abstract class来定义抽象类。抽象类除了不能实例化对象之外（只能继承），类的其它功能依然存在，成员变量、成员方法和构造方法的访问方式和普通类一样。

```
public abstract class Employee
```

继承抽象类

通过extends继承。抽象类和普通类的三点区别：

1）抽象方法必须为public或者protected（因为如果为private，则不能被子类继承，子类便无法实现该方法），缺省情况下默认为public。

2）抽象类不能用来创建对象；

3）抽象类的子类必须实现父类的抽象方法。如果没有实现，则必须将子类也定义为为abstract类。

抽象方法

抽象方法只包含一个方法名，而没有方法体。

```
   public abstract double computePay();
```

声明抽象方法：

- 如果一个类包含抽象方法，那么该类必须是抽象类。
- 任何子类必须重写父类的抽象方法，或者声明自身为抽象类。

抽象类和接口的对比

参数	抽象类	接口

默认的方法实现	它可以有默认的方法实现	接口完全是抽象的。它根本不存在方法的实现
实现	**extends**继承。如果子类不是抽象类的话，它需要提供抽象类中所有声明的方法的实现。	**implements**实现接口。它需要提供接口中所有声明的方法的实现
构造器	抽象类可以有构造器	接口不能有构造器
与正常Java类的区别	除了你不能实例化抽象类之外，它和普通Java类没有任何区别	接口是完全不同的类型
访问修饰符	抽象方法可以有**public**、**protected**和**default**这些修饰符	接口方法默认修饰符是**public**。
main方法	抽象方法可以有main方法并且我们可以运行它	接口没有main方法，因此不能运行它。
多继承	可以继承一个类和实现多个接口	只能继承一个或多个其它接口
速度	快	慢，需要时间去寻找在类中实现的方法。
添加新方法	可以给它提供默认的实现。不需要改变现有代码。	必须改变实现该接口的类。

语法层面上的区别

1）抽象类可以提供成员方法的实现细节，而接口中只能存在public abstract 方法；

2）抽象类中的成员变量可以是各种类型的，而接口中的成员变量只能是public static final类型的；

3）接口中不能含有静态代码块以及静态方法，而抽象类可以有静态代码块和静态方法；

4）一个类只能继承一个抽象类，而一个类却可以实现多个接口。