User Id : weizhe.goh@digipen.edu | Started : 2020.08.06 16:31:37 | Score : 100%

DigiPen INSTITUTE OF TECHNOLOGY

Polymorphism Assignment

Rules

Read carefully and check all rules you agree with:

- ☐ Your code must represent your own individual work. If something is not clear, ask your instructor for help.
- ☐ Each exercise has description which must be strictly followed.
- ☐ All programs must pass all tests in the main function (when given) to get the final grade. **You are not allowed to make any change in the main function in this case.**
- ☐ Keep the code proper formatted (correct indentation, max line width is 40 characters).

Specs

- ☐ This assignment is all about polymorphism. Keep it in mind when implementing following specs to produce the correct code and output.
- ☐ Implement a base class named Shape with a member function draw(). The function must output "Draw Shape" with the newline character at the end.
- ☐ Implement class Circle derived from Shape with a member function draw(). The function must call the base class draw() and then output "Draw Circle" with the newline character at the end.
- ☐ Implement class Polygon derived from Shape with a member function draw(). The function must call the base class draw() and then output "Draw Polygon" with the newline character at the end.
- ☐ Implement class Triangle derived from Polygon with a member function draw(). The function must call the base class draw() and then output "Draw Triangle" with the newline character at the end.
- ☐ All classes must have default constructors and destructors with output for testing purpose, for example, "Constructor Circle" with the newline character at the end.
- ☐ Make the base class destructor **virtual** to produce the correct output. (Will be explained next class)
- ☐ Test your classes in the given main function to make sure that all classes produce the correct output. Correct output is (without ") is:

```
"Constructor Shape
Constructor Circle
Constructor Shape
Constructor Polygon
Constructor Triangle
Draw Shape
Draw Circle
Draw Shape
Draw Polygon
Draw Triangle
Destructor Circle
Destructor Shape
Destructor Triangle
Destructor Polygon
Destructor Shape
"
```

★ Code 1 ☑

Run

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

class Shape
{
public:
    Shape()
    {
        cout << "Constructor Shape"
        << endl;
    }

    virtual void draw()
    {
        cout << "Draw Shape" << endl;
    }

    virtual ~Shape()
    {
        cout << "Destructor Shape"
        << endl;
    }
};

class Circle : public Shape
{
public:
    Circle()
    {
        cout << "Constructor Circle"
        << endl;
    }

    void draw()
    {
        Shape::draw();
        cout << "Draw Circle" << endl;
    }

    ~Circle()
    {
        cout << "Destructor Circle"
        << endl;
    }
};

class Polygon : public Shape
{
public:
    Polygon()
    {
        cout << "Constructor Polygon"
        << endl;
    }

    void draw()
    {
        Shape::draw();
        cout << "Draw Polygon" << endl;
    }

    ~Polygon()
    {
        cout << "Destructor Polygon"
        << endl;
    }
};

class Triangle : public Polygon
{
public:
    Triangle()
    {
        cout << "Constructor Triangle"
        << endl;
    }

    void draw()
    {
        Polygon::draw();
        cout << "Draw Triangle" << endl;
    }

    ~Triangle()
    {
        cout << "Destructor Triangle"
        << endl;
    }
};
```

```
int main() {
  Shape * shapes[] = {new Circle(),
                      new Triangle()};
  shapes[0]->draw();
  shapes[1]->draw();
  delete shapes[0];
  delete shapes[1];
  return 0;
}
```

```
Constructor Shape
Constructor Circle
Constructor Shape
Constructor Polygon
Constructor Triangle
Draw Shape
Draw Circle
Draw Shape
Draw Polygon
Draw Triangle
Destructor Circle
Destructor Shape
Destructor Triangle
Destructor Polygon
Destructor Shape
```

By signing this document you fully agree that all information provided therein is complete and true in all respects.	Responder sign: