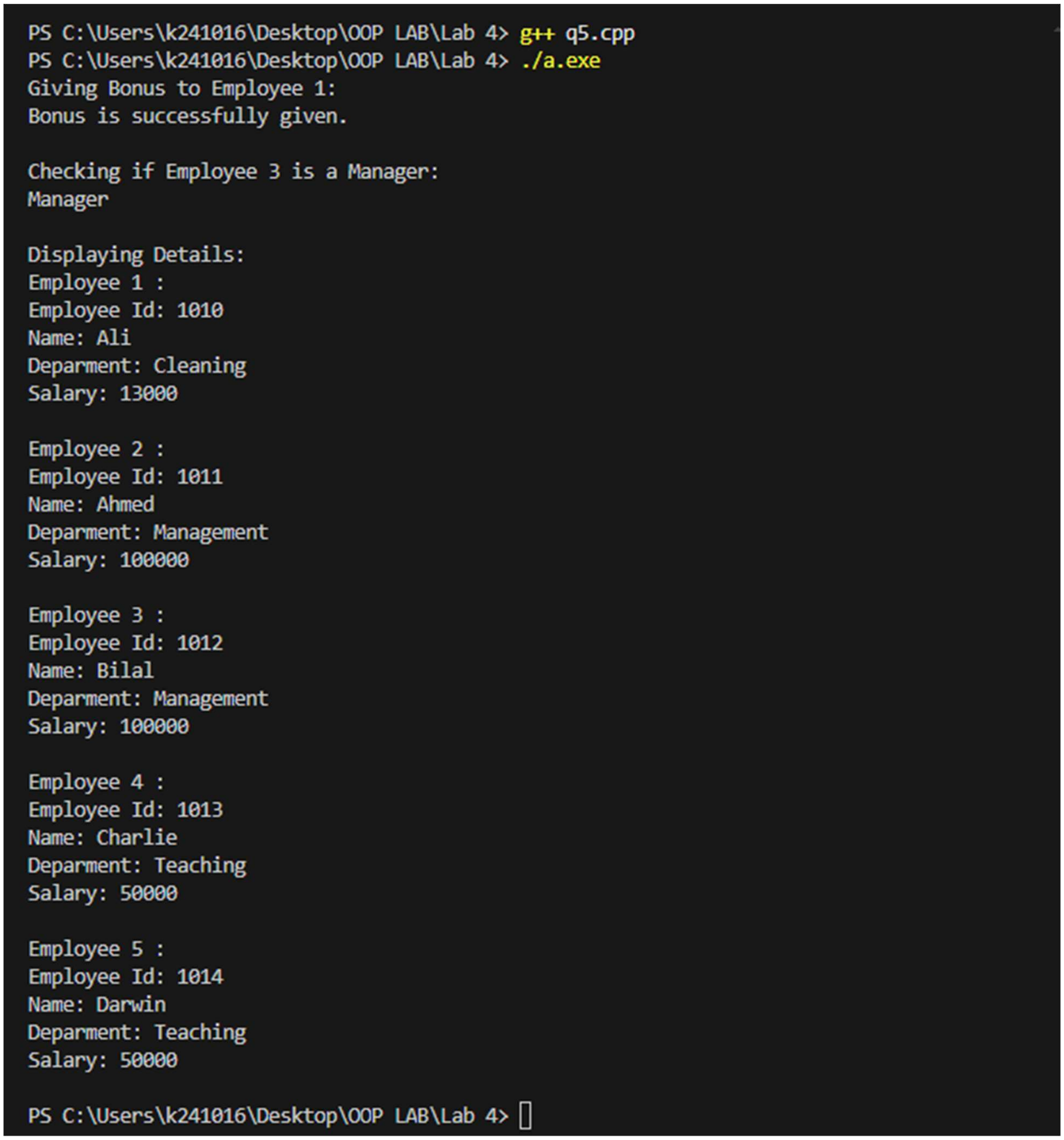
Source Code:

```
/*
You are building an employee management system. Create a class `Employee` to
manage employee
records.
