### JAVA第一阶段—DAY07- JAVA案例

1. 定义一个方法，实现n个整数的求和。

* 参考答案

public class VariableParamTest {

public static void main(String[] args) {

int rs1= getSum();

System.out.println(rs1);

int rs2= getSum(1,2);

System.out.println(rs2);

}

// 定义一个可变参数方法，求取n个整数之和

public static int getSum(int... numbers) {

int sum=0;

for (int i = 0; i < numbers.length; i++) {

sum+=numbers[i];

}

return sum;

}

}

1. 递归求n！。

* 参考答案

public class RecursiveTest {

public static void main(String[] args) {

// 调用递归方法，求n的阶乘

int res=getRecursive(5);

System.out.println(res);

}

// 递归方法，求n的阶乘

public static int getRecursive(int n) {

if(n==1) { // 如果为1的时候，就跳出，不再递归

return 1;

}else {

// n非1的情况下，继续调用本身，获取前一个数的递归结果

int res=n\*getRecursive(n-1);

return res;

}

}

}

1. 定义一个类及其相关的构造方法，并进行调用演示。

* 参考答案

public class Student {

// 成员变量

String username;

int age;

// 构造方法

public Student() {

System.out.println("这是一个无参构造方法");

}

public Student(String username, int age) {

this.username = username;

this.age = age;

System.out.println("这是一个有参构造方法");

}

// 成员方法

public void study(String username) {

System.out.println(username+"在学习");

}

}

public class ConstructorTest {

public static void main(String[] args) {

// 1、通过无参构造方法创建对象

Student stu1 =new Student();

System.out.println(stu1.username);

System.out.println(stu1.age);

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

// 2、通过有参构造方法创建对象

Student stu2 =new Student("李师师",20);

System.out.println(stu2.username);

System.out.println(stu2.age);

}

}

1. 比赛打分。

* 参考答案

public class ScoreTest {

public static void main(String[] args) {

//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];

//键盘录入评委分数

Scanner sc =new Scanner(System.in);

// 收评委分数的操作，用循环改进

for(int i=0; i<arr.length; i++) {

System.out.println("请输入第" + (i + 1) + "个评委的打分：");

arr[i] = sc.nextInt();

}

// 获取最大值

int max =getMax(arr);

// 获取最小值

int min =getMin(arr);

// 获取平均分

int avg =getAvg(arr,max,min);

//输出平均分

System.out.println("选手的最终得分是：" + avg);

}

// 定义获取最大值方法

public static int getMax(int[] arr) {

int max = arr[0];

for(int x=1; x<arr.length; x++) {

if(arr[x] > max) {

max = arr[x];

}

}

return max;

}

// 定义获取最小值方法

public static int getMin(int[] arr) {

int min = arr[0];

for(int x=1; x<arr.length; x++) {

if(arr[x] < min) {

min = arr[x];

}

}

return min;

}

// 定义求平均分方法

public static int getAvg(int[] arr,int max,int min) {

int total = 0;

for(int i=0; i<arr.length; i++) {

total+=arr[i];

}

int avg=(total-max-min)/(arr.length-2);

return avg;

}

}

1. 递归实现二分查找。

* 参考答案

public class BinarySearchTest {

public static void main(String[] args) {

// 定义一个有序数组

int[] arr = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

// 调用递归算法查找元素位置

int key=4;

int index= binSearch(arr, 0, arr.length-1, key);

// 输出结果

System.out.println(key+"数字在数组的角标位置是："+index);

}

// 定义递归二分算法

public static int binSearch(int arr[], int start, int end, int key) {

int mid = start + (end - start) / 2;

// 找到对应元素

if (arr[mid] == key) {

return mid;

}

if (key > arr[mid]) {

// 递归调用二分查找

return binSearch(arr, mid + 1, end, key);

} else if (key < arr[mid]) {

// 递归调用二分查找

return binSearch(arr, start, mid - 1, key);

}

// 没有找到，返回-1标志

if (start >= end) {

return -1;

}

return -1;

}

}

1. 兔子数列。

* 参考答案

public class FibonacciSequenceTest {

public static void main(String[] args) {

// 调用方法，获取12个月后的对数

int total =fun(12);

System.out.println("一年后兔子总对数是："+total);

}

// 定义递归放获取对应月数的兔子总对数

public static int fun(int m){

if(m==1 || m==2) {

return 1;

}else {

// 递归调用，求取前2个月对数之和

return fun(m-1)+fun(m-2);

}

}

}