4. 数组

4.1 数组概念：

一组连续的存储空间，存储多个相同数据类型的值。
特点：类型相同，长度固定，可自动将数组中的元素从0开始编号。

4.2 数组定义：

- 数组创建：`元素类型[] 数组名 = new 元素类型[元素个数或数组长度]`
- 声明数组变量须在程序中声明，并指定数据类型；
- 创建数组：要使用一个新的关键字`new`，`new`用来在内存中产生一个容器实体

4.3 数组初始化：

- 数组的声明与赋值

```
int[] a = new int[2];
a[0] = 10;
a[1] = 20;
```

初始化数组，必须将声明、创建、初始化都放在一条语句中，分开会导致语法错误。

4.4 数组下标的有效范围与常见异常

- Array Index Out Of Bounds Exception 索引值越界异常。
- Null Pointer Exception 空指针异常，引用类型变量没有指向任何对象，而访问了对象的属性或者是调用了对象的方法。

4.5 数组内存分析：

- 栈内存：栈内存存储的均为局部变量，变量一旦有其作用域，就会从内存消失，释放内存空间。
- 堆内存：堆内存存储的均为对象内存，对象一旦被使用完，并不会马上从内存中消失，而是等待垃圾回收器不定时地把垃圾对象回收，这时候该对象才会消失，释放内存。
- 凡是以`new`关键字创建的对象，`JVM`都会在堆内存中开辟一个新的空间，创建一个新的对象。
- 对象如果没有变量引用，那么该对象就是一个垃圾对象。

4.6 二维数组：

二维表结构可用二维数组来表示

4.6.1 创建二维数组

声明方式（两种）：

```
数组元素类型 数组名字[][];
数组元素类型[][] 数组名字;

int arr[][];
int [][] arr;
```

同一维数组一样，二维数组在声明时也没有分配内存空间，同样要使用关键字`new`来分配内存，然后才可以访问每个元素。为二维数组分配内存有两种方式：

```
int a[][];
a = new int[2][4];  //直接分配行列

int b[][];
b = new int[2][];   //先分配行，再分配列
b[0] = new int[2];  //给第一行分配列
b[1] = new int[2];  //给第二行分配列
```

4.6.2 二维数组的赋值：

二维数组的初始化方法与一维数组类似，也有`3`种方式。但不同的是，二维数组有两个索引（即下标），构成由行和列组成的一个矩阵。

4.6.3 多维数组：

比一维数组维数高的叫多维数组，理论上二维数组也属于多维数组。`Java`也支持三维、四维等多维数组，创建其他多维数组的方法与创建二维数组类似。

4.6.4 通过二维数组输出不同版式的古诗

创建`Poetry`类，声明一个字符型二维数组，将古诗《春晓》的内容赋值于二维数组，然后分别用横版和竖版两种方式输出

4.7 不规则数组

`Java`支持不规则数组，例如二维数组中，不同行的元素个数可以不同

4.8 数组的基本操作：

4.8.1 数组遍历：

遍历数组可使用`for`循环来实现；遍历二维数组，使用嵌套`for`循环，数组`length`属性可获得数组长度。

4.8.3 填充和替换数组元素

数组中的元素定义完成后，可通过`Arrays`类的静态方法`fill()`方法来对数组中的元素进行分配，起到填充和替换的效果。`fill()`方法可将指定的`int`值分配给`int`型数组的每个元素。语法如下：

```
Arrays.fill()(int[] a ,int value) //a:要进行元素分配的数组。value:要存储数组中所有元素的值。
```