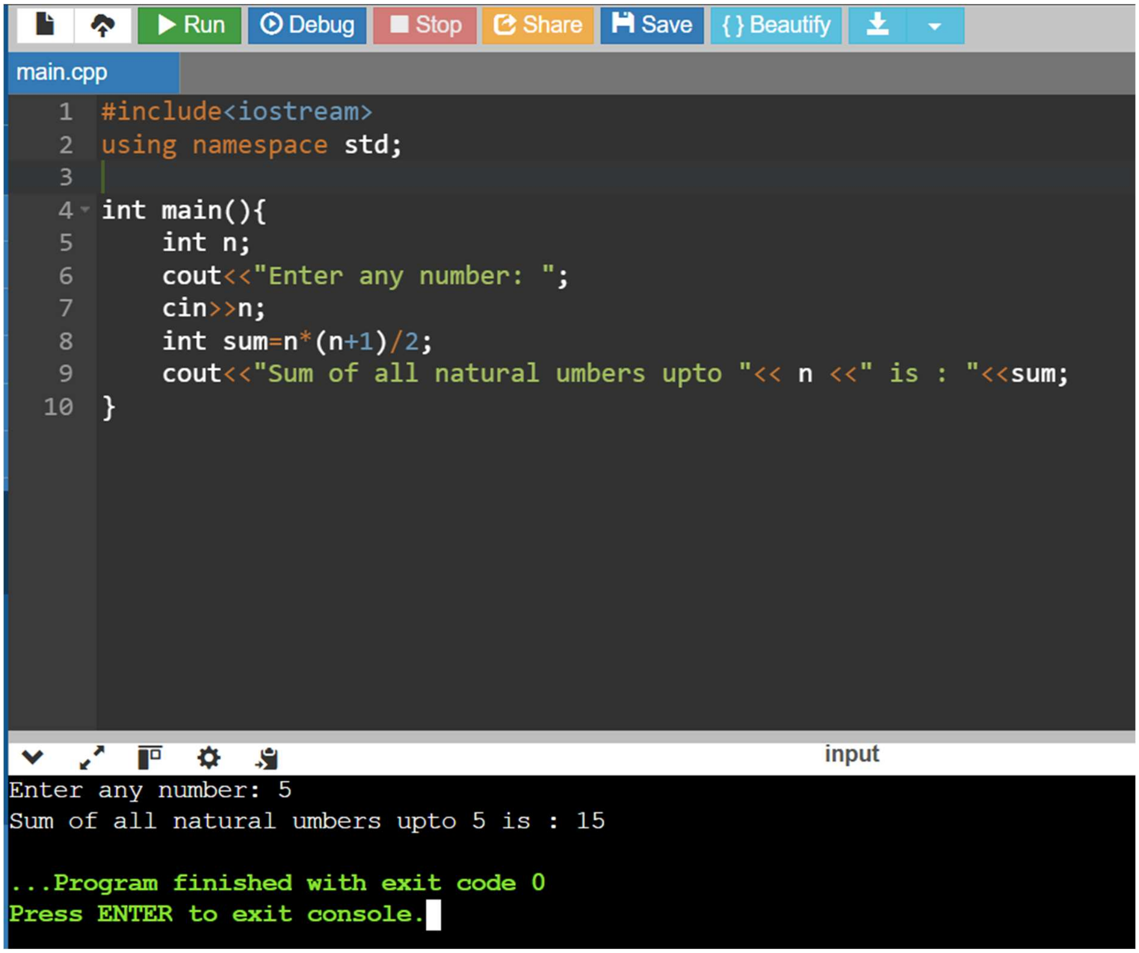
WWC

Name: Siddharth Lal Sec: KPIT-901(B)

