

Programming Two

Tutorial 5: Polymorphism and Virtual Functions

Compiling the Example Code

Compile the example code online, there is quite a bit, and make sure you understand what is going on.

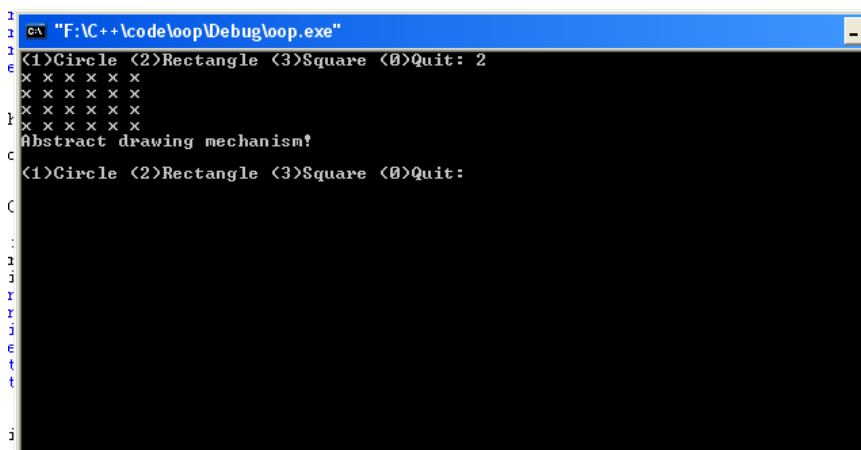
Exercise

Consider the following class hierarchy: *CShape* is the common base class. It has a constructor, with no arguments, and is an abstract class, containing a pure virtual function called *Draw()*. Consider the derived classes *CTriangle*, *CCircle* and *CRectangle*, all of them implementing their own *Draw* method (mandatory). Write down the corresponding files.

To advance this, use the following as your base class:

```
class Shape
{
public:
    Shape(){}
    virtual ~Shape(){}
    virtual long GetArea() = 0;
    virtual long GetPerim()= 0;
    virtual void Draw() = 0;
private:
};
```

Define the functions and main so that a similar output to this is produced



```

1  c:\ "F:\C++\code\loop\Debug\loop.exe"
2
3  (1)Circle (2)Rectangle (3)Square (0)Quit: 2
4  x x x x x
5  x x x x x
6  x x x x x
7  x x x x x
8  x x x x x
9  Abstract drawing mechanism!
10 (1)Circle (2)Rectangle (3)Square (0)Quit:
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

In main you will have to use the keyword **new** to define a new instance based on the user's selection and have a pointer to the base class which then points to the new instance of a derived class. Using the pointer then call the derived classes *Draw()* function.

For the circle function you don't actually have to define a drawing routine, just get it to print a relevant message to the screen.