JAVA基础知识

一、流程控制

（一）输出与输入

输出：

输出用法	解释
System.out.println()	println = print line,意为输出后自动换行
System.out.print()	输出后不换行
System.out.printf()	格式化输出，使用占位符%（与C语言类似

```
System.out.printf("%.2f\n", d); // 显示两位小数
```

占位符	说明
%d	格式化输出整数
%x	格式化输出十六进制整数
%f	格式化输出浮点数
%e	格式化输出科学计数法表示的浮点数
%s	格式化字符串

输入：

使用import导入输入的类，然后创建输入对象，读入后scanner会自动转换数据类型

读取输入字符串：**scanner.nextLine()**

读入输入的整数：**scanner.nextInt()**

（其余可以百度java输出）

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in); // 创建Scanner对象
        System.out.print("Input your name: "); // 打印提示
        String name = scanner.nextLine(); // 读取一行输入并获取字符串
        System.out.print("Input your age: "); // 打印提示
        int age = scanner.nextInt(); // 读取一行输入并获取整数
        System.out.printf("Hi, %s, you are %d\n", name, age); // 格式化输出
    }
}
```

二、数组

数组知识与C语言基本相同，定义有些许不同。

Java的数组有几个特点：

- 数组所有元素初始化为默认值，整型都是`0`，浮点型是`0.0`，布尔型是`false`；
- 数组一旦创建后，大小就不可改变。

（一）一维数组

1.基础知识

定义：

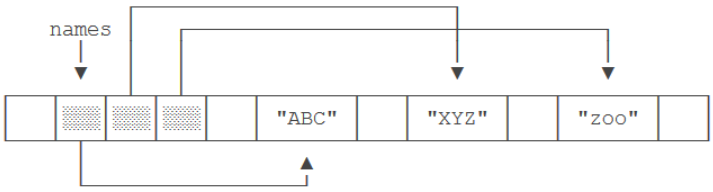
```
数据类型 数组名称 = new 数据类型[长度];
int score[] = new int[3];
//定义时直接指定元素
int score[] = new int[]{4,5,52,8};
int score[] = {4,5,52,8};
<<<<<<< HEAD
int score[4] = {4,5,52,8};//错误，4与{}不能同时出现
int score[];//错误
```

数组元素可以是各种数据类型，数组元素可以是值类型（如int）或引用类型（如String），但是数组本身是引用类型，即score为操作数组元素的地址，存在栈内存中，而数组元素存储在堆内存中（有点对象的**句柄和属性**那味了）。值得注意的是，**int型数组变化相对应内存会变化，而字符串只是指向变化，内存不变。**

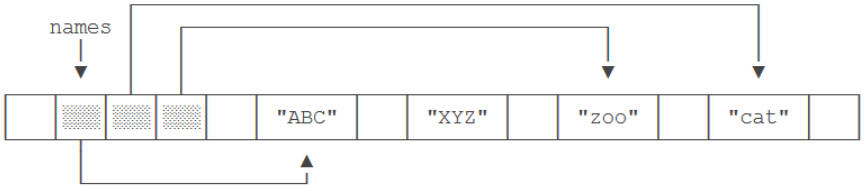
字符串是引用类型，因此我们先定义一个字符串数组：

```
String[] names = {
    "ABC", "XYZ", "zoo"
};
```

对于`String[]`类型的数组变量`names`，它实际上包含3个元素，但每个元素都指向某个字符串对象：



对`names[1]`进行赋值，例如`names[1] = "cat";`，效果如下：



其内存并没有改变，而是names[1]指向的内存块发生了变化。而"XYZ"因为再也无法被访问而“死亡”。

以及在数组为空时，调用score[0]和使用score.length都是错误的

2.数组遍历

使用for循环，通过索引遍历，与C语言相同。

方法一：

```
public class main{
    public static void main(String args){
        int ns[] = {4,5,8,6,6};
        for(int i=0; i<ns.length ;i++){
            System.out.println(ns[i]);
        }
    }
}
```

方法二：使用for each循环，直接迭代数组的每个元素，在该循环中，变量n直接拿到ns数组的元素而不是索引