UID: 22bcs13410

DAY 1

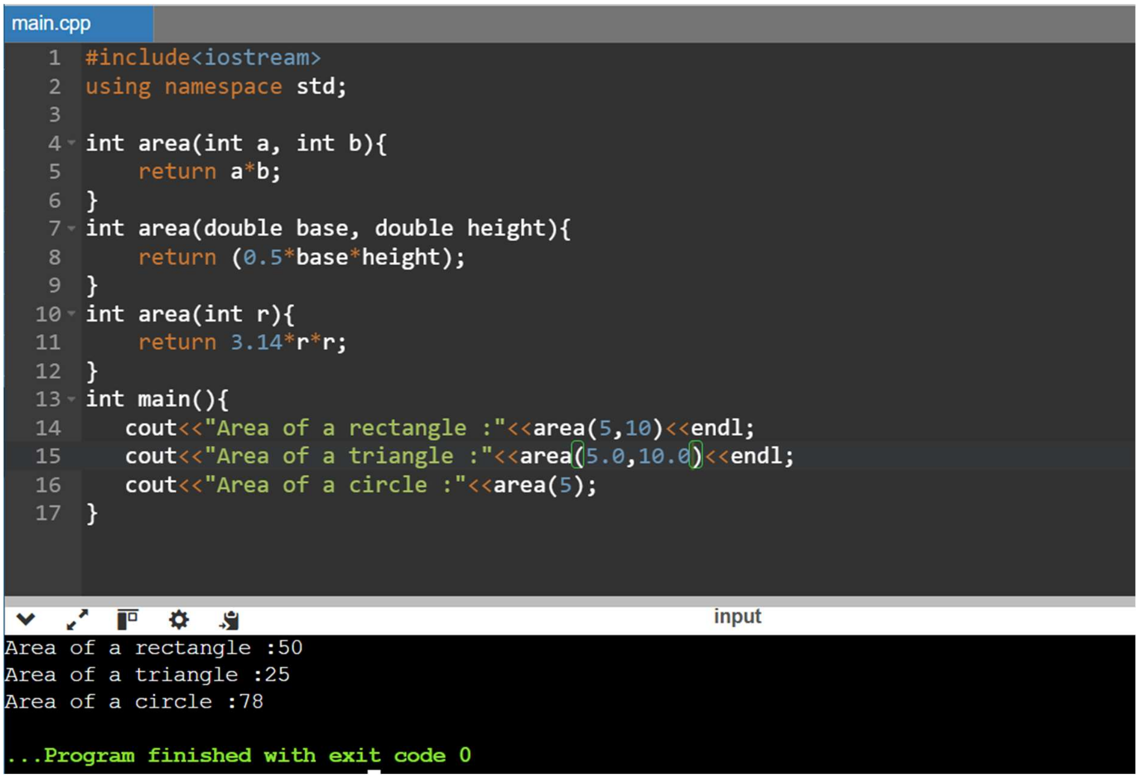
Problem 1



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

int main(){
    int n;
    cout<<"Enter any number: ";
    cin>>n;
    int sum=n*(n+1)/2;
    cout<<"Sum of all natural umbers upto "<< n <<" is : "<<sum;
}
```

```
Enter any number: 5
Sum of all natural umbers upto 5 is : 15

...Program finished with exit code 0
Press ENTER to exit console.
```

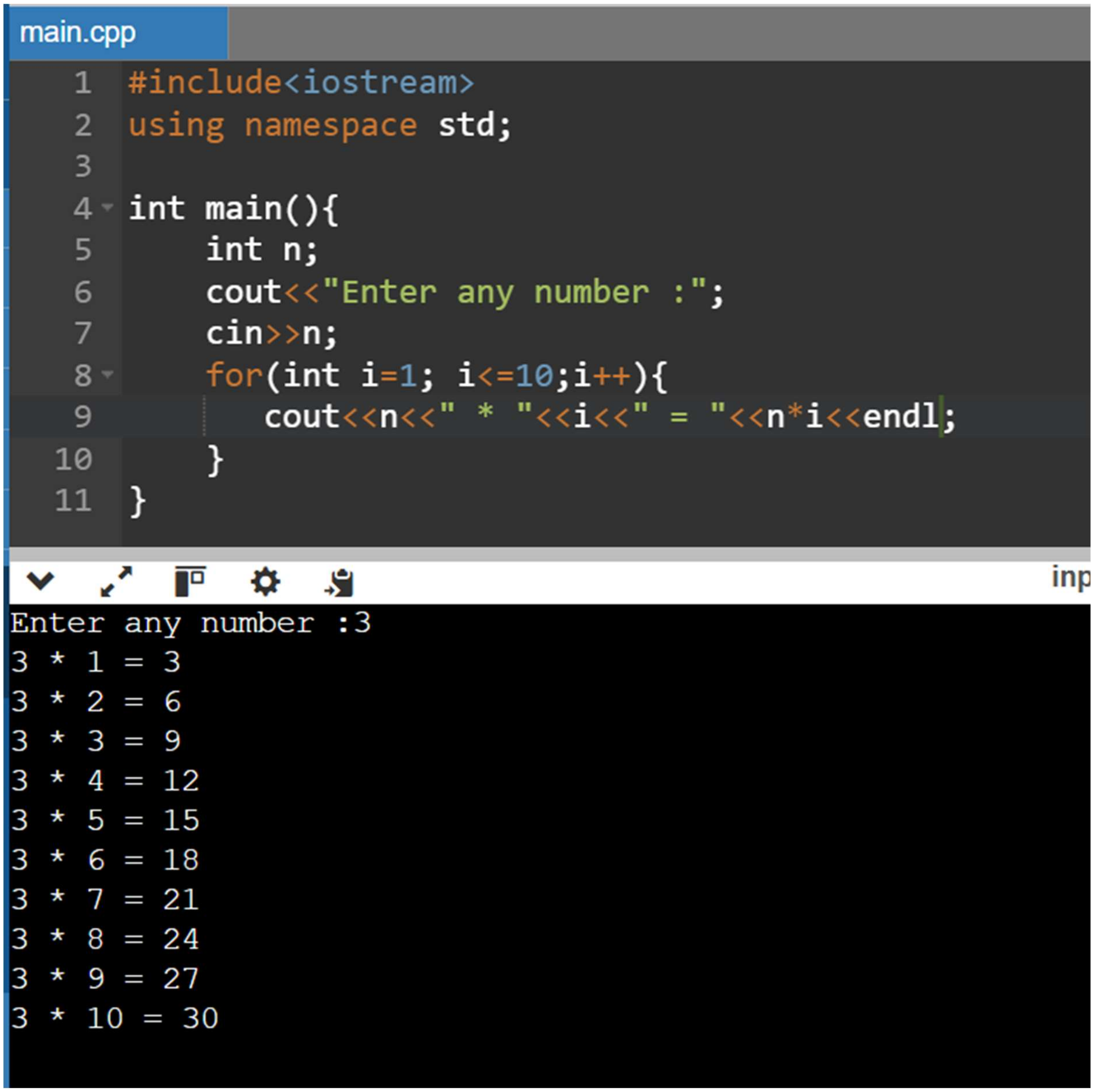
Problem 2



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

int area(int a, int b){
    return a*b;
}
int area(double base, double height){
    return (0.5*base*height);
}
int area(int r){
    return 3.14*r*r;
}
int main(){
   cout<<"Area of a rectangle :"<<area(5,10)<<endl;
   cout<<"Area of a triangle :"<<area(5.0,10.0)<<endl;
   cout<<"Area of a circle :"<<area(5);
}
```

```
Area of a rectangle :50
Area of a triangle :25
Area of a circle :78

...Program finished with exit code 0
```

