虚函数是用于实现类中多态的，类中多态是指，一个派生类对象通常具有两部分，一部分是基类，一部分是派生类对象自己，你可以用指针指向派生类，也可以用指针指向基类部分，当然他们是不等价的，但是可以互相转换的，并且有专门的c++风格的转换函数用于多态的转换。

虚函数具体是通过用虚表和虚表指针来实现的。每个类只有一个虚表，但是每个类的实例都有各自的虚表指针。

**在创建 Derived 对象时，先调用 Base 构造函数，此时对象虚表指针指向 Base 类虚表；再调用 Derived 构造函数，虚表指针更新为指向 Derived 类虚表。在销毁对象时，先调用 Derived 析构函数，虚表指针仍指向 Derived 类虚表；再调用 Base 析构函数，虚表指针更新为指向 Base 类虚表。**

**虚表指针指向虚表，在每个具有虚函数的对象实例中，虚表包含了虚函数的地址，每次调用虚函数，都是先去找虚表指针，然后找到虚表，再去找到对应的虚函数。**

**虚函数又分为三种，纯虚函数，包含纯虚函数的类成为抽象类，没有实例。纯虚函数一般不需要实现，但是纯虚析构函数必须提供具体实现，通常用于释放资源。纯虚函数必须被重写。**

**虚函数，虚函数可以提供默认实现，方便派生类调用，但是通常需要显示调用。在类外提供具体实现可以避免污染命名空间。**

**非虚函数，非虚函数不允许重写。**

**这三种虚函数方便代表了c++中声明，声明定义，定义的思想。**

**虚函数通常不是性能的瓶颈，但是会有三种开销，**

**时间上，运行时候会查找虚表，**

**空间上，有虚表和虚表指针**

第三个代价：虚函数是没有内联的。

从效率的角度来看，不太可能比编译器生成的实现更好地自己编写这些特性。