```
public class main{
    public static void main(String args){
        int ns[] = {4,5,8,6,6};
        for(int n : ns){
            System.out.println(n);
        }
    }
}
```

方法三：调用函数**Arrays.toString(数组名称)**迅速打印数组内容（可用于调试代码）

```
import java.util.Arrays;
      System.out.println(Arrays.toString(ns));
```

3.数组排序

排序是数组中常见的内容，其基本排序算法有冒泡排序、插入排序和快速排序等，当然还可以通过调用函数进行排序。排序也要注意int型与字符串型排序后内存变化的区别。

方法一：冒泡排序法

主要通过将最大的或者最小的冒泡到最后一个位置实现，下例为从小到大冒泡排序

```
   for (int i = 0; i < ns.length - 1; i++) {
            for (int j = 0; j < ns.length - i - 1; j++) {
                if (ns[j] > ns[j+1]) {
                    // 交换ns[j]和ns[j+1]:
                    int tmp = ns[j];
                    ns[j] = ns[j+1];
                    ns[j+1] = tmp;
                }
            }
        }
```

方法二：通过调用函数**Arrays.sort(数组名称)**实现从小到大排序

```
import java.util.Arrays;
    Arrays.sort(array);//array为需排序的数组名称
```

（其他方法暂时不做说明）

（二）多维数组

1.二维数组

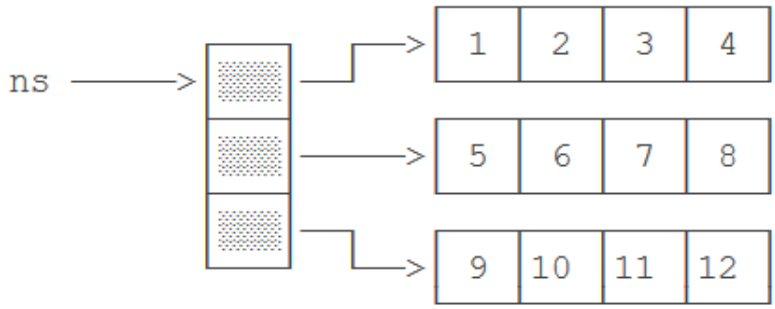
二维数组就像有行列的表格，跟C语言差不多。当然也可以理解为数组的数组。

定义：

```
数据类型 数组名[][] = new 数据类型[行的个数][列的个数];
int ns[][]= new int[3][4];
int ns[][]={
    { 1, 2, 3, 4 },
    { 5, 6, 7, 8 },
    { 9, 10, 11, 12 }

};
```

内存结构：



ns[0]对应第一行的数组，ns[0] [0]对应第一行的第一个元素;ns中包含三个数组，所以ns.length=3，而ns[0].length=4。

与C语言不同的是，二维数组每个数组的元素并不要求相同，可以定义如下数组：

```
int ns[][]={
    { 1, 2, 3, 4 },
    { 5, 6 },
    { 9, 10, 11 }

};
```

打印二维数组：

方法一：使用两层的for嵌套循环：

```
for (int[] arr : ns) {
    for (int n : arr) {
        System.out.print(n);
        System.out.print(', ');
    }
    System.out.println();
}
```

注意 int [] arr与int n的区别

方法二：使用函数**Arrays.deepToString()**

