

疑问：

- 1.创建类 定义方法（形参、返回值）
- 2.java实现非常复杂功能，SUN提供的jdk、找第三方的工具来实现
- 3.开发工作，面临自己的写方法

课程回顾

1 方法定义语法和调用语法***

```
1 public void 方法名(){
2
3 }
4 调用：对象.方法名()
5
6 public void 方法名(形参列表){
7
8 }
9 调用：对象.方法名(实参列表)
10
11 public 与返回值类型兼容的java数据类型 方法名(形参列表){
12     return 值; //将方法执行的结果返回给方法调用者
13 }
14 调用：与返回值类型兼容的java数据类型 变量=对象.方法名(实参列表)
15
16 public 与返回值类型兼容的java数据类型 方法名(){
17     return 值; //将方法执行的结果返回给方法调用者
18 }
19 调用：与返回值类型兼容的java数据类型 变量=对象.方法名()
```

2 重载 *****

- ```
1 同一个类中，方法名一样的，形参列表不一样（个数不一样、数据类型不一样，顺序不一样）
2
3 JVM如何区分使用的重载后的哪一个方法？主要看实参列表
```

### 3 形参和实参

- ```
1 出现位置不同
2 形参 出现定义方法时，
3 实参 出现方法调用时
4
5 赋值时机不同
6 形参不需要赋值，只需要指明形参的类型，
7 实参必须赋值，只有赋值的参数才能传递给方法使用
8
```

课后作业第四题

第一步：实现学生信息的添加功能

方案一：

- 学生管理类

```
1 public class StudentManagement {
2     /**
3      * 新增学生
4      */
5     public void add(){
6         Scanner input = new Scanner(System.in);
7         //定义数组保存学生信息
8         String[] names=new String[5];
9         for(int i=0;i<names.length;i++){
10             System.out.println("请输入学生的姓名: ");
11             names[i]=input.next();
12         }
13         //输出所有的学生的姓名
14         System.out.println("本班所有学生的姓名如下所示: ");
15         for (String name:names) {
16             System.out.print(name+"\t\t");
17         }
18     }
```

- 测试类

```
1 import java.util.Scanner;
2
3 /**
4  * @Author: lc
5  * @Date: 2022/3/21
6  * @Description: 测试学生管理类中添加学生信息的方法
7  * @Version: 1.0
8  */
9 public class TestStudentManagement {
10     public static void main(String[] args) {
11         StudentManagement sm = new StudentManagement();
12         //提交给学生管理类, 新增
13         sm.add();
14     }
15 }
```

方案二：

- 学生管理类

```
1 /**
2  * @Author: lc
3  * @Date: 2022/3/21
4  * @Description: 学生管理类, 主要提供: 添加学生和查找学生的功能
5  * @Version: 1.0
6  */
7 public class StudentManagement {
8     //定义数组, 全局变量, 属性!!!!
9     String[] names=new String[5];
```

```

10     /**
11      * 新增学生
12      */
13     public void add(String studentName){
14         for (int i=0;i<names.length;i++) {
15             //在数组找空位置，来一个学生，就往空位置存入一个学生
16             if (names[i] == null) {
17                 names[i] = studentName;
18                 break;
19             }
20         }
21     }
22
23     /**
24      * 显示数组所有的学生信息
25      */
26     public void showInfo(){
27         //输出所有的学生的姓名
28         System.out.println("本班所有学生的姓名如下所示：");
29         for (String name:names) {
30             System.out.print(name+"\t\t");
31         }
32     }
33 }

```

- 测试类

```

1  import java.util.Scanner;
2
3  /**
4   * @Author: lc
5   * @Date: 2022/3/21
6   * @Description: 测试学生管理类中添加学生信息的方法
7   * @Version: 1.0
8   */
9  public class TestStudentManagement {
10     public static void main(String[] args) {
11         Scanner input = new Scanner(System.in);
12         StudentManagement sm = new StudentManagement();
13         for(int i=0;i<5;i++){
14             System.out.println("请输入学生的姓名：");
15             String str=input.next();
16             //提交给学生管理类，新增
17             sm.add(str);
18         }
19
20         //查看存入数据后，数组的所有数据
21         sm.showInfo();
22     }
23 }

