链滴

学习笔记 --Java

作者：KioLuo

原文链接：https://ld246.com/article/1492567774759

来源网站：链滴

//输入输出

```
Scanner in = new Scanner(System.in);

int a = in.nextInt();

int b = in.next();

System.out.println(in.nextLine());
```

//输入流 InputStream

```
System.in.read(buffer);        //读取输入流存到buffer（byte[]类型）中

System.in.read();        //读一个字节，若结尾返回-1

System.in.read(buffer, int off, int len);        //    读len个字节，从buffer[off]开始存

System.in.skip(n);        //输入流跳过n个字节

System.in.available();

System.in.mark();        //标记

System.in.reset();        //返回标记处

System.in.markSupported;        //是否支持标记

System.in.close();        //关闭输入流
```

//类java.io.StreamTokenizer可以获取输入流并将其分析为Token(标记)。StreamTokenizer的nextToken方法将读取下一个标记,下一个标记的类型在ttype字段中返回。关于令牌的附加信息可能是在nva字段或标记生成器的sval 字段，结束标志为TT_EOF。

```
StreamTokenizer st = new StreamTokenizer(new BufferedReader(new InputStreamReader(System.in)));

st.nextToken();

int m=(int)st.nval;
```

//输出流 OutputStream

```
System.out.write(int b);

System.out.write(byte[] b);

System.out.write(byte[] b, int off, int len);

System.out.flush();

System.out.close();
```

//文件流

```
//二进制数据读写（字节流）
FileOutputStream out = new FileOutputStream("a.txt");       //根据字节读写数据
out.write(buffer);
out.close();
FileInputStream in = new FileInputStream("a.txt");
in.read();
in.close();
DataOutputStream out = new DataOutputStream(
                                new BufferedOutputStream(
                                    new FileOutputStream("a.txt")));       //读写基本类型数据
out.writeInt(int a);
DataInputStream in = new DataInputStream(
                                new BufferedInputStream(
                                    new FileInputStream("a.txt")));
in.readInt();
//文本数据读写（字符流）
PrintWriter out = new PrintWriter(
                                new BufferedWriter(
                                    new OutputStreamWriter(
                                        new FileOutputStream("a.txt"))));       //输出文本数据
out.println("abcdefg");
out.format("格式",...);
out.printf("格式", ...);
out.print(基本类型);
BufferedReader in = new BufferedReader(
                                new InputStreamReader(
```

```
                    new FileInputStream("a.txt")));                //读文本数据
String line = in.readLine();

in.getLineNumber();

FileReader in = new FileReader(String fileName);       //在给定从中读取数据的文件名的情况下新FileReader

FileReader in = new FileReader(File file);             //在给定从中读取数据的File的情况下新建FileReder
```

//转换编码

```
InputStreamReader(InputStream in, String charsetName);       //创建使用指定字符集的InputStreamReader
```

//网络端口连接读写数据

```
Socket  socket = new Socket(InetAddress.getByName("localhost"), 12345);       //建立与本地12345端口的连接

PrintWriter out = new PrintWriter(

                    new BufferedWriter(

                        new OutputStreamWriter(

                            socket.getOutputStream())));                    //建立对socket的输出流
```

//读写对象

```
class Student implements Serializable {

...

}                                                                   //类实现Serializable串行化接口

ObjectOutputStream out = new ObjectOutputStream(

        new FileOutputStream("a.dat"));                 //对象输出流

out.writeObject(Student stu1);

ObjectInputStream in = new ObjectInputStream(

        new FileInputStream("a.dat");                   //对象输入流

out.readObject(Student stu2);
```

