Java实践与练习4(计19-7、8)

/\*  
A. 数据子序列  
Description  
输入一个长度为 n 的整数序列（A1,A2,……An），从中找出一段连续的长度不超过 m 的子序列，使得这个子序列的和最大。  
Input  
有多组测试数据。  
  
对于每组测试数据的第一行，包含二个整数 n 和 m，表示本组有 n 个测试数据，子序列长度为 m，下一行为 n 个测试数据。  
Output  
对于每组测试数据，输出最大的子序列和，并换行。  
Sample Input  
3 1  
  
1 2 3  
  
3 2  
  
-1000 1000 1  
  
10 2  
  
3 8 9 7 4 8 9 2 3 7  
  
  
  
Sample Output  
3  
  
1001  
  
17  
   
\*/  
package A;  
import java.util.Scanner;  
//A  
public class Main {  
 public static void main(String[] args) {  
 Scanner cin = new Scanner(System.in);  
 while (cin.hasNext()) {  
 int n = cin.nextInt();  
 int m = cin.nextInt();  
 int result = 0;  
 int[] a = new int[n];  
 for (int i = 0; i < n; i++) {  
 a[i] = cin.nextInt();  
 if (i < m) {  
 result += a[i];  
 }  
 }  
 int temp = result;  
 for (int i = m; i < n; i++) {  
 temp = a[i] + temp - a[i - m];  
 if (temp > result) {  
 result = temp;  
 }  
 }  
 System.out.println(result);  
 }  
 }  
}

———————————————————————————————————————

/\*B. 字符串拆分   
 Description  
 给定一个字符串，将字符串中各个字符拆分出来并如 Sample Output 所示输出。  
 Input  
 有多组数据，每组数据由一个字符串组成。  
 Output  
 将字符串中的字符如 Sample Output 所示输出。  
 Sample Input  
 Hrbust  
 Computer  
  
  
 Sample Output  
 H  
  
 rr  
  
 bbb  
  
 uuuu  
  
 sssss  
  
 tttttt  
  
 C  
  
 oo  
  
 mmm  
  
 pppp  
  
 uuuuu  
  
 tttttt  
  
 eeeeeee  
  
 rrrrrrrr  
   
 \*/  
package B;  
  
import java.util.\*;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner scan =new Scanner(System.in);  
 while (scan.hasNext()) {  
  
 String str=scan.next();  
 for(int i=0;i<str.length();i++) {  
 for(int j = str.length()-i-1;j>0;j--) {  
 System.out.print(" ");  
 }  
 for(int j=0;j<=i;j++) {  
 System.out.print(str.charAt(i));  
 }  
 System.out.println();  
 }  
  
  
 }  
 }  
  
}

/\*C. 矩阵  
 Description  
 给定一个自然数，输出如 Sample Output 所示的矩阵。  
 Input  
 有多组数据，每组数据由一个自然数组成。  
 Output  
 根据给定的自然数，输出如 Sample Output 所示的矩阵。  
 Sample Input  
 5  
 10  
  
 Sample Output  
 1 1 1 1 1  
 1 2 1 1 1  
 1 2 3 1 1  
 1 2 3 4 1  
 1 2 3 4 5  
 1 1 1 1 1 1 1 1 1 1  
 1 2 1 1 1 1 1 1 1 1  
 1 2 3 1 1 1 1 1 1 1  
 1 2 3 4 1 1 1 1 1 1  
 1 2 3 4 5 1 1 1 1 1  
 1 2 3 4 5 6 1 1 1 1  
 1 2 3 4 5 6 7 1 1 1  
 1 2 3 4 5 6 7 8 1 1  
 1 2 3 4 5 6 7 8 9 1  
 1 2 3 4 5 6 7 8 9 10  
   
 \*/  
package C;  
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner cin = new Scanner(System.in);  
 while (cin.hasNext()) {  
 int num=cin.nextInt();  
 for(int i=1;i<=num;i++){  
 for(int j=1;j<=i;j++){  
 System.out.print(String.format("%4s",j));  
 }  
 for(int j=1;j<=num-i;j++){  
 System.out.print(String.format("%4s","1"));  
 }  
 if(i!=num) {  
 System.out.println();  
 }  
 }  
 }  
 }  
}

/\*D. 回数猜想  
  
Description  
数学中有个有名的猜想，叫 “回数猜想”，至今未解决。  
回数猜想是：给定一个自然数，再把它例过来，并将这两个数相加，然后把这个和数再倒过来，与原来的和数相加，重复此过程，一定能得到一个回文数。  
如：  
给定 153，倒过来是 351，相加：153+351=504  
504 倒过来：405，再相加：504+405=909  
909 是一个回数。  
编程序验证 “回数猜想”。为简便起见，如果上述过程重复 10 次仍无法得到回数，则认为该猜想为假。  
  
Input  
有多组数据，每组数据由一个自然数组成。  
Output  
如果给定的自然数最多经过 10 次运算后能得到一个回数，则输出该回数；如果不能，则输出 “not”。  
Sample Input  
153  
68  
18333  
  
Sample Output  
909  
1111  
not  
  
 \*/  
package D;  
/\*D. 回数猜想  
Time Limit: 1000 MS Memory Limit: 32768 K  
Total Submit: 78 (58 users) Total Accepted: 62 (58 users) Special Judge: No  
Description  
数学中有个有名的猜想，叫 “回数猜想”，至今未解决。  
回数猜想是：给定一个自然数，再把它例过来，并将这两个数相加，然后把这个和数再倒过来，与原来的和数相加，重复此过程，一定能得到一个回文数。  
如：  
给定 153，倒过来是 351，相加：153+351=504  
504 倒过来：405，再相加：504+405=909  
909 是一个回数。  
编程序验证 “回数猜想”。为简便起见，如果上述过程重复 10 次仍无法得到回数，则认为该猜想为假。  
  
Input  
有多组数据，每组数据由一个自然数组成。  
Output  
如果给定的自然数最多经过 10 次运算后能得到一个回数，则输出该回数；如果不能，则输出 “not”。  
Sample Input  
153  
68  
18333  
  
Sample Output  
909  
1111  
not  
  
 \*/  
package D;  
import java.util.Scanner;  
//D  
public class Main {  
 public static void main(String[] args) {  
 Scanner cin = new Scanner(System.in);  
 while (cin.hasNext()) {  
 int n =cin.nextInt();  
 int key = 10;  
 boolean flag = true;  
 while (key!=0){  
 String str = n+"";  
 int a=0;  
 int b=str.length()-1;  
 flag =true;  
 while (a<b){  
 if (str.charAt(a++)!=str.charAt(b--)){  
 flag=false;  
 break;  
 }  
 }  
 if (flag){  
 break;  
 }  
 String str2= new StringBuffer(str).reverse().toString();  
 int m=Integer.parseInt(str2);  
 n+=m;  
 key--;  
 }  
 if (flag){  
 System.out.println(n);  
 }else {  
 System.out.println("not");  
 }  
 }  
 }  
}

