第一部分 : IDEA开发工具

参见： IEDA的安装请参考文件夹PPT中的 04_IDEA.ppt

1.数组

1.1 数组介绍

数组就是存储数据长度固定的容器，存储多个数据的数据类型要一致。

1.2 数组的定义格式

1.2.1 第一种格式

数据类型[] 数组名

示例:

```
int[] arr;
double[] arr;
char[] arr;
```

1.2.2 第二种格式

数据类型 数组名[]

示例:

```
int arr[];
double arr[];
char arr[];
```

1.3 数组的动态初始化

1.3.1 什么是动态初始化

数组动态初始化就是只给定数组的长度，由系统给出默认初始化值

1.3.2 动态初始化格式

```
数据类型[] 数组名 = new 数据类型[数组长度];
```

```
int[] arr = new int[3];
```

1.3.3 动态初始化格式详解

- 等号左边:
  - int:数组的数据类型
  - []:代表这是一个数组
  - arr:代表数组的名称
- 等号右边:
  - new:为数组开辟内存空间
  - int:数组的数据类型
  - []:代表这是一个数组
  - 3:代表数组的长度

代码:

```
package com.itheima.array;

public class Demo2Array {
    /*
        数组的动态初始化:
                        在初始化的时候，需要手动指定数组的长度，系统会为数组容器分配初始值.

        动态初始化格式:
                        数据类型[] 数组名 = new 数据类型[数组的长度];

        注意:
                        打印数组变量的时候，会打印出数组的内存地址

        [I@10f87f48 :

                        @ : 分隔符
                        [ : 当前的空间是一个数组类型
                        I : 当前数组容器中所存储的数据类型
                        10f87f48 : 十六进制内存地址

                                0 1 2 3 4 5 6 7 8 9 a b c d e f
     */
    public static void main(String[] args) {
        // 数据类型[] 数组名 = new 数据类型[数组的长度];
```

```
        // 通过new关键字创建了一个int类型的数组容器，该容器可以存储5个int类型的整数，该容器被arr数组变量所记
录
        int[] arr = new int[5];
        // [I@10f87f48
        System.out.println(arr);

        byte[] bArr = new byte[3];
        // [B@b4c966a
        System.out.println(bArr);

    }
}
```

1.4 数组元素访问

1.4.1 什么是索引

每一个存储到数组的元素，都会自动的拥有一个编号，从0开始。

这个自动编号称为数组索引(index)，可以通过数组的索引访问到数组中的元素。

1.4.2访问数组元素格式

```
数组名[索引];
```

1.4.3示例代码

```
package com.itheima.array;

public class Demo3ArrayIndex {
    /*
        数组动态初始化：
                初始化的时候，手动指定数组长度，系统会为数组容器分配初始值.

        数组的元素访问格式：
                数组名[索引]

                索引：数组中数据的编号方式，编号从0开始
                作用：访问数组容器中的空间位置

        注意：
                数组在创建完毕后，即使没有赋值，也可以取出，但取出的元素都是默认初始化值.

     */
```

```
    public static void main(String[] args) {
        int[] arr = new int[3];         // 0 1 2
        System.out.println(arr);        // 数组的内存地址  [I@10f87f48

        // 数组名[索引]  访问数组容器中的空间位置
        System.out.println(arr[0]);     // 0  系统自动分配的默认初始化值
        System.out.println(arr[1]);
        System.out.println(arr[2]);

        System.out.println("--------------");

        // 数组名[索引]
        arr[0] = 11;
        arr[1] = 22;
        arr[2] = 33;

        System.out.println(arr[0]);
        System.out.println(arr[1]);
        System.out.println(arr[2]);
    }
}
```

1.5 内存分配

1.5.1 内存概述

内存是计算机中的重要原件，临时存储区域，作用是运行程序。

我们编写的程序是存放在硬盘中的，在硬盘中的程序是不会运行的。

必须放进内存中才能运行，运行完毕后会清空内存。

Java虚拟机要运行程序，必须要对内存进行空间的分配和管理。

1.5.2 java中的内存分配

- 目前我们只需要记住两个内存，分别是：栈内存和堆内存

区域名称	作用
寄存器	给CPU使用，和我们开发无关。
本地方法栈	JVM在使用操作系统功能的时候使用，和我们开发无关。
方法区	存储可以运行的class文件。
堆内存	存储对象或者数组，new来创建的，都存储在堆内存。