//字符串操作

```
String s = new String();
```

s.equals("abc");

s1.compareTo(s2);

s.substring(n);

s.substring(n1, n2);

s.charAt(index);

s.indexOf(c);

s.indexOf(c, n);

s.indexOf(t);　　//也可以寻找字符串位置

s.lastIndexOf(c);　　//从右边开始找

s.startsWith(t);　　//是否由字符串t开始

s.endsWith(t);　　//是否由字符串t结尾

s.trim();　　//把字符串两端的空格删掉

s.replace(c1, c2);　　//把s中所有的c1都换成c2

s.toLowerCase();　　//把s中所有的大写字母换成小写字母

s.toUpperCase();　　//把s中所有的小写字母换成大写字母

s.split(" ");　　//按某个字符（比如空格）将字符串提取出来

StringBuffer sb = new StringBuffer();　　//用作字符串缓存器

StringBuilder sb = new StringBuilder();　　//与StringBuffer相似，没有实现线程安全功能

sb.append("");　　//增加一个字符串

sb.insert(n, "");　　//插入字符串

sb.toString();　　//转换成字符串

String.format("Hello, %s. Next year, you'll be %d", name, age);　　//格式化字符串

//Java中的包装类

Integer类的常用方法：

byteValue()　　//将Integer转换为byte类型

doubleValue()　　//将Integer转换为double类型

...

基本类型转换为字符串：

Integer.toString(int n)　　//转换为字符串类型

String.valueOf(int n)　　//int类型转换为字符串类型

字符串转换为基本类型：

Integer.parseInt(Stirng s)　//将字符串转换为int类型

Integer.valueOf(String s)　　//将字符串转换为Integer类型

//时间类

Date d = new Date();　　//java.util包中，默认无参构建当前时间

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");　　//java.tex
包中，创建目标格式对象

String today = sdf.format(d);　　//将时间格式化为指定格式

Date date = sdf.parse(today);　　//将字符串按指定格式转化为时间

Calendar c = Calendar.getInstance();　　//java.util.Calendar，创建Calendar对象，初
化为当前时间

int year = c.get(Calendar.YEAR);

int month = c.get(Calendar.MONTH)+1;　　//0表示1月份

int day = c.get(Calendar.DAY_OF_MONTH);

int hour = c.get(Calendar.HOUR_OF_DAY);

int minute = c.get(Calendar.MINUTE);

int second = c.get(Calendar.SECOND);

Date date = c.getTime();　　//Date和Calendar类互转

Long time = c.getTimeInMillis();　　//获取当前毫秒数

//数学函数

Math.abs(a);　　// java.lang包，求绝对值

Math.pow(a);　　//指数运算

Math.random();　　//产生从0到1间的随机数

Math.round(a);　　//求近似

Math.floor();　　//返回小于参数的最大整数

Math.ceil();　　//返回大于参数的最小整数

//集合

//容器类

Collection接口，子接口有Set，List，Queue，实现类有ArrayList, LinkedList, HashSet

Map接口，实现类有HashMap

ArrayList list = new ArrayList();　　//定义ArrayList容器，ArrayList是List接口的一个实现类，List Collection的子接口

list.add(s);　　//增加元素

list.add(n, s);　//在位置n上插入元素

list.addAll(Array.asList(array));　　//将数组转换为list并添加到list中

list.addAll(n, Array.asList(array));　　//插入到指定位置

list.remove(s);　//删除指定元素或指定位置元素

list.removeAll(Array.asList(array));　　//删除list中指定的元素集合

list.set(n, s);　　//修改n位置的元素为s

list.size();　　//返回大小

Iterator it = list.iterator();　　//获取list的迭代器

it.hasNext();　　//判断迭代器是否有下一个元素

it.Next();　　//利用迭代器获取下一个元素

//HashMap散列表

HashMap coinnames = new HashMap();

coinnames.put(1, "penny");　　//放进键值对

coinnames.get(1);　　//返回键1对应的值

coinnames.containsKey(1);　　//是否包含某键

coinnames.keySet();　　//返回键的集合

//图形界面包

import javax.swing.JFrame;　　//图形窗口

import javax.swing.JPanel;　　//图形面板

import javax.swing.JButton;　　//图形按钮

import java.awt.Color;　　//颜色设置

```
import java.awt.Graphics;        //绘制图像
import java.awt.BorderLayout;        //组件布局管理
import java.awt.event.ActionEvent;        //事件处理
import java.awt.event.ActionListener;        //事件监听
import javax.swing.JScrollPane;        //滚动面板
import javax.swing.JTable;        //图形表格
import javax.swing.table.TableModel;        //表格的数据模型接口
```