复习：

1. Queue：队列。先进先出。

2. Set：元素存储无序不可重复。

HashSet---底层是基于HashMap

TreeSet---底层会对元素进行自然排序，要求元素对应的类实现Comparable

3. Comparator：比较器。指定比较规则，针对某一个对象使用

4. Iterator：迭代器。---本质上是通过指针的挪动来依次遍历每一个元素---底层会通过一个布尔值来标记每一个元素的状态，所以在迭代过程中不允许增删原集合

5. Collections：操作集合的工具类，本身不能创建对象，提供了大量的静态方法

6. 泛型：参数化类型---泛型的擦除发生在编译期。了解泛型的继承。

7. Map：映射。存储的元素是键值对。每一个键对应了一个值，键是唯一的。

HashMap---底层是以数组+链表的结构存储数据。在存储数据的时候是根据元素的哈希码来实现的。底层的数组的默认容量是16，数组中的每一个位置称之为是一个桶。默认加载因子是0.75f，每次扩容默认增加一倍。---每发生一次扩容都会导致所有的元素需要重新计算分布---rehash---允许键或者值为null，是一个异步式线程不安全的映射

Hashtable—默认初始容量是11，加载因子是0.75f。不允许键或者值为null，是一个同步式线程安全的映射。

## File

表示文件或者目录(文件夹)的类。

练习：删除目录

方法---先获取这个目录下的子目录和子文件(listFiles())，需要进行判断。如果是子文件(isFile)就直接删除，如果是子目录(isDirectory)---递归

练习：统计工作空间中Java文件和class文件的个数

方法---获取工作空间中所有的子文件和子目录，子目录---递归；如果是子文件，判断是否是一个Java文件还是一个class文件

### 路径

绝对路径：以盘符或者/开头的路径---直接跳转到指定的地址

相对路径：不以盘符或者/开头的路径---需要以当前的路径为基准进行计算 .. 表示上一次目录

目录分隔符-separatorChar/separator---在Windows自动变为\\,在Linux下是/

路径分隔符-pathSeparatorChar/pathSeparator在Windows中用 ; 分隔不同的路径，在Linux下是用 :

## IO

用于传输数据的。IO --- Input Output。数据从外部流向程序---输入流，数据从程序流向外部---输出流。 读取文件---数据从文件流向程序---输入流，向文件中写入“abc” --- 数据从程序流向了文件---输出流。

根据数据的传输方向：输入流和输出流

根据数据的传输形式：字节流和字符流

|  |  |  |
| --- | --- | --- |
|  | 输入流 | 输出流 |
| 字节流 | InputStream | OuputStream |
| 字符流 | Reader | Writer |

四个基本流都是抽象类。

数据的来源/目的地：硬盘、内存、输入设备、网络

向一个TXT文件中写入一个字符串--- 输出流、字符流、和文件有关的流 --- FileWriter

### 流的异常处理

1. 将流对象放在try之外声明并赋值为null，放在try之内实例化

2. 在关流之前需要判断流对象是否为null

3. 因为关流可能会产生失败，所以需要将流对象置为null，以防止关流失败导致流对象一直占用文件

4. 在关流之前需要手动冲刷缓冲区，为了防止关流失败导致一部分数据死在缓冲区中

练习：复制文件