第四章　循环结构

第一节 for语句

第二节 while语句

第三节 do-while语句

第四节 循环嵌套

第一节 for语句

一、语句格式

格式1

```
for (控制变量初始化表达式；条件表达式；增量表达式)
    语句1;
```

说明：语句1是for循环语句的循环体，它将在满足条件的情况下被重复执行。

格式2

```
for (控制变量初始化表达式；条件表达式；增量表达式)
{   语句1;
    语句2;
    ……
}
```

说明：循环体部分由多个语句构成，应由一对花括号括起来，构成一个语句块的形式程序风格提示：写for循环语句时，循环体的语句相对于for缩进两格。

第一节 for语句

二、语句执行过程

for语句的执行过程可由以下4步来描述。

(1)执行“控制变量初始化语句”，使控制变量获得一个初值。

(2)判断控制变量是否满足“条件表达式”，若满足条件则执行一遍循环体，否则结束整个for语句，继续执行for循环下面的句子。

(3)根据增量表达式，计算出控制变量所得到的新值

(4)自动转到第（2）步。

第一节 for语句

三、语句格式举例

(1)将控制变量从1变到100，增量为1

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

(2)将控制变量从100变到1，增量为 - 1

for(i=100;i>=1;--i)

(3)控制变量从7变到77，增量为7

for(i=7;i<=77;i+=7)

(4)控制变量从20变到2，增量为 - 2

for(int i=20;i>=2；i-=2)

(5)按所示数列改变控制变量值：99、88、77、66、55、44、33、22、11、0，增量为-11

for(int j=99;j>=0;j-=11)

(6)控制变量i和j共同进行循环控制，i从1变到99，j从2变到100，增量均为2。

for（int i=1,j=2;i<=99&&j<=100；i+=2,j+=2)

需要说明的是：可以在for循环“ 控制变量初始化语句”中声明变量（如上面最后3个例子），这些变量只在for循环结构中有效，离开了该for结构，变量就无效了。

第一节 for语句

例4.1 输出1—100之间所有偶数。

```
#include <iostream>
using namespace std;
int main (){
  for (int i=2; i<=100 ; i+=2)
  cout << i << "  ";
   return 0;
}
```

例4.2 利用for循环,计算输出1+2+…+100的和

```
#include <iostream>
using namespace std;
int main ()
{
    int sum=0;
    for (int i=1; i<=100 ; ++i)
       sum+=i;
     cout << sum;
    return 0;
}
```

第一节 for语句

例4.3 利用for循环计算n！的值。分析：n！＝1*2*3...*n

```
#include <cstdio>
using namespace std;
int main ()
{
  long long s;        //Noip2010开始C++语言中long long类型允许使用
  int n;              //n不能定义为long long，否则for语句死循环
  s=1;
  scanf("%d",&n);
  for (int i=1; i<=n ; ++i)//若s定义为int，当n=13时s的值就溢出了
    s*=i;
  printf("%lld\n",s);                    //低版本也可用printf("%I64d\n",s)
  return 0;
}
```

【说明】：当n>=13时，s值超过了int类型的表示范围。还有一种比int更大的类型，称为long long，它的表示范围是-2^{63}～$2^{63}-1$，比-10^{19}～10^{19}略窄，而我们一直使用的int范围是-2^{31}～$2^{31}-1$,只比$-2*10^9$～$2*10^9$略宽。

输入输出long long也可以借助于printf和scanf语句，但对应的占位符却是和平台与编译器相关的：在linux中，gcc很统一的用%lld；在windows中，**MinGW的gcc和VC6可用%I64d**；高版本编译器下**windows可以使用%lld**。

第一节 for语句

例4.4 利用for循环,分别计算1—100中奇数的和、偶数的和。

```
#include <iostream>
using namespace std;
int main ( ){
  int  jssum=0;
  int  ossum=0;
  for (int js=1,os=2;js<=99&&os<=100;js+=2 ,os+=2 )
    {
      jssum+=js;
      ossum+=os;
    }
  cout << "the sum of  odd  numbers  1 to 100 is : "
<<jssum<<endl;
  cout << "the sum of  even  numbers  1 to 100 is : "
<<ossum<<endl;
  return 0;
}
```

