内部类，简单点来说就是在类的里面定义的类。

在Java中主要就分为四种：

成员内部类，

静态内部类，

局部内部类，

匿名内部类（重点）。

成员内部类

```
public class Outer {

   //成员内部类
   class Edge{
      public void sys(){
         System.out.println("我是成员内部类的方法...");
      }
   }

   //使用成员内部类
   public void sysEdge(){
      Edge edge = new Edge();
      edge.sys();
   }

   public static void main(String[] args) {
      new Outer().sysEdge();
   }
}
```

如代码所见就是一个标准的成员内部类，在代码中可以看见我们又在外部类中定义了一个普通的方法用来调用该内部类，为什么要这样做呢？

因为成员内部类只能在所处外部类中使用，并且不能在static修饰的方法中使用。

静态内部类

```
public class Outer {
```

```
    //静态内部类
    static class Edge{
        public void sys(){
            System.out.println("我是静态内部类的方法...");
        }
    }

    //使用静态内部类
    public void sysEdge(){
        Edge edge = new Edge();
        edge.sys();
    }

    public static void main(String[] args) {
        new Outer().sysEdge();

        //使用静态内部类
        new Edge().sys();
    }
}
```

简单的说静态内部类就是在成员内部类前面加了个static修饰符，不过它的作用域就广泛了，如代码中在静态方法中也能使用，同样的，在其他类中也能使用。

局部内部类

```
public class Outer {


    public void sysEdge(){

        //局部内部类
        class Edge{
            public void sys(){
                System.out.println("我是局部内部类的方法...");
            }
        }

        //使用局部内部类
        new Edge().sys();
    }

    public static void main(String[] args) {
        new Outer().sysEdge();
    }
}
```

该内部类可以说是作用域最小的了，它定义在一个方法内部，只为该方法服务。

匿名内部类（重点）

首先这是一个接口

```
public interface Dog {

    void sayDog();
}
```

这是我们的类

```
public class Outer {


    public void sysEdge(){
        //使用匿名内部类
        new Dog(){

            @Override
            public void sayDog() {
                System.out.println("我是一个匿名内部类...");
            }
        }.sayDog();
    }

    public static void main(String[] args) {
        new Outer().sysEdge();
    }
}
```

可能这样看不是很直观，容我一步步解释。

如上代码我们最后是能调用sayDog方法并输出一句话的，Dog是一个接口，它本身定义的方法是不具备作用的，它本身也不能被当成类，我们都是通过实现类去实现接口以及接口方法的，这里是不是可以联想一下，有一个类实现了Dog接口并实现了sayDog方法呢，但是我们并没有看到这个类，但是实现了接口就肯定有一个类，所以这个类是匿名的。而且这个类是在外部类的里面实现的，所以又是内部类。这样理解匿名内部类是不是好理解点呢。当然，这里既然接口可以，抽象类也是可以的。