Requirements:
1. Attributes:
- `employeeID` (int)
- `name` (string)
- `department` (string)
- `salary` (double)
2. Define a default constructor that initializes `salary` to `0.0`.
3. Define a parameterized constructor to initialize all attributes.
4. Add methods:
- `giveBonus(double amount)`: Adds the bonus amount to the employee's salary.
- `isManager()`: Returns `true` if the employee's department is "Management".
- `displayDetails()`: Displays the employee's details.
5. Create a few `Employee` objects and test the methods.
*/

#include <iostream>
using namespace std;

class Employee{
    int employeeID;
    string name, department;
    double salary;
    public:
    Employee(int Employee_ID=0, string Name="", string Department="", double
Salary=0.0){
        employeeID=Employee_ID;
        name=Name;
        department=Department;
        salary=Salary;
    }
    void giveBonus(double amount){
        salary+=amount;
        cout<<"Bonus is successfully given.\n";
```



```cpp
    }
    bool isManager(){
        if (department=="Management") return true;
        return false;
    }
    void displayDetails(){
        cout<<"Employee Id: "<<employeeID<<endl;
        cout<<"Name: "<<name<<endl;
        cout<<"Deparment: "<<department<<endl;
        cout<<"Salary: "<<salary<<endl;
    }
};

int main(){
    Employee *e=new Employee[5];
    e[0]=Employee(1010, "Ali", "Cleaning", 10000);
    e[1]=Employee(1011, "Ahmed", "Management", 100000);
    e[2]=Employee(1012, "Bilal", "Management", 100000);
    e[3]=Employee(1013, "Charlie", "Teaching", 50000);
    e[4]=Employee(1014, "Darwin", "Teaching", 50000);
    cout<<"Giving Bonus to Employee 1: \n";
    e[0].giveBonus(3000);
    cout<<"\nChecking if Employee 3 is a Manager:\n";
    if (e[2].isManager()) cout<<"Manager\n";
    else cout<<"Not Manager\n";
    cout<<"\nDisplaying Details:\n";
    for (int i=0; i<5; i++){
        cout<<"Employee "<<i+1<<" : \n";
        e[i].displayDetails();
        cout<<endl;
    }
    delete[] e;
}
```

Output:



```
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> g++ q5.cpp
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> ./a.exe
