



山东大学

信息科学与工程学院

2022 – 2023 学年第一学期

实 验 报 告

课程名称: Java 编程技术

实验名称: Java 编程题

专 业 班 级 通信工程三班

学 生 学 号 202000120202

学 生 姓 名 李鑫

实 验 时 间 2022 年 10 月 1 日

实验报告

【实验目的】

1. 继续熟悉 Java 编程思想
2. 会使用各种循环和递归实现编程

【实验要求】

- 1、用 while 循环语句，计算 1--200 之间的所有 3 的倍数之和。
- 2、利用 switch 语句实现判断某年的某个月份有几天的程序。
- 3、水仙花数是指其个位、十位和百位上三个数的立方之和等于这个数本身。通过循环，判断 100-999 之间所有的数字，符合水仙花数条件的数字。

（例如： $1^3 + 5^3 + 3^3 = 153$ ）

- 4、已知 $XYZ + YZZ = 532$ ，其中 X、Y 和 Z 为数字，编程求出 X、Y 和 Z 的值。
- 5、编程实现“百钱买百鸡”问题。母鸡 5 分钱一只，公鸡三分钱一只，小鸡一分钱三只，现在有百钱欲买百鸡，有多少种买法？程序分别用三种方法来写，

第一方法是程序中有三层的循环，这个效率最低，第二方法是程序采用两层循环实现，第三方法是程序采用一层循环实现。

- 6、一个整数的各位数字之和能被 9 整除，则该数也能被 9 整除。验证这个定理的正确性。

注：验证从 0--2147483647 的整数中是否存在不符合该定理的数，若不存在即验证通过，验证时间可能需要 2-4 分钟，请耐心等待，但如果时间太长，务必思考为什么...

- 7、有一对兔子，从出生后第 3 个月起每个月都生一对兔子，小兔子长到第三个月后每个月又生一对兔子，假如兔子都不死，问 2 年后的兔子总数为多少？（尝试用递归编程实现）

【实验要求】

- 1、实验提交的程序代码中要加入注释，提高程序的阅读性，注意变量的命名规则，报告中对部分核心代码可以简单分析一下。
- 2、实验心得是自己实验过程的总结，可以帮助你再次分析你的问题，如果有帮助的资料也可以总结，方便以后查阅，学习语言的法宝：“多看，多练，多总结”。

【实验具体内容】

- 1、用 while 循环语句，计算 1--200 之间的所有 3 的倍数之和。

```
public class HomeWork1 {
    public static void main(String[] args) {
        // 累加变量定义
        int sum = 0;
        // 累加和判断
        for (int i = 1; i <= 200; i++) {
            if (i % 3 == 0) {
                sum += i;
            }
        }
        //输出最后的累加结果
        System.out.println("1-200 之间 3 的倍数的和为"+sum);
    }
}
```

编程思路：

本程序通过控制计数变量 i 在 for 循环中进行 1--200 的遍历，内部嵌入一个 if 判断语句 i 是否是 3 的倍数，如果满足倍数关系，则累加在 sum 变量上去，最后通过 println() 语句输出。

```

1  public class HomeWork1 {
2      public static void main(String[] args) {
3          int sum = 0;
4          for (int i = 1; i <= 200; i++) {
5              if (i % 3 == 0) {
6                  sum += i;
7              }
8          }
9          System.out.println(sum);
10     }
11 }
    
```

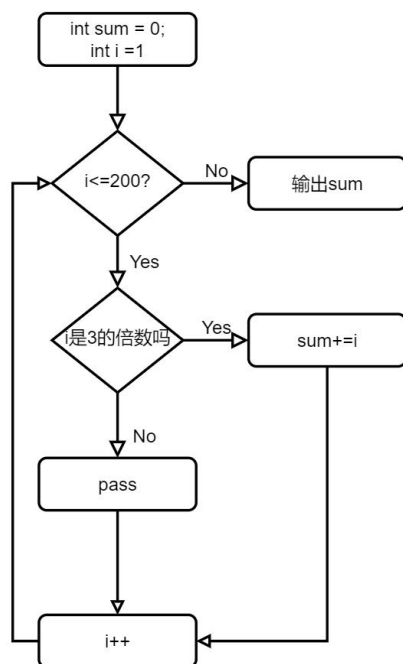
结果如下：

```

"C:\Program Files\Eclipse Foundation\jdk-8.0.302.8-hotspot\bin\java.exe" ...
1-200之间3的倍数的和为6633
    
```

进程已结束,退出代码0

流程图：



2、利用 *switch* 语句实现判断某年的某个月份有几天的程序。

```

import java.util.Scanner;

// 闰年的时候 2 月有 29 天，平年的时候 2 月有 28 天，除此之外，一月 31 天的有：1 3 5 7 8
// 10 12；一月 30 天的有 4 6 9 11

public class HomeWork2 {
    public static void main(String[] args) {
    
```

```

int inputyear;
int inputMonth;
Scanner input1 = new Scanner(System.in);
System.out.println("请输入年份:");
inputyear = input1.nextInt();
Scanner input = new Scanner(System.in); // 获取输入
System.out.println("请输入月份 (1~12): ");
// 判断输入数据
if (input.hasNextInt()) {
    inputMonth = input.nextInt(); // 赋值
    // 判断月份范围
    if ((inputyear % 4 == 0 && inputyear % 100 != 0) || inputyear %
400 == 0) {
        System.out.println(inputyear + "是闰年");
        if (inputMonth > 0 && inputMonth < 13) {
            switch (inputMonth) {
                case 1:
                case 3:
                case 5:
                case 7:
                case 8:
                case 10:
                case 12:
                    System.out.println(inputMonth + "月有 31 天");
                    break;
                case 4:
                case 6:
                case 9:
                case 11:
                    System.out.println(inputMonth + "月有 30 天");
                    break;
                case 2:
                    System.out.println(inputMonth + "月有 29 天");
                    break;
            }
        } else {
            System.out.println("你输入的月份超出范围 " + inputMonth);
        }
    } else {
        System.out.println(inputyear + "是平年");
        if (inputMonth > 0 && inputMonth < 13) {
            switch (inputMonth) {
                case 1:
                case 3:

```

```
        case 5:  
        case 7:  
        case 8:  
        case 10:  
        case 12:  
            System.out.println(inputMonth + "月有 31 天");  
            break;  
        case 4:  
        case 6:  
        case 9:  
        case 11:  
            System.out.println(inputMonth + "月有 30 天");  
            break;  
        case 2:  
            System.out.println(inputMonth + "月有 28 天");  
            break;  
    }  
} else {  
    System.out.println("你输入的月份超出范围 " + inputMonth);  
}  
  
}  
  
}
```

编程思路:

本题首先通过先确定年份是平年闰年，然后通过月份的规律：闰年的时候 2 月有 29 天，平年的时候 2 月有 28 天，除此之外，一月 31 天的有：1 3 5 7 8 10 12；一月 30 天的有 4 6 9 11，通过 switch 语句进行分类输出即可。

```
import java.util.Scanner;

// 闰年的时候2月有29天，平年的时候2月有28天，除此之外，一月31天的有：1 3 5 7 8 10 12；一月30天的有 4 6 9 11
public class HomeWork2 {

    public static void main(String[] args) {
        int inputyear;
        int inputMonth;
        Scanner input1 = new Scanner(System.in);
        System.out.println("请输入年份:");
        inputyear = input1.nextInt();
        Scanner input = new Scanner(System.in);// 获取输入
        System.out.println("请输入月份 (1~12) : ");
        // 判断输入数据
        if (input.hasNextInt()) {
            inputMonth = input.nextInt();// 赋值
            // 判断月份范围
            if ((inputyear % 4 == 0 && inputyear % 100 != 0) || inputyear % 400 == 0) {
                System.out.println(inputyear + "是闰年");
                if (inputMonth > 0 && inputMonth < 13) {
                    switch (inputMonth) {
                        case 1:
                        case 3:
                        case 5:
                        case 7:
                        case 8:
```

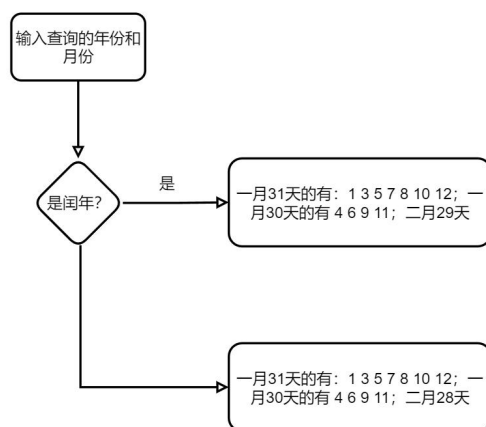
运行结果:

```

    HomeWork2 x
    "C:\Program Files\Eclipse Foundation\jdk-8.0.302.8-hotspot\bin\java.exe" ...
    请输入年份：
    2020
    请输入月份（1~12）：
    2
    2020是闰年
    2月有29天

    进程已结束,退出代码0
    
```

流程图：



3、水仙花数是指其个位、十位和百位上三个数的立方之和等于这个数本身。通过循环，判断100-999之间所有的数字，符合水仙花数条件的数字。

（例如： $1^3 + 5^3 + 3^3 = 153$ ）

实验代码：

```

public class HomeWork3 {
    /*注意一下程序入口main 方法的写法*/
    public static void main(String[] args) {
        int hundred, ten, one, sum;
        /*使用 for 指令进行循环*/
        for (int i = 100; i < 1000; i++) {
            /*百位数除以 100 求百位*/
            hundred = (i / 100);
            /*百位乘 100，再用 i 减去他，剩下的数除以 10 得到是的商是 i 的十位*/
            ten = (i - hundred * 100) / 10;
        }
    }
}
    
```



```

        /*和求十位一样的思路，全部减去再除以 1 得到的商就是 i 的各位*/
        one = i - hundred * 100 - ten * 10;
        /*个十百为分别求立方，然后相加，将结构存放在一个专门的变量中*/
        sum = (hundred * hundred * hundred) + (ten * ten * ten) + (one * one
* one);

        /*使用 if 语句进行判断输出*/
        if (sum == i) {
            /*若是各位的立方和和原来的数相等，就是水仙花，调用 out.println 方法实现输出*/

            System.out.println(i);
        }
    }
}
}

```

编程思路：

定义一个变量 i，进行从 100 到 999 的遍历，循环程序内部，通过提取 i 的百位十位个位，分别求立方加和赋值给 sum，之后通过 if 语句判定 sum 是否与 i 相等，若相等则为水仙花数，进行输出。

```

public class HomeWork3 {
    /*注意一下程序入口main 方法的写法*/
    public static void main(String[] args) {
        int hundred, ten, one, sum;
        /*使用 for 指令进行循环*/
        for (int i = 100; i < 1000; i++) {
            /*百位数除以100求百位*/
            hundred = (i / 100);
            /*百位乘 100，再用 i 减去他，剩下的数除以 10 得到的是的商是 i 的十位*/
            ten = (i - hundred * 100) / 10;
            /*和求十位一样的思路，全部减去再除以 1 得到的商就是 i 的各位*/
            one = i - hundred * 100 - ten * 10;
            /*个十百为分别求立方，然后相加，将结构存放在一个专门的变量中*/
            sum = (hundred * hundred * hundred) + (ten * ten * ten) + (one * one * one);
            /*使用if 语句进行判断输出*/
            if (sum == i) {
                /*若是各位的立方和和原来的数相等，就是水仙花，调用 out.println 方法实现输出*/
                System.out.println(i);
            }
        }
    }
}

```

运行结果：

```
"C:\Program Files\Eclipse Foundation\jdk-8.0.302.8-hotspot\bin\java.exe" ...
```

```
153
```

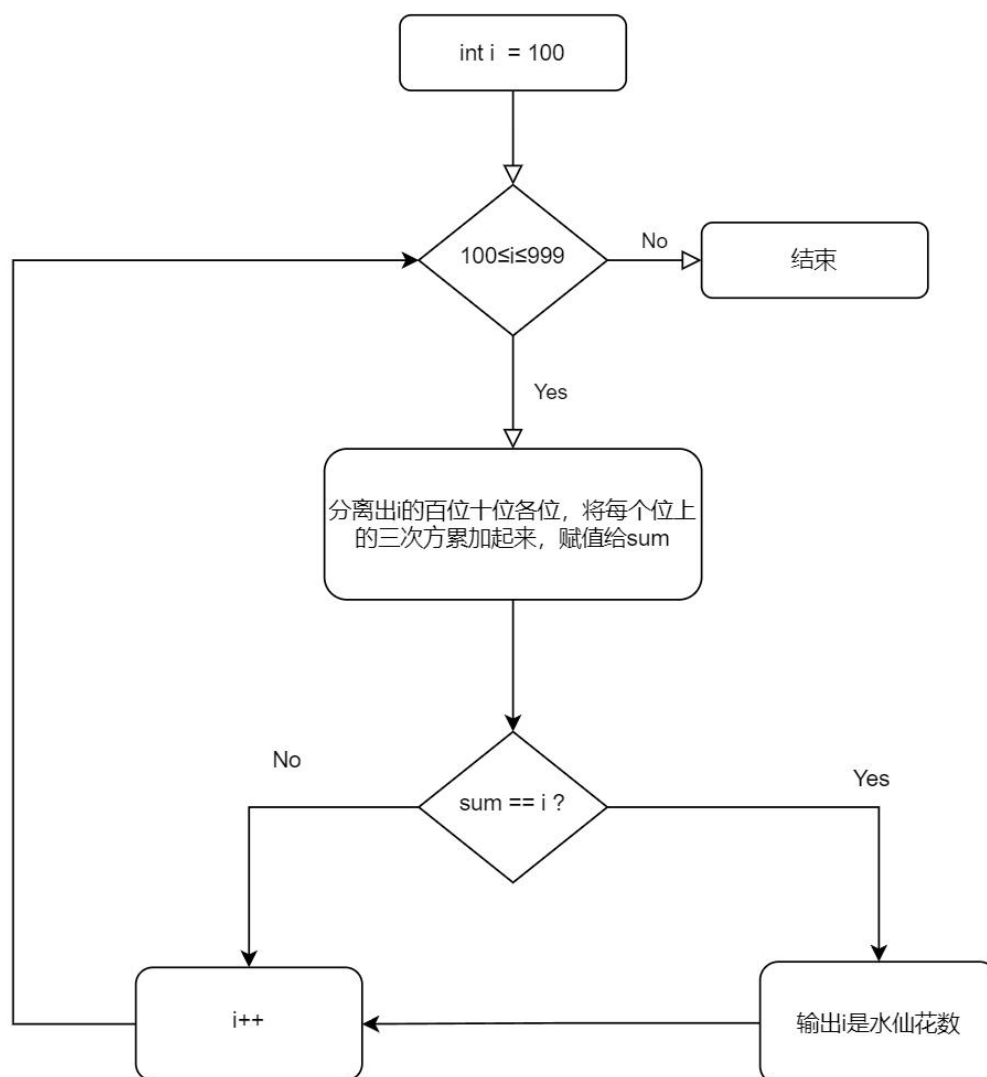
```
370
```

```
371
```

```
407
```

```
进程已结束,退出代码0
```

流程图:



4、已知 $XYZ + YZZ = 532$ ，其中 X 、 Y 和 Z 为数字，编程求出 X 、 Y 和 Z 的值。

```
public class HomeWork4 {
```

```

public static void main(String[] args) {
    int X,Y,Z;
    for(X=1;X<10;X++) {
        for(Y=1;Y<10;Y++) {
            for(Z=0;Z<10;Z++) {
                /*每取一个值，就要判断是否满足要求*/
                if(X*100+Y*10+Z+Y*100+Z*10+Z==532)
                    System.out.println("X = "+X+" Y = "+Y+" Z = "+Z);
            }
        }
    }
}

```

实验思路：

由于三个数都要确定，所以需要通过三重遍历来实现，之后将百十个位上的数分别乘以 100、10、1，然后累加起来，判断如果等于 532，则可输出（如果确定只有一个解，可以提前 break 结束所有循环）

运行结果：

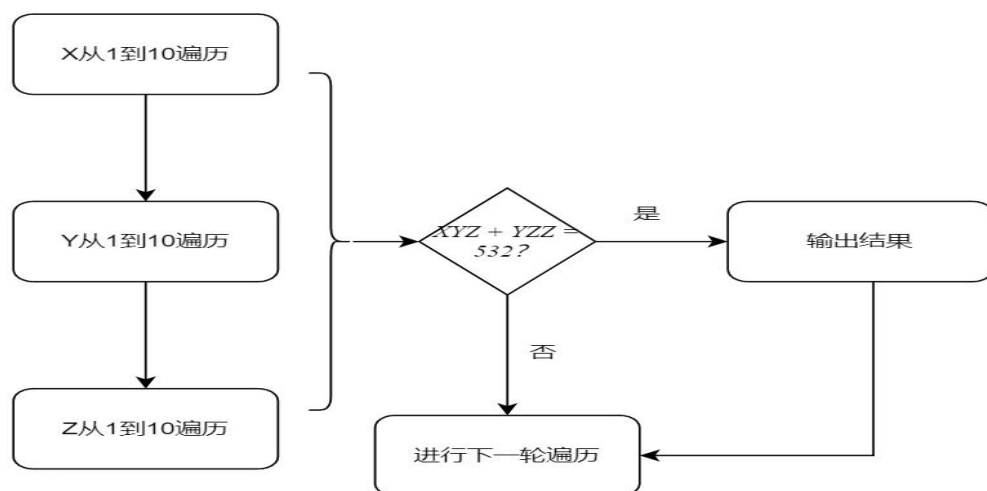
```

"C:\Program Files\Eclipse Foundation\jdk-8.0.302.8-hotspot\bin\java.exe" ...
X = 3 Y = 2 Z = 1

```

进程已结束,退出代码0

流程图：



5、百元百鸡

C#实现版本:

方式一：未知数为鸡的数目

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace chicken1

{

    class Chook100

    {

        static void Main(string[] args)

        {

            int x, y, z;

            x = y = z = 0;

            for (x = 1; x <= 100 / 5; x++)

                // 设 x 为公鸡数目, y 为母鸡数目, z 为小鸡数目

                for (y = 1; y <= 100 / 3; y++)
```

```

        {

            z = 100 - x - y;

            if (z % 3 == 0 && 5 * x + 3 * y + z / 3 == 100)

                goto end; // 直接从内循环跳出，跳了二层循环，这是 break 语句做不到的

        }

        end: Console.WriteLine("Cock={0}  Hen={1}  Chick={2}", x, y, z);

        // 满足上述的若干个约束条件分别则输出公鸡、母鸡、小鸡的数目

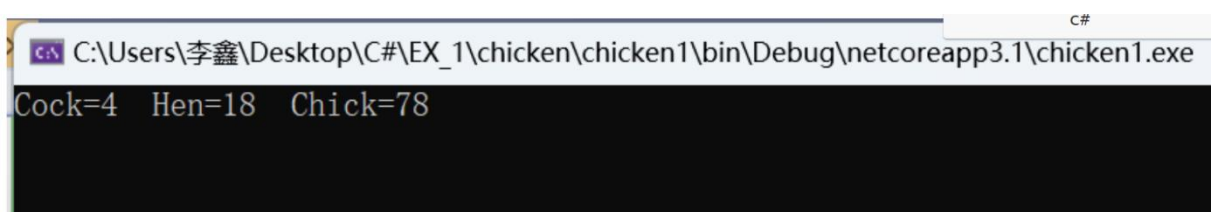
        Console.Read();

    }

}

}

```



方式二：设未知数为所买鸡的钱的数目

```

using System;

using System.Collections.Generic;

using System.Linq;

```

```

using System.Text;

using System.Threading.Tasks;

namespace Ex2_10
{
    class Chook100
    {
        static void Main(string[] args)
        {
            int x, y, z;

            //设 x 为买公鸡所用的钱数, y 和 z 分别为买母鸡和小鸡所用的钱数

            x = y = z = 0;

            for (x = 1; x <= 100; x++)

                for (y = 1; y <= 100; y++)

                    {

                        z = 100 - x - y;

                        if (x / 5 + y / 3 + 3 * z == 100 && x % 5 == 0 && y %
3 == 0)

                            goto end; // 直接从内循环跳出, 跳了二层循环, 这是 brea
    
```

k 语句做不到的

```

        }

        end: Console.WriteLine("Cock={0}   Hen={1}   Chick={2}", x / 5, y / 3,
3 * z);

// 满足上述的若干个约束条件分别则输出公鸡、母鸡、小鸡的数目

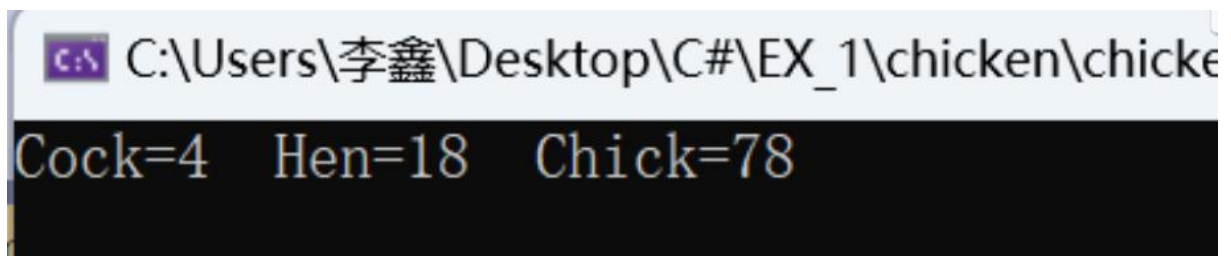
        Console.Read();

    }

}

}

```



方式三：一重循环

```

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

```

```
using System.Threading.Tasks;

namespace ConsoleApp99
{
    class Program
    {
        static void Main(string[] args)
        {
            int x, y, z;

            // 设 x 为公鸡数目, y 为母鸡数目, z 为小鸡数目

            x = y = z = 0;

            int count = 0;

            for (int k = 1; k <= 3; k++)
            {
                count++;

                x = 4 * k;

                y = 25 - 7 * k;

                z = 75 + 3 * k;
            }
        }
    }
}
```


//上述三条表达式由数学关系 $x+y+z==100$ 和 $x / 5 + y / 3 + 3 * z == 10$ 导出

```
        Console.WriteLine("Cock: " + x + "只, Hen: " + y + "只, Chick: " + z + "只");
```

// 满足上述的若干个约束条件分别则输出公鸡、母鸡、小鸡的数目

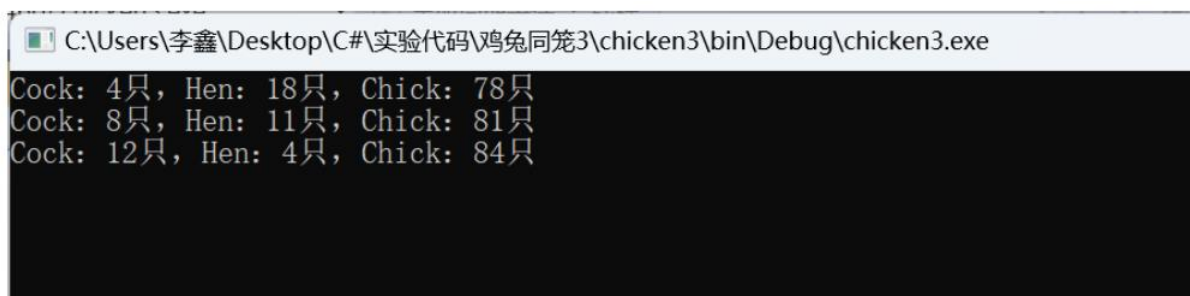
```
    }
```

```
    Console.Read();
```

```
}
```

```
}
```

```
}
```



```
C:\Users\李鑫\Desktop\C#\实验代码\鸡兔同笼3\chicken3\bin\Debug\chicken3.exe
Cock: 4只, Hen: 18只, Chick: 78只
Cock: 8只, Hen: 11只, Chick: 81只
Cock: 12只, Hen: 4只, Chick: 84只
```

Java 版本:

第一种: 三层循环

```
public class Chicken1 {
    public static void main(String[] args)
    {
```

```

int x, y, z;

x = y = z = 0;

for (x = 1; x <= 100 / 5; x++)
    // 设 x 为公鸡数目, y 为母鸡数目, z 为小鸡数目
    {for (y = 1; y <= 100 / 3; y++)
        {
            z = 100 - x - y;

            if (z % 3 == 0 && 5 * x + 3 * y + z / 3 == 100)
                continue;
        }

        continue;
    }

System.out.println("Cock="+x +" \t Hen=" +y +" \t Chick="+z );

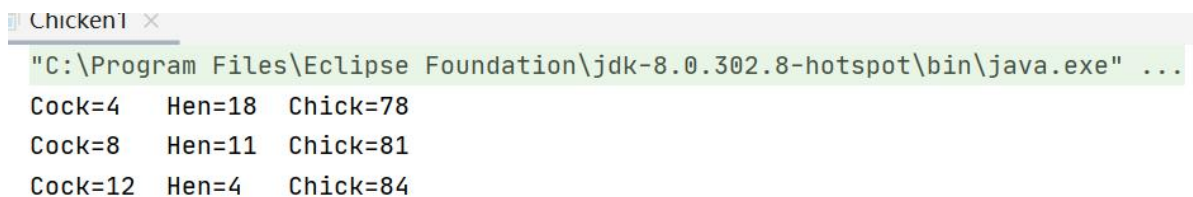
// 满足上述的若干个约束条件分别则输出公鸡、母鸡、小鸡的数目

}

}
    
```

实现思路:

本题是一种整数规划问题，一共有三个未知数，但是只有两个方程，所以可以通过利用一个未知数来表达其他两个未知数，这和我们线性代数里面的基底向量是有一定的相同之处的。



```

Chicken1 x
"C:\Program Files\Eclipse Foundation\jdk-8.0.302.8-hotspot\bin\java.exe" ...
Cock=4   Hen=18   Chick=78
Cock=8   Hen=11   Chick=81
Cock=12  Hen=4    Chick=84
    
```

如果需要输出所有结果，可以使用下面的程序:

```

public class Chicken1 {
    public static void main(String[] args)
    {
    
```

```

int x, y, z;

x = y = z = 0;

for (x = 1; x <= 100 / 5; x++)
    // 设 x 为公鸡数目, y 为母鸡数目, z 为小鸡数目
    {for (y = 1; y <= 100 / 3; y++)
        {
            z = 100 - x - y;
            if (z % 3 == 0 && 5 * x + 3 * y + z / 3 == 100)
                System.out.println("Cock="+x +"\\t Hen=" +y +"\\t Chick="+z );
        }
    }
}
}

```

本程序直接基于 C#第一个程序改版而来，通过三层循环，分别确定满足条件的 X、Y、Z 的值。

实验结果：

```

能买4公 12母84小鸡
能买11公 8母81小鸡
能买18公 4母78小鸡
能买25公 0母75小鸡

```

第二种:三层循环

```

public class Chicken2 {

```

```

public static void main(String[] args) {
    int Male, Female, Chick=0;
    /*三层循环，遍历每一个可能的取值*/
    for (Male = 0; Male <= 100; Male++) {
        for (Female = 0; Female <= 100; Female++) {
            for (Chick = 0; Chick <= 100; Chick++) {
                /*使用 if 语句输出满足条件的所有组合，首先是三者的数量加起来是 100 只*/ /*其次是三种鸡的钱加起来是 100 块*/
                if (Male + Female + Chick == 100) {
                    /*在这里需要注意的是，java 中的除法是取整除法，即只保留商，舍去余数，那这样的话不能保证所花费的钱正好是 100，所以需要增加一个限定条件，即除法所得即完整的结果，即除以 3 没有余数*/
                    if (5 * Female + 3 * Male + Chick / 3 == 100 && Chick % 3 == 0)
                        /*再复习一下，out 是 System 类的成员变量，同时也是 PrintStream 类的对象，多以通过他调用 println 方法以实现输出，并且通过+将字符串连接起来输出*/
                        System.out.println("能买" + Male + "公 " + Female + "母" + Chick + "小鸡"); }
            }
        }
    }
}

```

实现思路：

本题是一种整数规划问题，一共有三个未知数，但是只有两个方程，所以可以通过利用一个未知数来表达其他两个未知数，这和我们线性代数里面的基底向量是有一定的相同之处的。

实验结果：

```
"C:\Program Files\Eclipse Foundation\jdk-8.0.302.8-hotspot\bin\jav
能买25公 0母75小鸡
能买18公 4母78小鸡
能买11公 8母81小鸡
能买4公 12母84小鸡

进程已结束,退出代码0
```

第三种：

```
public class Chicken3 {
    public static void main(String[] args) {
        /*依旧是建立三个整型变量分别存放公鸡、母鸡和小鸡的数量*/
        int Male, Female, Chick = 0;
        /*选择用母鸡的个数分别表示出公鸡和小鸡的个数，这也就对循环时的终点做
        限制，即 不能大于 100/5，以保证求出来的为正整数 */
        for (Female = 0; Female <= 100 / 5; Female++) {
            Male = (100 - 7 * Female) / 4;
            Chick = (300 + 3 * Female) / 4;
            /*满足 if 语句的三个数值将会被输出，还是一定注意，除法不能有余数*
            /

            if ((Chick % 3 == 0) && Male > 0 && (Male + Female + Chick ==
            100) && (5 * Female + 3 * Male + Chick /
```

```

        3 == 100))

        System.out.println("能买" + Male + "公" + Female + "母" + Chick
        + "小鸡");
    }
}
}
}

```

实验思路：

除了用另外一个未知数表示另外两个以外，我们还可以通过引入新的变量，通过让新的变量表示我们的未知数，这样的话，因为的参变量的范围成了唯一的问题，这也就使得我们的程序循环可以减少两层

实验结果：

```
"C:\Program Files\Eclipse Foundation\jdk-8.0.302.8-hotspot\bin\java.exe" ...
```

```
能买25公0母75小鸡
```

```
能买18公4母78小鸡
```

```
能买11公8母81小鸡
```

```
能买4公12母84小鸡
```

```
进程已结束,退出代码0
```

6、一个整数的各位数字之和能被 9 整除，则该数也能被 9 整除。验证这个定理的正确性。

```

public class HomeWork6 {
    public static void main(String args[]) {

        /*声明两个整形的成员变量，并实现了自动装箱*/
        int number1;
        int number2;
        /*遍历所有的数，并对每一个数进行验证*/
        for (int i=0;i<2147483647;i++) {
            /*将该数显赋给变量在执行操作，防止对循环变量产生影响，从而导致错误的循环*/

```

```

        number1=i;
        number2=0;
        /*使用 while 控制循环取每一位的数,只要该数不是个位数,那就还可以提取低位,办法
        就是除以 10 取余,若是个位数,则直接参与求和即可*/
        while(number1>9){
            /*除以 10 取余,即从最低位开始取数*/
            /*使用该容器进行累加每一位*/
            number2 +=number1%10;
            /*进行取余操作后,原本的数不会发生改变,所以要自己把低位去掉*/
            /*取完低位之后,将这个数除以 10,因为 java 中的除法是取整,所以该操作就把
            低位的 数去掉了,进入下次循环的时候,就是取次低位的数*/
            number1=number1/10;
        }
        /*number2 中存放的是从次低位到个位每一位的数,剩下的不能进入 while 循环的一
        位 数就是原数的最高位*/
        number2=number2+number1;
        /*得到各位数的和之后,进行除法操作*/
        if(number2%9==0) {
            if(i%9!=0) {

                System.out.println("该数据"+i+"不符合定理");

            }
        }
        System.out.println("运行完毕,符合定理");
    }
}

```

实验思路：遍历整个范围，然后每取一个数，就把他的各位求和，然后能被 9 整除时，就再判断该 数能否被整除。

实验截图：

```

1  public class HomeWork6 {
2      public static void main(String args[]) {
3
4
5          /*声明两个整形的成员变量，并实现了自动装箱*/
6          int number1;
7          int number2;
8          /*遍历所有的数，并对每一个数进行验证*/
9          for (int i=0;i<2147483647;i++) {
10             /*将该数显赋给变量在执行操作，防止对循环变量产生影响，从而导致错误的循环*/
11             number1=i;
12             /*第二次报告也提到过，出现累加的话就需要一个容器*/
13             number2=0;
14             /*使用 while 控制循环取每一位的数，只要该数不是个位数，那就还可以提取低位，办法就是除以 10 取余，若是个位数，则直接参与求和即可*/
15             while(number1>9){
16                 /*除以 10 取余，即从最低位开始取数*/
17                 /*使用该容器进行累加每一位*/
18                 number2 +=number1%10;
19                 /*进行取余操作后，原本的数不会发生改变，所以要自己吧低位去掉*/
20                 /*取完低位之后，将这个数除以 10，因为java 中的除法是取整，所以该操作就把低位的 数去掉了，进入下次循环的时候，就是取次低位的数*/
21                 number1=number1/10;
22             }
23             /*number2 中存放的是从次低位到个位每一位的数，剩下的不能进入 while 循环的一位 数就是原数的最高位*/
24             number2=number2+number1;
25             /*得到各位数的和之后，进行除法操作*/
26             if(number2%9==0) {
27                 if(i%9!=0) {
28                     System.out.println("该数据"+i+"不符合定理");
29                 }
30             }
31         }
32     }
33 }
34 System.out.println("运行完毕，符合定理");
35 }
36
37 }
38
    
```

实验结果：

```

HomeWork6 x
"C:\Program Files\Eclipse Foundation\jdk-8.0.302.8-hotspot\bin\java.exe" ...
运行完毕，符合定理

进程已结束,退出代码0
    
```

7、有一对兔子，从出生后第 3 个月起每个月都生一对兔子，小兔子长到第三个月后每个月又生一对兔子，假如兔子都不死，问 2 年后的兔子总数为多少？（尝试用递归编程实现）

```

8、    public class HomeWork7 {
        public static void main(String[] args) {
            /*定义整型数据，并实现自动的装箱，用来指示两年即 24 个月*/
            int i = 24;
        }
    }
    
```



```

        /*调用 sum 方法进行递归计算，并且使用+将各种要输出的参数连接起来*/
        System.out.println("2 年后的兔子总数为" + sum(i)+"对");
    }

    /*借鉴 main 方法的修饰词，为了省事，于是将方法写成公共的，并且是被指向的即 每一个调用它的变量都能对他进行修改*/
    /*形参为整形变量 month，用于接收调用方法时传递过来的实参 i*/
    public static int sum(int month) {
        /*使用 if 语句进行判断，一开始只有一对兔子，并且第三个月才开始生兔子，所以月份

        如果是 1 或 2 的话，就是只有 1 对兔子*/
        if (month == 1 || month == 2)
            return 1;
        /*若是 3 个月以上，那么该月的兔子对数就是前两个月的对数之和*/
        else
            /*递归就是利用前面的结果，通过同样的方程（也就是方法）去求下一个数*/
            return sum(month - 1) + sum(month - 2);
        }
    }
}

```

实验思路：

通过递归的方式，在函数内部使用函数，得到斐波那契数列的前 n 项和。

实验截图

```

public class HomeWork7 {
    public static void main(String[] args) {
        /*定义整型数据，并实现自动的装箱，用来指示两年即24 个月*/
        int i = 24;
        /*调用sum 方法进行递归计算，并且使用+将各种要输出的参数连接起来*/
        System.out.println("2 年后的兔子总数为" + sum(i)+"对");
    }
    /*借鉴 main 方法的修饰词，为了省事，于是将方法写成公共的，并且是被指向的即 每一个调用它的变量都能对他进行修改*/
    /*形参为整形变量 month，用于接收调用方法时传递过来的实参 i*/
    public static int sum(int month) {
        /*使用if 语句进行判断，一开始只有一对兔子，并且第三个月才开始生兔子，所以月份
        如果是 1 或 2 的话，就是只有 1 对兔子*/
        if (month == 1 || month == 2)
            return 1;
        /*若是 3 个月以上，那么该月的兔子对数就是前两个月的对数之和*/
        else
            /*递归就是利用前面的结果，通过同样的方程（也就是方法）去求下一个数*/
            return sum( month: month_ - 1) + sum( month: month - 2);
    }
}

```

实验结果：

```

"C:\Program Files\Eclipse Foundation\jdk-8.0.302.8-hotspot\bin\java.exe" ...
2 年后的兔子总数为46368对

进程已结束,退出代码0

```

【思考题】

在一个 class 里面是不是一定要有 main？如果没有 main，那这个 class 的入口地址是什么，在哪呢？

回答：

不是的,main 是程序的入口, java 程序是以类为基础实现的, 每个项目都要有一个主类, 这个主类中必须得有 main, 用于程序的入口. 在程序测试时, 一般每个类中都有一个 main, 用于方便测试人员对类成员进行测试

如果没有 main，那这个 class 的入口地址是主类的 main。

JAVA 中的斜杠

有正斜杠与反斜杠之分, 正斜杠, 一般就叫做斜杠, 符号为 “/” ;反斜杠的符号为“\”。

斜杠(/)在 JAVA 中没有什么特别的意义, 就是代表一个字符/;

反斜杠(\)则不然, 它和紧跟着它的那个字符构成转义字符, 如 “\n” (换行)、\” (字符”)等, 所以在字符串中要表示字符\要用“\”来表示, 例:如果你这样定义一个字符串

`String s = "name\\sex"`是错误的, 要这样定义 `String s = "name\\\\sex"`;

注意:在正则表达式中的`\`表示和后面紧跟着的那个字符构成一个转义字符(姑且先这样命名), 代表着特殊的意义;所以

如果你要在正则表达式中表示一个反斜杠`\`, 应当写成`\\`。如果你这样获得一个 `Matcher`, `Matcher m=`

`Pattern.compile("\\").matcher("I\\")` 将会报错, 你应当这样写 `Matcher m = Pattern.compile("\\\\").matcher("\\")` 才是正确且匹配的

Java 的内存机制

Java 把内存划分成两种:一种是栈内存, 另一种是堆内存。在函数中定义的一些基本类型的变量和对象的引用变量都是在函数的栈内存中分配, 当在一段代码块定义一个变量时, Java 就在栈中为这个变量分配内存空间, 当超过变量的作用域后(比如, 在函数 A 中调用函数 B, 在函数 B 中定义变量 a, 变量 a 的作用域只是函数 B, 在函数 B 运行完以后, 变量 a 会自动被销毁。分配给它的内存会被回收), Java 会自动释放掉为该变量分配的内存空间, 该内存空间可以立即被另作它用。

堆内存用来存放由 `new` 创建的对象和数组, 在堆中分配的内存, 由 Java 虚拟机的自动垃圾回收器来管理。在堆中产生了一个数组或者对象之后, 还可以在栈中定义一个特殊的变量, 让栈中的这个变量的取值等于数组或对象在堆内存中的首地址, 栈中的这个变量就成了数组或对象的引用变量, 以后就可以在程序中使用栈中的引用变量来访问堆中的数组或者对象, 引用变量就相当于为数组或者对象起的一个名称。引用变量是普通的变量, 定义时在栈中分配, 引用变量在程序运行到其作用域之外后被释放。而数组和对象本身在堆中分配, 即使程序运行到使用 `new` 产生数组或者对象的语句所在的代码块之外, 数组和对象本身占据的内存不会被释放, 数组和对象在没有引用变量指向它的时候, 才变为垃圾, 不能在堆中使用, 但仍然占据内存空间不放, 在随后的一个不确定的时间被垃圾回收器收走(释放掉)。这也是 Java 比较占内存的原因, 实际上, 栈中的变量指向堆内存中的变量, 这就是 Java 中的指针!

【实验心得】

本次实验之前, 我花了半天的时间看了廖雪峰的 Java 教程, 内容包括下图中的知识, 截止到我完成实验的时候, 类和方法还没有深入进去, 本次实验大多都是通过流控实现的, 相对来说有点偏向于 C 语言的处理方式, 没有使用到 JAVA 最核心的面向对象编程, 当然原因是因为截至完成时的我的个人能力还不够。本次实验主要对于各种类型的算法题进行了训练, 其中百元百鸡程序采用了两种语言编写, 并分别给出了三种方法及其对应的原理, 由于时间原因, 没有采用 python 编程实现, 但是相对来说 python 的简单程度是我认为三个语言里面最为简洁的, 也有可能是我对 JAVA 和 C# 中的部分函数没有良好的掌握, 但是我认为随着本套课程的进行, 我的三门语言能力都会有所上升。

T Java笔记.md - Typora

文件(F) 编辑(E) 段落(P) 格式(O) 视图(V) 主题(T) 帮助(H)

文件

大纲

Java笔记

参考资料

> JAVA优势及名词解释

> 第一个Java程序

Java程序基础

Java程序基本结构

> 变量和数据类型

var关键字

变量的作用范围

> 整数运算

> 浮点数运算

> 字符和字符串

数组类型

字符串数组

> 流程控制

> 数组操作

廖雪峰 JAVA教程

JAVA优势及名词解释

为什么Java应用最广泛?

从互联网到企业平台，Java是应用最广泛的编程语言，原因在于：

- Java是基于JVM虚拟机的跨平台语言，一次编写，到处运行；
- Java程序易于编写，而且有内置垃圾收集，不必考虑内存管理；
- Java虚拟机拥有工业级的稳定性和高度优化的性能，且经过了长时期的考验；
- Java拥有最广泛的开源社区支持，各种高质量组件随时可用。

Java的版本

随着Java的发展，SUN给Java又分出了三个

< </>

16979 词