ASSIGNMENT #7

Name : Akash Raju Deshmane

Div : A

Roll NO : 51

PRN : 2324000819

```
#include <iostream>
#define PI 3.1415
using namespace std;

// Abstract base class
class Shape {
public:
        // Pure virtual functions
        virtual double getArea() const = 0;
        virtual double getPerimeter() const = 0;
```

```
};

// Derived class for Circle
class Circle : public Shape {
private:
    double radius;

public:
    Circle(double radius){
    this->radius = radius;
    }

    // Override getArea() and getPerimeter()
    double getArea() const override {
        return 3.1415 * radius * radius;
    }

    double getPerimeter() const override {
        return 2 * 3.1415 * radius;
    }
};

// Derived class for Rectangle
class Rectangle : public Shape {
private:
    double height, width;
```

```
public:
    Rectangle(double height, double width){
    this->height = height;
    this->width = width;
    }

    // Override getArea() and getPerimeter()
    double getArea() const override {
        return height * width;
    }

    double getPerimeter() const override {
        return 2 * (height + width);
    }
};

// Derived class for Square
class Square : public Shape {
private:
    double size;

public:
    Square(double size){
    this->size = size;
    }
    // Override getArea() and getPerimeter()
    double getArea() const override {
        return size * size;
    }

    double getPerimeter() const override {
        return 4 * size;
    }
};

// Main function
int main() {
    // Array of pointers to Shape
    Shape* shapes[3];

    // Create objects of Circle, Rectangle, and Square
    shapes[0] = new Circle(5);
    shapes[1] = new Rectangle(4, 6);
    shapes[2] = new Square(3);

    // Loop through the array and display area and perimeter
    for (int i = 0; i < 3; ++i) {
        if(i==0){
            cout<<"circle : "<<endl;
        }
        else if(i==1){
            cout<<"Rectangle : "<<endl;
        }
        else{
            cout<<"Square : "<<endl;
        }
        cout << "Area is : " << shapes[i]->getArea() << endl;
        cout << "Perimeter is : " << shapes[i]->getPerimeter() << endl;
        cout << endl;
    }

    // Clean up memory
    for (int i = 0; i < 3; ++i) {
        delete shapes[i];
    }

    return 0;
}
```