CPP PROGRAMS
REG.NO:24BBS0060

Q1.Write a C++ program to dynamically allocate an integer, a character and a string and assign a value to them.

Solution:

```
#include <iostream>  // Including the Input/Output Stream Library
#include <string>    // Including the String Library

int main() {
  // Dynamically allocate an integer
  int * dynamicInt = new int;  // Allocating memory for an integer and storing its address in dynamicInt
  * dynamicInt = 20;  // Assigning a value of 20 to the dynamically allocated integer

  // Dynamically allocate a character
  char * dynamicChar = new char;  // Allocating memory for a character and storing its address in dynamicChar
  * dynamicChar = 'C';  // Assigning the character 'C' to the dynamically allocated char

  // Dynamically allocate a string
  std::string * dynamicString = new std::string;  // Allocating memory for a string and storing its address in dynamicString
  * dynamicString = "C++ Dynamically allocated string.";  // Assigning a string value to the dynamically allocated string

  // Display the values
  std::cout << "Dynamically allocated integer: " << * dynamicInt << std::endl;  // Output the dynamically allocated integer value
  std::cout << "Dynamically allocated character: " << * dynamicChar << std::endl;  // Output the dynamically allocated character
  std::cout << "Dynamically allocated string: " << * dynamicString << std::endl;  // Output the dynamically allocated string

  // Deallocate the memory
```

```
  delete dynamicInt;  // Deallocating the memory occupied by the dynamically allocated integer
  delete dynamicChar;  // Deallocating the memory occupied by the dynamically allocated character
  delete dynamicString;  // Deallocating the memory occupied by the dynamically allocated string

  return 0;  // Returning 0 to indicate successful execution of the program
}
```

Q2.Write a C++ program to implement a class called Circle that has private member variables for radius. Include member functions to calculate the circle's area and circumference.

Solution:

```
#include <iostream> // Include necessary header for input/output stream

#include <cmath> // Include necessary header for mathematical functions

const double PI = 3.14159; // Define the value of PI as a constant

class Circle { // Define a class named Circle
  private:
    double radius; // Private member to store the radius

  public:
    // Constructor to initialize the Circle object with a radius
    Circle(double rad): radius(rad) {}

    // Member function to calculate the area of the circle
    double calculateArea() {
      return PI * pow(radius, 2); // Formula to calculate the area of a circle
    }

    // Member function to calculate the circumference of the circle
    double calculateCircumference() {
      return 2 * PI * radius; // Formula to calculate the circumference of a circle
```

```
    }
};

int main() {
  // Create a circle object
  double radius;
  std::cout << "Input the radius of the circle: ";
  std::cin >> radius; // Input the radius from the user
  Circle circle(radius); // Create a Circle object with the given radius

  // Calculate and display the area of the circle
  double area = circle.calculateArea(); // Calculate the area using the Circle
object
  std::cout << "Area: " << area << std::endl; // Output the calculated area

  // Calculate and display the circumference of the circle
  double circumference = circle.calculateCircumference(); // Calculate the
circumference using the Circle object
  std::cout << "Circumference: " << circumference << std::endl; // Output the
calculated circumference

  return 0; // Return 0 to indicate successful completion
}
```

q3.Write a C++ program to implement a class called Shape with virtual member functions for calculating area and perimeter. Derive classes such as Circle, Rectangle, and Triangle from the Shape class and override virtual functions accordingly.

Solution:

```
#include <iostream> // Include necessary header for input/output stream
#include <cmath> // Include necessary header for mathematical functions

const double PI = 3.14159; // Define constant value for PI

class Shape { // Define a base class named Shape
  public:
    // Virtual member function to calculate the area (pure virtual function)
    virtual double calculateArea() const = 0;

    // Virtual member function to calculate the perimeter (pure virtual function)
    virtual double calculatePerimeter() const = 0;
};

class Circle: public Shape { // Define a derived class named Circle inheriting from Shape
  private:
    double radius; // Private member variable to store the radius of the circle

  public:
    // Constructor for Circle class
    Circle(double rad): radius(rad) {}

    // Override the virtual member function to calculate the area
    double calculateArea() const override {
      return PI * pow(radius, 2); // Calculate the area of the circle using the radius
    }

    // Override the virtual member function to calculate the perimeter
    double calculatePerimeter() const override {
      return 2 * PI * radius; // Calculate the perimeter of the circle using the radius
    }
```

```
};

class Rectangle: public Shape { // Define a derived class named Rectangle inheriting from Shape
  private:
    double length; // Private member variable to store the length of the rectangle
    double width; // Private member variable to store the width of the rectangle

  public:
    // Constructor for Rectangle class
    Rectangle(double len, double wid): length(len), width(wid) {}

    // Override the virtual member function to calculate the area
    double calculateArea() const override {
      return length * width; // Calculate the area of the rectangle using its length and width
    }

    // Override the virtual member function to calculate the perimeter
    double calculatePerimeter() const override {
      return 2 * (length + width); // Calculate the perimeter of the rectangle using its length and width
    }
};

class Triangle: public Shape { // Define a derived class named Triangle inheriting from Shape
  private:
    double side1; // Private member variable to store the first side of the triangle
    double side2; // Private member variable to store the second side of the triangle
    double side3; // Private member variable to store the third side of the triangle

  public:
    // Constructor for Triangle class
    Triangle(double s1, double s2, double s3): side1(s1), side2(s2), side3(s3) {}

    // Override the virtual member function to calculate the area
```

```
    double calculateArea() const override {
      // Using Heron's formula to calculate the area of a triangle
      double s = (side1 + side2 + side3) / 2; // Calculate the semi-perimeter of
the triangle
      return sqrt(s * (s - side1) * (s - side2) * (s - side3)); // Calculate the area
using Heron's formula
    }

    // Override the virtual member function to calculate the perimeter
    double calculatePerimeter() const override {
      return side1 + side2 + side3; // Calculate the perimeter of the triangle
using its sides
    }
};

int main() {
  // Create instances of different shapes: Circle, Rectangle, and Triangle
  Circle circle(7.0); // Create a Circle object with radius 7.0
  Rectangle rectangle(4.2, 8.0); // Create a Rectangle object with length 4.2
and width 8.0
  Triangle triangle(4.0, 4.0, 3.2); // Create a Triangle object with sides 4.0, 4.0,
and 3.2

  // Calculate and display the area and perimeter of each shape
  std::cout << "Circle: " << std::endl;
  std::cout << "Area: " << circle.calculateArea() << std::endl; // Calculate and
output the area of the circle
  std::cout << "Perimeter: " << circle.calculatePerimeter() << std::endl; //
Calculate and output the perimeter of the circle

  std::cout << "\nRectangle: " << std::endl;
  std::cout << "Area: " << rectangle.calculateArea() << std::endl; // Calculate
and output the area of the rectangle
  std::cout << "Perimeter: " << rectangle.calculatePerimeter() << std::endl; //
Calculate and output the perimeter of the rectangle

  std::cout << "\nTriangle: " << std::endl;
  std::cout << "Area: " << triangle.calculateArea() << std::endl; // Calculate and
output the area of the triangle
```

```
  std::cout << "Perimeter: " << triangle.calculatePerimeter() << std::endl; // Calculate and output the perimeter of the triangle

  return 0; // Return 0 to indicate successful completion
}
```