说明：我们也可以在for循环初始化或增值表达式部分中放一条以上的语句，中间用逗号隔开。

【上机练习】

1.求平均年龄【1.5编程基础之循环控制01】

班上有学生若干名，给出每名学生的年龄（整数），求班上所有学生的平均年龄，保留到小数点后两位。

输入:

第一行有一个整数n（1<= n <= 100），表示学生的人数。其后n行每行有1个整数，表示每个学生的年龄，取值为15到25。

输出:

输出一行，该行包含一个浮点数，为要求的平均年龄，保留到小数点后两位。

样例输入:

2

18

17

样例输出:

17.50

【上机练习】

2.均值【1.5编程基础之循环控制02】

给出一组样本数据，包含n个浮点数，计算其均值，精确到小数点后4位。

输入:

输入有两行，第一行包含一个整数n（n小于100），代表样本容量；第二行包含n个绝对值不超过1000的浮点数，代表各个样本数据。

输出:

输出一行，包含一个浮点数，表示均值，精确到小数点后4位。

样例输入:

2

1.0 3.0

样例输出:

2.0000

【上机练习】

3.求整数的和与均值【1.5编程基础之循环控制03】

读入n(1<=n<=10000)个整数，求它们的和与均值。

输入:

输入第一行是一个整数n，表示有n个整数。

第2~n+1行每行包含1个整数。每个整数的绝对值均不超过10000。

输出:

输出一行，先输出和，再输出平均值(保留到小数点后5位)，两个数间用单个空格分隔。

样例输入:

```
4
344
222
343
222
```

样例输出:

```
1131 282.75000
```

【上机练习】

4.最高的分数【1.5编程基础之循环控制04】

孙老师讲授的《计算概论》这门课期中考试刚刚结束，他想知道考试中取得的最高分数。因为人数比较多，他觉得这件事情交给计算机来做比较方便。你能帮孙老师解决这个问题吗？

输入:

输入两行，第一行为整数n（1 <= n < 100），表示参加这次考试的人数.第二行是这n个学生的成绩，相邻两个数之间用单个空格隔开。所有成绩均为0到100之间的整数。

输出:

输出一个整数，即最高的成绩。

样例输入:

5

85 78 90 99 60

样例输出:

99

【上机练习】

5.最大跨度值【1.5编程基础之循环控制05】

给定一个长度为n的非负整数序列，请计算序列的最大跨度值(最大跨度值 = 最大值减去最小值)。

输入:

一共2行，第一行为序列的个数n（1 <= n <= 1000)，第二行为序列的n个不超过1000的非负整数，整数之间以一个空格分隔。

输出:

输出一行，表示序列的最大跨度值。

样例输入:

6

3 0 8 7 5 9

样例输出:

9

【上机练习】

6.奥运奖牌计数【1.5编程基础之循环控制06】

2008年北京奥运会，A国的运动员参与了n天的决赛项目(1≤n≤17)。现在要统计一下A国所获得的金、银、铜牌数目及总奖牌数。输入第1行是A国参与决赛项目的天数n，其后n行，每一行是该国某一天获得的金、银、铜牌数目。输出4个整数，为A国所获得的金、银、铜牌总数及总奖牌数。

输入:

输入n＋1行，第1行是A国参与决赛项目的天数n，其后n行，每一行是该国某一天获得的金、银、铜牌数目，以一个空格分开。

输出:

输出1行，包括4个整数，为A国所获得的金、银、铜牌总数及总奖牌数，以一个空格分开。

样例输入:

3
1 0 3
3 1 0
0 3 0

样例输出:

4 4 3 11

【上机练习】

7.奇数求和【1.5编程基础之循环控制07】

计算非负整数m到n(包括m和n)之间的所有奇数的和，其中，m 不大于 n，且 n 不大于300。例如 m=3, n=12, 其和则为：3+5+7+9+11=35。

输入:

两个数 m 和 n，两个数以一个空格分开，其中 0 <= m <= n <= 300 。

输出:

输出一行，包含一个整数，表示m 到 n（包括m 和 n ）之间的所有奇数的和

样例输入:

7 15

样例输出:

55

【上机练习】

8.满足条件的数【1.5编程基础之循环控制08】

将正整数m和n之间(包括m和n)能被17整除的数累加，其中0<m<n<1000。

输入:

一行，包含两个整数m和n，其间，以一个空格间隔。

输出:

输出一行，包行一个整数，表示累加的结果。

样例输入:

50 85

样例输出:

204

【上机练习】

9.整数的个数【1.5编程基础之循环控制09】

给定k(1<k<100)个正整数，其中每个数都是大于等于1，小于等于10的数。写程序计算给定的k个正整数中，1，5和10出现的次数。

输入:

输入有两行：第一行包含一个正整数k，第二行包含k个正整数，每两个正整数用一个空格分开。

输出:

输出有三行，第一行为1出现的次数，，第二行为5出现的次数，第三行为10出现的次数。

样例输入:

5

1 5 8 10 5

样例输出;

1

2

1

【上机练习】

10.与指定数字相同的数的个数【1.5编程基础之循环控制10】

输出一个整数序列中与指定数字相同的数的个数。输入包含2行：第1行为N和m，表示整数序列的长度(N<=100)和指定的数字；第2行为N个整数，整数之间以一个空格分开。输出为N个数中与m相同的数的个数。

输入:

第1行为N和m，表示整数序列的长度(N<=100)和指定的数字， 中间用一个空格分开；

第2行为N个整数，整数之间以一个空格分开。

输出:

输出为N个数中与m相同的数的个数。

样例输入:

3 2

2 3 2

样例输出:

2

【上机练习】

11.乘方计算【1.5编程基础之循环控制11】

给出一个整数a和一个正整数n(-1000000<=a<=1000000，1<=n<=10000)，求乘方a^n，即乘方结果。最终结果的绝对值不超过1000000。

输入:

一行，包含两个整数a和n。-1000000 <= a <= 1000000，1 <= n <= 10000。

输出:

一个整数，即乘方结果。题目保证最终结果的绝对值不超过1000000。

样例输入:

2 3

样例输出:

8

【上机练习】

12.人口增长【1.5编程基础之循环控制12】

我国现有x亿人口，按照每年0.1%的增长速度，n年后将有多少人？保留小数点后四位。

输入:

一行，包含两个整数x和n，分别是人口基数和年数，以单个空格分隔。

输出:

输出最后的人口数，以亿为单位，保留到小数点后四位。1 <=x<= 100, 1<=n<=100。

样例输入:

13 10

样例输出:

13.1306

【上机练习】

13.菲波那契数【1.5编程基础之循环控制13】

菲波那契数列是指这样的数列：数列的第一个和第二个数都为1，接下来每个数都等于前面2个数之和。给出一个正整数k，要求菲波那契数列中第k个数是多少。

输入：

输入一行，包含一个正整数k。（1 <= k <= 46）

输出：

输出一行，包含一个正整数，表示菲波那契数列中第k个数的大小

样例输入：

19

样例输出：

4181

【上机练习】

14.鸡尾酒疗法【1.5编程基础之循环控制15】

鸡尾酒疗法，指“高效抗逆转录病毒治疗”。人们在鸡尾酒疗法的基础上又提出了很多种改进的疗法。为了验证这些治疗方法是否在疗效上比鸡尾酒疗法更好，可用通过临床对照实验的方式进行。假设鸡尾酒疗法的有效率为x，新疗法的有效率为y，如果y-x大于5%，则效果更好，如果x-y大于5%，则效果更差，否则称为效果差不多。下面给出n组临床对照实验，其中第一组采用鸡尾酒疗法，其他n-1组为各种不同的改进疗法。请写程序判定各种改进疗法效果如何。

输入：

第一行为整数n（1<n<=20）；其余n行每行两个整数，第一个整数是临床实验的总病例数(小于等于10000)，第二个疗效有效的病例数。这n行数据中，第一行为鸡尾酒疗法的数据，其余各行为各种改进疗法的数据。

输出：

有n-1行输出，分别表示对应改进疗法的效果：如果效果更好，输出better；如果效果更差，输出worse；否则输出same。

样例输入:

```
5
125 99
112 89
145 99
99 97
123 98
```

样例输出:

```
same
worse
better
same
```

【上机练习】

15.救援【1.5编程基础之循环控制16】

救生船从大本营出发，营救若干屋顶上的人回到大本营，屋顶数目以及每个屋顶的坐标和人数都将由输入决定，求出所有人都到达大本营并登陆所用的时间。

在直角坐标系的原点是大本营，救生船每次从大本营出发，救了人之后将人送回大本营。坐标系中的点代表屋顶，每个屋顶由其位置坐标和其上的人数表示。救生船每次从大本营出发，以速度50 米/分钟驶向下一个屋顶，达到一个屋顶后，救下其上的所有人，每人上船1分钟，船原路返回，达到大本营，每人下船0.5分钟。假设原点与任意一个屋顶的连线不穿过其它屋顶。

