LAB SESSION 9

Question 1: Shape Area Calculation

Problem Statement:

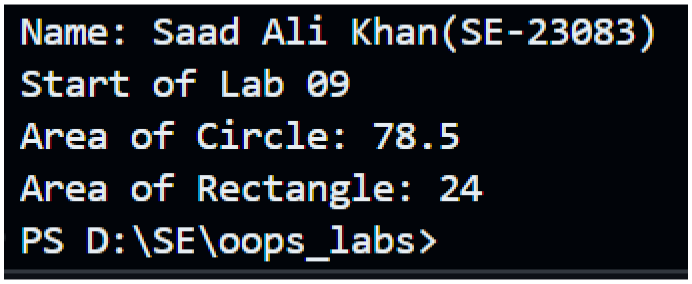
Create an abstract class Shape with a pure virtual function area(). Derive two classes Circle and Rectangle from Shape. Implement the area() function in both derived classes. Write a program to calculate the area of a circle and a rectangle.

Code:

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

int startlab9()
{
    cout << "Name: Saad Ali Khan(SE-23083)" << endl;
    cout << "Start of Lab 09" << endl;
    return 0;
}

class Shape
{
public:
    virtual double area() const = 0;
    virtual ~Shape() = default;
};

class Circle : public Shape
{
private:
    double radius;

public:
    Circle(double r) : radius(r) {}

    double area() const override
    {
        return 3.14 * radius * radius;
    }
};

class Rectangle : public Shape
{
```

```cpp
private:
    double width;
    double height;

public:
    Rectangle(double w, double h) : width(w), height(h) {}

    double area() const override
    {
        return width * height;
    }
};

int l9q1()
{
    Circle c(5.0);
    Rectangle r(4.0, 6.0);

    cout << "Area of Circle: " << c.area() << endl;
    cout << "Area of Rectangle: " << r.area() << endl;
    return 0;
}

int main()
{
    startlab9();
    l9q1();
    return 0;
}
```

Output:



```
Name: Saad Ali Khan(SE-23083)
Start of Lab 09
Area of Circle: 78.5
Area of Rectangle: 24
PS D:\SE\oops_labs>
```

Question 2: Employee Salary Calculation

Problem Statement:

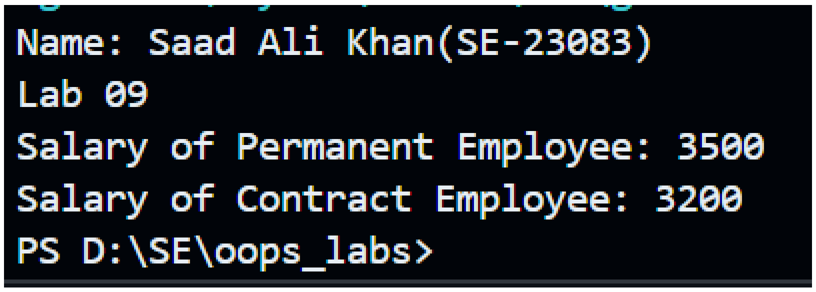
Create an abstract class Employee with a pure virtual function calculateSalary(). Derive two classes PermanentEmployee and ContractEmployee from Employee. Implement the calculateSalary() function in both derived classes. Write a program to calculate the salary of a permanent employee and a contract employee.

Code:

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

int startlab9()
{
    cout << "Name: Saad Ali Khan(SE-23083)" << endl;
    cout << "Lab 09" << endl;
    return 0;
}

class Employee
{
public:
    virtual double calculateSalary() const = 0;
    virtual ~Employee() = default;
};

class PermanentEmployee : public Employee
{
private:
    double basicSalary;
    double bonus;

public:
    PermanentEmployee(double basic, double bon) : basicSalary(basic), bonus(bon)
{}

    double calculateSalary() const override
    {
        return basicSalary + bonus;
    }
};

class ContractEmployee : public Employee
```

```
{
private:
    double hourlyRate;
    int hoursWorked;

public:
    ContractEmployee(double rate, int hours) : hourlyRate(rate),
hoursWorked(hours) {}

    double calculateSalary() const override
    {
        return hourlyRate * hoursWorked;
    }
};

int l9q2()
{
    PermanentEmployee pe(3000.0, 500.0);
    ContractEmployee ce(20.0, 160);

    cout << "Salary of Permanent Employee: " << pe.calculateSalary() << endl;
    cout << "Salary of Contract Employee: " << ce.calculateSalary() << endl;
    return 0;
}

int main()
{
    startlab9();
    l9q2();
    return 0;
}
```

Output:



```
Name: Saad Ali Khan(SE-23083)
Lab 09
Salary of Permanent Employee: 3500
Salary of Contract Employee: 3200
PS D:\SE\oops_labs>
```

Question 3: Vehicle Fuel Efficiency

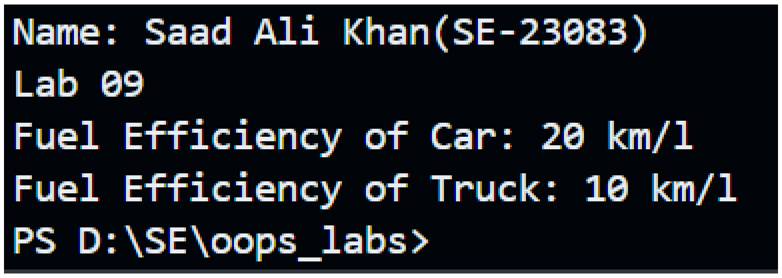
Problem Statement:

Create an abstract class Vehicle with a pure virtual function fuelEfficiency(). Derive two classes Car and Truck from Vehicle. Implement the fuelEfficiency() function in both derived classes. Write a program to calculate the fuel efficiency of a car and a truck.