/\*E. 数字统计  
Description  
给定一个字符串，统计其中每个数字出现的次数。  
Input  
有多组数据，每组数据由一个字符串组成。  
Output  
每个数字出现的次数，每个数占 3 位宽度，不出现为 0。  
Sample Input  
Harbin,Xuefu Load 52  
Now Time:2018-06-28 11:22:33  
  
Sample Output  
0 0 1 0 0 1 0 0 0 0  
2 3 4 2 0 0 1 0 2 0  
  
 \*/  
package E;  
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args){  
 Scanner scan=new Scanner(System.in);  
 while (scan.hasNext()){  
 String str = scan.nextLine();  
 int a[]=new int[10];  
 for(int i=0;i<=9;i++){  
 a[i]=0;  
 }  
  
 for(int i=0;i<str.length();i++){  
  
 if(str.charAt(i)=='0'){  
 a[0]++;  
 }  
  
  
 if(str.charAt(i)=='1'){  
 a[1]++;  
 }  
  
 if(str.charAt(i)=='2'){  
 a[2]++;  
 }  
  
 if(str.charAt(i)=='3'){  
 a[3]++;  
 }  
  
 if(str.charAt(i)=='4'){  
 a[4]++;  
 }  
  
 if(str.charAt(i)=='5'){  
 a[5]++;  
 }  
  
 if(str.charAt(i)=='6'){  
 a[6]++;  
 }  
  
 if(str.charAt(i)=='7'){  
 a[7]++;  
 }  
  
 if(str.charAt(i)=='8'){  
 a[8]++;  
 }  
  
 if(str.charAt(i)=='9'){  
 a[9]++;  
 }  
  
 }  
  
 for(int i=0;i<=9;i++){  
 if(i<9) {  
 System.out.print(a[i] + " ");  
 }  
  
 if(i==9){  
 System.out.print(a[i]);  
 }  
  
 }  
  
 }  
 }  
}  
/\* 参考代码  
auhor：于喆庚  
import java.util.Scanner;  
//E  
public class charNumber {  
 public static void main(String[] args) {  
 Scanner cin = new Scanner(System.in);  
 while (cin.hasNext()) {  
 String str = cin.nextLine();  
 int[] num = new int[10];  
 for (int i=0;i<str.length();i++){  
 if (Character.isDigit(str.charAt(i))){  
 num[Integer.parseInt(String.valueOf(str.charAt(i)))]++;  
 }  
 }  
 System.out.print(num[0]);  
 for (int i = 1; i < 10; i++) {  
 System.out.print(String.format("%3s",num[i]));  
 }  
 System.out.println();  
 }  
 }  
}  
  
 \*/

Java 实践与练习 3 (计 19-7、8）

/\*A. 个位数统计  
Description  
给定一个 k 位整数 N=dk-110k-1+....+d1101+d0 (0≤di ≤9, i=0,?,k?1, dk?1>0), 请编写程序统计每种不同的个位数字出现的次数。例如：给定 N=100311，则有 2 个 0，3 个 1，和 1 个 3。  
Input  
输入一个不超过 1000 位的正整数 N。  
Output  
对 N 中每一种不同的个位数字，以 D:M 的格式在一行中输出该位数字 D 及其在 N 中出现的次数 M。要求按 D 的升序输  
Sample Input  
100311  
Sample Output  
0:2  
1:3  
3:1  
  
 \*/  
package A;  
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args){  
 Scanner scan =new Scanner(System.in);  
 while (scan.hasNext()){  
 int DIGITS[]={0,0,0,0,0,0,0,0,0,0};  
// int num = scan.nextInt();  
 String s=scan.nextLine();  
 int len=s.length();  
 for(int i=0;i<len;i++){  
 DIGITS[s.charAt(i)-'0']++;  
 }  
  
 for(int i=0;i<10;i++){  
 if (DIGITS[i] == 0) {  
 continue;  
 }  
 System.out.println(i+":"+DIGITS[i]);  
 }  
 }  
 }  
}

/\*  
package B;  
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args){  
 Scanner scan=new Scanner(System.in);  
 while (scan.hasNext()){  
 int num= scan.nextInt();  
 if(num<0){  
 System.out.print("fu ");  
 num=-num;  
  
  
 }  
  
 if(num>=0){  
  
 }  
 }  
}  
\*/  
/\*B. 念数字  
Description  
输入一个整数，输出每个数字对应的拼音。当整数为负数时，先输出 fu 字。十个数字对应的拼音如下：  
0: ling 1: yi 2: er 3: san 4: si 5: wu 6: liu 7: qi 8: ba 9: jiu  
Input  
输入在一行中给出一个整数，如：1234。提示：整数包括负数、零和正数。  
Output  
在一行中输出这个整数对应的拼音，每个数字的拼音之间用空格分开，行末没有最后的空格。如 yi er san si。  
Sample Input  
-600  
Sample Output  
fu liu ling ling  
  
 \*/  
package B;  
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner input=new Scanner(System.in);  
 while (input.hasNext()){  
 String str=input.nextLine();  
 int a1=0;int a2=0;int a3=0;int a4=0;  
 int a5=0;int a6=0;int a7=0;int a8=0;  
 int a9=0;int a0=0;  
 for(int i=0;i<str.length()-1;i++){  
 if(str.charAt(i)=='0'){  
 System.out.print("ling");  
 System.out.print(" ");  
 }  
 if(str.charAt(i)=='1'){System.out.print("yi");System.out.print(" ");}  
 if(str.charAt(i)=='2'){System.out.print("er");System.out.print(" ");}  
 if(str.charAt(i)=='3'){System.out.print("san");System.out.print(" ");}  
 if(str.charAt(i)=='4'){System.out.print("si");System.out.print(" ");}  
 if(str.charAt(i)=='5'){System.out.print("wu");System.out.print(" ");}  
 if(str.charAt(i)=='6'){System.out.print("liu");System.out.print(" ");}  
 if(str.charAt(i)=='7'){System.out.print("qi");System.out.print(" ");}  
 if(str.charAt(i)=='8'){System.out.print("ba");System.out.print(" ");}  
 if(str.charAt(i)=='9'){System.out.print("jiu");System.out.print(" ");}  
 if(str.charAt(i)=='-'){System.out.print("fu");System.out.print(" ");}  
 }  
 if(str.charAt(str.length()-1)=='0'){System.out.print("ling");}  
 if(str.charAt(str.length()-1)=='1'){System.out.print("yi");}  
 if(str.charAt(str.length()-1)=='2'){System.out.print("er");}  
 if(str.charAt(str.length()-1)=='3'){System.out.print("san");}  
 if(str.charAt(str.length()-1)=='4'){System.out.print("si");}  
 if(str.charAt(str.length()-1)=='5'){System.out.print("wu");}  
 if(str.charAt(str.length()-1)=='6'){System.out.print("liu");}  
 if(str.charAt(str.length()-1)=='7'){System.out.print("qi");}  
 if(str.charAt(str.length()-1)=='8'){System.out.print("ba");}  
 if(str.charAt(str.length()-1)=='9'){System.out.print("jiu");}  
 if(str.charAt(str.length()-1)=='-'){System.out.print("fu");}  
 System.out.println();  
  