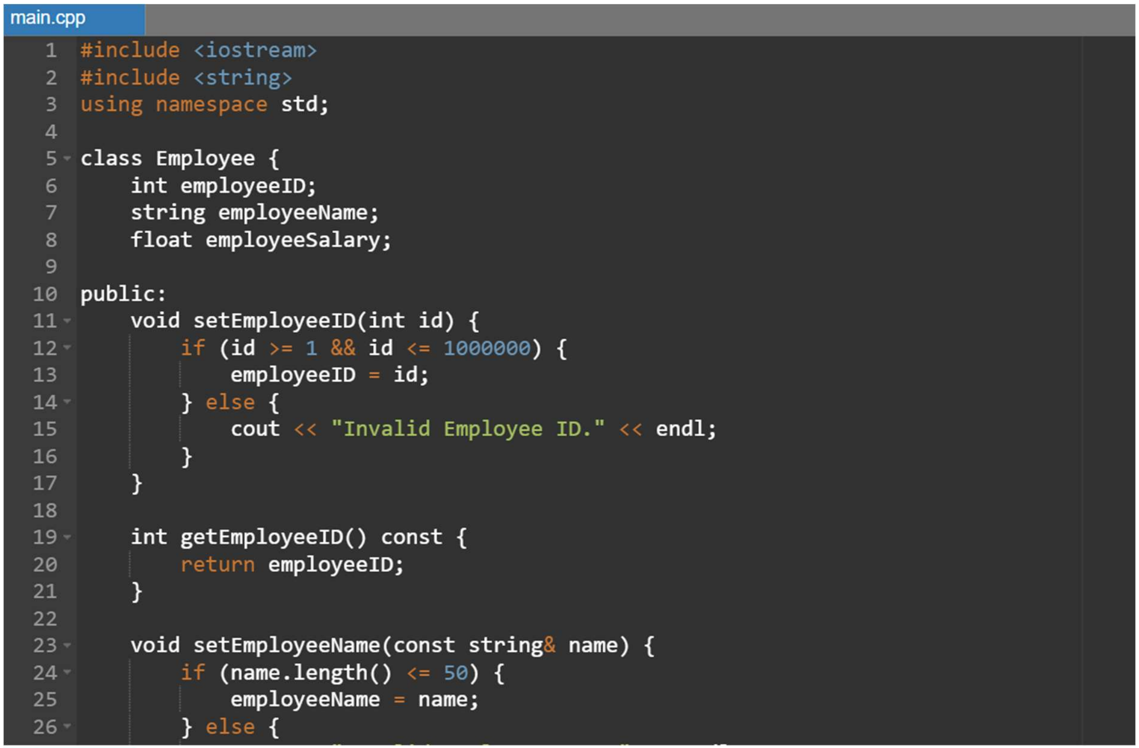
Problem 3



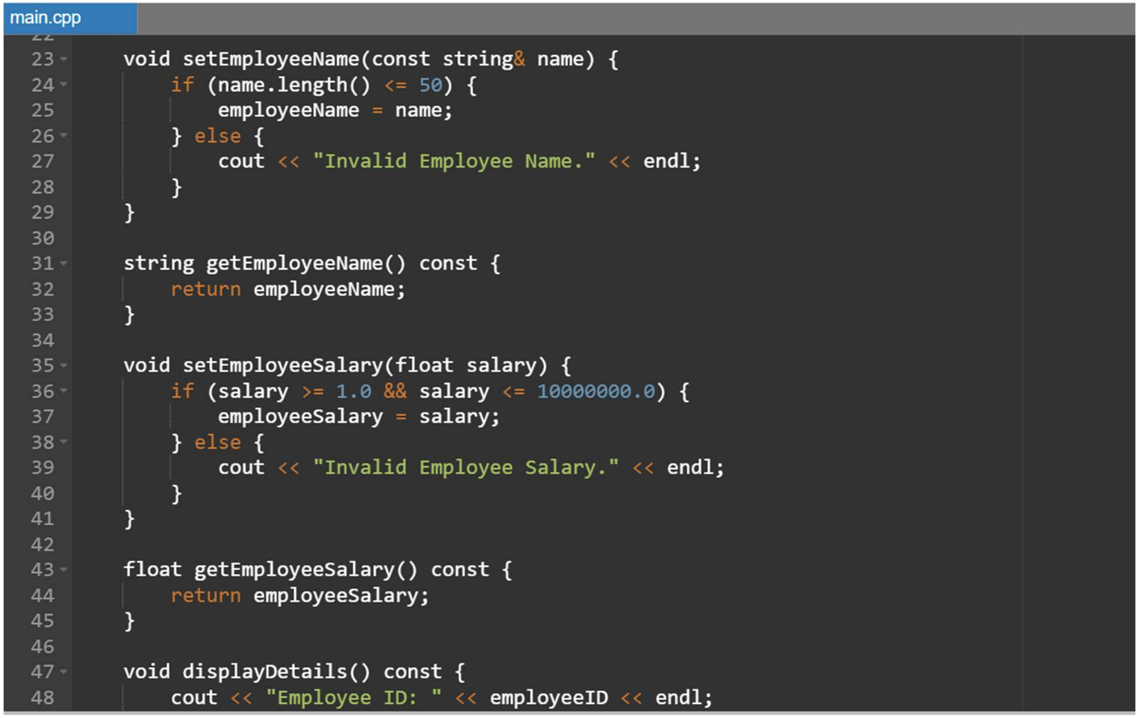
```cpp
#include<iostream>
using namespace std;

int main(){
    int n;
    cout<<"Enter any number :";
    cin>>n;
    for(int i=1; i<=10;i++){
        cout<<n<<" * "<<i<<" = "<<n*i<<endl;
    }
}
```

```
Enter any number :3
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
3 * 7 = 21
3 * 8 = 24
3 * 9 = 27
3 * 10 = 30
```