输入：

第一行，一个整数，表示屋顶数n。接下来依次有n 行输入，每一行上包含两个表示屋顶相对于大本营的平面坐标位置的实数(单位是米)、一个表示人数的整数。

输出：

救援需要的总时间，精确到分钟(向上取整)。

样例输入:

```
1
30 40 3
```

样例输出:

```
7
```

【上机练习】

16.津津的储蓄计划【1.5编程基础之循环控制19】Noip2012提高组第1题

津津的零花钱一直都是自己管理。每个月的月初妈妈给津津300元钱，津津会预算这个月的花销，并且总能做到实际花销和预算的相同。

为了让津津学习如何储蓄，妈妈提出，津津可以随时把整百的钱存在她那里，到了年末她会加上20%还给津津。因此津津制定了一个储蓄计划：每个月的月初，在得到妈妈给的零花钱后，如果她预计到这个月的月末手中还会有多于100元或恰好100元，她就会把整百的钱存在妈妈那里，剩余的钱留在自己手中。

例如11月初津津手中还有83元，妈妈给了津津300元。津津预计11月的花销是180元，那么她就会在妈妈那里存200元，自己留下183元。到了11月月末，津津手中会剩下3元钱。

现在请你根据2004年1月到12月每个月津津的预算，判断会不会出现这种情况。如果不会，计算到2004年年末，妈妈将津津平常存的钱加上20%还给津津之后，津津手中会有多少钱。

输入：

包括12行数据，每行包含一个小于350的非负整数，分别表示1月到12月津津的预算。

输出：

只包含一个整数。如果储蓄计划实施过程中出现某个月钱不够用的情况，输出-X，X表示出现这种情况的第一个月；否则输出到2004年年末津津手中会有多少钱。

【上机练习】

样例 #1输入：

290
230
280
200
300
170
340
50
90
80
200
60

样例 #1输出：

-7

样例 #2输入：

290
230
280
200
300
170
330
50
90
80
200
60

样例 #2输出：

1580

【上机练习】

17.药房管理【1.5编程基础之循环控制20】

随着信息技术的蓬勃发展，医疗信息化已经成为医院建设中必不可少的一部分。计算机可以很好地辅助医院管理医生信息、病人信息、药品信息等海量数据，使工作人员能够从这些机械的工作中解放出来，将更多精力投入真正的医疗过程中，从而极大地提高了医院整体的工作效率。

对药品的管理是其中的一项重要内容。现在药房的管理员希望使用计算机来帮助他管理。假设对于任意一种药品，每天开始工作时的库存总量已知，并且一天之内不会通过进货的方式增加。每天会有很多病人前来取药，每个病人希望取走不同数量的药品。如果病人需要的数量超过了当时的库存量，药房会拒绝该病人的请求。管理员希望知道每天会有多少病人没有取上药。

输入：

共3行,第一行是每天开始时的药品总量m。

第二行是这一天取药的人数n(0<n<=100)。

第三行共有n个数，分别记录了每个病人希望取走的药品数量(按照时间先后的顺序)。

输出：

只有1行，为这一天没有取上药品的人数。

样例输入:

30

6

10 5 20 6 7 8

样例输出:

2

【上机练习】

18.正常血压【1.5编程基础之循环控制21】

监护室每小时测量一次病人的血压，若收缩压在90-140之间并且舒张压在60-90之间(包含端点值)则称之为正常，现给出某病人若干次测量的血压值，计算病人保持正常血压的最长小时数。