区域名称	作用
方法栈	方法运行时使用的内存，比如main方法运行，进入方法栈中执行。

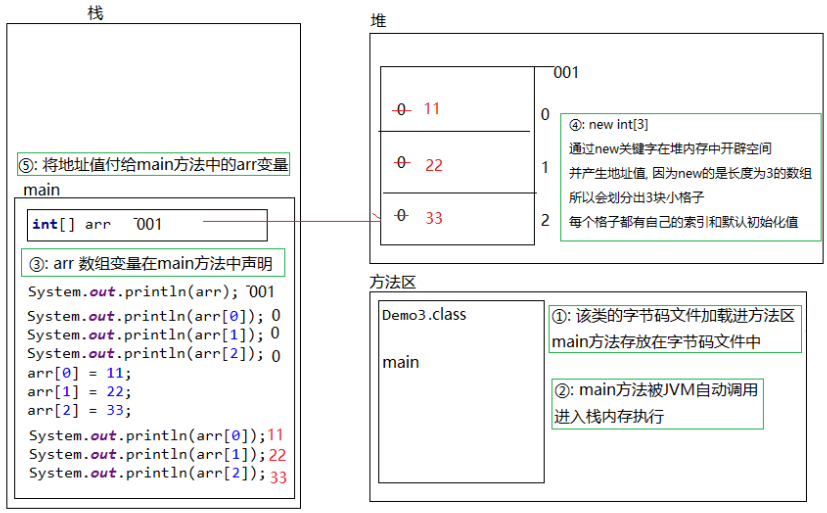
1.6 Java内存分配-一个数组内存图

```
public class Demo3{
    public static void main(String[] args) {
        int[] arr = new int[3];
        System.out.println(arr);
        System.out.println(arr[0]);
        System.out.println(arr[1]);
        System.out.println(arr[2]);
        arr[0] = 11;
        arr[1] = 22;
        arr[2] = 33;
        System.out.println(arr[0]);
        System.out.println(arr[1]);
        System.out.println(arr[2]);
    }
}
```