 }  
 }}

/\*C. 比较大小  
Description  
本题要求将输入的任意 3 个整数从小到大输出。  
Input  
输入在一行中给出 3 个整数，其间以空格分隔。  
Output  
在一行中将 3 个整数从小到大输出，其间以 "->" 相连。  
Sample Input  
4 2 8  
Sample Output  
2->4->8  
  
 \*/  
package C;  
import java.util.Scanner;  
import static java.util.Arrays.sort;  
  
public class Main {  
 public static void main(String[] args){  
 Scanner scan = new Scanner(System.in);  
 while (scan.hasNext()){  
 int a[]={0,0,0};  
 for (int i=0;i<=2;i++){  
 a[i]=scan.nextInt();  
 }  
 sort(a);  
 System.out.println(a[0]+"->"+a[1]+"->"+a[2]);  
 }  
 }  
}

/\*D. 字符串处理  
Description  
给定一个不含标点的英文字符串，两个单词间以空格分割，不改变整句中单词的顺序，将其中的每一个单词倒序输出，并统计字符串中出现次数最多的字母（不区分大小写），如果出现次数相同，则输出其中字典序最小的字母的小写。  
Input  
多组输入，每组输入是一个不含标点的英文字符串。  
Output  
对于每组输入，第一行输出按要求倒序输出的结果，第二行按要求输出出现次数最多的字母。  
Sample Input  
I have an apple  
Sample Output  
I evah na elppa  
a  
Hint  
每组输入长度 L：0<L<=100，仅由大小写字母构成，输入有可能只含一个单词。  
  
 \*/  
package D;  
import java.util.\*  
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args){  
 Scanner scan=new Scanner(System.in);  
 while (scan.hasNext()){  
 String s=scan.nextLine();  
 int count[]=new int[50];  
  
 StringBuilder s2=new StringBuilder();  
 for(int i=0;i<s.length();i++){  
 if(i+1==s.length()||s.charAt(i+1)==' ')  
 {  
 for(int j=i;j>=0&&s.charAt(j)!=' ';j--){  
 s2.append(s.charAt(j));  
 }  
 if(i+1!=s.length())  
 {  
 s2.append(' ');  
 }  
 }  
 }  
 System.out.println(s2);  
 String s1=s.toLowerCase();  
 for(int i=0;i<s1.length();i++){  
 if(s.charAt(i)!=' '){  
 count[s1.charAt(i)-'a']++;  
 }  
 }  
 }  
 }  
}  
  
/\*  
package D;  
import java.util.\*;  
  
import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.in);  
 while(scan.hasNext()){  
 String s = scan.nextLine();  
 int count[] = new int[50];  
  
 StringBuilder s2 = new StringBuilder();  
 for(int i = 0; i < s.length(); i++)  
 {  
 if(i+1 == s.length() || s.charAt(i+1) == ' ')  
 {  
 for(int j = i; j >= 0 && s.charAt(j) != ' ' ; j--)  
 {  
 s2.append(s.charAt(j));  
 }  
 if(i+1 != s.length())  
 {  
 s2.append(' ');  
 }  
 }  
 }  
 System.out.println(s2);  
 String s1 = s.toLowerCase();  
 for (int i = 0; i < s1.length(); i++) {  
 if(s.charAt(i) != ' ') {  
 count[s1.charAt(i) - 'a']++;  
 }  
 }  
 int pos = 0;  
 int max = 0;  
 for(int i=0; i<50; i++){  
 if(count[i] > max){  
 pos = i;  
 max = count[i];  
 }  
 }  
 char ch = (char) (pos + 'a');  
 System.out.println(ch);  
 }  
 }  
}  
  
 \*/

/\*F. 正整数 A+B  
Description  
求两个正整数 A 和 B 的和，其中 A 和 B 都在区间 [1,1000]。稍微有点麻烦的是，输入并不保证是两个正整数。  
Input  
输入在一行给出 A 和 B，其间以空格分开。问题是 A 和 B 不一定是满足要求的正整数，有时候可能是超出范围的数字、负数、带小数点的实数、甚至是一堆乱码。  
注意：我们把输入中出现的第 1 个空格认为是 A 和 B 的分隔。题目保证至少存在一个空格，并且 B 不是一个空字符串。  
Output  
如果输入的确是两个正整数，则按格式 A + B = 和输出。如果某个输入不合要求，则在相应位置输出？，此时和也是？。  
Sample Input  
123 456  
-100 blabla bla...33  
22. 18  
Sample Output  
123 + 456 = 579  
? + ? = ?  
? + 18 = ?  
  
 \*/  
package B;  
import java.util.Scanner;  
import java.util.regex.Pattern;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.in);  
 while (scan.hasNext()) {  
 String A=scan.next();  
 String B=scan.next();  
 String afterB=scan.nextLine();  
 String result ="?";  
  
 Pattern pattern=Pattern.compile("[0-9]\*");  
 boolean JUDGE\_WHETHER\_A\_OK=false;  
 boolean JUDGE\_WHETHER\_B\_OK=false;  
  
 if(A.length()>4||!pattern.matcher(A).matches()||  
 Integer.parseInt(A)<1||Integer.parseInt(A)>1000){  
 A="?";  
 }  
 else {  
 JUDGE\_WHETHER\_A\_OK=true;  
 }  
 if(B.length()>4||!pattern.matcher(B).matches()||  
 Integer.parseInt(B)<1||Integer.parseInt(B)>1000){  
 B="?";  
 }  
 else {  
 JUDGE\_WHETHER\_B\_OK=true;  
 }  
  
 if(JUDGE\_WHETHER\_A\_OK&&JUDGE\_WHETHER\_B\_OK){  
 int sum=Integer.parseInt(A)+Integer.parseInt(B);  
 result=String.valueOf(sum);  
 }  
  