```

第二步：实现学生信息在指定范围内查找

- 学生管理类

```

1  /**

```

```

2  * @Author: lc
3  * @Date: 2022/3/21
4  * @Description: 学生管理类，主要提供：添加学生和查找学生的功能
5  * @Version: 1.0
6  */
7  public class StudentManagement {
8      //定义数组，全局变量,属性!!!!，默认null
9      String[] names=new String[5];
10
11     /**
12      * 查找学生
13      * 回忆: int findIndex=Arrays.binarySearch(int[],key)
14      * 自定义方法必须保证每一条执行的路线都有返回值!! 1
15      * @param start 开始查找的位置
16      * @param end 结束查找的位置
17      * @param findName 要查找的学生姓名
18      */
19     public boolean search(int start,int end,String findName){
20         //查找范围的合法性验证
21         if(start<=0){
22             System.out.println("起始位置必须是大于等于0的整数!");
23             return false;
24         }
25         if(end>names.length){
26             System.out.println("结束位置的值必须小于等于"+names.length);
27             return false;
28         }
29         if(end<start){
30             System.out.println("结束位置必须大于等于开始位置!");
31             return false;
32         }
33         //定义保存查找到的学生所在位置下标的变量
34         int findIndex=-1;
35
36         for(int i=start-1;i<end;i++){
37             //避免NullPointerException, 建议将明确值的变量放在前面
38             if(findName.equals(names[i])){//names[i].equals(findName)
39                 findIndex=i;
40                 break;
41             }
42         }
43         return findIndex>=-1;//true-找到了 false--未找到
44     }
45 }
46

```

- 测试类，测试学生查找的功能

```

1  import java.util.Scanner;
2
3  /**
4   * @Author: lc
5   * @Date: 2022/3/21
6   * @Description: 测试学生管理类中的查找学生信息的方法
7   * @Version: 1.0
8   */
9  public class TestStudentManagement {

```

```

10     public static void main(String[] args) {
11         Scanner input = new Scanner(System.in);
12         StudentManagement sm = new StudentManagement();
13
14         System.out.println("请输入开始查找的位置: ");
15         int start = input.nextInt();
16         System.out.println("结束查找的位置: ");
17         int end = input.nextInt();
18         System.out.println("请输入要查找的学生姓名: ");
19         String findName = input.next();
20         boolean isOk = sm.search(start, end, findName);
21         if(isOk==true){//if(isOk)等价的
22             System.out.println("找到了");
23         }else{
24             System.out.println("未找到");
25         }
26     }
27 }
28

```

课程目标

1 全局变量和局部变量的区别 ===== 理解

2 按值传递和按引用传递区别 ===== 理解

3 构造方法 ===== 重点

课程实施

1 按值传递和按引用传递

传参是什么意思？调用方法并传入实参过程

1-1 传参的形式分为两类

按值传递：

调用方法时，传入的实参类型都是基本类型。String虽然是引用类型，但是参数传递的时候，遵循按值传递的特点

特点：按值传递时，将实参的值复制一份交给形参使用。方法对形参的数据无论做什么操作，跟实参没有任何关系！！

形参的值发生变化，不影响实参。

按引用（即地址）传递

调用方法时，传入的实参类型都是引用类型。

特点：按引用传递时，将实参的地址复制一份交给形参使用。方法对形参地址指向的位置的数据进行操作，实参也地址也是指向当前堆中的数据，堆中的数据变化，所有引用该地址的变量值都会变化

形参的值发生变化，实参同步发生变化

1-2 案例分析：按值传递和按引用传递的区别

按值传递

```
1  /**
2   * @Author: lc
3   * @Date: 2022/3/21
4   * @Description: PACKAGE_NAME
5   * @Version: 1.0
6   */
7  public class Calculator {
8      /**
9       * 就是将用户传入的int类型数据进行修改
10     * @param i 用户调用方法时传入的实际值
11     */
12     public void changeNum(int i){
13         i=100;
14     }
15 }
16
```

