**每日作业卷**

**JavaSE第8天**

传智播客.黑马程序员

# 基础题

## 练习一:相对路径和绝对路径的使用

描述:创建两个文件对象，分别使用相对路径和绝对路径创建。

答案

操作步骤:

绝对路径创建文件对象：使用File类一个参数的构造方法。

相对路径创建文件对象：使用File类两个参数的构造方法。

代码:

**public class** Test01\_01 {

**public static void** main(String[] args) {

// 创建文件对象：绝对路径

File f1 = **new** File("d:/aaa/a.txt");

// 创建文件对象：相对路径

File f2 = **new** File("a.txt");

}

}

## 练习二:检查文件是否存在,文件的创建

描述:检查D盘下是否存在文件a.txt,如果不存在则创建该文件。

答案

操作步骤:

1. 使用绝对路径创建对象关联到D盘的a.txt。
2. 通过文件对象方法判断文件是否存在。
3. 不存在则调用创建文件的方法创建文件。

代码:

**public class** Test01\_02 {

**public static void** main(String[] args) **throws** IOException{

// 创建文件对象：绝对路径

File f = **new** File("d:/a.txt");

// 如果文件不存在，则创建文件

**if**(!f.exists()) {

f.createNewFile();

}

}

}

## 练习三:单级文件夹的创建

描述:在D盘下创建一个名为bbb的文件夹。

答案

操作步骤:

1. 创建文件对象指定路径为d:/bbb
2. 调用文件对象创建文件夹的方法

代码:

**public class** Test01\_03 {

**public static void** main(String[] args) {

// 创建文件对象

File f = **new** File("d:/bbb");

// 创建单级文件夹

f.mkdir();

}

}

## 练习四:多级文件夹的创建

描述:在D盘下创建一个名为ccc的文件夹，要求如下：

1.ccc文件夹中要求包含bbb子文件夹

2.bbb子文件夹要求包含aaa文件夹

答案:

操作步骤:

1. 创建文件对象关联路径：d:/ccc/bbb/aaa
2. 调用文件对象创建多级文件夹的方法

代码:

**public class** Test01\_04 {

**public static void** main(String[] args) {

// 创建文件对象

File f = **new** File("d:/ccc/bbb/aaa");

// 创建多级文件夹

f.mkdirs();

}

}

## 练习五:删除文件和文件夹

描述:

将D盘下a.txt文件删除

将D盘下aaa文件夹删除,要求文件夹aaa是一个空文件夹。

答案:

操作步骤:

1. 创建文件对象关联路径：d:/a.txt
2. 调用文件对象删除文件的方法
3. 创建文件对象关联路径：d:/aaa
4. 调用文件对象删除文件夹的方法

代码:

**public class** Test01\_05 {

**public static void** main(String[] args) {

// 创建文件对象

File f = **new** File("d:/a.txt");

// 删除文件

f.delete();

// 创建文件夹对象

File dir = **new** File("d:/aaa");

// 删除文件夹

dir.delete();

}

}

## 练习六:获取文件信息:文件名,文件大小,文件的绝对路径,文件的父路径

描述:

获取D盘aaa文件夹中b.txt文件的文件名，文件大小，文件的绝对路径和父路径等信息，并将信息输出在控制台。

答案:

操作步骤:

1. 在D盘aaa文件夹中创建一个b.txt文件并输入数据
2. 创建文件对象关联路径：d:/aaa/b.txt
3. 调用文件对象的相关方法获得信息并输出。可以通过API帮助文档查询方法。

代码:

**public class** Test01\_06 {

**public static void** main(String[] args) {

// 创建文件对象

File f = **new** File("d:/aaa/b.txt");

// 获得文件名

String filename = f.getName();

// 获得文件大小

**long**filesize = f.length();

// 获得文件的绝对路径

String path = f.getAbsolutePath();

// 获得父文件夹路径，返回字符串

String parentPath = f.getParent();

// 获得父文件夹路径，返回文件对象

File parentFile = f.getParentFile();

// 输出信息

System.***out***.println("文件名：" + filename);

System.***out***.println("文件大小：" + filesize);

System.***out***.println("文件路径：" + path);

System.***out***.println("文件父路径：" + parentPath);

System.***out***.println("文件父路径：" + parentFile);

}

}

## 练习七:文件夹或文件的判断

描述:

1.判断File对象是否是文件,是文件则输出：xxx是一个文件，否则输出：xxx不是一个文件。

2.判断File对象是否是文件夹,是文件夹则输出：xxx是一个文件夹，否则输出：xxx不是一个文件夹。(xxx是文件名或文件夹名)

答案:

操作步骤:

1. 创建两个文件对象分别关联到不同的文件，比如：d:/a.txt，d:/aaa
2. 调用文件对象的判断是否是文件或是否是文件夹的方法
3. 获得文件名，根据判断结果输出信息。

代码:

**public class** Test01\_07 {

**public static void** main(String[] args) {

// 创建文件对象

File f1 = **new** File("d:/b.txt");

// 判断是否是一个文件

**if**(f1.isFile()) {

System.***out***.println(f1.getName()+"是一个文件");

} **else** {

System.***out***.println(f1.getName()+"不是一个文件");

}

// 创建文件对象

File f2 = **new** File("d:/aaaa");

// 判断是否是一个文件夹

**if**(f2.isDirectory()) {

System.***out***.println(f2.getName()+"是一个文件夹");

} **else** {

System.***out***.println(f2.getName()+"不是一个文件夹");

}

}

}

## 练习八:文件夹的获取方法

描述:

获取指定文件夹下所有的文件，并将所有文件的名字输出到控制台。

注意：不包含子文件夹下的文件

答案

操作步骤:

1. 创建文件对象关联到指定文件夹，比如：c:/aaa
2. 调用文件对象的listFiles方法获得文件数组
3. 遍历文件数组将每一个文件的名字输出到控制台

代码:

**public class** Test01\_08 {

**public static void** main(String[] args) {

// 创建文件对象

File f = **new** File("d:/aaa");

// 获得文件夹下所有文件

File[] files = f.listFiles();

// 遍历文件数组

**for** (File file :files) {

// 将文件的名字打印到控制台

System.***out***.println(file.getName());

}

}

# 扩展题

## 练习一:文件创建、判断功能、获取功能

描述:

键盘录入一个文件路径，根据文件路径创建文件对象，判断是文件还是文件夹

如果是文件，则输出文件的大小

如果是文件夹，则计算该文件夹下所有文件大小之和并输出(不包含子文件夹)。

答案

操作步骤:

1. 创建键盘录入对象
2. 接收键盘录入的字符串路径
3. 根据字符串路径创建文件对象
4. 判断文件对象是文件还是文件夹，如果是文件，则直接输出文件大小
5. 如果是文件夹，则获得该文件夹下所有的文件，定义一个求和变量，遍历文件数组获得每一个文件的大小并累加到求和变量中，最后输出求和变量的值。

代码:

**public class** Test02\_01 {

**public static void** main(String[] args) {

// 创建键盘录入对象

Scanner sc = **new** Scanner(System.***in***);

// 提示用户输入一个文件路径

System.***out***.println("请输入一个文件路径:");

// 接收用户输入的文件路径

String filePath = sc.nextLine();

// 根据文件路径创建文件对象

File file = **new** File(filePath);

// 判断是否是文件

**if**(file.isFile()) {

// 输出文件大小

System.***out***.println("文件大小为：" + file.length());

} **else** {

// 是文件夹

// 定义变量统计文件大小之和

**long**length = 0;

// 获得该文件夹下的所有文件

File[] files = file.listFiles();

// 遍历文件数组

**for** (File f :files) {

**if**(f.isFile()) {

// 累加文件大小

length += f.length();

}

}

// 输出文件夹文件的总大小

System.***out***.println("文件总大小为：" + length);

}

}

}

## 练习二:递归的使用(一)

描述:

用递归实现不死神兔

故事得从西元1202年说起，话说有一位意大利青年，名叫斐波那契。

在他的一部著作中提出了一个有趣的问题：假设一对刚出生的小兔一个月后就能长成大兔，

再过一个月就能生下一对小兔，并且此后每个月都生一对小兔，没有发生死亡，

问：现有一对刚出生的兔子2年后(24个月)会有多少对兔子?

答案

解题思路:

用递归法求斐波那契数列第n项

斐波那契数列，又称黄金分割数列，指的是这样一个数列：

1、1、2、3、5、8、13、21、……

这个数列大家很容易的就推算出来后面好几项的值，那么到底有什么规律，简单说，就是前两项的和是第三项的值

代码:

import java.util.Scanner;

public class Test02\_02 {

public static void main(String[] args) {

// 键盘输入并接收

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

// 调用递归方法并打印

System.out.println(fun(n));

}

public static int fun(int n) {

// 判断如果是第1个或者第2个月就返回1

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

return 1;

} else {

// 如果是第3个月就返回前两个月的值

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

}

}

}

## 练习三:递归的使用(二)

描述:猴子吃桃子问题，猴子第一天摘下若干个桃子，当即吃了快一半，还不过瘾，又多吃了一个。第二天又将仅剩下的桃子吃掉了一半，又多吃了一个。以后每天都吃了前一天剩下的一半多一个。到第十天，只剩下一个桃子。试求第一天共摘了多少桃子？

答案

解题思路:采用逆向思维，从后往前推断，发现其中有相同的地方，即出现递推公式，可以采用递归方法。

令S10=1,容易看出S9=2(S10+1),简化一下

S9=2S10+2

S8=2S9+2

…..

