**泛型的基本概念：**

1. 泛型（Generics）是指在定义类的时候不指定类中属性的类型，而是在声明和实例

化对象时指定类型。但不能指定为基本数据类型。

1. 泛型设置的目的是为了解决对象强制向下转型所带来的的安全隐患。
2. 泛型的主要原理是在定义类时通过标识符来表示类中属性的类型或者某个

方法的返回值及参数类型。

**泛型类与泛型对象的定义格式：**

1. 泛型类的定义：

[访问权限] class 类名称<泛型类型标识1,泛型类型标识2…>{

[访问权限] 泛型类型标识 属性名;

[访问权限] 泛型类型标识 方法名( ){ };

[访问权限] 返回值类型 方法名( ){ };

}

1. 泛型对象的定义：

类名称<具体类型1> 对象名称=new 类名称<具体类型>( );

**泛型的安全警告：**

在泛型应用中，声明泛型类对象时最好指定好泛型类型，如果不指定泛型类型而给对象 的属性赋值，则在编译时会出现安全警告，但并不影响程序的运行。因为不指定类型的 话，泛型会被擦除，会使用Object类表示。

**泛型通配符：**

在涉及泛型对象的引用传递时，泛型类型必须匹配才可以传递。虽然可以使用没有指定

泛型类型的对象进行接收，但不妥，为解决这样的问题，Java引入了通配符“？”，表 示可以接收任意泛型类型的对象。但如果使用了“？”接收泛型对象，则不能改变该泛 型对象的属性内容，只能将其属性内容设置成null值。

**受限泛型：**

在泛型操作中，可以设置一个泛型类或泛型对象的泛型类型的范围上限和范围下限。

1. 泛型上限使用extends关键字声明，表示泛型的类型可能是所指定的类型或者是它

的子类。格式如下：

声明对象：类名称<? extends 类> 对象名称

声明类：[访问权限] 类名称<泛型标识 extends 类>{ }

1. 泛型下限使用super关键字声明，表示泛型的类型可能是所指定的类型，或者是此

类型的父类，或是Object类。格式如下：

声明对象：类名称<? super 类> 对象名称

声明类：[访问权限] 类名称<泛型标识 super 类>{ }

**泛型对象间的限制：**

继承关系中，子类可以通过对象的多态性为父类实例化，但在泛型操作中，子类的泛型 类型无法使用父类的泛型类型接收，即子类泛型类型对象不能为父类泛型类型对象实例 化。只能使用通配符“?”接收。因为子类泛型变为父类泛型，则表示扩大了子类的内 容。

即不同泛型类型的同一泛型类的泛型对象不能相互赋值。

在两个泛型对象是同一泛型类的前提下，虽然一个泛型对象的泛型类型是另个泛型对象

的泛型类型的子类的话，但两者不能相互赋值。

**泛型接口的定义：**

声明泛型接口和声明泛型类的语法相似，格式如下：

[访问权限] interface 接口名称<泛型标识>{ }

**实现泛型接口的实现：**

定义泛型接口后，就要定义此接口的子类，定义泛型接口的子类有两种方式。一是直接

在子类之后声明泛型；另一种是直接在子类实现的接口中明确地给出泛型类型。

泛型方法：

泛型方法

泛型方法：

1. 泛型方法的定义与其所在的类是否是泛型类没有任何关系，所在的类可以是泛型类，

也可以不是泛型类。在泛型方法中可以定义泛型参数，此时参数的类型就是传入数

据的类型。定义格式如下：

[访问方法] <泛型标识> 返回值 方法名称（[泛型标识 参数名称]）

1. 如果要通过泛型方法返回一个泛型类的实例化对象，则必须在方法的返回类型声明

处明确地指定泛型标识。

1. 多个泛型对象作为同一方法的参数时，他们所指定的泛型类型必须一致。如果说一

方法与要求传入的泛型对象的泛型类型是一致的，也可以通过泛型方法指定。

泛型数组：

使用泛型方法时，也可以向方法中传递数组或返回一个数组。

泛型的嵌套设置：

在一个泛型类中可以将另一个泛型类作为本泛型类属性的泛型。