Giving Bonus to Employee 1:
Bonus is successfully given.

Checking if Employee 3 is a Manager:
Manager

Displaying Details:
Employee 1 :
Employee Id: 1010
Name: Ali
Deparment: Cleaning
Salary: 13000

Employee 2 :
Employee Id: 1011
Name: Ahmed
Deparment: Management
Salary: 100000

Employee 3 :
Employee Id: 1012
Name: Bilal
Deparment: Management
Salary: 100000

Employee 4 :
Employee Id: 1013
Name: Charlie
Deparment: Teaching
Salary: 50000

Employee 5 :
Employee Id: 1014
Name: Darwin
Deparment: Teaching
Salary: 50000

PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> 
```

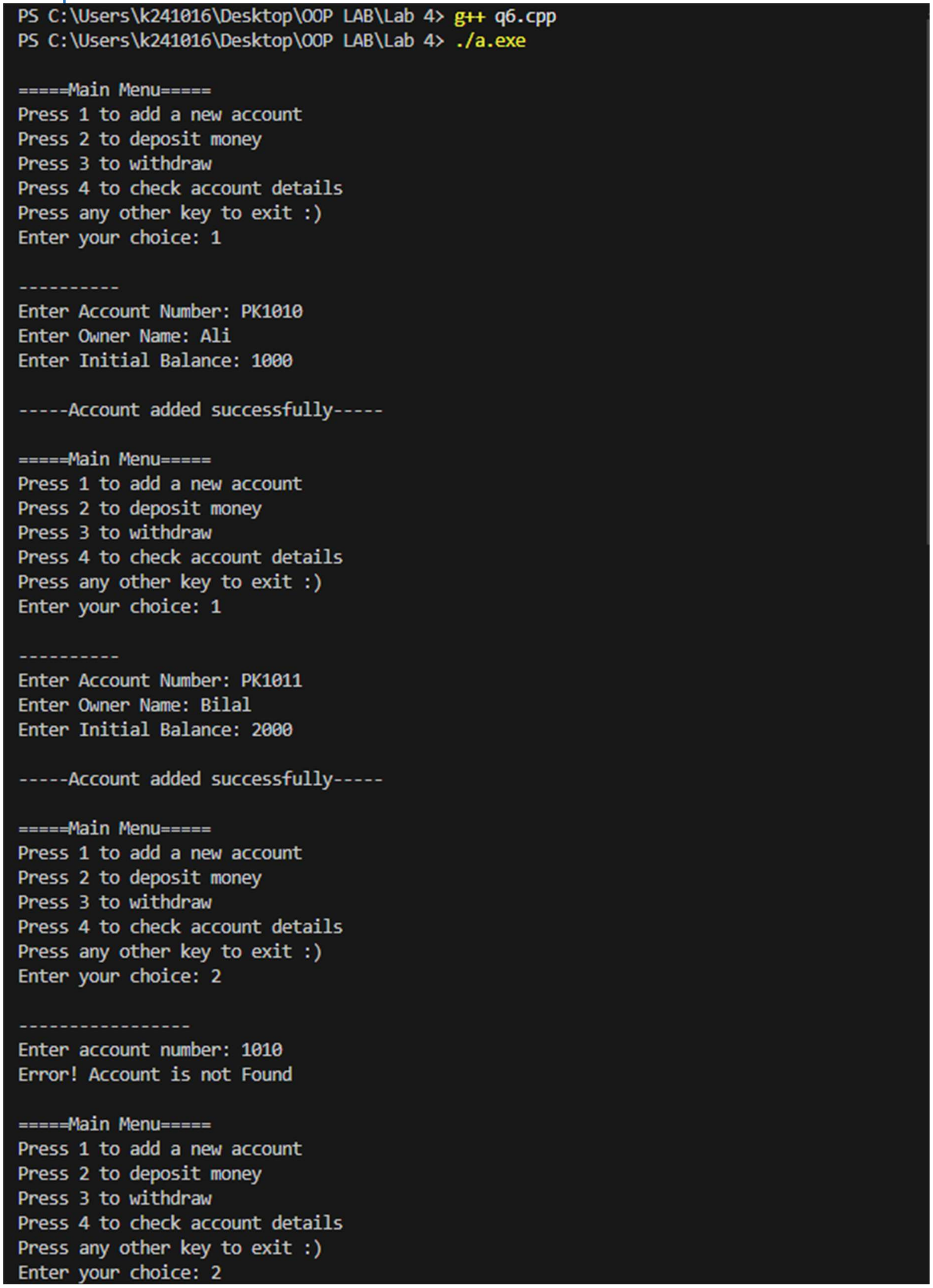
Question 6:

Source Code:

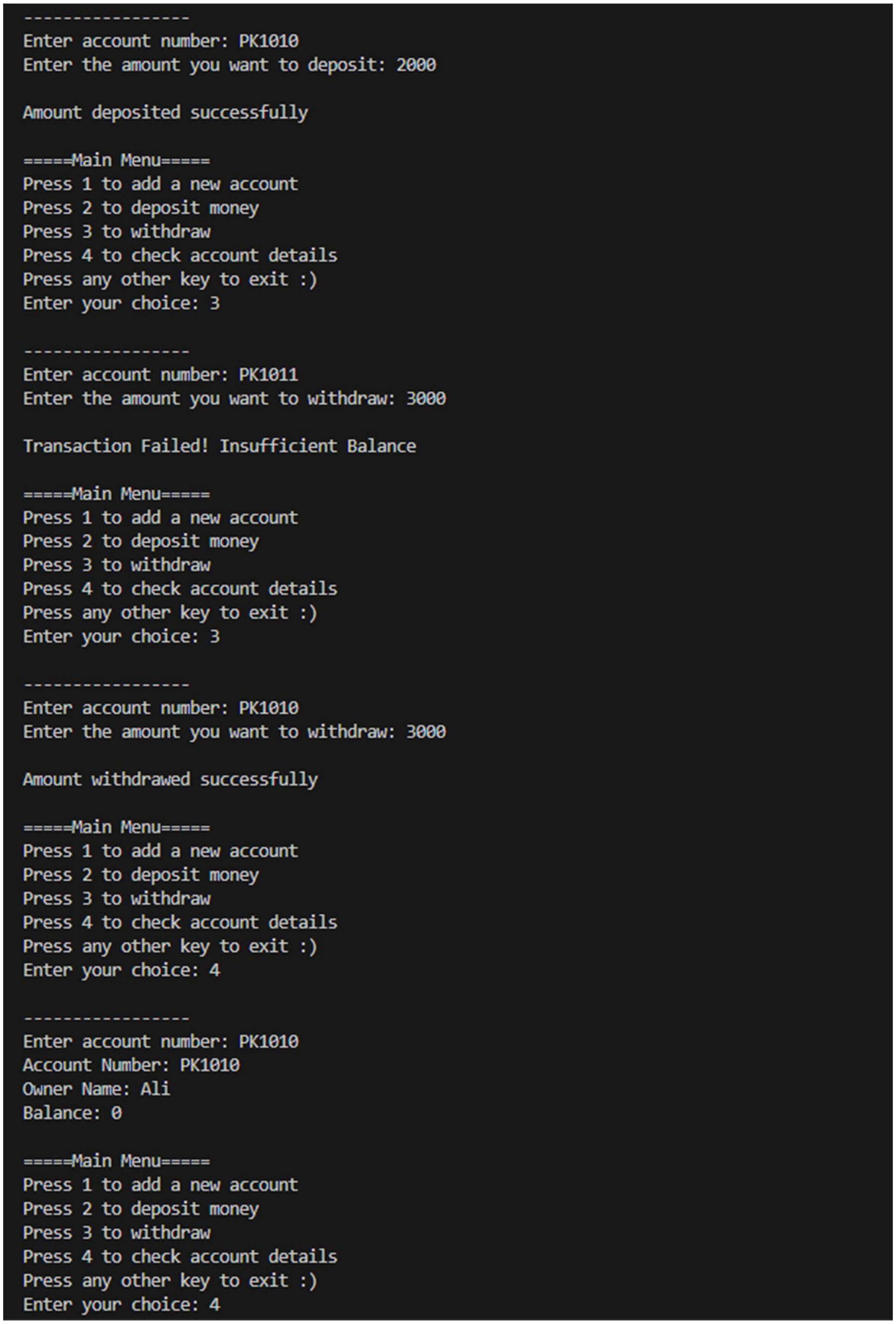
```
/*
Scenario: A bank wants to develop a system for managing customer accounts. The
system
should allow customers to:
1. Create a new account with an account number, owner’s name, and initial balance
(default balance is 0 if not provided).
2. Deposit money into their account.
3. Withdraw money from their account, ensuring they cannot withdraw more than the
available balance.
4. Display account details including account number, owner’s name, and current
balance.
Your task is to implement a C++ program that fulfills these requirements.
*/

#include <iostream>
using namespace std;

class Account{
    string accountNumber;
    string accountHolderName;
    double balance;
    public:
    Account(string Account_Number="", string Holder_Name="", double Balance=0.0){
        accountNumber=Account_Number;
        accountHolderName=Holder_Name;
        balance=Balance;
    }
    string get_ID(){
        return accountNumber;
    }
    void deposit(double amount){
        balance+=amount;
        cout<<"\nAmount deposited successfully"<<endl;
    }
    void withdraw(double amount){
        if (amount<=balance){
            balance-=amount;
            cout<<"\nAmount withdrawed successfully"<<endl;
        }
        else{
            cout<<"\nTransaction Failed! Insufficient Balance"<<endl;
        }
    }
    void displayDetails(){
        cout<<"Account Number: "<<accountNumber<<endl;