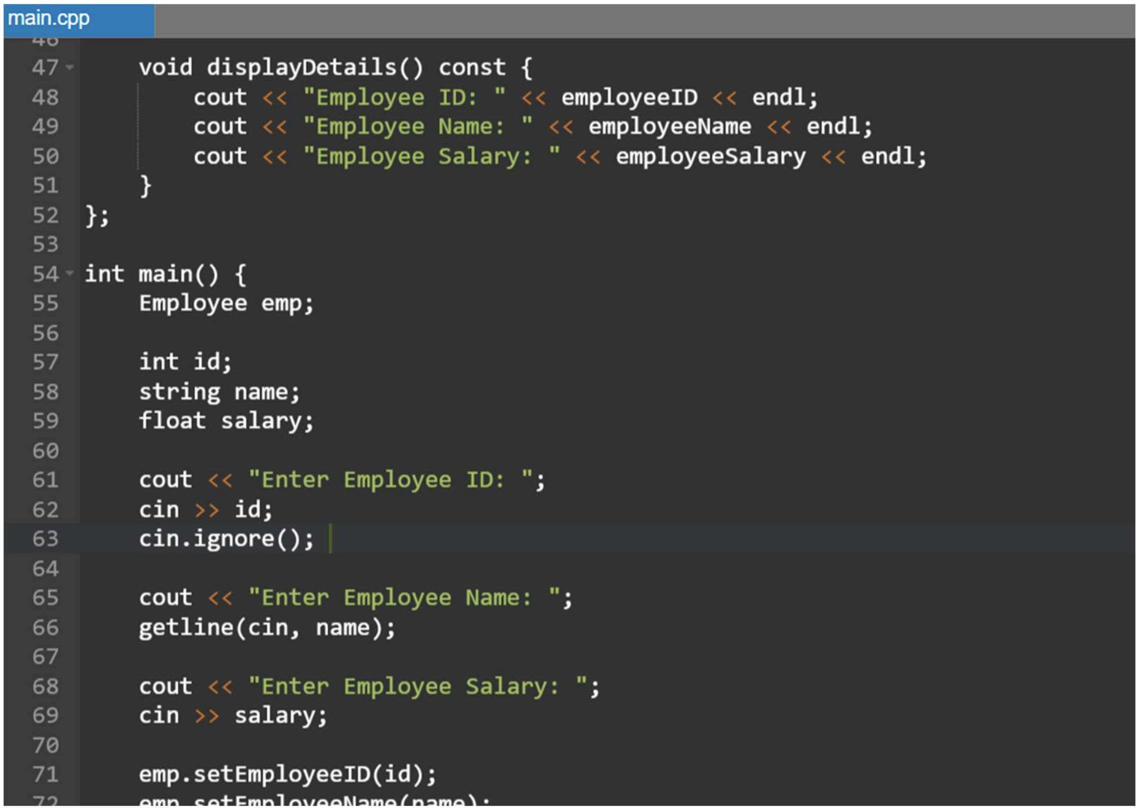
Problem 4



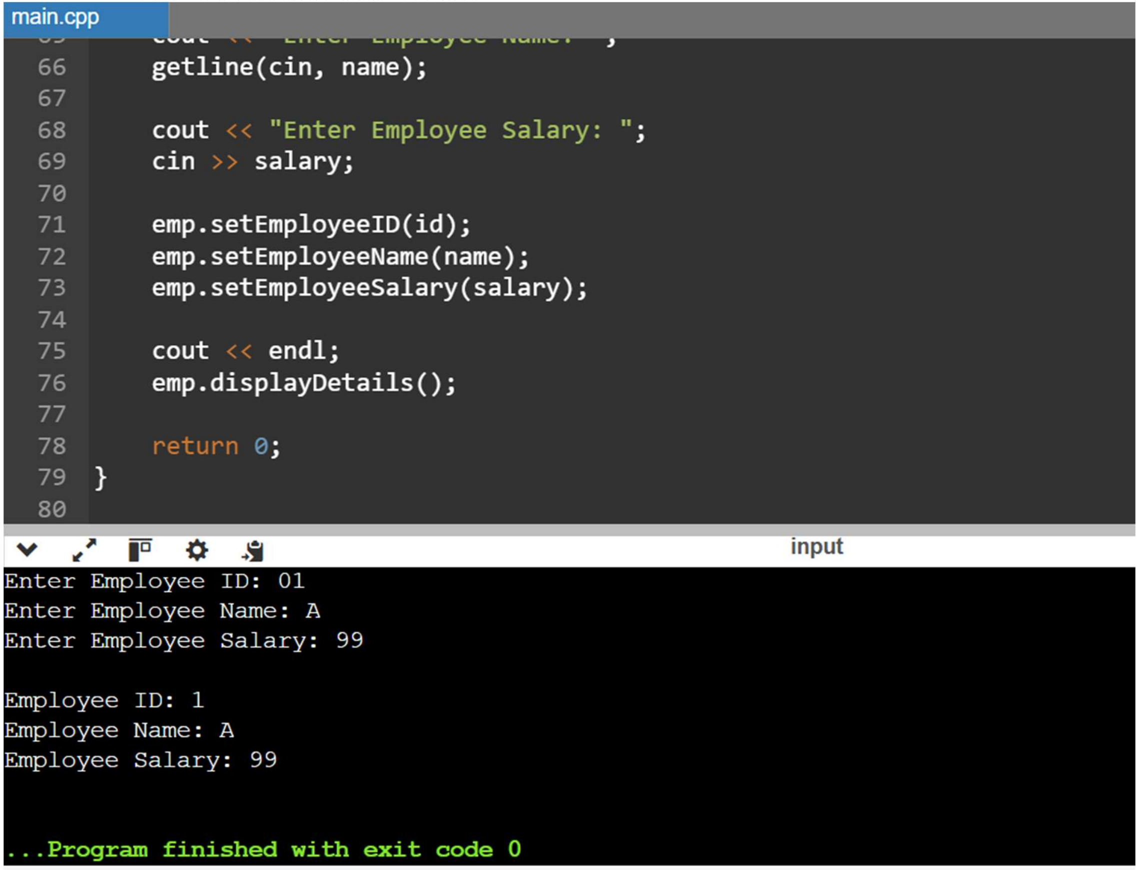
```cpp
#include <iostream>
#include <string>
using namespace std;

class Employee {
    int employeeID;
    string employeeName;
    float employeeSalary;

public:
    void setEmployeeID(int id) {
        if (id >= 1 && id <= 1000000) {
            employeeID = id;
        } else {
            cout << "Invalid Employee ID." << endl;
        }
    }

    int getEmployeeID() const {
        return employeeID;
    }

    void setEmployeeName(const string& name) {
        if (name.length() <= 50) {
            employeeName = name;
        } else {
```



```cpp
    void setEmployeeName(const string& name) {
        if (name.length() <= 50) {
            employeeName = name;
        } else {
            cout << "Invalid Employee Name." << endl;
        }
    }

    string getEmployeeName() const {
        return employeeName;
    }

    void setEmployeeSalary(float salary) {
        if (salary >= 1.0 && salary <= 10000000.0) {
            employeeSalary = salary;
        } else {
            cout << "Invalid Employee Salary." << endl;
        }
    }

    float getEmployeeSalary() const {
        return employeeSalary;
    }

    void displayDetails() const {
        cout << "Employee ID: " << employeeID << endl;
```