输入：

第一行为一个正整数n(n<100)，其后有n行，每行2个正整数，分别为一次测量的收缩压和舒张压。

输出：

输出仅一行，血压连续正常的最长小时数。

样例输入:

```
4
100 80
90 50
120 60
140 90
```

样例输出:

```
2
```

【上机练习】

19.统计满足条件的4位数【1.5编程基础之循环控制23】

给定若干个四位数，求出其中满足以下条件的数的个数：个位数上的数字减去千位数上的数字，再减去百位数上的数字，再减去十位数上的数字的结果大于零。

输入：

输入为两行，第一行为四位数的个数n，第二行为n个的四位数。(n<=100)

输出：

输出为一行，包含一个整数，表示满足条件的四位数的个数。

样例输入:

5

1234 1349 6119 2123 5017

样例输出:

3

【上机练习】

20.求分数序列和【1.5编程基础之循环控制29】

有一个分数序列 q1/p1,q2/p2,q3/p3,q4/p4,q5/p5,.... ,其中qi+1= qi+ pi, pi+1=qi, p1= 1, q1= 2。比如这个序列前6项分别是2/1,3/2,5/3,8/5,13/8,21/13。求这个分数序列的前n项之和。

输入:

输入有一行，包含一个正整数n(n <= 30)。

输出:

输出有一行，包含一个浮点数，表示分数序列前n项的和，精确到小数点后4位。

样例输入:

2

样例输出:

3.5000

【上机练习】

21.计算分数加减表达式的值【1.5编程基础之循环控制30】

编写程序，输入n的值，求 1/1 - 1/2 + 1/3 - 1/4 + 1/5 - 1/6 + 1/7 - 1/8 + ... + (-1)^(n-1)·1/n 的值。

输入:

输入一个正整数n。1 <= n <= 1000。

输出:

输出一个实数，为表达式的值，保留到小数点后四位。

样例输入:

2

样例输出:

0.5000

【上机练习】

22．7647 余数相同问题【小学奥数7647】

已知三个正整数a，b，c。现有一个大于1的整数x，将其作为除数分别除a，b，c，得到的余数相同。

请问满足上述条件的x的最小值是多少？数据保证x有解。

输入：

一行，三个不大于1000000的正整数a，b，c，两个整数之间用一个空格隔开。

输出：

一个整数，即满足条件的x的最小值。

样例输入:

300 262 205

样例输出:

19

【上机练习】

23．分苹果【小学奥数7826】

把一堆苹果分给n个小朋友，要使每个人都能拿到苹果，而且每个人拿到的苹果数都不同的话，这堆苹果至少应该有多少个？

输入:

一个不大于1000的正整数n，代表小朋友人数。

输出:

一个整数，表示满足条件的最少苹果个数。

样例输入:

8

样例输出:

36

【上机练习】

24. 求小数的某一位【小学奥数7830】

分数a/b化为小数后，小数点后第n位的数字是多少？

输入:

三个正整数a，b，n，相邻两个数之间用单个空格隔开。0<a<b< 100，1<=n<=10000。

输出:

一个数字。

样例输入:

1 2 1

样例输出:

5

【上机练习】

25. 计算星期几【小学奥数7831】

假设今天是星期日，那么过ab天之后是星期几？

输入:

两个正整数a，b，中间用单个空格隔开。0<a<=100, 0<b<=10000。

输出:

一个字符串，代表过ab天之后是星期几。

其中，Monday是星期一，Tuesday是星期二，Wednesday是星期三，Thursday是星期四，Friday是星期五，Saturday是星期六，Sunday是星期日。

样例输入:

3 2000

样例输出:

Tuesday

【上机练习】

26．幂的末尾【小学奥数7833】

幂ab的末3位数是多少？

输入：

两个正整数a，b。1<=a<=100，1<=b<=10000。

输出：

从高位到低位输出幂的末三位数字，中间无分隔符。若幂本身不足三位，在前面补零。

样例输入:

7 2011

样例输出:

743

第二节 while语句

一、语句格式

格式1

```
while (条件表达式)
  语句1;
```

说明：语句1是while循环语句的循环体，它将在满足条件的情况下被重复执行。

格式2

```
while (条件表达式)
  {  语句1;
     语句2;
     ……
   }
```

说明：循环体部分由多个语句构成，应由一对花括号括起来，构成一个语句块的形式。

程序风格提示：写while循环语句时，循环体的语句相对于while缩进两格。

第二节 while语句

二、语句执行过程

(1)计算作为循环控制条件表达式的值，得到逻辑真或假，假定用M表示。

(2)若M为真，则执行了一遍循环体，否则离开循环，结束整个while语句的执行。

(3)循环体的所有语句执行结束后，自动转向第(1)步执行。

三、格式举例

(1) i=0;

```
while (i<10)
  ++i;
```

功能：当i的值小于10，重复执行++i语句

(2) cin>>x;

```
while（x<0)
  cin>>x;
```

功能：当输入的数据小于0时，重复读数据。

第二节 while语句

例4.5 求s=1 +2 +3……+n，当加到第几项时，s的值会超过1000?