按引用传递

```
1  /**
2   * @Author: lc
3   * @Date: 2022/3/21
4   * @Description: PACKAGE_NAME
5   * @Version: 1.0
6   */
7  public class Calculator {
8      /**
9       * arr数组里面只有一个值
10     * @param arr
11     */
12     public void changeNum2(int[] arr){
13         arr[0]=100;
14     }
15 }
16
```

程序测试的结果

```
1  /**
2   * @Author: lc
3   * @Date: 2022/3/21
4   * @Description: PACKAGE_NAME
5   * @Version: 1.0
6   */
7  public class TestCalculator {
8      public static void main(String[] args) {
9          Calculator c = new Calculator();
10         int num=0;
11         //没有起作用!!!??? 按值传递
12         c.changeNum(num);
13         System.out.println("num="+num); //num=0
14     }
15 }
16
```

```
14
15      //按引用传递
16      int[] temp=new int[1];
17      temp[0]=num;
18      //技巧: int[] arr=new int[]{0}
19      c.changeNum2(temp);
20      System.out.println("num="+temp[0]); //num=100
21  }
22 }
```

2 全局变量和局部变量

2-1 概念

类里面，方法外面定义的变量，统称为全局变量

方法的形参以及方法内部定义变量，统称为局部变量

2-2 区别

```
1  定义位置不同
2  全局变量：类里面，方法外面
3  局部变量：方法形参和方法内部
4
5  初始值不同
6  全局变量：程序员可以不赋值，创建对象时JVM给默认值
7  局部变量：程序员必须赋值，遵循变量使用规则：先定义、再赋值，才能使用
8
9  使用范围不同：
10 全局变量：类中所有的方法都可以引用全局变量
11 局部变量：哪个方法定义，只能由该方法使用，其他方法访问不到！！
12
13 生命周期不同：
14 全局变量：随着对象产生而产生，对象销毁内部全局变量也会被销毁    活得时间长
15 局部变量：随着方法调用而产生，方法弹栈，内部的局部变量就会全部销毁    活得时间短
16 优先考虑使用局部变量。
17 一个类中，很多方法使用同一个数据时，考虑使用全局变量
18
19 变量内存保存位置不同：
20 全局变量：都在堆中
21 局部变量：变量名都在栈中
22
23
24 使用关系：
25 全局变量和局部变量可以同名。一个方法如果使用了同名变量，如何判断使用的是全局还是局部？就近原则
26 优先使用局部变量。
27
28 如果使用全局变量，解决方案是？this访问全局变量
```

3 课后作业第十题

3-1 学生类

```
1  /**
2   * @Author: lc
3   * @Date: 2022/3/21
4   * @Description: 没有任何功能，仅提供学生该有的属性和属性值的输出
5   * @Version: 1.0
6   */
7  public class Student {
8      /**
9       * 姓名
10      */
11     public String name;
12     /**
13      * 成绩
14      */
15     public int score;
16
17     /**
18      * 输出属性的值
19      */
20     public void showInfo(){
21         System.out.println(name+"的成绩是: "+score);
22     }
23 }
```

3-2 学生成绩修改类

```
1  /**
2   * @Author: lc
3   * @Date: 2022/3/21
4   * @Description: PACKAGE_NAME
5   * @Version: 1.0
6   */
7  public class StudentModifer {
8      /**
9       * 修改学生成绩
10      * 条件: 有一个学生成绩小于60分，全体+2分
11      * @param students 全班学生的成绩
12      */
13     public void changeScore(Student[] students){
14         //students数组jvm执行到的时候，一定有值
15         //1-1 判断有没有人小于60分
16         boolean isAdd=false;//没有人小于60分
17         for(int i=0;i<students.length;i++){
18             //students[i]:对象数组 里面每一个下标位置对应的都是一个学生对象
19             if(students[i].score<60) {
20                 isAdd=true;//有人小于60
21                 break;
22             }
23         }
24         //1-2 基于第一步的结果，全体+2分
25         if (isAdd) {
26             //全体+2
27             for (int j=0;j<students.length;j++) {
28                 students[j].score+=2;
```



```

29         }
30     }
31
32     //2 输出所有学生的成绩
33     for(Student s:students){
34         s.showInfo();
35     }
36 }
37 }

