继承

独立的类

animal-1.cpp

基类和派生类

重用：name、my_name()
改造：say_hello()
扩充：birth、my_birth、color、my_color()

基类：animal
派生类：dog、cat

animal-2.cpp

查看对象的内存空间分配情况：

```
(gdb) p a
(gdb) p sizeof(a)
(gdb) p &a
(gdb) p &a.name
...
(gdb) p a.my_name
(gdb) p b.my_name
(gdb) p c.my_name
...
(gdb) p a.say_hello
(gdb) p b.say_hello
(gdb) p c.say_hello
...
```

访问被覆盖的成员变量、成员函数：

```
b.say_hello();
b.dog::say_hello();
b.animal::say_hello();

c.say_hello();
c.cat::say_hello();
c.animal::say_hello();
```

构造函数、析构函数的调用顺序

animal-3.cpp

访问权限控制

能够继承，不代表能够访问。

以基类中的成员变量name为例，有3个位置可能对它进行访问。

```
char name[8];
```

1. 基类内部

```
void say_hello()
{
    cout << "Hello" << name << endl;
}
```

2. 基类外部、派生类内部

```
void say_hello()
{
        cout << "Wang Wang ... " << name << endl;
}
```

3. 基类外部、派生类外部

```
dog b("WangCai", 2020);
cout << b.name;
```

是否能访问成功，取决于两点：

1. 继承方式（public、protected、private）

```
class dog : public animal { ... }
```

```
class dog : protected animal { ... }
```

```
class dog : private animal { ... }
```

2. 基类成员（public、protected、private）

```
public:
char name[8];
```

```
protected:
char name[8];
```

```
private:
char name[8];
```

$$3 \times 3 = 9$$

继承方式	public成员	protected成员	private成员
public继承	public成员	protected成员	不可访问成员
protected继承	protected成员	protected成员	不可访问成员
private继承	private成员	private成员	不可访问成员

多继承

一个派生类可以有多个基类。

animal-4.cpp

```
class shenshou : public dog, public cat
{
    ...
};
```

```
shenshou s("ShenShou", 2022, "GOLD");
s.my_birth();
s.my_color();
```

以下代码是否正确?

```
cout << s.name << endl;
s.my_name();
```

改成以下代码：

```
cout << s.dog::name << endl;
s.dog::my_name();

cout << s.cat::name << endl;
s.cat::my_name();
```

查看对象的内存空间分配情况（name长度改为100）：

```
(gdb) p s
(gdb) p sizeof(s)
```

animal > dog

animal > cat

dog、cat > shenshou

如何更加优美的解决问题?

虚基类

animal-5.cpp

```
class dog : public virtual animal
class cat : public virtual animal
class shenshou : public dog, public cat
{
    ...
};
```

以下代码是否正确?

```
shenshou s("ShenShou", 2022, "GOLD");
cout << s.name << endl;
s.my_name();
```

以下代码是否正确?

```
s.say_hello();
```

改成以下代码：

```
s.animal::say_hello();
s.dog::say_hello();
s.cat::say_hello();
```

查看对象的内存空间分配情况（name长度改为100）：

```
(gdb) p s
(gdb) p sizeof(s)
```