程序如下:

```
#include <iostream>
using namespace std;
int main ()
{
  int n=0,s=0;
  while (s<=1000)
  {
    ++n;
    s+=n;
  }
  cout<<n;
  return 0;
}
```

第二节 while语句

例4.6 求两个正整数m，n的最大公约数。

分析：求两个整数的最大公约数可以采用辗转相除法。以下是辗转相除法的算法：分别用m，n，r表示被除数、除数、余数；

1)求m除以n的余数r；

2)当r!=0,执行第3)步；若r==0，则n为最大公约数,算法结束。

3)将n的值赋给m，将r的值赋给n；再求m除以n的余数r。

4)转到第2)步

```
#include <iostream>
using namespace std;
int main ()
{
  int m,n,r;
  cin>>m>>n;
  r =m % n;
  while (r!=0)                  //也可以使用 while (r),c++中 非0即真
  {
    m=n;
    n=r;
    r=m % n;
  }
  cout<<"最大公约数="<<n<<endl;
  return 0;
}
```

第二节 while语句

例4.7　编一程序求满足不等式1+ 1/2 + 1/3 ...+1/n >=5的最小n值。

分析：此题不等式的左边是一个求和的算式，该和式中的数据项个数是未知的，也正是要求出的。对于和式中的每个数据项，对应的通式为1/i，i=1，2，...n。

所以可采用循环累加的方法来计算出它的值。设循环变量为i，它应从1开始取值，每次增加1，直到和式的值不小于5为止，此时的i值就是所求的n。设累加变量为s，在循环体内把1/i的值累加到s上。

根据以上分析，采用while循环编写出程序如下：

```
#include <iostream>
using namespace std;
int main ()
{
  int i=0;
  float s=0;
  while(s<5) //当s的值还未超过5时
  {
    ++i;
    s+=1.0/i;
  }
  cout<<i;
  return 0;
}
```

若采用for循环来写，则如下所示：

```
#include <iostream>
using namespace std;
int main ()
{
  int i;
  float s=0;
  for(i=1;s<5;++i)
    s+=1.0/i;
  cout<<i-1;
  return 0;
}
```

第二节 while语句

例4.8 数据统计

输入一些整数，求出它们的最小值、最大值和平均值（保留3位小数）。输入保证这些数都是不超过1000的整数。

样例输入：2 8 3 5 1 7 3 6

样例输出：1 8 4.375

【参考程序】

```
#include<cstdio>
int main()
{
  int x,n=0,min,max,s=0;
  while (scanf("%d",&x)==1)
  {
    s+=x;
    if (x<min) min=x;
    if (x>max) max=x;
    ++n;
  }
  printf("%d %d %.3lf\n",min,max,(double)s/n);
  return 0;
}
```

第二节 while语句

【优化程序】

```
#include<cstdio>
#define INF 100000000
int main()
{
    int x,n=0,min=INF,max=-INF,s=0;
    while (scanf("%d",&x)==1)
            //scanf("%d",&x)!=EOF，如果没数据可读，scanf返回EOF
    {
        s+=x;
        if (x<min) min=x;
        if (x>max) max=x;
        ++n;
    }
    printf("%d %d %.3lf\n",min,max,(double)s/n);
    return 0;
}
```

第二节 while语句

最后，我们来更仔细地研究一下输入输出。研究对象就是经典的“A+B”问题：输入若干对整数，输出每对之和。假设每个整数不超过109，一共不超过106个数对。

第1种方法是：

```
#include<cstdio>
int main()
{
  int a,b;
  while(scanf("%d%d",&a,&b)==2)
  printf("%d\n",a+b);
  return 0;
}
```