 System.out.println(A+" + " +B+" = "+result);  
  
 }  
 }  
}  
  
  
  
  
/\*import java.util.Scanner;  
 import java.util.regex.Pattern;  
  
public class Main {  
  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
 // 输入A，到第一个空格就停止  
 String A = scanner.next();  
 // 输入B，到第二个空格  
 // 如果直接使用 String B = scanner.nextLine(); 会出错  
 // 它把 A 后面的那个空格也一起读取了，会对B的判断出错  
 // 修改一下，分开读，如果B后面还有字符串，就是不合法  
 String B = scanner.next();  
 String afterB = scanner.nextLine();  
 String result = "?";  
  
  
 // 编辑正则表达式，匹配数字 0-9  
 Pattern pattern = Pattern.compile("[0-9]\*");  
 boolean A\_ok = false;  
 boolean B\_ok = false;  
  
 // 因为正常的 A,B in [1, 1000]，所以先从字符串长度判断，然后判断是不是数字  
 // 如果是数字，再判断是不是在 [1, 1000]  
 if (A.length() > 4 || !pattern.matcher(A).matches() ||  
 Integer.parseInt(A) < 1 || Integer.parseInt(A) > 1000) {  
 // 只要一个条件满足 ，A 就是错误字符  
 A = "?";  
 } else {  
 // 否则 A 就是符合条件的整数，标记一下  
 A\_ok = true;  
 }  
  
 // 如果B的长度大于4，或者包含非数字，或不在[1, 1000]范围内，或者B后面还有字符，都算非法  
 if (B.length() > 4 || !pattern.matcher(B).matches() ||  
 Integer.parseInt(B) < 1 || Integer.parseInt(B) > 1000 ||  
 afterB.length() > 0) {  
 B = "?";  
 } else {  
 B\_ok = true;  
 }  
  
 // 如果 A 、B 都符合条件，就计算和  
 if (A\_ok && B\_ok) {  
 int sum = Integer.parseInt(A) + Integer.parseInt(B);  
 result = String.valueOf(sum);  
 }  
  
 // 输出结果  
 System.out.println(A + " + " + B + " = " + result);  
  
 }  
}  
\*/

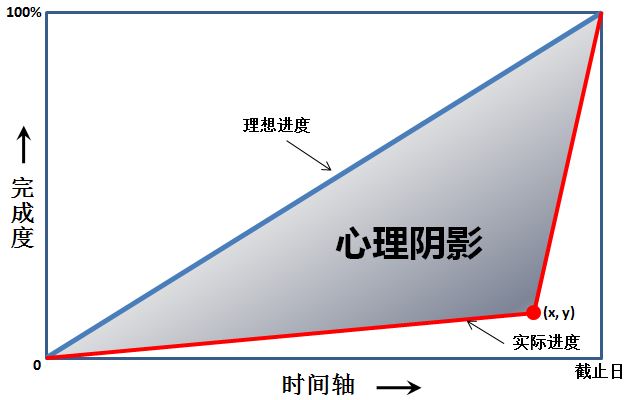
/\*G. 统计奇偶数  
  
Description  
给定 N 个正整数，请统计奇数和偶数各有多少个？  
Input  
输入第一行给出一个正整 N（≤1000）；第 2 行给出 N 个非负整数，以空格分隔。  
Output  
在一行中先后输出奇数的个数、偶数的个数。中间以 1 个空格分隔。  
Sample Input  
9  
88 74 101 26 15 0 34 22 77  
Sample Output  
3 6  
  
 \*/  
package C;  
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.in);  
 while (scan.hasNext()) {  
 int n=scan.nextInt();  
 int ODD\_NUM=0;  
 int EVEN\_NUM=0;  
 while (n!=0){  
 int input = scan.nextInt();  
 if(input%2==0){  
 EVEN\_NUM++;  
 }  
 if(input%2==1){  
 ODD\_NUM++;  
 }  
 n--;  
 }  
 System.out.println(ODD\_NUM+" "+EVEN\_NUM);  
 ODD\_NUM=0;  
 EVEN\_NUM=0;  
 }  
 }  
}

/\*H. 实现学生成绩的排序  
Description  
一个班级有 n 个学生，每个学生有姓名、学号和一门课程的考试成绩，现在按成绩从高到低重新排序输出，如果成绩相同则按学号从小到大顺序输出。  
Input  
第一个数据 n 表示学生人数，后面跟 n 组数据，每组数据包括学生姓名、学号、考试成绩。  
Output  
将各学生信息按要求重新排序后输出。  
Sample Input  
3  
a 110102 86  
b 110103 89  
c 110101 86  
Sample Output  
name:b;number:110103;score:89  
name:c;number:110101;score:86  
name:a;number:110102;score:86  
  
 \*/  
package H;  
import java.util.\*;  
class student{  
 String name;  
 String num;  
 int score;  
 public student(String name,String num,int score){  
 this.name=name;  
 this.num=num;  
 this.score=score;  
 }  
}  
public class Main {  
 public static void main(String arg[]){  
 int n;  
 Scanner sc=new Scanner(System.in);  
 student s[]=new student[1005];  
 while(sc.hasNext()){  
 n=sc.nextInt();  
 String k,m;  
 int q;  
 for(int i=1;i<=n;i++){  
 k=sc.next();  
 m=sc.next();  
 q=sc.nextInt();  
 s[i]=new student(k,m,q);  
 }  
 for(int i=1;i<n;i++){  
 for(int j=1;j<=n-i;j++){  
 if(s[j].num.compareTo(s[j+1].num)>0){  
 student tmp=s[j];  
 s[j]=s[j+1];  
 s[j+1]=tmp;  
 }  
 }  
 }  
 for(int i=1;i<n;i++){  
 for(int j=1;j<=n-i;j++){  
 if(s[j].score<s[j+1].score){  
 student tmp=s[j];  
 s[j]=s[j+1];  
 s[j+1]=tmp;  
 }  
 }  
 }  
 for(int i=1;i<=n;i++){  
 System.out.println("name:"+s[i].name+";number:"+s[i].num+";score:"+s[i].score);  
 }  
 }  
 }  
}