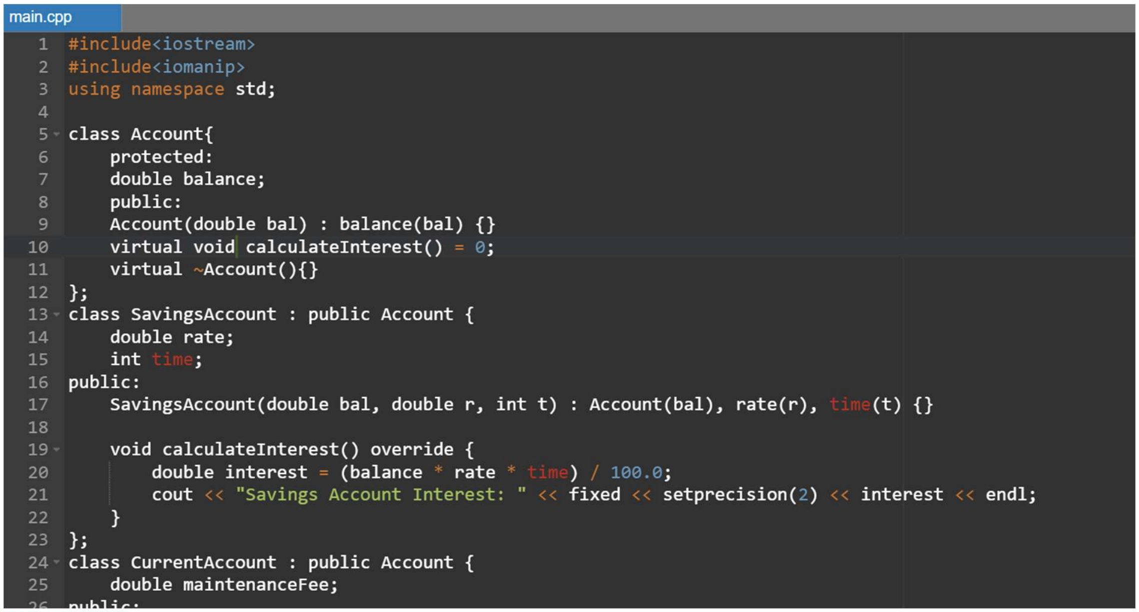
```cpp
    void displayDetails() const {
        cout << "Employee ID: " << employeeID << endl;
        cout << "Employee Name: " << employeeName << endl;
        cout << "Employee Salary: " << employeeSalary << endl;
    }
};

int main() {
    Employee emp;

    int id;
    string name;
    float salary;

    cout << "Enter Employee ID: ";
    cin >> id;
    cin.ignore();

    cout << "Enter Employee Name: ";
    getline(cin, name);

    cout << "Enter Employee Salary: ";
    cin >> salary;

    emp.setEmployeeID(id);
    emp.setEmployeeName(name);
```



```cpp
    cout << "Enter Employee Name: ";
    getline(cin, name);

    cout << "Enter Employee Salary: ";
    cin >> salary;

    emp.setEmployeeID(id);
    emp.setEmployeeName(name);
    emp.setEmployeeSalary(salary);

    cout << endl;
    emp.displayDetails();

    return 0;
}
```

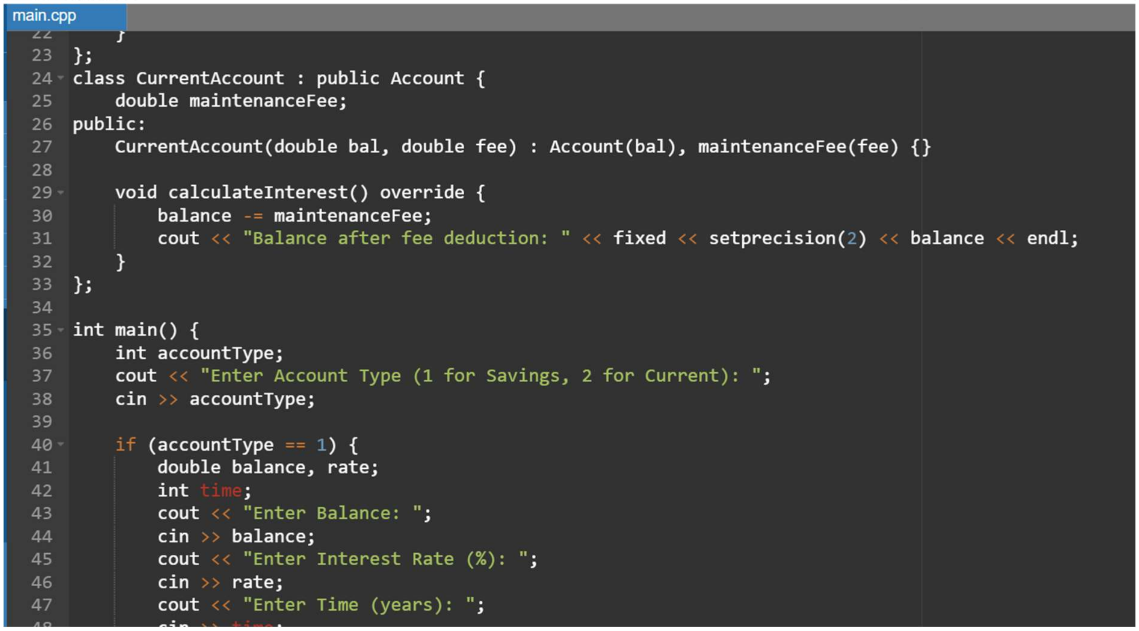
```
Enter Employee ID: 01
Enter Employee Name: A
Enter Employee Salary: 99

Employee ID: 1
Employee Name: A
Employee Salary: 99


...Program finished with exit code 0
```