        cout<<"Owner Name: "<<accountHolderName<<endl;
        cout<<"Balance: "<<balance<<endl;
    }
};

class Bank{
    Account *accounts;
    int noOfAccounts;
    public:
    Bank(){
        accounts=nullptr;
        noOfAccounts=0;
    }
    ~Bank(){
        delete[] accounts;
    }
    void newAccount(Account account){
        Account *temp_shifting=new Account[++noOfAccounts];
        int i, found=0;
        for (i=0; i<noOfAccounts-1; i++){
            if (accounts[i].get_ID()==account.get_ID()){
                found=1; break;
            }
            temp_shifting[i]=accounts[i];
        }
        if (found==0){
            temp_shifting[i]=account;
            delete[] accounts;
            accounts=temp_shifting;
            cout<<"\n-----Account added successfully-----\n";
        }
        else{
            delete[] temp_shifting;
            noOfAccounts--;
            cout<<"\n-----Error! Account must have a unique Account Number-----
\n";
        }
    }
    void deposit_money(){
        cout<<"\n-----------------\n";
        cout<<"Enter account number: ";
        string ac;
        cin>>ac;
        for (int i=0; i<noOfAccounts; i++){
            if (ac==accounts[i].get_ID()){
                cout<<"Enter the amount you want to deposit: ";
                double temp;
                cin>>temp;
                accounts[i].deposit(temp);
                return;
            }
        }
        cout<<"Error! Account is not Found\n";
    }
    void withdraw_money(){
        cout<<"\n-----------------\n";
        cout<<"Enter account number: ";
        string ac;
        cin>>ac;
        for (int i=0; i<noOfAccounts; i++){
            if (ac==accounts[i].get_ID()){
                cout<<"Enter the amount you want to withdraw: ";
                double temp;
                cin>>temp;
                accounts[i].withdraw(temp);
                return;
            }
        }
```

```cpp
        cout<<"Error! Account is not Found\n";
    }
    void display_account_details(){
        cout<<"\n-----------------\n";
        cout<<"Enter account number: ";
        string ac;
        cin>>ac;
