HAM JING YI A19EC0048

PROGRAM 9.3

4a) output:

|  |
| --- |
| Area = 0  Area = 0  Area = 0 |

b)

Suppose that there are 3 different answers for the 3 area. The first one is area of the shape =0. Is correct. However, the second area and the third area should be area from class Circle and Rectangle. But three of the area shown are all from the parent class Shape. This happened is because there is no “virtual” word in front of the int getArea() const in base class, it is static binding. It is redefined but not overridden. The code which the method to be bound with is decided by the compiler at compile time and it is unchanged throughout the execution of the program. During the compilation, the compiler already know that the p at line 33 is actually pointer for the class Shape. The object p will then refer all the methods in the class Shape. So, the p->getArea() in line 40, 43 and 46 here will refer the method (int getArea() ) in the class Shape since it is static binding. The method getArea associated with p is pre-bound at compile time and remain unchanged at runtime. So, every time calling to the method area (line 40, 43 and 46) will always call method getArea of class Shape. Since the method area in class Shape will always return 0, so such result will be printed.

c) *code is also uploaded.*

I will insert “virtual” in front of the int getArea () const { return 0;} in the base class (class Shape). This is because overridden method is used and dynamic binding is applied. The C++ will determine the type of object making the call and binds the methods to the appropriate version of the function. The code which the method to be bounded with is decided by the program at run time. The getArea in line 40 will be referred to the getArea in the class Shape. The getArea method in line 43 will be referred to the getArea in class Circle. The getArea method in line 46 will be referred to the getArea in class Rectangle. So, we will get different area when the area is displayed.