栈内存：方法运行时，进入的内存，局部变量都存放于这块内存当中

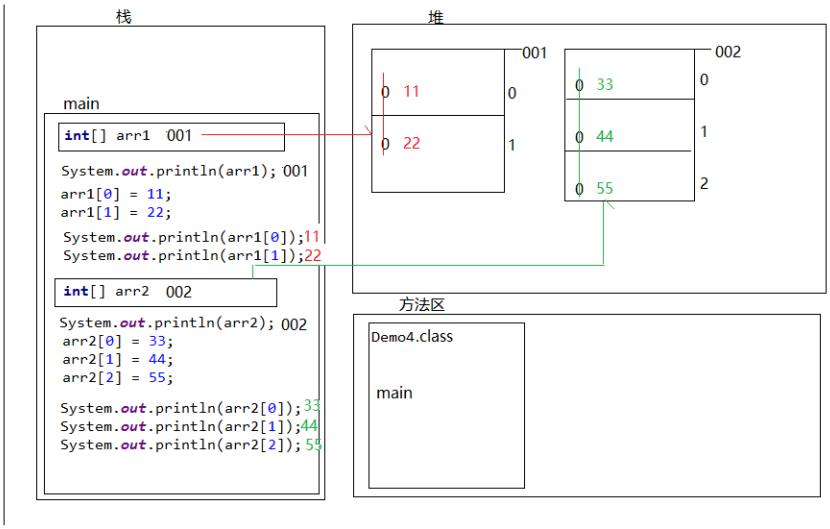
堆内存：new出来的内容都会进入堆内存，并且会存在地址值

方法区：字节码文件(.class文件)加载时进入的内存



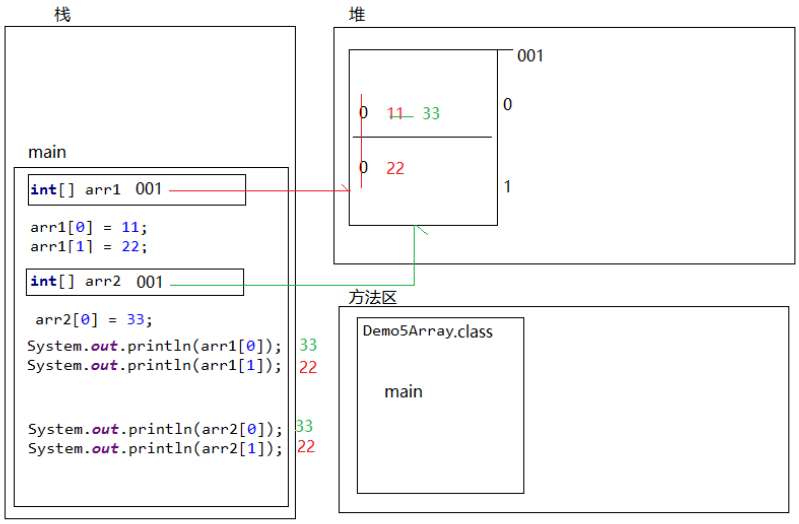
1.7 两个数组内存图

```
public class Demo4 {
    public static void main(String[] args) {
        int[] arr1 = new int[2];
        System.out.println(arr1);
        arr1[0] = 11;
        arr1[1] = 22;
        System.out.println(arr1[0]);
        System.out.println(arr1[1]);

        System.out.println("-------------");

        int[] arr2 = new int[3];
        System.out.println(arr2);
        arr2[0] = 33;
        arr2[1] = 44;
        arr2[2] = 55;
        System.out.println(arr2[0]);
        System.out.println(arr2[1]);
        System.out.println(arr2[2]);
    }
}
```



1.8 多个数组指向相同内存图

```java
public class Demo5Array {

    public static void main(String[] args) {
        int[] arr1 = new int[2];
        arr1[0] = 11;
        arr1[1] = 22;

        int[] arr2 = arr1;
        arr2[0] = 33;

        System.out.println(arr1[0]);
        System.out.println(arr1[1]);

        System.out.println("----------");

        System.out.println(arr2[0]);
        System.out.println(arr2[1]);

    }
}
```



1.9 数组的静态初始化

1.9.1 什么是静态初始化

在创建数组时，直接将元素确定

1.9.2 静态初始化格式

- 完整版格式

```
数据类型[] 数组名 = new 数据类型[]{元素1,元素2,...};
```

- 简化版格式

```
数据类型[] 数组名 = {元素1,元素2,...};
```

1.9.3示例代码

```
package com.itheima.array2;

public class Demo1Array {
    /*
        数组静态初始化 ： 初始化时指定每个数组元素的初始值，由系统决定数组长度

        完整格式:
                    数据类型[] 数组名 = new 数据类型[]{数据1,数据2,数据3...};
        简化格式:
```

```
                    数据类型[] 数组名 = {数据1,数据2,数据3...};
     */
    public static void main(String[] args) {
        // 数据类型[] 数组名 = new 数据类型[]{数据1,数据2,数据3...};
        int[] arr = new int[]{11,22,33};
        System.out.println(arr[0]);
        System.out.println(arr[1]);
        System.out.println(arr[2]);

        // 数据类型[] 数组名 = {数据1,数据2,数据3...};
        int[] arr2 = {44,55,66};
        System.out.println(arr2);
        System.out.println(arr2[0]);
        System.out.println(arr2[1]);
        System.out.println(arr2[2]);

    }
}
```

1.10 数组操作的两个常见问题

1.10.1 索引越界异常

- 出现原因

```
public class ArrayDemo {
    public static void main(String[] args) {
        int[] arr = new int[3];
        System.out.println(arr[3]);
    }
}
```

数组长度为3，索引范围是0~2，但是我们却访问了一个3的索引。

程序运行后，将会抛出ArrayIndexOutOfBoundsException 数组越界异常。在开发中，数组的越界异常是不能出现的，一旦出现了，就必须要修改我们编写的代码。

