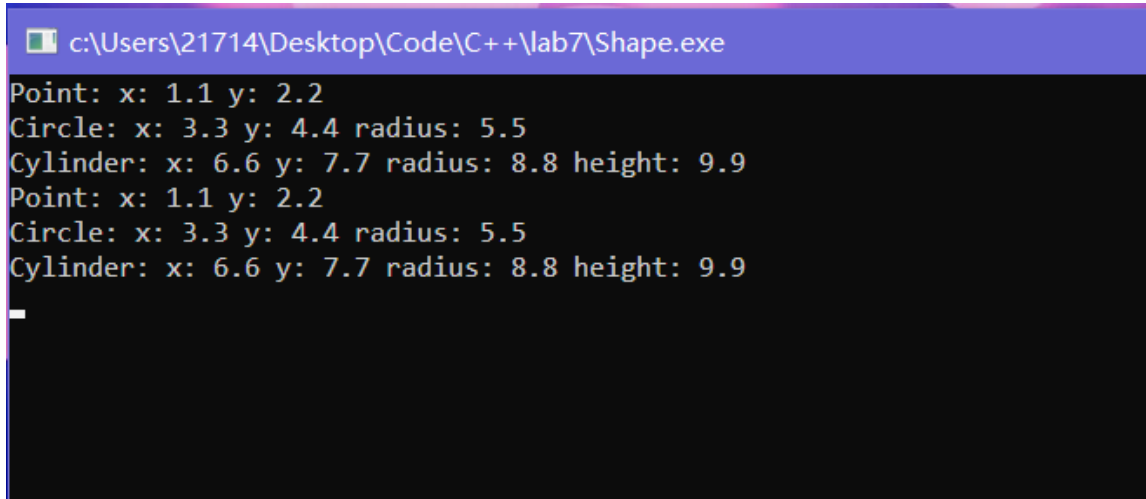


# Lab7

## 一、运行结果：



```
c:\Users\21714\Desktop\Code\C++\lab7\Shape.exe
Point: x: 1.1 y: 2.2
Circle: x: 3.3 y: 4.4 radius: 5.5
Cylinder: x: 6.6 y: 7.7 radius: 8.8 height: 9.9
Point: x: 1.1 y: 2.2
Circle: x: 3.3 y: 4.4 radius: 5.5
Cylinder: x: 6.6 y: 7.7 radius: 8.8 height: 9.9
_
```

## 二、代码思路：

(1) 首先声明一个 Shape 类，来表示图形的一些基础性质，如名称，面积，体积。由于 shapeName 是纯虚函数，故 Shape 类是一个抽象类，无法定义对象。(2) 由 Shape 类作为抽象基类，经过公有继承后派生出 Point 类，表示一个点的基本性质。Point 相对于 Shape 类多了 x 坐标和 y 坐标这两个 float 型数据成员，以及与其相关的各种函数。同时对虚函数 shapeName 进行了重新定义，使其与 Point 类相契合。同时还通过友元函数为 Point 类的对象重载了 << 运算符；(3) 由 Point 类再通过共有继承派生出 Circle 类，表示一个圆的基本性质。Circle 类又添加了 float 型数据成员 radius，以及与其相关的各种函数，同时对 shapeName 又进行了重新定义，使其与 Circle 类相契合。同样，也为 Circle 类通过友元函数重载了 << 运算符。(4) 由 Circle 类再通过公有继承派生出 Cylinder 类，表示一个圆柱的基本性质。Cylinder 相对于 Circle 类又多了 float 型数据成员 height，以及与其相关的各种函数，同时也对 shapeName 进行了重新定义，使其与 Cylinder 类相契合。同样

也重载了<<运算符用于 Cylinder 类型成员的基本性质的输出。

在 main()函数中, 首先分别构造一个 Point, 一个 Circle, 一个 Cylinder 对象, 然后再先通过静态关联的方式直接用对象调用 shapeName()函数, 并用重载的<<运算符输出对象的其他信息。之后再通过使用基态指针进行动态关联的方式分别让 pt 指向不同的对象来调用不同的 shapeName 函数, 并利用各个对象的成员函数获取信息并输出。

### 三、回答问题:

(1) 静态关联是指在编译时就能够确定某处调用的函数属于哪个类以及函数的具体内容是什么。而动态关联是在运行的过程中根据当时的情况来确定某处调用的函数属于哪个类以及函数的具体内容是什么。以本 lab 为例, 前三行的 shapeName()函数的调用就属于静态关联, 因为是使用各个对象直接进行调用, 故在编译时就能确定调用的函数具体为哪里定义的函数。而后三行的 shapeName()函数的调用就属于动态关联, 因为是使用基态指针 pt 进行调用, pt 可以指向不同类的对象, 故在编译时无法确定 pt 调用的 shapeName()函数究竟属于哪个类, 只有在运行过程中确定 pt 具体指向的对象的类型时才能够确定其所调用的 shapeName()函数。

(2) 纯虚函数是一种特殊的虚函数, 纯虚函数在基类中没有定义, 含有纯虚函数的类称为抽象类, 无法建立抽象类的对象, 故纯虚函数不能被调用, 只有在派生类中对该函数进行定义之后, 才可以在派生类中使用该函数, 如果在派生类中也未对该函数进行定义, 则该函数在派生类中仍为纯虚函数, 不能被调用。而虚函数在基类中有定义, 如果该基类不为抽象类, 则可以直接通过该基类的对象调用该虚函数, 同时也可以在派生类中重新对该虚函数进行定义, 并在派生类中使用新定义的虚函数。以本 lab 为例, shapeName()属于纯虚函数, 故在基类 Shape 中无定义, Shape 类属于抽象类, 无法在 Shape 中调用该函数, 只有在 Point 以及之后的派生类中提供了 shapeName()的定义之后才能调用该函数。

(3) 对于上述设计, 我认为 Shape 类中的 area()和 volume()完全可以定义为纯虚函数, 因为 Shape 类没有数据成员, 显然是无法给出 area()以及 volume()这两个函数的定义的。同时在 Point 类中可以考虑加入 area()和 volume(), 在 Circle 类中可以考虑加入 volume()作为 Shape 类 area()和 volume()的定义 (如果在 Shape 中把 area()和 volume()定义为纯虚函数的话), 并让这些加入的函数都直接返回 0.0, 这样也是符合逻辑的。然后除了以上我感觉不妥的地方之外, 其他地方设计的都很合理, 可以说是演示静态关联和动态关联的一个很好的案例了。