        for (int i=0; i<noOfAccounts; i++){
            if (ac==accounts[i].get_ID()){
                accounts[i].displayDetails();
                return;
            }
        }
        cout<<"Error! Account is not Found\n";
    }
};

int main(){
    string choice, id, title;
    double balance;
    Bank State_Bank_Of_Pakistan;
    while (1){
        cout<<"\n=====Main Menu=====\n";
        cout<<"Press 1 to add a new account\nPress 2 to deposit money\nPress 3 to withdraw\nPress 4 to check account details\nPress any other key to exit :)\n";
        cout<<"Enter your choice: ";
        cin>>choice;
        if (choice=="1"){
            cout<<"\n----------\n";
            cout<<"Enter Account Number: ";

            cin>>id;

            cout<<"Enter Owner Name: ";

            cin.ignore();

            getline(cin, title);

            cout<<"Enter Initial Balance: ";

            cin>>balance;

            State_Bank_Of_Pakistan.newAccount(Account(id, title, balance));

        }

        else if (choice=="2"){

            State_Bank_Of_Pakistan.deposit_money();

        }

        else if (choice=="3"){

            State_Bank_Of_Pakistan.withdraw_money();

        }

        else if (choice=="4"){

            State_Bank_Of_Pakistan.display_account_details();

        }

        else{

            cout<<"\n=====Thank You for using our service=====";

            break;

        }

    }

}
```

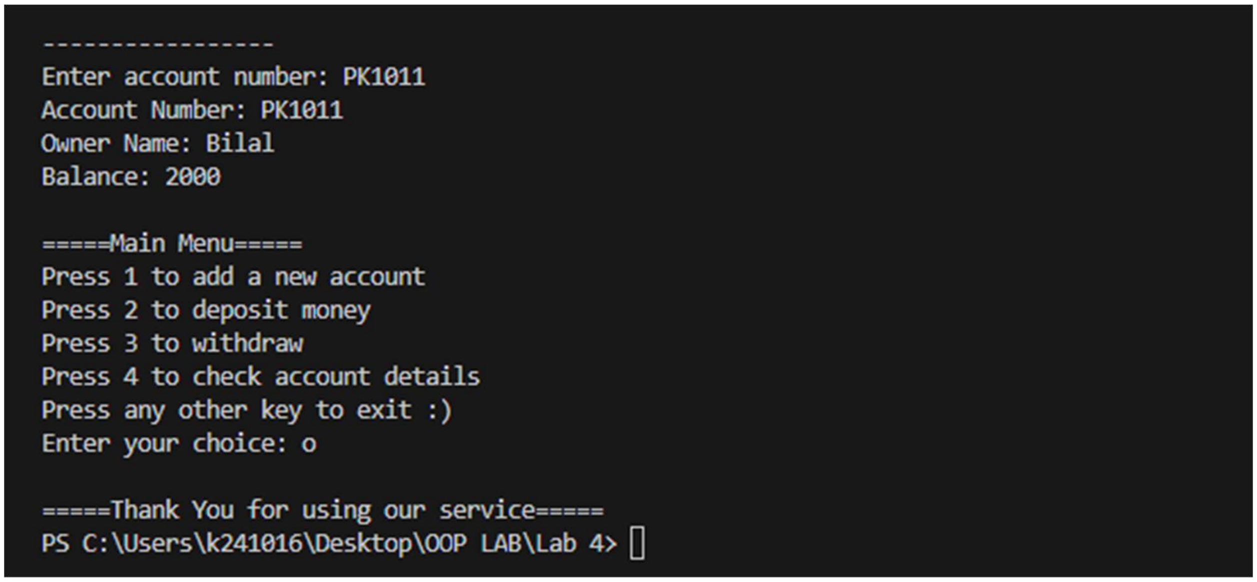
Output:



```
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> g++ q6.cpp
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4> ./a.exe

=====Main Menu=====
Press 1 to add a new account
Press 2 to deposit money
Press 3 to withdraw
Press 4 to check account details
Press any other key to exit :)
Enter your choice: 1

----------
Enter Account Number: PK1010
Enter Owner Name: Ali
Enter Initial Balance: 1000

-----Account added successfully-----

=====Main Menu=====
Press 1 to add a new account
Press 2 to deposit money
Press 3 to withdraw
Press 4 to check account details
Press any other key to exit :)
Enter your choice: 1

----------