```

3-3 测试学生成绩修改类

```

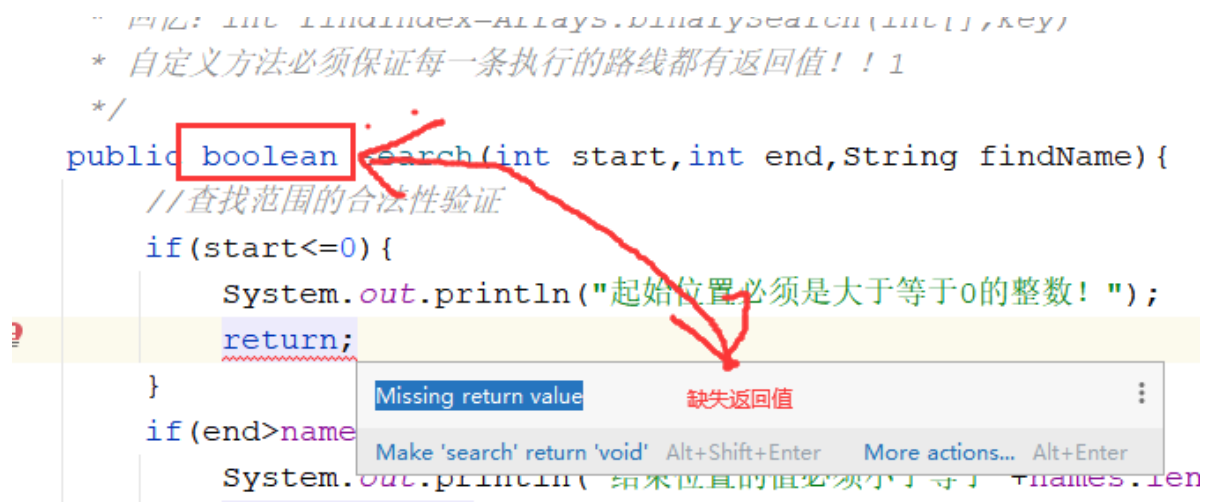
1  /**
2   * @Author: lc
3   * @Date: 2022/3/21
4   * @Description: PACKAGE_NAME
5   * @Version: 1.0
6   */
7  public class TestStudentModifier {
8      public static void main(String[] args) {
9          //1.准备一个班的学生数据 对象数组，jvm给的默认值是NULL
10         Student[] students=new Student[3];
11         //分别对象赋值
12         students[0]=new Student();
13         students[0].name="张三";
14         students[0].score=66;
15
16         students[1]=new Student();
17         students[1].name="李三";
18         students[1].score=96;
19
20         students[2]=new Student();
21         students[2].name="王三";
22         students[2].score=86;
23
24         //2.交给对象，实现+2功能
25         StudentModifier sm = new StudentModifier();
26         //Student[] students=students;
27         sm.changeScore(students);// Student[] students=实参
28     }
29 }

```

常见的程序问题

1 方法因为if没有返回值而报错

```
...  public int findIndex-Arrays.BinarySearch(int[],key)
* 自定义方法必须保证每一条执行的路线都有返回值！！1
*/
public boolean search(int start,int end,String findName){
    //查找范围的合法性验证
    if(start<=0){
        System.out.println("起始位置必须是大于等于0的整数！");
        return;
    }
    if(end>name
        System.out.println("结束位置的值必须小于等于"+names.length
```



课程总结

1 按值传递和按引用传递

2 全局变量和局部变量

3

预习安排