Sn=2S(n+1)+2

代码:

public class Test02\_03 {

public static void main(String[] args) {

// 定义桃子数量调用递归方法

int sum = sumPeach(1);

// 调用递归函数并打印

System.out.println(sum);

}

public static int sumPeach(int day) {

// 判断如果是第10天就返回1

if (day == 10) {

return 1;

} else {

// 递归计算前一天的数量

return 2 \* sumPeach(day + 1) + 2;

}

}

}

## 练习四:文件过滤器的使用

描述:

获得指定文件夹下所有的java文件(不考虑子文件夹的)并输出到控制台

答案

操作步骤:

1. 创建一个类实现文件过滤器FileFilter接口并重新accept方法，在该方法根据传入的文件判断是否文件Java文件，如果是则返回true，否则返回false。
2. 根据文件夹路径字符串创建文件对象和创建文件过滤器接口实现类对象
3. 调用文件对象的listFiles(FileFilter f) 方法，传递文件过滤器实现类对象。
4. 遍历文件数组，输入每一个文件对象

代码:

public class Test02\_04 {  
public static void main(String[] args) {  
// 创建文件夹对象  
File file= new File("d:/aaa");  
// 获得该文件夹下的所有java文件  
File[] files = file.listFiles(new FileFilter() {  
 public boolean accept(File pathname) {  
 // 判断文件后缀名是否.java文件  
 if (pathname.getName().endsWith(".java")) {  
 return true;  
 } else {  
 return false;  
 }  
 }

});  
// 遍历文件数组,打印所有的Java文件路径  
 for (File f : files) {  
 System.out.println(f.getAbsolutePath());  
 }  
 }  
}

## 练习五:文件和文件夹删除

描述:

键盘录入一个文件夹路径，删除该文件夹以及文件夹路径下的所有文件。

要求：录入的文件夹里面要有多个文件，不能包含有子文件夹。

提示：如果文件夹里面有文件，则需要先将文件删除才能删除文件夹。

答案

操作步骤:

1. 创建键盘录入对象Scanner
2. 定义字符串接收用户输入的文件夹路径
3. 根据文件夹路径创建文件对象
4. 调用文件对象的listFiles方法获得文件数组
5. 遍历文件数组，删除每一个文件
6. 最后调用删除方法删除文件夹。

代码:

**public class** Test02\_05 {

**public static void** main(String[] args) {

// 创建键盘录入对象

Scanner sc = **new** Scanner(System.***in***);

// 提示用户输入一个文件路径

System.***out***.println("请输入一个文件夹路径:");

// 接收用户输入的文件夹路径

String dirPath = sc.nextLine();

// 根据文件夹路径创建文件对象

File dir = **new** File(dirPath);

// 获得该文件夹下的所有文件

File[] files = dir.listFiles();

// 遍历文件数组删除每一个文件

**for** (File file :files) {

// 删除文件

file.delete();

}

// 删除文件夹

System.***out***.println(dir.delete()?"删除成功":"删除失败");

}

}

## 练习六:文件获取方法,递归的使用

描述:从键盘接收一个文件夹路径,获得该文件夹大小并输出到控制台。

答案

操作步骤:

1. 创建键盘录入对象Scanner
2. 定义字符串接收用户输入的文件夹路径
3. 根据文件夹路径创建文件对象
4. 定义一个方法calculate用来计算指定文件夹的大小，接收文件参数，返回long类型数值表示文件夹的大小。
5. 调用calculate方法传入文件夹对象，在该方法内部获得文件夹中所有文件，得到一个文件数组，定义一个变量size累加每一个文件的大小，遍历文件数组，判断是否是文件，如果是文件则获得文件大小并累加到变量size中，如果是文件夹，继续递归调用当前方法。

代码:

**public class** Test02\_06 {

**public static void** main(String[] args) {

// 创建键盘输入对象

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("请输入一个文件夹路径：");

// 接收用户输入的路径

String filePath = sc.nextLine();

// 根据路径字符串创建文件对象

File dir = **new** File(filePath);

// 调用calculate方法获得文件大小

**long** length = *calculate*(dir);

System.***out***.println("文件夹总大小：" + length);

}

/\*

\* 获得文件夹的大小

\*/

**public static long** calculate(File dir){

**long** length = 0;

// 获得文件数组

File files[] = dir.listFiles();

// 遍历数组

**for** (File file :files) {

// 判断是否是文件

if(file.isFile()) {

length += file.length();

} **else** {

length += *calculate*(file);

}

}

**return** length;

}

}