Enter Account Number: PK1011
Enter Owner Name: Bilal
Enter Initial Balance: 2000

-----Account added successfully-----

=====Main Menu=====
Press 1 to add a new account
Press 2 to deposit money
Press 3 to withdraw
Press 4 to check account details
Press any other key to exit :)
Enter your choice: 2

-----------------
Enter account number: 1010
Error! Account is not Found

=====Main Menu=====
Press 1 to add a new account
Press 2 to deposit money
Press 3 to withdraw
Press 4 to check account details
Press any other key to exit :)
Enter your choice: 2
```



```
-----------------
Enter account number: PK1010
Enter the amount you want to deposit: 2000

Amount deposited successfully

=====Main Menu=====
Press 1 to add a new account
Press 2 to deposit money
Press 3 to withdraw
Press 4 to check account details
Press any other key to exit :)
Enter your choice: 3

-----------------
Enter account number: PK1011
Enter the amount you want to withdraw: 3000

Transaction Failed! Insufficient Balance

=====Main Menu=====
Press 1 to add a new account
Press 2 to deposit money
Press 3 to withdraw
Press 4 to check account details
Press any other key to exit :)
Enter your choice: 3

-----------------
Enter account number: PK1010
Enter the amount you want to withdraw: 3000

Amount withdrawed successfully

=====Main Menu=====
Press 1 to add a new account
Press 2 to deposit money
Press 3 to withdraw
Press 4 to check account details
Press any other key to exit :)
Enter your choice: 4

-----------------
Enter account number: PK1010
Account Number: PK1010
Owner Name: Ali
Balance: 0

=====Main Menu=====
Press 1 to add a new account
Press 2 to deposit money
Press 3 to withdraw
Press 4 to check account details
Press any other key to exit :)
Enter your choice: 4
```



```
-----------------
Enter account number: PK1011
Account Number: PK1011
Owner Name: Bilal
Balance: 2000

=====Main Menu=====
Press 1 to add a new account
Press 2 to deposit money
Press 3 to withdraw
Press 4 to check account details
Press any other key to exit :)
Enter your choice: o

=====Thank You for using our service=====
PS C:\Users\k241016\Desktop\OOP LAB\Lab 4>
```