java泛型（Generic）

一、什么叫泛型

泛型：

字面意思：广泛的类型，各种各样的类型。

专业定义：参数化的类型。

参数：方法的参数仅仅是一个声明，可以接收指定类型的任意数据。

参数化的类型：将类型变成参数，可以传不同的类型。

二、java中的泛型

2.1 泛型的概念：

泛型是jdk1.5的新特性。

> 泛型，即“参数化类型”。一提到参数，最熟悉的就是定义方法时有形参，然后调用此方法时传递实参。那么参数化类型怎么理解呢？顾名思义，就是将类型由原来的具体的类型参数化，类似于方法中的变量参数，此时类型也定义成参数形式（可以称之为类型形参），然后在使用/调用时传入具体的类型（类型实参）。

泛型的本质是为了参数化类型（在不创建新的类型的情况下，通过泛型指定的不同类型来控制形参具体限制的类型）。也就是说在泛型使用过程中，操作的数据类型被指定为一个参数。

泛型可以使用在类、接口和方法中，分别被称为泛型类、泛型接口、泛型方法。

2.2 泛型的作用

1、 类型安全。

泛型的主要目标是提高 Java 程序的类型安全。通过知道使用泛型定义的变量的类型限制，编译器可以在一个高得多的程度上验证类型假设。

2、 消除强制类型转换。

泛型的一个附带好处是，消除源代码中的许多强制类型转换。这使得代码更加可读，并且减少了出错机会。

3、 潜在的性能收益。

泛型为较大的优化带来可能。在泛型的初始实现中，编译器将强制类型转换（没有泛型的话，程序员会指定这些强制类型转换）插入生成的字节码中。

三、泛型类

语法：

```
class 类名称 <泛型标识：可以随便写任意标识号，标识指定的泛型的类型>{
  private 泛型标识 /*（成员变量类型）*/ var;
  .....

  }
}
```

实例：

```
//此处T可以随便写为任意标识，常见的如T、E、K、V等形式的参数常用于表示泛型
//在实例化泛型类时，必须指定T的具体类型
public class Generic<T>{
    //key这个成员变量的类型为T,T的类型由外部指定
    private T key;

    public Generic(T key) { //泛型构造方法形参key的类型也为T，T的类型由外部指定
        this.key = key;
    }

    public T getKey(){ //泛型方法getKey的返回值类型为T，T的类型由外部指定
        return key;
    }
}
```