* **处理流之四：打印流**
* 在整个IO包中，打印流是输出信息最方便的类。
  + **PrintStream**(字节打印流)和**PrintWriter**(字符打印流)。
  + 提供了一系列重载的print和println方法，用于多种数据类型的输出。
  + PrintStream和PrintWriter的输出不会抛出异常。
  + PrintStream和PrintWriter有自动flush功能。
  + System.out返回的是PrintStream的实例。

@Test  
public void test1() {  
  
 FileOutputStream fos = null;  
  
 try {  
 fos = new FileOutputStream("print.txt");  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 }  
 //创建打印输出流，设置为自动刷新模式

//（写入换行符或字节'\n'时都会刷新出缓冲区）  
 PrintStream ps = new PrintStream(fos, true);  
  
 //把标准输出流（控制台输出）改成文件  
 if (ps != null) {  
 System.setOut(ps);  
 }  
  
 //输出ASCII字符  
 for (int i = 0; i <= 225; i++) {  
 System.***out***.println((char) i);  
  
 //每50个数据一行  
 if (i % 50 == 0) {  
 //换行  
 System.***out***.println();  
 }  
 }  
  
 ps.close();  
}

* **处理流之五：数据流**
* 为了方便地操作Java语言的基本数据类型的数据，可以使用数据流。
* 数据流有两个类：(用于读取和写出基本数据类型的数据）。
  + **DataInputStream**和**DataOutputStream**。
  + 分别“套接”在InputStream和OutputStream节点流上。
* DataInputStream中的方法。
  + boolean readBoolean() byte readByte()
  + char readChar() float readFloat()
  + double readDouble() short readShort()
  + long readLong() int readInt()
  + String readUTF() void readFully(byte[]b)
* DataOutputStream中的方法。
  + 将上述的方法的read改为相应的write即可。

**写、读**

@Test  
 public void test2() {  
  
 DataOutputStream dos = null;  
  
 try {  
 FileOutputStream fos = new FileOutputStream("data.txt");  
 dos = new DataOutputStream(fos);  
  
 dos.writeUTF("我爱你，而你却不知道！");  
 dos.writeBoolean(true);  
 dos.writeLong(1453514);  
 } catch (IOException e) {  
 e.printStackTrace();  
 } finally {  
 if (dos != null) {  
 try {  
 dos.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 }  
  
 @Test  
 public void test3() {  
  
 DataInputStream dis = null;  
 try {  
 dis = new DataInputStream(new FileInputStream(new File("data.txt")));  
  
 String str = dis.readUTF();  
 System.***out***.println(str);  
 boolean b= dis.readBoolean();  
 System.***out***.println(b);  
 Long l = dis.readLong();  
 System.***out***.println(l);  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 if (dis != null) {  
 try {  
 dis.close();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}