

## 五、编程题

266、1 已知 Linux 系统为 CentOS7 系统,已知 hadoop 启动目录/simple/hadoop2.7.3/sbin,请依次写出命令。启动 hadoop、查看 hadoop 进程是否全部启动、递归查看当前 HDFS 有那些文件、在 HDFS 的根目录创建 input 目录、在本地/simple 目录下创建文件 data.txt 并上传到 HDFS 的/input 目录下、将集群上的/input/data.txt 下载到/appdata(该目录已存在)、最后关闭 hadoop

```
[root@master /]#cd /simple/hadoop2.7.3/sbin
```

```
[root@master sbin]#start-all.sh
```

```
[root@master sbin]#jps
```

```
[root@master sbin]#hadoop fs -ls -R /
```

```
[root@master sbin]#hadoop fs -mkdir /input
```

```
[root@master sbin]#cd /simple
```

```
[root@master simple]#touch data.txt
```

```
[root@master simple]#hadoop fs -put data.txt /input
```

```
[root@master simple]#hadoop fs -get /input/data.txt /appdata
```

```
[root@master simple]#cd /simple/hadoop2.7.3/sbin
```

```
[root@master sbin]#stop-all.sh
```

267、已知当前 hadoop 已全部正常启动,且 HDFS 的根目录下不存在 hdfs-test 的目录,用到的 ip 和端口写 192.168.1.26:9000

使用 java api,在 HDFS 的根目录下,创建名为 hdfs-test 的目录

```
import org.apache.hadoop.fs.FileSystem;
```

```
import java.net.URI;
```

```
import org.apache.hadoop.conf.Configuration;
```

```

import org.apache.hadoop.fs.Path;

import java.io.IOException;

public class MakeDir{

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

        FileSystem fs = FileSystem.get(new URI("hdfs://192.168.1.26:9000"), new
Configuration(), "root");

        Boolean flag = fs.mkdirs(new Path("/hdfstest"));

        System.out.println(flag?"创建成功":"创建失败");

    }

}

```

**268、**问题:编写一个程序，该程序接受控制台以逗号分隔的数字序列，并生成包含每个数字的列表和元组。假设向程序提供以下输入:

34 岁,67 年,55 岁,33 岁,12 日,98 年

则输出为:['34', '67', '55', '33', '12', '98']

( '34', '67', '55', '33', '12', '98')

提示:在为问题提供输入数据的情况下，应该假设它是控制台输入。方法可以将列表转换为元组

解决方案:

```

import re

print('请输入一组数字: ')

values=input()

l=values.split(",")

k=re.findall(r'[0-9]+',values)

t=tuple(k)

print (k)

print (t)

```

269、请在下面程序的下划线中补充完整程序（共 8 处）。

```
public class WordCount {
    public static class TokenizerMapper extends
        Mapper<__Object____, __Text __, __Text____, __IntWritable____> {
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();
        public void map(LongWritable key, Text value, Context context) {
            StringTokenizer itr = new StringTokenizer(value.toString());
            while (itr.hasMoreTokens()) {
                word.set(itr.nextToken());
                context.write(word, one);
            }
        }
    }

    public static class IntSumReducer extends
        Reducer< __Text____, __IntWritable____, Text, IntWritable> {
        private IntWritable result = new IntWritable();
        public void reduce( __Texy____ key, Iterable< __IntWritable____> values, Context context)
        {
            int sum = 0;
            for (IntWritable val : values) {
                sum += val.get();
            }
            result.set(sum);
            context.write(key, result);
        }
    }

    public static void main(String[] args) throws Exception {
        略.....
    }
}
```

270、某个公司采用公用电话传递数据，数据是四位的整数，在传递过程中是加密的，加密规则如下：每位数字都加上 5,然后用和除以 10 的余数代替该数字，再将第一位和第四位交换，第二位和第三位交换。

```
public class Prog48{

    public static void main(String[] args){

        int n = 1234;

        int[] a = new int[4];
```

```

        for(int i=3;i>=0;i--){

            a[i] = n%10;

            n /= 10;

        }

        for(int i=0;i<4;i++)

            System.out.print(a[i]);

        System.out.println();

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

            a[i] += 5;

            a[i] %= 10;

        }

        int temp1 = a[0];

        a[0] = a[3];

        a[3] = temp1;

        int temp2 = a[1];

        a[1] = a[2];

        a[2] = temp2;

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

            System.out.print(a[i]);

    }}

```

**271**、海滩上有一堆桃子，五只猴子来分。第一只猴子把这堆桃子凭据分为五份，多了一个，这只猴子把多的一个扔入海中，拿走了一份。第二只猴子把剩下的桃子又平均分成五份，又多了一个，它同样把多的一个扔入海中，拿走了一份，第三、第四、第五只猴子都是这样做的，问海滩上原来最少有多少个桃子？

```

public class Prog41{

    public static void main(String[] args){

        int n;

        n = fun(0);

        System.out.println("原来有"+n+"个桃子");

    }

    private static int fun(int i){

        if(i==5)

            return 1;

        else

            return fun(i+1)*5+1;

    }}

```

272、编写一个函数，输入 n 为偶数时，调用函数求  $1/2+1/4+\dots+1/n$ ,当输入 n 为奇数时，调用函数  $1/1+1/3+\dots+1/n$ (利用指针函数)

```

import java.util.Scanner;public class Prog39{

    public static void main(String[] args){

        System.out.print("请输入一个整数: ");

        Scanner scan = new Scanner(System.in);

        int n = scan.nextInt();

        scan.close();

        if(n%2==0)

            System.out.println("结果: "+even(n));

        else

```

```

        System.out.println("结果: "+odd(n));
    }

    //奇数

    static double odd(int n){

        double sum = 0;

        for(int i=1;i<n+1;i+=2){

            sum += 1.0/i;

        }

        return sum;

    }

    //偶数

    static double even(int n){

        double sum = 0;

        for(int i=2;i<n+1;i+=2){

            sum += 1.0/i;

        }

        return sum;

    }

}

```

273、输入数组，最大的与第一个元素交换，最小的与最后一个元素交换，输出数组。

```

import java.util.Scanner;public class Prog35{

    public static void main(String[] args){

        System.out.print("请输入一组数: ");

        Scanner scan = new Scanner(System.in).useDelimiter("\\s");
    }
}

```

```

int[] a = new int[50];

int m = 0;

while(scan.hasNextInt()){

    a[m++] = scan.nextInt();

}

scan.close();int[] b = new int[m];

for(int i=0;i<m;i++)

    b[i] = a[i];

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

    for(int j=0;j<b.length-i-1;j++)

        if(b[j]<b[j+1]){

            int temp = b[j];

            b[j] = b[j+1];

            b[j+1] = temp;

        }

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

    System.out.print(b[i]+" ");

}}

```

**274、** 有 n 个孩子站成一圈，从第一个孩子开始顺时针方向报数，报到 3 的人出列，下一个人继续从 1 报数，直到最后剩下一个孩子为止。问剩下第几个孩子。下面的程序以 10 个孩子为例，模拟了这个过程，请完善之（提示：报数的过程被与之逻辑等价的更容易操作的过程所代替）。

```
Vector a = new Vector();
```

```

for(int i=1; i<=10; i++)

{

a.add("第" + i + "个孩子");

}

for(;;)

{

if(a.size()==1) break;

for(int k=0; k<2; k++)

_____

a.remove(0);

}

System.out.println(a);

```

结果：

[第 4 个孩子]

填写：

```
a.add(a.remove(0))
```

过程：

每一次把更新的序列的第一个和第二个丢到后面，然后接下来的第三个给删去……………

代码：

```
import java.util.Scanner;
```

```
import java.util.Vector;
```

```
public class Main {
```

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

```
        Vector a = new Vector();
```



```

        for(int i=1; i<=10; i++)

        {

            a.add("第" + i + "个孩子");//赋值

        }

        for(;;)

        {

            if(a.size()==1) break;//剩下最后一个孩子

            //remove 返回值为移除的元素,add 把元素添加向量的末尾

            for(int k=0; k<2; k++)//先把前面的两个元素放在后面

                a.add(a.remove(0));//填空

            a.remove(0);//再把第三个元素给删除了

        }

        System.out.println(a);

    }

}

```

275、下列代码运行结果为：

12345

23456

89

23456789

即把一个串从数字不连续的位置断开。试完善之。

```
String s = "12345234568923456789";
```

```
String t = "1";
```

```
for(int i=1; i<s.length(); i++)
```

```
{
```

```

if(s.charAt(i)==s.charAt(i-1)+1)

{

t += s.charAt(i);

}

else

{

System.out.println(t);

_____

}

}

System.out.println(t);

```

结果：

12345

23456

89

23456789

填写：

```
t += s.charAt(i)
```

代码如下：

```
import java.util.Scanner;
```

```
import java.util.Vector;
```

```
public class Main {
```

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

```
        String s = "12345234568923456789";
```

```

String t = "1";

for(int i=1; i<s.length(); i++)

{

    if(s.charAt(i)==s.charAt(i-1)+1)//如果是连续的

    {

        t += s.charAt(i);//子串

    }

    else//如果是不连续,清除 t 字符串，为下一次做准备

    {

        System.out.println(t);

        t="" +s.charAt(i);//填空

    }

}

System.out.println(t);

}

}

```

**276、**很多人都玩过这个游戏：甲在心中想好一个数字，乙来猜。每猜一个数字，甲必须告诉他猜大了，猜小了，还是刚好猜中了。下列的代码模拟了这个过程。其中用户充当甲的角色，计算机充当乙的角色。为了更快地猜中，计算机使用了二分法。

阅读分析代码，填写缺失的部分。

把填空的答案（仅填空处的答案，不包括题面）存入考生文件夹下对应题号的“解答.txt”中即可。

```
System.out.println("请在心中想好一个数字（1~100），我来猜");
```

```
System.out.println("我每猜一个数字，你要告诉我是“猜大了”，“猜小了”，还是“猜中”");
```

```
Scanner scan = new Scanner(System.in);
```

```

int v1 = 1;

int v2 = 100;

for(;;)

{

int m = (v1 + v2)/2;

System.out.println("我猜是: " + m);

System.out.println("1.猜得太大了");

System.out.println("2.猜得太小了");

System.out.println("3.猜中! ");

System.out.print("请选择: ");

int user = Integer.parseInt(scan.nextLine());

if(user==3) break;

if(user==1) _____;

if(user==2) _____;

}

```

填空:

v2=m-1

v1=m+1

**277、**我们把“cba”称为“abc”的反转串。

求一个串的反转串的方法很多。下面就是其中的一种方法，代码十分简洁（甚至有些神秘），请聪明的你通过给出的一点点线索补充缺少的代码。

把填空的答案（仅填空处的答案，不包括题面）存入考生文件下对应题号的“解答.txt”中即可。

```

public static String reverseString(String x)

{

```

```

if(x==null || x.length()<2) return x;

return _____ + x.charAt(0);

}

```

填空：

reverseString(x.substring(1))+ x.charAt(0)

过程：

```

reverseString("abcde")=reverseString("bcde")+a      =edcba

reverseString("bcde")=reverseString("cde")+b        =edcb

reverseString("cde")=reverseString("de")+c          =edc

reverseString("de")=reverseString("e")+d            =ed

```

**278、**股票交易上的投机行为往往十分危险。假设某股票行为十分怪异，每天不是涨停（上涨 10%）就是跌停（下跌 10%）。

假设上涨和下跌的概率均等（都是 50%）。再假设交易过程没有任何手续费。某人在开始的时候持有总价值为 x 的该股股票，

那么 100 个交易日后，他盈利的可能性是多少呢？

以下程序通过计算机模拟了该过程，一般的输出结果在 0.3 左右。请填写缺失的代码。

把填空的答案（仅填空处的答案，不包括题面）存入考生文件夹下对应题号的“解答.txt”中即可。

```

int N = 10000;

int n = 0;

for(int i=0; i<N; i++)

{

double value = 1000.0;

for(int k=0; k<100; k++)

{

if(Math.random() > _____)

```

```

value = value * 1.1;

else

value = value * 0.9;

}

if(_____) n++;

}

System.out.println(1.0*n/N);

```

填空：

0.5      value>1000

结果：

0.3079

**279、** 下面的代码用于判断一个串中的括号是否匹配

所谓匹配是指不同类型的括号必须左右呼应，可以相互包含，但不能交叉

例如：

..(..[...]).. 是允许的

..(...[...]).... 是禁止的

对于 main 方法中的测试用例，应该输出：

false

true

false

false

```
import java.util.*;
```

```
public class A22
```

```
{
```

```
    public static boolean isGoodBracket(String s)
```

```

{

Stack<Character> a = new Stack<Character>();

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

{

char c = s.charAt(i);

if(c=='(') a.push('(');

if(c=='[') a.push('[');

if(c=='{' ) a.push('{');

if(c==' ' || c==' ' || c=='}')

{

if(_____) return false;    // 填空

if(a.pop() != c) return false;

}

}

if(_____) return false;    // 填空

return true;

}

public static void main(String[] args)

{

System.out.println( isGoodBracket("...(.[.].){(..).}..."));

System.out.println( isGoodBracket("...(.[...].(.).){(..).}..."));

System.out.println( isGoodBracket(".....[...].(.).){(..).}..."));

System.out.println( isGoodBracket("...(.[...].(.).){(..)....}"));

}

}

```

请分析代码逻辑，并推测划线处的代码。

答案写在 “解答.txt” 文件中

注意：只写划线处应该填的内容，划线前后的内容不要抄写。

填写：

a. `empty()`    `!a.empty()`

280、许多人都曾经玩过“拍七”游戏。规则是：大家依次从 1 开始顺序数数，数到含有 7 或 7 的倍数的要拍手或其它规定的方式表示越过（比如：7,14,17 等都不能数出），

下一人继续数下面的数字。违反规则者受罚。下面的程序模拟这个过程，拍 7 的情况输出“\*”，请完善之。

```
for(int i=1; i<100; i++)  
  
{  
  
    if(i % 7 == 0)  
  
        printf("*\n");  
  
    else if(_____  
  
        printf("*\n");  
  
    else  
  
        printf("%d\n", i);  
  
}
```

填空：

`i%10==7`

281、利用条件运算符的嵌套来完成此题：学习成绩 $\geq 90$  分的同学用 A 表示，60-89 分之间的用 B 表示，60 分以下的用 C 表示。

程序分析：`(a>b)?a:b` 这是条件运算符的基本例子。

```
public class Programme5 {  
  
    public static void main(String[] args) {  
  
        System.out.println("请输入你的分数：");
```



```

Scanner scanner=new Scanner(System.in);

int input=scanner.nextInt();//获取输入

//等级判断

String belong=input>=90?"A":(input>=60?"B":"c");

System.out.println(input+"分属于: "+belong);

scanner.close();

}

```

### 282、计算字符串中子串出现的次数

```

public class Prog49{
public static void main(String[] args){
String str = "I come from County DingYuan Province AnHui.";
char[] ch = str.toCharArray();
int count = 0;
for(int i=0;i<ch.length;i++){
if(ch[i]==' ')
count++;
}
count++;
System.out.println("共有"+count+"个字符串");
}
}

```

### 283、 $809 * ?? = 800 * ?? + 9 * ?? + 1$

其中??代表的两位数, $8 * ??$ 的结果为两位数,  $9 * ??$ 的结果为 3 位数。求??代表的两位数, 及  $809 * ??$ 后的结果。

```

public class Prog42{
public static void main(String[] args){
int n = 0;
boolean flag = false;
for(int i=10;i<100;i++){
if(809*i==800*i+9*i+1){

```

```

        flag = true;
        n = i;
        break;
    }
    if(flag)
        System.out.println(n);
    else
        System.out.println("无符合要求的数！");
}
}

```

**284、**海滩上有一堆桃子，五只猴子来分。第一只猴子把这堆桃子凭据分为五份，多了一个，这只猴子把多的一个扔入海中，拿走了一份。第二只猴子把剩下的桃子又平均分成五份，又多了一个，它同样把多的一个扔入海中，拿走了一份，第三、第四、第五只猴子都是这样做的，问海滩上原来最少有多少个桃子？

```

public class Prog41{
    public static void main(String[] args){
        int n;
        n = fun(0);
        System.out.println("原来有"+n+"个桃子");
    }
    private static int fun(int i){
        if(i==5)
            return 1;
        else
            return fun(i+1)*5+1;
    }
}

```

**285、**字符串排序。

```

public class Prog40{
    public static void main(String[] args){
        String[] str = {"abc","cad","m","fa","f"};
        for(int i=str.length-1;i>=1;i--){
            for(int j=0;j<=i-1;j++){
                if(str[j].compareTo(str[j+1])<0){
                    String temp = str[j];
                    str[j] = str[j+1];
                    str[j+1] = temp;
                }
            }
        }
    }
}

```

```

}
for(String subStr:str)
    System.out.print(subStr+" ");
}
}

```

**286、**打印出杨辉三角形（要求打印出 10 行如下图）

程序分析：

```

      1
    1 1
  1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1

```

```

public class Prog33{
public static void main(String[] args){
int[][] n = new int[10][21];
n[0][10] = 1;
for(int i=1;i<10;i++)
    for(int j=10-i;j<10+i+1;j++)
        n[i][j] = n[i-1][j-1]+n[i-1][j+1];
for(int i=0;i<10;i++){
for(int j=0;j<21;j++){
if(n[i][j]==0)
    System.out.print("  ");
else{
    if(n[i][j]<10)
        System.out.print(" "+n[i][j]);//空格为了美观需要
    else if(n[i][j]<100)
        System.out.print(" "+n[i][j]);
    else

```

```

        System.out.print(n[i][j]);
    }
}
System.out.println();
}
}
}

```

**287、**取一个整数 **a** 从右端开始的 4~7 位。

程序分析：可以这样考虑：

(1)先使 **a** 右移 4 位。

(2)设置一个低 4 位全为 1,其余全为 0 的数。可用  $\sim(\sim 0 << 4)$

(3)将上面二者进行&运算

```

import java.util.Scanner;
public class Prog32{
    public static void main(String[] msg){
        //输入一个长整数
        Scanner scan = new Scanner(System.in);
        long l = scan.nextLong();
        scan.close();
        //以下截取字符
        String str = Long.toString(l);
        char[] ch = str.toCharArray();
        int n = ch.length;
        if(n<7)
            System.out.println("输入的数小于 7 位！");
        else
            System.out.println("截取的 4~7 位数字： "+ch[n-7]+ch[n-6]+ch[n-5]+ch[n-4]);
    }
}

```

**288、**求一个 3\*3 矩阵对角线元素之和

程序分析：利用双重 **for** 循环控制输入二维数组，再将 **a[i][i]**累加后输出。

```

public class Prog29{

```

```

public static void main(String[] args){
    int[][] a = new int[][] {{100,2,3},{4,5,6},{17,8,9}};
    matrSum(a);
}
private static void matrSum(int[][] a){
    int sum1 = 0;
    int sum2 = 0;
    for(int i=0;i<a.length;i++){
        for(int j=0;j<a[i].length;j++){
            if(i==j) sum1 += a[i][j];
            if(j==a.length-i-1) sum2 += a[i][j];
        }
    }
    System.out.println("矩阵对角线之和分别是: "+sum1+"和"+sum2);
}
}

```

289、给一个不多于 5 位的正整数，要求：一、求它是几位数，二、逆序打印出各位数字。

```

public class Prog24{
    public static void main(String[] args){
        int n = Integer.parseInt(args[0]);
        int i = 0;
        int[] a = new int[5];
        do{
            a[i] = n%10;
            n /= 10;
            ++i;
        }while(n!=0);
        System.out.print("这是一个"+i+"位数，从个位起，各位数字依次为: ");
        for(int j=0;j<i;j++){
            System.out.print(a[j]+" ");
        }
    }
}

```

290、有 5 个人坐在一起，问第五个人多少岁？他说比第 4 个人大 2 岁。问第 4 个人岁数，他说比第 3 个人大 2 岁。问第三个人，又说比第 2 人大两岁。问第 2 个人，说比第一个人大两岁。最后问第一个人， he 说是 10 岁。请问第五个人多大？

程序分析：利用递归的方法，递归分为回推和递推两个阶段。要想知道第五个人岁数，需知道第四人的岁数，依次类推，推到第一人（10 岁），再往回推。

```
public class Prog23{
public static void main(String[] args){
System.out.println(getAge(5,2));
}
//求第 m 位同志的年龄
private static int getAge(int m,int n){
if(m==1)
return 10;
else
return getAge(m-1,n)+n;
}
}
```

291、利用递归方法求 5!。

程序分析：递归公式：  $fn=fn\_1*4!$

```
public class Prog22{
public static void main(String[] args){
System.out.println(fact(10));
}
//递归求阶乘
private static long fact(int n){
if(n==1)
return 1;
else
return fact(n-1)*n;
}
}
```

292、求  $1+2!+3!+\dots+20!$  的和

程序分析：此程序只是把累加变成了累乘。

```
public class Prog21{
public static void main(String[] args){
long sum = 0;
for(int i=0;i<20;i++){
sum += factorial(i+1);
}
```

```

System.out.println(sum);
}
//阶乘
private static long factorial(int n){
int mult = 1;
for(int i=1;i<n+1;i++){
    mult *= i;
}
return mult;
}
}

```

293、打印出如下图案（菱形）

```

    *

   ***

  *****

 *****

 *****

    ***

    *

```

```

public class Prog19{
public static void main(String[] args){
int n = 5;
printStar(n);
}
//打印星星
private static void printStar(int n){
//打印上半部分
for(int i=0;i<n;i++){
for(int j=0;j<2*n;j++){
    if(j<n-i)
        System.out.print(" ");
    if(j>=n-i && j<=n+i)
        System.out.print("*");
}
System.out.println();
}
}

```

```

}
//打印下半部分
for(int i=1;i<n;i++){
    System.out.print(" ");
    for(int j=0;j<2*n-i;j++){
        if(j<i)
            System.out.print(" ");
        if(j>=i && j<2*n-i-1)
            System.out.print("*");
    }
    System.out.println();
}
}
}
}

```

294、有 1、2、3、4 个数字，能组成多少个互不相同且无重复数字的三位数？都是多少？

```

public class Prog11{
    public static void main(String[] args){
        int count = 0;
        int n = 0;
        for(int i=1;i<5;i++){
            for(int j=1;j<5;j++){
                if(j==i)
                    continue;
                for(int k=1;k<5;k++){
                    if(k!=i && k!=j){
                        n = i*100+j*10+k;
                        System.out.print(n+" ");
                        if((++count)%5==0)
                            System.out.println();
                    }
                }
            }
        }
        System.out.println();
        System.out.println("符合条件的数共: "+count+"个");
    }
}

```

295、输入一个字符，判断它是否为小写字母，如果是，将它转换成大写字母，否则，不转换。



```

package HomeWork03;

import java.util.Scanner;

public class HomeWork03 {

    public static void main(String[] args) {

        //小写字母的 ascll 值为 97-122

        //大写字母的 ascll 值为 65-90

        System.out.println("请输入一个字母： \n");

        Scanner input = new Scanner(System.in);

        char zimu=input.next().charAt(0);

        if (zimu>=97&&zimu<=122){           //判断是否是小写字母

            System.err.println("该字母是小写字母");

            zimu=(char) (zimu-32);           //如果是小写字母则 将其转换成大写字母

            System.err.println("转换之后的大写字母是： "+zimu);

        }

        else{

            System.out.println("该字母不是小写字母！ ");

        }

    }

}

```

295、用 while 循环，计算 1~200 之间所有 3 的倍数之和

```

package HomeWork09;

public class HomeWork09 {

```

```

public static void main(String[] args) {

    // 用 while 循环，计算 1~200 之间所有 3 的倍数之和。

    int a=1;

    int sum=0;

    while(a<=200){

        if(a%3==0){

            sum=sum+a;

        }

        a++;

    }

    System.out.println("1~200 之间所有 3 的倍数之和为:"+sum);

}
}

```

296、编写程序，输出 200~500 之间的所有素数。

```

package HomeWork10;

public class HomeWork10 {

    public static void main(String[] args) {

        int num=200;

        while (num<=500) {

            boolean tag=true;    //素数标记

            for(int d=2;d<=num-1;d++){

                if(num % d==0){

```

```

        tag=false;

        break;

    }

}

if(tag){           //如果是素数

    System.out.println(num);

}

num++;

}

}

}

```

297、使用循环语句输出下面的图形。

```

#

# # #

# # # # #

# # # # # # #

# # # # # # # # #

```

```

package HomeWork12;

public class HomeWork12 {

    public static void main(String[] args) {

        int aa=-1;

        for( int a=0;a<5;a++){

            aa+=2;

            for(int b=1;b<=aa;b++){

                System.out.print( "#" );

            }

            System.out.println();}

```

```
}  
}
```

298、输入一行字符，分别统计出其中英文字母、空格、数字和其它字符的个数。

1.程序分析：利用 **while** 语句,条件为输入的字符不为'\n'.

```
package cn.edu.hit;  
  
import java.util.Scanner;  
  
public class strIdentify {  
  
    public static void main(String[] args) {  
  
        int abcCount = 0;  
  
        int spaceCount = 0;  
  
        int numCount = 0;  
  
        int otherCount = 0;  
  
        Scanner sc = new Scanner(System.in);  
  
        String str = sc.nextLine();  
  
        char[] ch = str.toCharArray();  
  
        for (int i = 0; i < ch.length; i++) {  
  
            if (Character.isDigit(ch[i])) {  
  
                numCount++;  
  
            } else if (Character.isSpaceChar(ch[i])) {  
  
                spaceCount++;  
  
            } else if (Character.isLetter(ch[i])) {  
  
                abcCount++;  
  
            } else {
```

```

        otherCount++;

    }

}

System.out.println("字母个数"+abcCount);

System.out.println("数字个数"+numCount);

System.out.println("空格个数"+spaceCount);

System.out.println("其他字符个数"+otherCount);

}

}

```

299、一个数如果恰好等于它的因子之和，这个数就称为“完数”。例如

6=1 + 2 + 3. 编程 找出 1000 以内的所有完数。

```

public class wanShu {
    public static void main(String[] args) {
        int k = 2;
        int num = 0;
        int temp = 1;
        int j = 0;
        for (num = 1; num <= 1000; num++) {
            k = 2;
            temp = 1;
            j = num;
            while (j >= k) {
                if (j % k == 0) {
                    temp += k;
                    j = j / k;
                } else {
                    k++;
                }
            }
            if (temp == num) {

```

```

        System.out.println(temp);
    }

}

}

```

300、一球从 100 米高度自由落下，每次落地后反跳回原高度的一半；再落下，求它在第 10 次落地时，共经过多少米？第 10 次反弹多高？

```

public class testBall {
    public static void main(String[] args) {
        double a = 100;
        double sum = 100;
        for(int i =2 ;i<=10;i++){
            a = a*0.5;
            sum += a*2;
        }
        System.out.println("a="+a);
        System.out.println("sum="+sum);
    }
}

```

# 实践题题库

第一题: hdfs

1. hdfs

有三种 shell 命令方式。

`hadoop fs`

`hadoop dfs` (已过期)

`hdfs dfs` (常用)

`hadoop fs` 适用于任何不同的文件系统，比如本地文件系统和 HDFS 文件系统

`hadoop dfs` 只能适用于 HDFS 文件系统

`hdfs dfs` 跟 `hadoop dfs` 的命令作用一样，也只能适用于 HDFS 文件系统

1. 在 hdfs 创建一个在 user 下的目录 hadoop

`hadoop fs -mkdir /user/hadoop`

2. 写 “Hello world” 到 test.txt 文件

```
echo "Hello world" ->test.txt
```

3. 将 test.txt 放入 hdfs 在 user 下的目录 hadoop

```
hadoop fs -put test.txt /user/hadoop
```

4. 在 hdfs 中查看 test.txt

```
hadoop fs -cat /user/test.txt
```

5. 从 hdfs 上下载 test.txt 到本地

```
hadoop fs -get /user/test.txt
```

6. 从 hdfs 上删除 test.txt

```
hadoop fs -rm /user/test.txt
```

第二题：mongodb 数据库相关

show dbs:显示数据库列表

show collections: 显示当前数据库中的集合（类似关系数据库中的表 table）

show users: 显示所有用户

use yourDB: 切换当前数据库至 yourDB

db.help() : 显示数据库操作命令

db.yourCollection.help() : 显示集合操作命令，yourCollection 是集合名

MongoDB 没有创建数据库的命令，如果你想创建一个“School”的数据库，先运行 use School 命令，之后做一些操作

1. 创建一个 “student” 的数据库

```
use student
```

2. 插入数据

```
db.student.insert({_id:1, sname: 'zhangsan', sage: 20})
```

或者



```
db.student.save({_id:1, sname: 'zhangsan', sage: 22})
```

这两种方式，其插入的数据中\_id 字段均可不写，会自动生成一个唯一的\_id 来标识本条数据。而 insert 和 save 不同之处在于：在手动插入\_id 字段时，如果\_id 已经存在，insert 不做操作，save 做更新操作；如果不加\_id 字段，两者作用相同都是插入数据。

### 3. 查找数据

```
db.student.find(criteria, filterDisplay)
```

criteria : 查询条件，可选

filterDisplay: 筛选显示部分数据，如显示指定列数据，可选（当选择时，第一个参数不可省略，若查询条件为空，可用 {} 做占位符，如下例第三句）

```
db.student.find() #查询所有记录。相当于：select * from student
```

```
db.student.find({sname: 'lisi'}) #查询 sname='lisi' 的记录。相当于：select * from student where sname= 'lisi'
```

```
db.student.find({}, {sname:1, sage:1}) #查询指定列 sname、sage 数据。相当于：select sname, sage from student。sname:1 表示返回 sname 列，默认_id 字段也是返回的，可以添加_id:0（意为不返回_id）写成 {sname: 1, sage: 1, _id:0}，就不会返回默认_id 字段了
```

```
db.student.find({sname: 'zhangsan', sage: 22}) #and 与条件查询。相当于：select * from student where sname = 'zhangsan' and sage = 22
```

```
db.student.find({$or: [{sage: 22}, {sage: 25}]}) #or 条件查询。相当于：select * from student where sage = 22 or sage = 25
```

### 4. 修改数据

```
db.student.update({sname: 'lisi' }, {$set: {sage: 30}}, false, true) #相当于：update student set sage =30 where sname = 'lisi' ;
```

### 5. 删除数据

```
db.student.remove({sname: 'zhaoliu' }) #相当于：delete from student where sname= 'zhaoliu'
```

### 6. 删除集合

```
db.student.drop()
```

### 第三题：HBASE

1. 创建一个 student 表，属性有：name, sex, age, dept, course。
2. 为 student 表添加了学号为 18001，名字为 zhangsan 的一行数据，其行键为 18001。
3. 查看数据
4. 删除 student 表中 95001 行下的 sex 列的所有数据。

`delete 'student', '18001', 'sex'`

5. 删除 student 表中的 18001 行的全部数据。
6. 删除表

### 第四题：关系模式

设有一个记录各个球队队员每场比赛进球数的关系模式 R（队员编号，比赛场次，进球数，球队名，队长名）

如果规定每个队员只能属于一个球队，每个球队只有一个队长。

- (1)试写出关系模式 R 的基本 FD 和关键码。
- (2)说明 R 不是 2NF 模式的理由，并把 R 分解成 2NF 模式集。
- (3)进而把 R 分解成 3NF 模式集，并说明理由。

### 第五题：关系模式

设有关系模式 R（职工名，项目名，工资，部门名，部门经理）

如果规定每个职工可参加多个项目，各领一份工资；每个项目只属于一个部门管理；每个部门只有一个经理。

- (1)试写出关系模式 R 的基本 FD 和关键码。
- (2)说明 R 不是 2NF 模式的理由，并把 R 分解成 2NF 模式集。
- (3)进而把 R 分解成 3NF 模式集，并说明理由。

#### 第六题：ER 图绘制

设某汽车运输公司数据库中有三个实体集。一是“车队”实体集，属性有车队号、车队名等；二是“车辆”实体集，属性有牌照号、厂家、出厂日期等；三是“司机”实体集，属性有司机编号、姓名、电话等。

设车队与司机之间存在“聘用”联系，每个车队可聘用若干司机，但每个司机只能应聘于一个车队，车队聘用司机有个聘期；车队与车辆之间存在“拥有”联系，每个车队可拥有若干车辆，但每辆车只能属于一个车队；司机与车辆之间存在着“使用”联系，司机使用车辆有使用日期和公里数两个属性，每个司机可使用多辆汽车，每辆汽车可被多个司机使用。

- (1)试画出 ER 图，并在图上注明属性、联系类型、实体标识符；
- (2)将 ER 图转换成关系模型，并说明主键和外键。
- (3)将 ER 图转换成对象联系图。
- (4)将 ER 图转换成 UML 的类图。

#### 第七题：ER 图绘制

设大学里教学数据库中有三个实体集。一是“课程”实体集，属性有课程号、课程名称；二是“教师”实体集，属性有教师工号、姓名、职称；三是“学生”实体集，属性有学号、姓名、性别、年龄。

设教师与课程之间有“主讲”联系，每位教师可主讲若干门课程，但每门课程只有一位主讲教师，教师主讲课程将选用某本教材；教师与学生之间有“指导”联系，每位教师可指导若干学生，但每个学生只有一位指导教师；学生与课程之间有“选课”联系，每个学生可选修若干课程，每门课程可由若干学生选修，学生选修课程有个成绩。

- (1) 试画出 ER 图，并在图上注明属性、联系类型、实体标识符；

(2) 将 ER 图转换成关系模型，并说明主键和外键。

(3) 将 ER 图转换成对象联系图。

(4) 将 ER 图转换成 UML 的类图。

#### 第八题：ER 图绘制

设大学教学数据库中有下面一些数据：

- Dept（系）有属性 dno（系编号）和 dname（系名）；
- Student（学生）有属性 sno（学号）和 sname（学生姓名）；
- Course（课程）有属性 cno（课程号）、cname（课程名）和 teacher（任课教师）；
- 学生选修课程有个 grade（成绩）。

如果规定：每个系有若干学生，每个学生只能属于一个系；每个系开设了若干课程，每门课程由一个系开设；每个学生可以选修若干课程，每门课程可以有若干学生选修。

(1) 试画出 ER 图，并在图上注明属性、联系类型、实体标识符；

(2) 将 ER 图转换成关系模型，并说明主键和外键。

(3) 试画出第 38 题数据库的对象联系图。

(4) 试画出第 38 题数据库的 UML 类图。

#### 第九题：Linux 基本操作

1. 在当前目录下建立文件 exam.c，将文件 exam.c 拷贝到/tmp 这个目录下，并改名为 shiyan.c。

```
touch exam.c
```

```
cp /root/exam.c /tmp/shiyan.c
```

2. 在任何目录下回到用户主目录。

```
cd /tmp  
cd
```

3. 打印当前目录（隐藏文件也显示）。

```
ll -a
```

4. 在当前目录中新建文件 text 并设置文件的属性为文件属主(u)增加执行权限与文件属主同组用户(g)增加写权限其他用户(o) 删除读权限。

```
touch text  
chmod u+x text  
chmod g+w text  
chmod o-r text
```

5. 创建用户 xu 和 liu 并将/home/xu 目录中的所有文件拷贝到目录/home/liu 中。

```
useradd xu  
useradd liu  
ll /home/xu  
cd /home/xu  
touch a1.c  
cp -r /home/xu/* /home/liu
```

#### 第十题：intellij idea 实操

1、生成并运行应用程序

2、完成“YUE,I LOVE YOU”的输出

3、将应用程序打包到 JAR 中

4、运行打包的应用程序

5、为打包的应用程序创建运行配置

## 十一、备份数据：

5. 备份单个数据库结构（sakila 为数据库名，-d）
6. 备份单个数据库数据（sakila 为数据库名，-t）
7. 备份多个表的结构和数据（table1,table2 为表名）
8. 一次备份多个数据库

## 十二、还原数据库：

1. 系统命令行：
2. source 方法：