- 解决方案

将错误的索引修改为正确的索引范围即可！

1.10.2 空指针异常

- 出现原因

```
public class ArrayDemo {
    public static void main(String[] args) {
        int[] arr = new int[3];

        //把null赋值给数组
        arr = null;
        System.out.println(arr[0]);
    }
}
```

arr = null 这行代码，意味着变量arr将不会在保存数组的内存地址，也就不允许再操作数组了，因此运行的时候会抛出 NullPointerException 空指针异常。在开发中，数组的越界异常是不能出现的，一旦出现了，就必须要修改我们编写的代码。

- 解决方案

  给数组一个真正的堆内存空间引用即可！

1.11 数组遍历

- 数组遍历：就是将数组中的每个元素分别获取出来，就是遍历。遍历也是数组操作中的基石。

```
public class ArrayTest01 {
        public static void main(String[] args) {
                int[] arr = { 1, 2, 3, 4, 5 };
                System.out.println(arr[0]);
                System.out.println(arr[1]);
                System.out.println(arr[2]);
                System.out.println(arr[3]);
                System.out.println(arr[4]);
        }
}
```

以上代码是可以将数组中每个元素全部遍历出来，但是如果数组元素非常多，这种写法肯定不行，因此我们需要改造成循环的写法。数组的索引是 0 到 lenght-1 ，可以作为循环的条件出现。

```
public class ArrayTest01 {
    public static void main(String[] args) {
        //定义数组
        int[] arr = {11, 22, 33, 44, 55};

        //使用通用的遍历格式
        for(int x=0; x<arr.length; x++) {
            System.out.println(arr[x]);
        }
    }
}
```

1.12 数组获取最大值

- 最大值获取：从数组的所有元素中找出最大值。
- 实现思路：
  - 定义变量，保存数组0索引上的元素
  - 遍历数组，获取出数组中的每个元素
  - 将遍历到的元素和保存数组0索引上值的变量进行比较
  - 如果数组元素的值大于了变量的值，变量记录住新的值
  - 数组循环遍历结束，变量保存的就是数组中的最大值
- 代码实现：

```
package com.itheima.test;

import java.util.Scanner;

public class Test2Array {
    /*
        需求：从数组中查找最大值

                int[] arr = {12,45,98,73,60};

        实现步骤：
                1. 假设数组中的第一个元素为最大值
                2. 遍历数组，获取每一个元素，准备进行比较
                3. 如果比较的过程中，出现了比max更大的，让max记录更大的值
                4. 循环结束后，打印最大值.
     */
    public static void main(String[] args) {
        int[] arr = {12,45,98,73,60};
        // 1. 假设数组中的第一个元素为最大值
        int max = arr[0];
        // 2. 遍历数组，获取每一个元素，准备进行比较
        for(int i = 1; i < arr.length; i++){
```

```
            // 3. 如果比较的过程中，出现了比max更大的，让max记录更大的值
            if(arr[i] > max){
                max = arr[i];
            }
        }
        //  4. 循环结束后，打印最大值.
        System.out.println("max:" + max);
    }
}
```

1.13 数组元素求和

- 需求：键盘录入5个整数，存储到数组中，并对数组求和
- 思路： 1.创建键盘录入对象，准备键盘录入 2.定义一个求和变量，准备记录累加后的结果 3.动态初始化一个长度为5的int数组，准备存储键盘录入的数值 4.将键盘录入的数值存储到数组中 5.遍历数组，取出每一个元素，并求和 6.输出总和
- 代码实现：

```
package com.itheima.test;

import java.util.Scanner;

public class Test3Array {
    /*
        需求：键盘录入5个整数，存储到数组中，并对数组求和

        思路:
            1.创建键盘录入对象，准备键盘录入
            2.定义一个求和变量，准备记录累加后的结果
            3.动态初始化一个长度为5的int数组，准备存储键盘录入的数值
            4.将键盘录入的数值存储到数组中
            5.遍历数组，取出每一个元素，并求和
            6.输出总和
     */
    public static void main(String[] args) {
        // 1.创建键盘录入对象，准备键盘录入
        Scanner sc = new Scanner(System.in);
        // 2.定义一个求和变量，准备记录累加后的结果
        int sum = 0;
        // 3.动态初始化一个长度为5的int数组，准备存储键盘录入的数值
        int[] arr = new int[5];
        // 4.将键盘录入的数值存储到数组中
        for(int i = 0; i < arr.length; i++){
            System.out.println("请输入第" + (i+1) + "个整数:");
            //arr[i] = 10;
            arr[i] = sc.nextInt();
```

```
        }

        // 5.遍历数组，取出每一个元素，并求和
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }

        // 6.输出总和
        System.out.println("sum:" + sum);

    }
}
```

