HW10 - 1 Document

R07521603 蔡松霖

People* p1 = new People();

可想而知，一定是執行 People talking，以及 People thinking。其餘的 cases，我起初的猜測是依照 new 後面來決定執行的函式，因此分別是執行後面 class 的函式。

但結果與我原先的猜測有些出入：

```
People* p1 = new People();
People* p2 = new SmartPeople();
SmartPeople* p3 = new SmartPeople();
People* p4 = new VerySmartPeople();
SmartPeople* p5 = new VerySmartPeople();
```

output：

```
People talking.
People thinking.
SmartPeople talking.
People thinking.
SmartPeople talking.
SmartPeople thinking.
VerySmartPeople talking.
People thinking.
VerySmartPeople talking.
VerySmartPeople thinking.
```

```
class People{

public:
    // talk() can be overrided starting in People
    virtual void talk(){
        std::cout<<"People talking.\n";
    }
    void think(){
        std::cout<<"People thinking.\n";
    }
};

class SmartPeople : public People{
public:
    void talk(){
        std::cout<<"SmartPeople talking.\n";
    }
    // think() can be overrided starting in SmartPeople
    virtual void think(){
        std::cout<<"SmartPeople thinking.\n";
    }
};


class VerySmartPeople : public SmartPeople{
public:
    void talk(){
        std::cout<<"VerySmartPeople talking.\n";
    }
    void think(){
        std::cout<<"VerySmartPeople thinking.\n";
    }
};
```

了解到 derived class 要能夠 override 掉 base class 的函式，需要在其之前(或是更之前)的 base class 中的該函式前加上 virtual 的字樣。因此以 p2 為例，因為 People 的 think()前面沒有 virtual，所以並不行執行 SmartPeople 的 override 函式。因為 pointer p2 指向的是 People，所以會執行 People 的 think()，而因為 talk()可以 override，結果會是執行 SmartPeople 中的 talk()。