Java 实践与练习 2 (计 19-7、8）

/\*A.6 翻了  
Time Limit: 1000 MS Memory Limit: 32768 K  
Total Submit: 202 (60 users) Total Accepted: 63 (60 users) Special Judge: No  
Description  
“666” 是一种网络用语，大概是表示某人很厉害、我们很佩服的意思。最近又衍生出另一个数字 “9”，意思是 “6 翻了”，实在太厉害的意思。如果你以为这就是厉害的最高境界，那就错啦 —— 目前的最高境界是数字 “27”，因为这是 3 个 “9”！  
本题就请你编写程序，将那些过时的、只会用一连串 “6666……6” 表达仰慕的句子，翻译成最新的高级表达。  
Input  
输入在一行中给出一句话，即一个非空字符串，由不超过 1000 个英文字母、数字和空格组成，以回车结束。  
Output  
从左到右扫描输入的句子：如果句子中有超过 3 个连续的 6，则将这串连续的 6 替换成 9；但如果有超过 9 个连续的 6，则将这串连续的 6 替换成 27。其他内容不受影响，原样输出。  
Sample Input  
it is so 666 really 6666 what else can I say 6666666666  
  
  
  
Sample Output  
it is so 666 really 9 what else can I say 27  
  
  
  
Author  
天梯赛  
  
 \*/  
package A;  
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.in);  
 while (scan.hasNext()) {  
 String str = scan.nextLine();  
 int left = 0;  
 for (int i = 0; i < str.length(); i++) {  
 int count = 0;  
 left = i;  
 while (str.charAt(i) == '6') {  
 i++;  
 count++;  
 if (i > str.length() - 1) {  
 break;  
 }  
 }  
 if (count > 3 && count <= 9) {  
 str = str.replaceFirst(str.substring(left, i), "9");  
 i = left;  
 } else if (count > 9) {  
 str = str.replaceFirst(str.substring(left, i), "27");  
 i = left;  
 }  
 }  
 System.out.println(str);  
 }  
 }  
}

/\*B. 吃鱼还是吃肉  
Description  
国家给出了 8 岁男宝宝的标准身高为 130 厘米、标准体重为 27 公斤；8 岁女宝宝的标准身高为 129 厘米、标准体重为 25 公斤。  
现在你要根据小宝宝的身高体重，给出补充营养的建议。  
  
Input  
输入在第一行给出一个不超过 10 的正整数 N，随后 N 行，每行给出一位宝宝的身体数据：  
性别 身高 体重  
其中性别是 1 表示男生，0 表示女生。身高和体重都是不超过 200 的正整数。  
  
Output  
对于每一位宝宝，在一行中给出你的建议：  
如果太矮了，输出：duo chi yu!（多吃鱼）；  
如果太瘦了，输出：duo chi rou!（多吃肉）；  
如果正标准，输出：wan mei!（完美）；  
如果太高了，输出：ni li hai!（你厉害）；  
如果太胖了，输出：shao chi rou!（少吃肉）。  
先评价身高，再评价体重。两句话之间要有 1 个空格。  
Sample Input  
4  
0 130 23  
1 129 27  
1 130 30  
0 128 27  
Sample Output  
ni li hai! duo chi rou!  
duo chi yu! wan mei!  
wan mei! shao chi rou!  
duo chi yu! shao chi rou!  
Author  
天梯赛  
\*/  
  
package B;  
   
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args){  
 Scanner scan=new Scanner(System.in);  
 while (scan.hasNext()){  
 int n=scan.nextInt();  
 int[][]arr=new int[n][3];  
 for(int i=0;i<n;i++){  
 for(int j=0;j<3;j++){  
 arr[i][j]= scan.nextInt();  
 }  
 }  
  
 for(int i=0;i<n;i++) {  
  
 if (arr[i][0] == 0) {  
 if (arr[i][1] == 129) {  
 System.out.print("wan mei! ");  
 }  
 if (arr[i][1] > 129) {  
 System.out.print("ni li hai! ");  
 }  
 if (arr[i][1] < 129) {  
 System.out.print("duo chi yu! ");  
 }  
  
  
 if (arr[i][2] == 25) {  
 System.out.println("wan mei!");  
 }  
 if (arr[i][2] > 25) {  
 System.out.println("shao chi rou!");  
 }  
 if (arr[i][2] < 25) {  
 System.out.println("duo chi rou!");  
 }  
  
 }  
  
 if (arr[i][0] == 1) {  
 if (arr[i][1] == 130) {  
 System.out.print("wan mei! ");  
 }  
 if (arr[i][1] > 130) {  
 System.out.print("ni li hai! ");  
 }  
 if (arr[i][1] < 130) {  
 System.out.print("duo chi yu! ");  
 }  
  
 if (arr[i][2] == 27) {  
 System.out.println("wan mei!");  
 }  
 if (arr[i][2] > 27) {  
 System.out.println("shao chi rou!");  
 }  
 if (arr[i][2] < 27) {  
 System.out.println("duo chi rou!");  
 }  
 }  
  
 }  
 }  
  
 }  
}

/\*  
C. 心理阴影面积  
Description  
  
Input  
这是一幅心理阴影面积图。我们都以为自己可以匀速前进（图中蓝色直线），而拖延症晚期的我们往往执行的是最后时刻的疯狂赶工（图中的红色折线）。由红、蓝线围出的面积，就是我们在做作业时的心理阴影面积。  
  
现给出红色拐点的坐标 （x,y），要求你算出这个心理阴影面积。  
  
Output  
输入在一行中给出 2 个不超过 100 的正整数 x 和 y，并且保证有 x>y。这里假设横、纵坐标的最大值（即截止日和最终完成度）都是 100。  
Sample Input  
在一行中输出心理阴影面积。  
  
友情提醒：三角形的面积 = 底边长 x 高 / 2；矩形面积 = 底边长 x 高。嫑想得太复杂.  
  
Sample Output  
90 10  
Hint  
4000  
 \*/  
package C;  
   
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner scan = new Scanner(System.in);  
 while (scan.hasNext()) {  
 int x = scan.nextInt();  
 int y = scan.nextInt();  
 System.out.println(50 \* Math.abs(x - y));  
 }  
 }  
}

Java 实践与练习 1 (计 19-7、8）

/\*A.HelloWorld  
Description  
编写程序，输出一个短句。  
Input  
从键盘输入任意一个字符串。  
Output  
将这个字符串显示出来。  
Sample Input  
Hello World!  
Sample Output  
Hello World!  
  