Code:

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

int startlab9()
{
    cout << "Name: Saad Ali Khan(SE-23083)" << endl;
    cout << "Lab 09" << endl;
    return 0;
}

class Vehicle
{
public:
    virtual double fuelEfficiency() const = 0;
    virtual ~Vehicle() = default;
};

class Car : public Vehicle
{
private:
    double distance;
    double fuelConsumed;

public:
    Car(double d, double f) : distance(d), fuelConsumed(f) {}

    double fuelEfficiency() const override
    {
        return distance / fuelConsumed;
    }
};

class Truck : public Vehicle
{
```

```
private:
    double distance;
    double fuelConsumed;

public:
    Truck(double d, double f) : distance(d), fuelConsumed(f) {}

    double fuelEfficiency() const override
    {
        return distance / fuelConsumed;
    }
};

int l9q3()
{
    Car c(500.0, 25.0);
    Truck t(600.0, 60.0);

    cout << "Fuel Efficiency of Car: " << c.fuelEfficiency() << " km/l" << endl;
    cout << "Fuel Efficiency of Truck: " << t.fuelEfficiency() << " km/l" <<
endl;
    return 0;
}

int main()
{
    startlab9();
    l9q3();
    return 0;
}
```
Output:



```
Name: Saad Ali Khan(SE-23083)
Lab 09
Fuel Efficiency of Car: 20 km/l
Fuel Efficiency of Truck: 10 km/l
PS D:\SE\oops_labs>
```

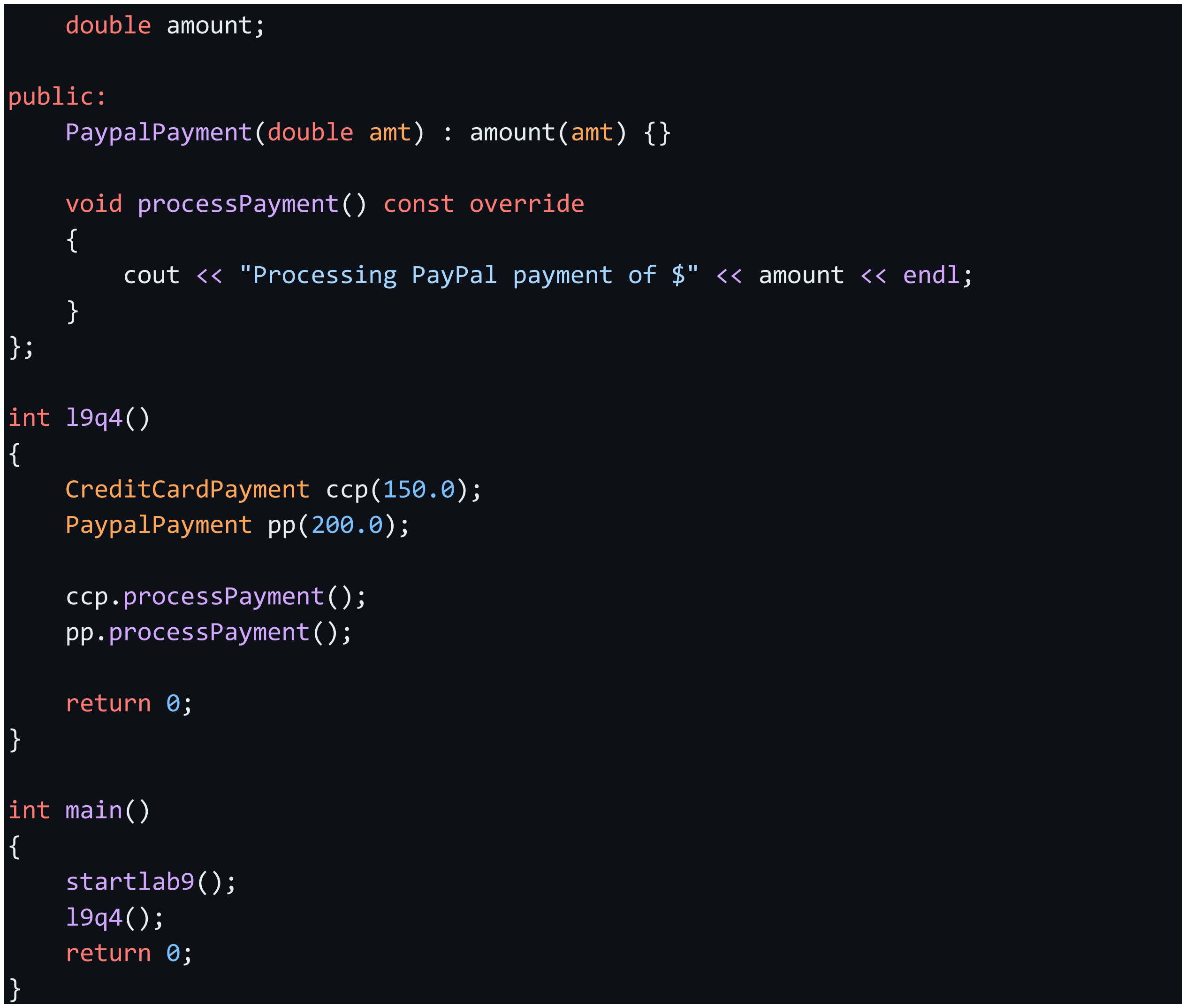
Question 4: Payment Processing System

Problem Statement:

Create an abstract class Payment with a pure virtual function processPayment(). Derive two classes CreditCardPayment and PaypalPayment from Payment. Implement the processPayment() function in both derived classes. Write a program to process a payment through credit card and PayPal.

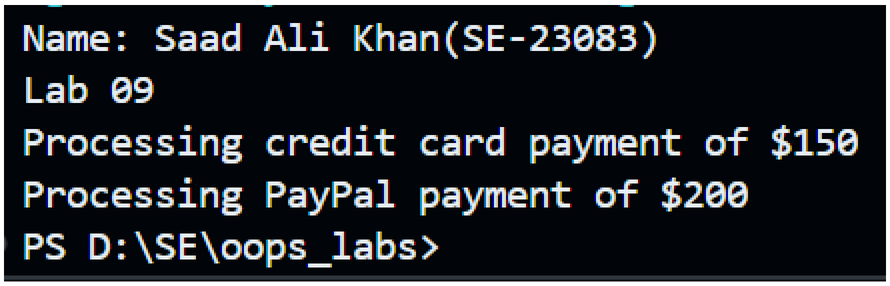
Code:

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

int startlab9()
{
    cout << "Name: Saad Ali Khan(SE-23083)" << endl;
    cout << "Lab 09" << endl;
    return 0;
}

class Payment
{
public:
    virtual void processPayment() const = 0;
    virtual ~Payment() = default;
};

class CreditCardPayment : public Payment
{
private:
    double amount;

public:
    CreditCardPayment(double amt) : amount(amt) {}

    void processPayment() const override
    {
        cout << "Processing credit card payment of $" << amount << endl;
    }
};

class PaypalPayment : public Payment
{
private:
```



```cpp
    double amount;

public:
    PaypalPayment(double amt) : amount(amt) {}

    void processPayment() const override
    {
        cout << "Processing PayPal payment of $" << amount << endl;
    }
};

int l9q4()
{
    CreditCardPayment ccp(150.0);
    PaypalPayment pp(200.0);

    ccp.processPayment();
    pp.processPayment();

    return 0;
}

int main()
{
    startlab9();
    l9q4();
    return 0;
}
```