```
System.out.println(Arrays.deepToString(ns));
```

2.三维数组

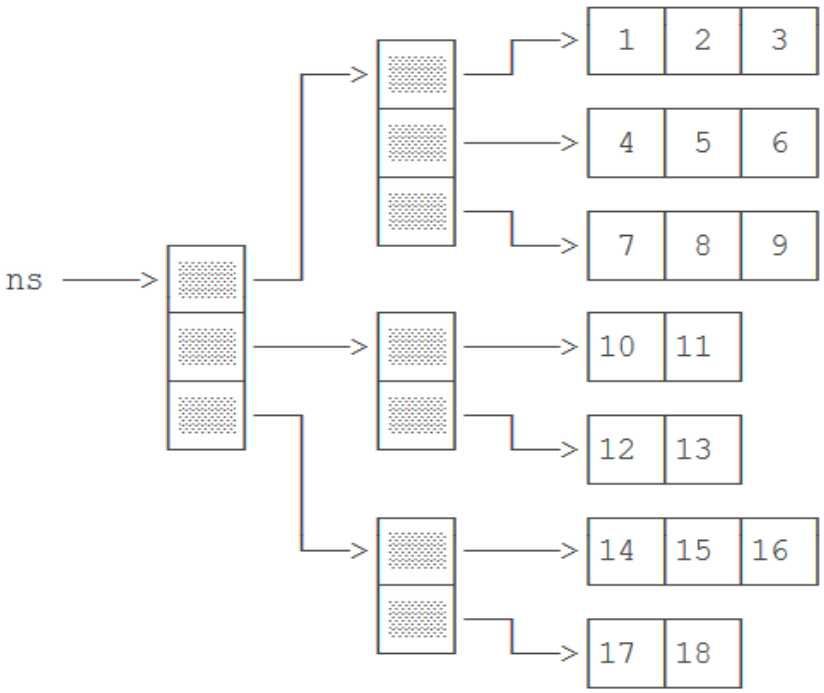
三维数组就是二维数组的数组，多维数组依次类推。

定义：

```
int ns[][][] = new int[4][5][5];
int[][][] ns = {
    {
        {1, 2, 3},
        {4, 5, 6},
        {7, 8, 9}
```

```
    },
    {
        {10, 11},
        {12, 13}
    },
    {
        {14, 15, 16},
        {17, 18}
    }
};
```

内存结构：



访问元素及length与二维数组无差。

三、String类与字符串

1.实例化对象

实例化对象的方法有两种：

```
直接赋值：String str1 = "hello";
调用构造方法，new一个对象：String str1 = new String("hello");
```

两种实例化方法的比较：

首先明确字符串是String类的匿名对象，匿名对象就是已经开辟了堆内存空间的并可以直接使用的对象。

第一种实例方法实际是把一个在堆中开辟好的堆内存空间的使用权给了str1这个对象,如果下次还有相同的字符串声明，不会再重新开辟空间，而是把使用权同时给str3这个对象（引用分配）。所采用的模式是共享设计模式，jvm底层实际有个对象池（理解为存储在常量区也可以），在池子里以后就被保存好，下次来了同名的就把他调出来。

第二种实例方法是new，new无论如何都会开辟一个新空间，实际上开辟了**两个空间？？**，但是真正使用的只有使用new开辟的这一个空间。

2.String对象内容比较

比较方法也有两种：

```
使用"==":==进行的是数值的比较，无法透过地址看本质，实际是比较地址是否相同
例如：
String str1 = "hello";
String str2 = new String("hello");
String Str3 = "hello";
String str4 = str1;
String str5 = new String("hello");
str1 != str2:因为他们的地址不相同
str1 == str3:注意咯！定义一样内容的字符串，他们也是指向同一个地址
str1 == str4:因为他们的句柄指向的本来就是同一块地址，同一块空间
str2 ！= str5：他们是不同的对象，所指示地址空间不一样
```

```
 使用Sring里的类equals():可以深入地址比较字符串的内容 ！！注意区分大小写
 equalslgnoreCase()不区分
str1.equals(str2):true
str1.equals(str4):true
```

3.字符串内容不可变

在源代码中，字符串使用final关键字定义，它是常量，不可被继承，不可被修改，且底部实质为数组。于是对于每次改字符串的操作，它都要生成新的数组，不过表面还是字符串修改。

4.一些基本方法

length（）：字符串中的长度为方法，需要使用str.length()；而数组中的长度为属性，使用array.length。