* **RandomAccessFile 类**
* RandomAccessFile类支持“随机访问”的方式，程序可以直接跳到文件的任意地方来读、写文件。
* 既可以充当一个输入流，又可以充当输出流。
* 支持只**访问文件的部分内容**。
* 可以**向已存在的文件后追加内容**。
* RandomAccessFile对象包含一个记录指针，用以标示当前读写处的位置。RandomAccessFile类对象可以自由移动记录指针：
  + **long getFilePointer()：获取文件记录指针的当前位置。**
  + **void seek(long pos)：将文件记录指针定位到pos位置。**
* 构造器
* public RandomAccessFile(File file,String mode)
* public RandomAccessFile(String name,String mode)
* **创建RandomAccessFile类实例需要指定一个mode参数，该参数指定RandomAccessFile的访问模式：**
* **r:以只读方式打开。**
* **rw：打开以便读取和写入。**
* **rwd:打开以便读取和写入；同步文件内容的更新。**
* **rws:打开以便读取和写入；同步文件内容和元数据的更新。**
* 读取文件内容

/\*\*  
 \* RandomAccessFile：支持随机访问  
 \* 1）既可以充当一个输入流，又可以充当一个输出流。  
 \* 2）支持从文件的开头读取、写入。  
 \* 3）支持从任意位置的读取、写入（插入）。  
 \*/  
@Test  
public void test1() {  
  
 RandomAccessFile raf1 = null;  
 RandomAccessFile raf2 = null;  
 try {  
 raf1 = new RandomAccessFile("wcy.txt", "r");  
 raf2 = new RandomAccessFile("wcy1.txt", "rw");  
  
 byte[] b = new byte[20];  
 int len;  
 while ((len = raf1.read(b)) != -1) {  
 raf2.write(b, 0, len);  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 } finally {  
 if (raf2 != null) {  
 try {  
 raf2.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 if (raf1 != null) {  
 try {  
 raf1.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}  
  
/\*\*  
 \* 实现的实际上时覆盖的效果  
 \*/  
@Test  
public void test2() {  
  
 RandomAccessFile raf = null;  
 try {  
 raf = new RandomAccessFile("wcy1.txt", "rw");  
  
 raf.seek(4);  
 raf.write("wcy".getBytes());  
 } catch (IOException e) {  
 e.printStackTrace();  
 } finally {  
 if (raf != null) {  
 try {  
 raf.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}  
  
@Test  
public void test3() {  
  
 RandomAccessFile raf = null;  
 try {  
 raf = new RandomAccessFile("wcy1.txt", "rw");  
  
 raf.seek(4);  
 //efg123456，指针会移到别的位置。  
 String str = raf.readLine();  
  
 raf.seek(4);  
 raf.write("wcy".getBytes());  
 raf.write(str.getBytes());  
  
 } catch (IOException e) {  
 e.printStackTrace();  
 } finally {  
 if (raf != null) {  
 try {  
 raf.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}