### JAVA第一阶段—DAY07- JAVA作业答案

1. 简答题：简述递归方法的作用及优缺点。

**答案：**

递归方法可以使用重复方法、少量代码，即可实现复杂功能的实现；

递归方法一定要有出口，否则无限递归调用，直到出现StackOverflowError（栈内存溢出）错误；

递归方法的递归次数也不能过多，否则同样出现StackOverflowError（栈内存溢出）错误；

递归方法都可以使用非递归方法解决，比如循环。

1. 简答题：简述this关键字的使用。

**答案：**

this.属性名：来访问类中的成员变量，用来区分成员变量和局部变量（重名问题，如前面章节的有参构造方法属性赋值）；

this.方法名(参数值...)：用来访问本类的成员方法，通常直接省略；

this(参数值...)：访问本类的构造方法。

1. 简答题：简述Java中静态代码块的特点。

**答案：**

静态初始化块，用于类的初始化操作。

static代码块随着类的加载而加载，并且只初始化执行一次；

静态初始化块中不能直接访问非static成员；

静态初始化块可以置于类中的任何地方，类中可以有多个静态初始化块，多个静态代码块会按照书写顺序加载。

1. 编程题：使用递归实现求n！。

**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. 编程题：完成评委打分。

**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. 编程题：使用递归实现二分查找。

**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. 编程题：实现兔子数列。

**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);

}

}

}

1. 编程题：多种方式实现猴子吃桃问题。

**思路：我们可以采用逆向思维，第十天剩一个，前一天则为 S9 = （S10 +1）\*2，以此推算前一天。**

**答案：**

public class PeachTest {

public static void main(String[] args) {

forTest();

whileTest();

System.out.println(recursiveTest(10,1));

}

// 1、使用for循环实现

public static void forTest() {

int cur = 1; // 当前第10天 剩1个

for(int day = 9;day >=1; day--){

cur = (1+cur)\*2;

}

System.out.println(cur);

}

// 2、使用while循环实现

public static void whileTest() {

int cur = 1; // 第10天 剩1个

int day = 9;

while(day > 0){

cur = (1+cur)\*2;

day--;

}

System.out.println(cur);

}

// 3、使用递归方法实现

public static int recursiveTest(int day,int total){

if(day==1) {

return total;

}else {

total=(total+1)\*2;

return recursiveTest(--day,total);

}

}

}

1. 编程题：有一分数序列：2/1，3/2，5/3，8/5，13/8，21/13....求出这个数列的前20项之和。

**思路：后一个数的分子是前一个数的分母，同时后一个数的分母是前一个数的分子、分母之和，以此类推。**

**答案：**

public class FractionalSequenceTest {

public static void main(String[] args) {

System.out.println("数列的和为:" + getValue(20));

}

// 获取第i项的值 2/1，3/2，5/3，8/5，13/8

public static double getValue(int n) {

double a=2; //分子

double b = 1;//分母

double sum = 0; //值

double temp = 0;//临时变量

for(int i = 0; i < n; i++) {

sum += a/b;

temp = a;

a += b;

b = temp;

}

return sum;

}

}