1.14 数组基本查找【应用】

- 需求： 已知一个数组 arr = {19, 28, 37, 46, 50}; 键盘录入一个数据，查找该数据在数组中的索引，并在控 制台输出找到的索引值。
- 思路： 1.定义一个数组，用静态初始化完成数组元素的初始化 2.键盘录入要查找的数据，用一个变量接收 3.定义一个索引变量，初始值为-1 4.遍历数组，获取到数组中的每一个元素 5.拿键盘录入的数据和数组中的每一个元素进行比较，如果值相同，就把该值对应的索引赋值给索引变量，并结束循环 6.输出索引变量
- 代码实现：

```
public static void main(String[] args) {
        // 1.定义一个数组，用静态初始化完成数组元素的初始化
        int[] arr = {19, 28, 37, 46, 50};
        // 2.键盘录入要查找的数据，用一个变量接收
        Scanner sc = new Scanner(System.in);
        System.out.println("请输入您要查找的元素:");
        int num = sc.nextInt();
        // 3.定义一个索引变量，初始值为-1
        // 假设要查找的数据，在数组中就是不存在的
        int index = -1;
        // 4.遍历数组，获取到数组中的每一个元素
        for (int i = 0; i < arr.length; i++) {
            // 5.拿键盘录入的数据和数组中的每一个元素进行比较，如果值相同，就把该值对应的索引赋值给索引变量，并结束循环
            if(num == arr[i]){
                // 如果值相同，就把该值对应的索引赋值给索引变量，并结束循环
                index = i;
                break;
            }
        }
        //  6.输出索引变量
        System.out.println(index);
    }
```

```
}
```

1.15 评委打分【应用】

- 需求：在编程竞赛中，有6个评委为参赛的选手打分，分数为0-100的整数分。 选手的最后得分为：去掉一个最高分和一个最低分后 的4个评委平均值 (不考虑小数部分)。
- 思路： 1.定义一个数组，用动态初始化完成数组元素的初始化，长度为6 2.键盘录入评委分数 3.由于是6个评委打分，所以，接收评委分数的操作，用循环 4.求出数组最大值 5.求出数组最小值 6.求出数组总和 7.按照计算规则进行计算得到平均分 8.输出平均分
- 代码实现：

```
public static void main(String[] args) {
    // 1.定义一个数组，用动态初始化完成数组元素的初始化，长度为6
    int[] arr = new int[6];
    // 2.键盘录入评委分数
    Scanner sc = new Scanner(System.in);
    //  3.由于是6个评委打分，所以，接收评委分数的操作，用循环
    for (int i = 0; i < arr.length; i++) {
        System.out.println("请输入第" + (i+1) + "个评委的打分:");
        int score = sc.nextInt();
        if(score >= 0 && score <= 100){
            // 合法的分值
            arr[i] = score;
        }else{
            // 非法的分值
            System.out.println("您的打分输入有误，请检查是否是0-100之间的");
            i--;
        }
    }

    // 4.求出数组最大值
    int max = arr[0];
    for (int i = 1; i < arr.length; i++) {
        if(max < arr[i]){
            max = arr[i];
        }
    }

    // 5.求出数组最小值
    int min = arr[0];
    for (int i = 1; i < arr.length; i++) {
        if(min > arr[i]){
            min = arr[i];
        }
    }
```

```
        // 6.求出数组总和
        int sum = 0;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }

        // 7.按照计算规则进行计算得到平均分
        int avg = (sum - max - min ) / 4;

        // 8.输出平均分
        System.out.println(avg);
    }
}
```