Problem 5



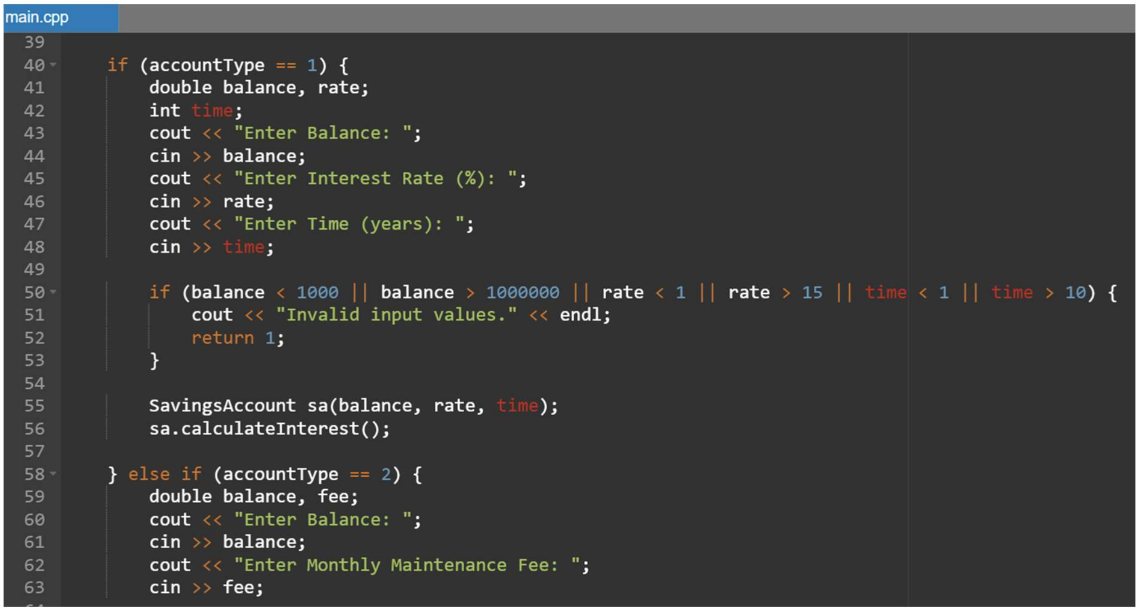
main.cpp

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

class Account{
    protected:
    double balance;
    public:
    Account(double bal) : balance(bal) {}
    virtual void calculateInterest() = 0;
    virtual ~Account(){}
};
class SavingsAccount : public Account {
    double rate;
    int time;
public:
    SavingsAccount(double bal, double r, int t) : Account(bal), rate(r), time(t) {}