 \*/  
package A;  
import java.util.Scanner;  
  
/\*\*  
 \* @author 86188  
 \*/  
public class HelloWorld {  
 public static void main(String[] args) {  
 Scanner cin = new Scanner(System.in);  
 while (cin.hasNext()) {  
// Scanner sc = new Scanner(System.in);  
 String str = "";  
 str = cin.nextLine();  
 System.out.println(str);  
 }  
 }  
}

/\*B. 比较大小  
Description  
本题输入任意 3 个整数，按从小到大输出。  
Input  
输入 3 个整数，其间以空格分隔。  
Output  
在一行中将 3 个整数从小到大输出，其间以空格分隔。  
Sample Input  
4 2 8  
Sample Output  
2 4 8  
  
 \*/  
package B;  
import java.util.\*;  
  
/\*\*  
 \* @author 86188  
 \*/  
public class Main {  
 public static void main (String[] args){  
 int a=0,b=0,c=0;  
 Scanner cin=new Scanner(System.in);  
 while (cin.hasNext()) {  
 a = cin.nextInt();  
 b = cin.nextInt();  
 c = cin.nextInt();  
 if (a > b) {  
 int temp = 0;  
 temp = a;  
 a = b;  
 b = temp;  
 }  
  
 if (b > c) {  
 int temp2 = 0;  
 temp2 = b;  
 b = c;  
 c = temp2;  
 }  
  
 if (a > b) {  
 int temp3 = 0;  
 temp3 = a;  
 a = b;  
 b = temp3;  
 }  
 System.out.print(a + " ");  
 System.out.print(b + " ");  
 System.out.println(c);  
 }  
 }  
}

/\*C. 比较大小（2）  
Description  
将输入的任意 3 个整数从小到大输出。  
Input  
输入一行 3 个任意整数，其间以空格分隔。  
Output  
在一行中将 3 个整数按从小到大输出，其间以 “->" 相连  
Sample Input  
4 2 8  
Sample Output  
2->4->8  
  
 \*/  
package C;  
import java.util.\*;  
  
/\*\*  
 \* @author 86188  
 \*/  
public class Main {  
 public static void main (String[] args){  
 int a=0,b=0,c=0;  
 Scanner cin=new Scanner(System.in);  
 while (cin.hasNext()) {  
 a = cin.nextInt();  
 b = cin.nextInt();  
 c = cin.nextInt();  
 if (a > b) {  
 int temp = 0;  
 temp = a;  
 a = b;  
 b = temp;  
 }  
  
 if (b > c) {  
 int temp2 = 0;  
 temp2 = b;  
 b = c;  
 c = temp2;  
 }  
  
 if (a > b) {  
 int temp3 = 0;  
 temp3 = a;  
 a = b;  
 b = temp3;  
 }  
 System.out.print(a + "->");  
 System.out.print(b + "->");  
 System.out.println(c);  
 }  
 }  
}

/\*D. 计算给定正整数的累加和  
Description  
从键盘读取一组正整数，输出这组正整数的累加和。其中第一个数为这组数的个数.  
Input  
输入一组正整数，其中第一个数为这组数的个数，各数之间用空格隔开。  
Output  
这组正整数的累加和。  
Sample Input  
5 1 3 4 6 1  
3 2 4 6  
  
Sample Output  
15  
12  
  
 \*/  
package D;  
import java.util.\*;  
  
/\*\*  
 \* @author 86188  
 \*/  
public class Main {  
 public static void main (String[] args){  
 int num=0;  
 int integer=0;  
 int sum=0;  
 Scanner cin=new Scanner(System.in);  
 while (cin.hasNext()) {  
 num= cin.nextInt();  
 while (num!=0){  
 integer=cin.nextInt();  
 sum+=integer;  
 num--;  
 }  
 System.out.println(sum);  
 sum=0;  
 }  
 }  
}

/\*E. 计算给定整数的累加和  
Description  
从键盘读取一组整数，分别输出这组整数中正整数和负整数的累加和。其中第一个数为这组数的个数.  
Input  
输入一组整数，其中第一个数为这组数的个数。各数之间用空格隔开。  
Output  
分别输出这组数中正数的累加和，以及负数的累加和。  
Sample Input  
5 -1 3 -4 6 1  
3 2 4 6  
Sample Output  
positive sum=10  
negtive sum=-5  
positive sum=12  
negtive sum=0  
  
 \*/  
  
package E;  
import java.util.\*;  
  
/\*\*  
 \* @author 86188  
 \*/  
public class Main {  
 public static void main (String[] args){  
 int num=0;  
 int integer=0;  
 int positiveSum=0;  
 int negtiveSum=0;  
 Scanner cin=new Scanner(System.in);  
 while (cin.hasNext()) {  
 num= cin.nextInt();  
 while (num!=0){  
 integer=cin.nextInt();  
 if(integer>=0) {  
 positiveSum += integer;  
 }  
 else if (integer<0){  
 negtiveSum += integer;  
 }  
 num--;  
 }  
 System.out.println("positive sum="+positiveSum);  
 System.out.println("negtive sum="+negtiveSum);  
 positiveSum=0;  
 negtiveSum=0;  
 }  
  
 }  
}

/\*F. 找出唯一数  
Description  
在一组数中找出唯一出现一次的数字。  
Input  
每组数组第一行是一个数字 n。  
Output  
第二行恰好有（2 \* n + 1）个数字，保证有 n 个数字恰好出现两次，有一个数字只出现了一次。  
0 <= n <= 10000000。  
0 <= 其他数字 <= 1000000000。  
Sample Input  
输出你找到的数字。  
Sample Output  
5  
1 2 3 4 5 6 5 4 3 2 1  
0  
1  
1  
1 1 2  
  
  
  
Hint  
6  
1  
2  
  
  
 \*/  
package F;  
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
  
 Scanner cin = new Scanner(System.in);  
 int n;  
 while (cin.hasNext()) {  
 n = cin.nextInt();  
 int ans = 0;  
 int[] a = new int[2 \* n + 1];  
 for (int i = 0; i < 2 \* n + 1; i++) {  
 a[i]=cin.nextInt();  
 ans ^= a[i];  
 }  
 System.out.println(ans);  
  
 }  
 }  
}

Java 常用函数

substring (): 截取字符串

1、str＝str.substring (int beginIndex); 截取掉 str 从下标为 beginIndex 开始的字符串，将剩余字符串赋值给 str；

例：String str = “abcdefg”;

String s = str.substring (1); 返回 "bcdefg";

str＝str.substring (int beginIndex，int endIndex); 截取 str 中从 beginIndex 开始至 endIndex 结束时的字符串，并将其赋值给 str;

例：String str = “abcdefg”;

String s = str.substring (1,3); 返回 "bc"; 不包含下标 3 的字符

concat ()：连接字符串

str = str.concat (“aaa”); 连接连个字符串，相当于 “+” 号的作用。

replace (): 替换字符

Str = str.replace (“a”,”b”); 将 str 字符串中的 a 替换成 b。

Str = str.replaceAll (“a”,”b”); 将 str 字符串中的所有的 a 替换成 b。

Str = str.replaceFirst (“a”,”b”); 将 str 字符串中的第一个 a 替换成 b。

例：String str = “abac”; 将 “a” 替换成 “\a”, 由于 “\” 是 java 的转义字符，所以在替换时，需再加个 “\” 转义字符：

Str = str.replace(“a”,”\a”); -->\ab\ac

Str = str.replaceAll(“a”,”\\a”); -->\ab\ac

Str = str.replaceFirst(“a”,”\\a”);–>\abac

trim (): 去掉字符串首尾的空格

例：str = “abc”;

Str = str.trim(); -->”abc”

valueof (): 将其他类型的字符转换成指定类型

例：int i = 12;

String str = String.valueof(i); -->”12”

Integer i = Integer.valueof(str); --> 12

toLowerCase (): 转为小写

例：String str = “a12AsB”;

Str = str.toLowerCase(); -->”a12asb”

toUpperCase (): 转为大写

例：String str = “a12AsB”;

Str = str.toUpperCase();–>”A12ASB”

length (): 取字符长度

例：String str = “abcd”;

int i = str.length(); --> i=4

charAt (): 去一个字符

例：String str = “abcd”;

char c = str.charAt(2); --> c=’c’

getChars (): 截取多个字符

void getChars(int sourceStart,int sourceEnd,char target[],int targetStart)

sourceStart: 开始截取位置的下标

sourceEnd：结束截取位置的下标

target []：接收截取后字符的数组

targetStart：定义 target 数组开始的下标值

例： String s=”this is a demo of the getChars method.”;

char buf[]=new char[20];

s.getChars(10,14,buf,0); -->buf = “demo”

getBytes (): 将字符转换成字节

例： String s = “Hello! 你好！”;

byte[] bytes = s.getBytes();

indexOf (): 查找字符或者子串第一次出行的位置

lastIndexOf (): 查找字符或者子串最后一次出行的位置

例：String str = “abcd”;

int position = str.indexOf(“b”); -->position = 1

String str = “abcdechgscla”;

int position = str.lastIndexOf(“c”); -->position = 9

split (): 分割字符串

如果用 “.” “|” “+” “\*” 作为分割的话，必须得加 "\";

例：String [] aa = “aaa|bbb|ccc”.split ("\|"); 这样才能得到正确的结果

还有如果想在串中使用 "“字符，则也需要转义。首先要表达"aaaa\bbbb"这个串就应该用"aaaa\bbbb”, 如果要分隔就应该这样才能得到正确结果：

String[] aa = “aaa\bbb\bccc”.split("\\");



**System.out.printf**

简介

System.out.printf 与 C 语言中的 printf 使用方法类似，可以向控制台 (Console) 输出指定格式的内容。

System.out.printf 的方法比使用操作符 ‘+’进行拼接的方法要更加清晰、简介，前者只需要定义好输出格式和占位置的值，而后者需要用 ‘+’进行拼接，在显示变量较多的情况下需要用到大量的’+’, 一旦非变量内容需要进行大量修改，后者改动量更大。

控制符



示例

String name = "Tony";  
int age = 23;  
System.out.printf("name : %s, age : %d", name, age);

输出：  
name : Tony, age : 23

·字符串

整数转字符串：

1 String str = String. valueOf(a);

字符串转数组

1 char[] chars str. toCharArray（）;

字符串长度：

1 int lenght str. length（）;

字符串获取某个字符：

1 char c = str. charAt(i);

字符串替换

1 str. replace('a', 'b');

字符串去除空格

1 str. trim（）;

字符串拼接

1 StringBuilder sb = new StringBuilder( )；

2 sb.append(a);

3 sb.length(a);

4 sb.deleteCharAt(index);

截取字符串：

1 str.substring(start,end)含前不含后，也可以只有

2 str. substring(start)

字符串获取某个字符的位置

1 str.indexof('a')

字符串分割

1 str. split("\\s+");按空格分割

list/dequeString重组成String

1 String.join("", list);

2 String.join("", deque);

数组初始化

1 char[ ] array new char[len];

2 int[ ] arr={1,2,3}

3 int[ ] arr = new int[ ]{1,2,3,4,5,6};

数组长度：

1 int length arr.length;

数组拷贝：

T[ ] target Arrays.copyOfRange(T[ ] original,int start,int to)

2 含前不含后

3 public static void arraycopy(Object src, int srcPos, Object dest, int destPos, int length)

4

5 src:源数组；

6

7 srcPos:源数组要复制的起始位置；

8

9 dest:目的数组；

10

11 destPos:目的数组放置的起始位置；

12

13 length:复制的长度。

数组自定义排序：

1 Arrays.sort(nums,new Comparator<Integer>（）{

2 public int compare(Integer a, Integer b){

3 return b-a;

4 }

5});

函数

1.Math.max(a,b);

2.Math.min(a,b);

3.Math.pow(a,digit);

4.Random random = new Random();

5.random.nextInt(2);返回0/1

//找出唯一数

class Solution {

public int singleNumber(int[] nums) {

int ans=0;

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

ans^=nums[i]; //异或运算

}

return ans;

}

}

/JAVA将字符串拆分放入数租.(截取字符串：截取第i个位置到第i+1个位置的字符串)

String a = "abc";

String[] arr = new String[a.length()];

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

{ arr[i] = a.substring(i, (i+1));}

//数组输出：System.out.println(Arrays.toString(arr));

//如何将字串 String 转换成整数 int

1). int i = Integer.parseInt([String]);

2). int i = Integer.valueOf(my\_str).intValue();

//int 转换成字串 String

1.) String s = String.valueOf(i);

2.) String s = Integer.toString(i);

3.) String s = "" + i;

//String型转换为float||float型转换为String型

s.floatValue();||f.toString();

十进制转成十六进制：

Integer.toHexString(int i)

十进制转成八进制

Integer.toOctalString(int i)

十进制转成二进制

Integer.toBinaryString(int i)

十六进制转成十进制

Integer.valueOf("FFFF",16).toString()

八进制转成十进制

Integer.valueOf("876",8).toString()

二进制转十进制

Integer.valueOf("0101",2).toString()

进制转换如何写（二，八，十六）不用算法

Integer.toBinaryString

Integer.toOctalString

Integer.toHexString

字串出现的次数

public class SubStringCount {

　　public static void main(String[] args) {

　　　　String str1 = "abcjavanbajavasrcjavammmjava";

　　　　String str2 = "java";

　　　　int count = 0;

　　　　int index=str1.indexOf(str2);

　　　　while(index!=-1){

　　　　　　index = str2.length() + index;

　　　　　　count++;

　　　　　　index = str1.indexOf(str2, index);

　　　　}

　　　　System.out.println(count);

　　}

}

public class SearchStringEmp {

public static void main(String[] args) {

String strOrig = "Hello World!Hello World!";

int intIndex = strOrig.indexOf("Hello");

if(intIndex == - 1){

System.out.println("没有找到字符串 Hello");

}else{

System.out.println("Hello 字符串位置:" + intIndex);

}

}

}

质数查找

import java.util.Scanner;

public class MyFun2 {

public static void main(String[] args) {

// TODO Auto-generated method stub

Scanner in = new Scanner(System.in);

int x;

System.out.println("你要求取那个值的范围素数:");

x = in.nextInt();//统计x范围有多少素数；

boolean isPrime = true;//布尔类型;判断一个数是不是素数的标志位

int primeNUM = 0;

for(int i=2;i<=x;i++){

for(int j=2;j<i;j++){

if(i%j==0){

isPrime = false;

break;

}

}

if(isPrime){

System.out.print(i+"\t");

primeNUM++;

if(primeNUM%5==0)

System.out.println();

}

isPrime = true;

}

System.out.println();

System.out.println("素数"+ x +"范围内,一共有："+ primeNUM +"个素数");

}

}

1，charAt()

通过String类的charAt()的方法来获取字符串中的每一个字符，然后将其拼接为一个新的字符串

public static String CharAtreverse(String s) {

int length = s.length();

String reverse = "";

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

reverse = s.charAt(i) + reverse;

return reverse;

}

2，toCharArray()

通过String的toCharArray()方法可以获得字符串中的每一个字符并转换为字符数组，然后用一个空的字符串从后向前一个个的拼接成新的字符串。

public static String reverseCharArray(String s) {

char[] array = s.toCharArray();

String reverse = "";

for (int i = array.length - 1; i >= 0; i--) {

reverse += array[i];

}

return reverse;

}

3，reverse()

通过StringBuiler或StringBuffer的reverse()的方法，此方法最快

public static String reverseStringBuilder(String s) {

StringBuilder sb = new StringBuilder(s);

String afterReverse = sb.reverse().toString()；

return afterReverse;

}

字符串替换：

1.replace（）

该方法的作用是替换字符串中所有指定的字符，然后生成一个新的字符串。经过该方法调用以后，原来的字符串不发生改变。例如：

String s = “abcat”；

String s1 = s.replace（‘a’，‘1’）；

该代码的作用是将字符串s中所有的字符a替换成字符1，生成的新字符串s1的值是“1bc1t”，而字符串s的内容不发生改变。

2. replaceAll（）

如果需要将字符串中某个指定的字符串替换为其它字符串，则可以使用replaceAll（），例如：

String s = “abatbac”；

String s1 = s.replaceAll（“ba”，“12”）；

该代码的作用是将字符串s中所有的字符串“ab”替换为“12”，生成新的字符串“a12t12c”，而字符串s的内容也不发生改变。

3.replaceFirst（）

如果只需要替换第一个出现的指定字符串时，可以使用replaceFirst（），例如：

String s = “abatbac”；

String s1 = s. replaceFirst （“ba”，“12”）；

该代码的作用是只将字符串s中第一次出现的字符串“ab”替换为字符串“12”，则字符串s1的值是“a12tbac”，字符串s的内容也不发生改变。

正则

只能输入整数，正数负数零都可以："^[+-]?[0-9]+$"或\d\*

整数或者小数：^[0-9]+\.{0,1}[0-9]{0,2}$

只能输入数字："^[0-9]\*$"。

只能输入n位的数字："^\d{n}$"。

只能输入至少n位的数字："^\d{n,}$"。

只能输入m~n位的数字：。"^\d{m,n}$"

只能输入零和非零开头的数字："^(0|[1-9][0-9]\*)$"。

只能输入有两位小数的正实数："^[0-9]+(.[0-9]{2})?$"。

只能输入有1~3位小数的正实数："^[0-9]+(.[0-9]{1,3})?$"。

只能输入非零的正整数："^\+?[1-9][0-9]\*$"。

只能输入非零的负整数："^\-[1-9][]0-9"\*$。

只能输入长度为3的字符："^.{3}$"。

只能输入由26个英文字母组成的字符串："^[A-Za-z]+$"。

只能输入由26个大写英文字母组成的字符串："^[A-Z]+$"。

只能输入由26个小写英文字母组成的字符串："^[a-z]+$"。

只能输入由数字和26个英文字母组成的字符串："^[A-Za-z0-9]+$"。

只能输入由数字、26个英文字母或者下划线组成的字符串："^\w+$"。

public class三角形面积 {

public static void main(String[] args) {

Scanner scan =new Scanner(System.in);

System.out.println("输入三角形的三边");

int a = scan.nextByte();

int b = scan.nextByte();

int c = scan.nextByte();

float s =(a+b+c)/2f;

float S = (float) Math.sqrt(s\*(s-a)\*(s-b)\*(s-c));

if (a+b>c && b+c>a && a+c>b){

System.out.println(S);

}

else{

System.out.println("不成立三角形");

}

}

}

import java.util.Scanner;

public class CommonDiviser

{

private static Scanner input = new Scanner(System.in);

public static void main(String[] args)

{

int n = input.nextInt();

int m = input.nextInt();

int zdgys = result(n, m);

System.out.println("最大公约数是:" + zdgys);

System.out.println("最大公倍数是:" + n \* m / zdgys);

input.close();

}

public static int result(int n, int m)

{

// 余数

int k = 0;

// 使n>m

if (n < m)

{

int t = 0;

t = n;

n = m;

m = t;

}

// 直到余数为0

while (m != 0)

{

k = n % m;

n = m;

m = k;

}

System.out.println(n);

return n;

}

}

Eclipse用法

Ctrl+1 快速修复

Ctrl+Shift+F 格式化当前代码

Debug 全局 继续 F8

导包 ctrl+Shift+/

eclipse中补全代码快捷键,默认Alt+/

找到左侧菜单栏

1.首先，点击导航栏中的window选项。

2.在该选项中，点击show view选项。

3.在show view选项中，如果发现有project Exlorer选项就直接点击该选项，就会自动出现左侧的项目栏了。如...

4.在show view的右拉菜单中点击最下方的other选项。

5.找到general选项，并在该选项中就可以找到projectexplorer选项