Output:



```
Name: Saad Ali Khan(SE-23083)
Lab 09
Processing credit card payment of $150
Processing PayPal payment of $200
PS D:\SE\oops_labs>
```

Question 5: Animal Sound Simulation

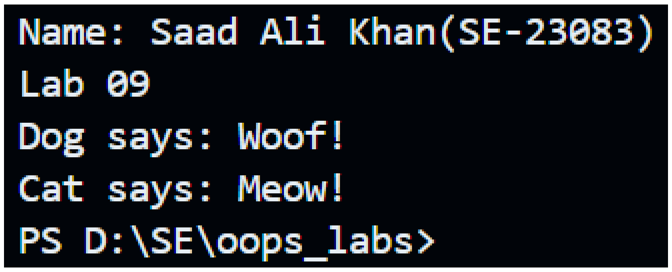
Problem Statement:

Create an abstract class Animal with a pure virtual function makeSound(). Derive two classes Dog and Cat from Animal. Implement the makeSound() function in both derived classes. Write a program to simulate the sounds of a dog and a cat.

Code:

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

int startlab9()
{
    cout << "Name: Saad Ali Khan(SE-23083)" << endl;
    cout << "Lab 09" << endl;
    return 0;
}

class Animal
{
public:
    virtual void makeSound() const = 0;
    virtual ~Animal() = default;
};

class Dog : public Animal
{
public:
    void makeSound() const override
    {
        cout << "Dog says: Woof!" << endl;
    }
};

class Cat : public Animal
{
public:
    void makeSound() const override
    {
        cout << "Cat says: Meow!" << endl;
    }
};

int l9q5()
```

```
{
    Dog d;
    Cat c;

    d.makeSound();
    c.makeSound();
    return 0;
}

int main()
{
    startlab9();
    l9q5();
    return 0;
}
```
Output:



```
Name: Saad Ali Khan(SE-23083)
Lab 09
Dog says: Woof!
Cat says: Meow!
PS D:\SE\oops_labs>
```