    void calculateInterest() override {
        double interest = (balance * rate * time) / 100.0;
        cout << "Savings Account Interest: " << fixed << setprecision(2) << interest << endl;
    }
};
class CurrentAccount : public Account {
    double maintenanceFee;
public:
```

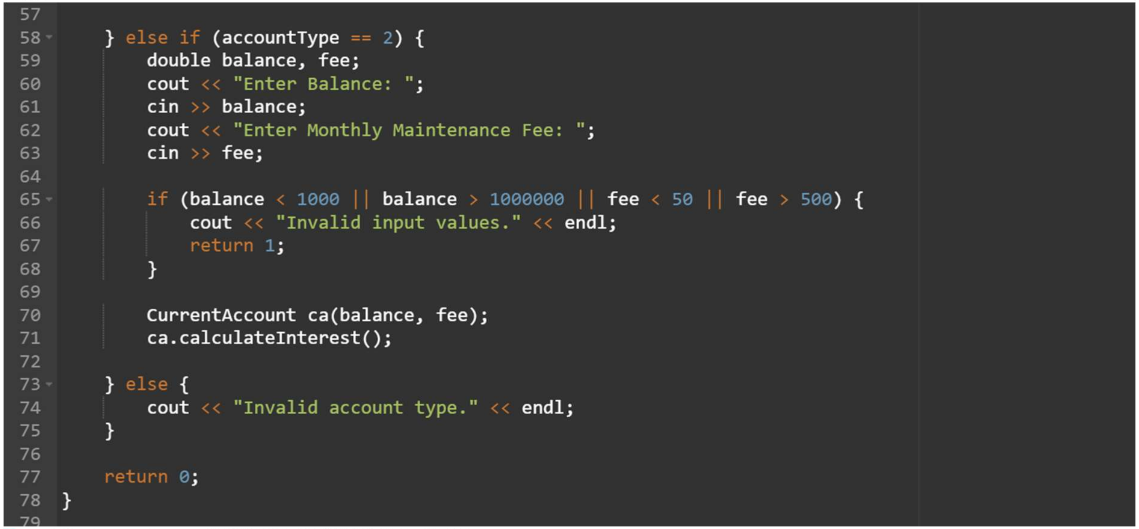


main.cpp

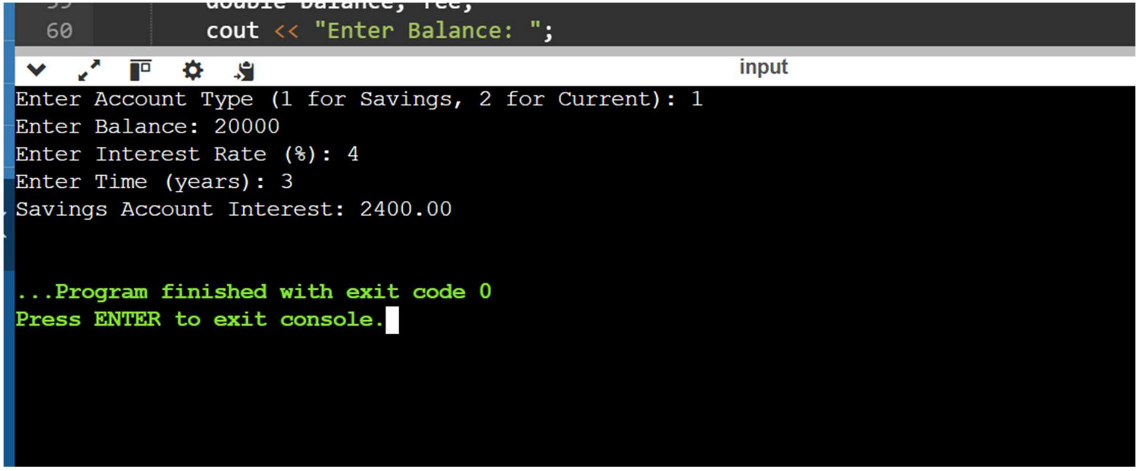
```cpp
    }
};
class CurrentAccount : public Account {
    double maintenanceFee;
public:
    CurrentAccount(double bal, double fee) : Account(bal), maintenanceFee(fee) {}

    void calculateInterest() override {
        balance -= maintenanceFee;
        cout << "Balance after fee deduction: " << fixed << setprecision(2) << balance << endl;
    }
};

int main() {
    int accountType;
    cout << "Enter Account Type (1 for Savings, 2 for Current): ";
    cin >> accountType;

    if (accountType == 1) {
        double balance, rate;
        int time;
        cout << "Enter Balance: ";
        cin >> balance;
        cout << "Enter Interest Rate (%): ";
        cin >> rate;
        cout << "Enter Time (years): ";
        cin >> time;
```



```cpp

    if (accountType == 1) {
        double balance, rate;
        int time;
        cout << "Enter Balance: ";
        cin >> balance;
        cout << "Enter Interest Rate (%): ";
        cin >> rate;
        cout << "Enter Time (years): ";
        cin >> time;

        if (balance < 1000 || balance > 1000000 || rate < 1 || rate > 15 || time < 1 || time > 10) {
            cout << "Invalid input values." << endl;
            return 1;
        }

        SavingsAccount sa(balance, rate, time);
        sa.calculateInterest();

    } else if (accountType == 2) {
        double balance, fee;
        cout << "Enter Balance: ";
        cin >> balance;
        cout << "Enter Monthly Maintenance Fee: ";
        cin >> fee;
```



```cpp

    } else if (accountType == 2) {
        double balance, fee;
        cout << "Enter Balance: ";
        cin >> balance;
        cout << "Enter Monthly Maintenance Fee: ";
        cin >> fee;

        if (balance < 1000 || balance > 1000000 || fee < 50 || fee > 500) {
            cout << "Invalid input values." << endl;
            return 1;
        }

        CurrentAccount ca(balance, fee);
        ca.calculateInterest();

    } else {
        cout << "Invalid account type." << endl;
    }

    return 0;
}
```



```cpp
        double balance, fee;
        cout << "Enter Balance: ";
```

```
Enter Account Type (1 for Savings, 2 for Current): 1
Enter Balance: 20000
Enter Interest Rate (%): 4
Enter Time (years): 3
Savings Account Interest: 2400.00


...Program finished with exit code 0
Press ENTER to exit console.
```