第2种方法也许更加常用（你再也不用记住%d、%lf等恼人的占位符了）：

```
#include<iostream>
using namespace std;
int main()
{
  int a,b;
  while(cin >> a >> b )
    cout << a+b <<endl;
  return 0;
}
```

【上机练习】

1.球弹跳高度的计算【1.5编程基础之循环控制17】

一球从某一高度h落下(单位米)，每次落地后反跳回原来高度的一半，再落下。编程计算气球在第10次落地时，共经过多少米？第10次反弹多高？

输出包含两行，第1行：到球第10次落地时，一共经过的米数。第2行：第10次弹跳的高度。

输入:

输入一个整数h，表示球的初始高度。

输出:

第1行：到球第10次落地时，一共经过的米数。

第2行：第10次弹跳的高度。

注意：结果可能是实数，结果用double类型保存。

提示：输出时不需要对精度特殊控制，用cout << ANSWER，或者printf("%g", ANSWER)即可。

样例输入:

20

样例输出:

59.9219

0.0195313

【上机练习】

2.角谷猜想【1.5编程基础之循环控制18】

谓角谷猜想，是指对于任意一个正整数，如果是奇数，则乘3加1，如果是偶数，则除以2，得到的结果再按照上述规则重复处理，最终总能够得到1。如，假定初始整数为5，计算过程分别为16、8、4、2、1。程序要求输入一个整数，将经过处理得到1的过程输出来。

输入:

一个正整数N(N <= 2,000,000)

输出:

从输入整数到1的步骤，每一步为一行，每一部中描述计算过程。最后一行输出"End"。如果输入为1，直接输出"End"。

样例输入:

5

样例输出:

5*3+1=16

16/2=8

8/2=4

4/2=2

2/2=1

End

【上机练习】

3.级数求和【1.5编程基础之循环控制24】Noip2002普及组第1题

已知：Sn= 1＋1/2＋1/3＋...＋1/n。显然对于任意一个整数K，当n足够大的时候，Sn大于K。现给出一个整数K（1<=k<=15），要求计算出一个最小的n，使得Sn＞K。

输入:

一个整数K。

输出:

一个整数n。

样例输入:

1

样例输出:

2

【上机练习】

4.分离整数的各个数【1.5编程基础之循环控制25】

给定一个整数n(1<=n<=100000000)，要求从个位开始分离出它的每一位数字。从个位开始按照从低位到高位的顺序依次输出每一位数字。

输入:

输入一个整数，整数在1到100000000之间。

输出:

从个位开始按照从低位到高位的顺序依次输出每一位数字。数字之间以一个空格分开。

样例输入:

123

样例输出:

3 2 1

【上机练习】

5.数字反转【1.5编程基础之循环控制26】Noip2011普及组第1题

给定一个整数，请将该数各个位上数字反转得到一个新数。新数也应满足整数的常见形式，即除非给定的原数为零，否则反转后得到的新数的最高位数字不应为零，例如输入-380，反转后得到的新数为-83。

输入:

输入共 1 行，一个整数N。

-1,000,000,000 ≤ N≤ 1,000,000,000。

输出:

输出共 1 行，一个整数，表示反转后的新数。

样例输入:

样例 #1:

123

样例 #2:

-380

样例输出

样例 #1:

321

样例 #2:

-83

【上机练习】

6.含k个3的数【1.5编程基础之循环控制27】

输入两个正整数m和k，其中1<m<100000，1<k<5 ，判断m 能否被19整除，且恰好含有k个3，如果满足条件，则输出YES，否则，输出NO。 例如，输入：43833 3，满足条件，输出YES。如果输入：39331 3，尽管有3个3，但不能被19整除，也不满足条件，应输出NO。

输入:

m 和 k 的值，中间用单个空格间隔。

输出:

满足条件时输出 YES，不满足时输出 NO。

样例输入:

43833 3

样例输出:

YES

第三节 do-while语句

一、语句格式

格式1

```
do
  语句1;
while (条件表达式);
```

说明：语句1是do-while的循环体。

格式2

```
do
 {
    语句1;
    语句2;
     ......
 }
while (条件表达式);
```

说明：循环体部分由多个语句构成，应由一对花括号括起来，构成一个语句块的形式。

二、语句执行过程

(1)执行一遍循环体。

(2)求出作为循环条件的“条件表达式”的值，若为逻辑值真则自动转向第(1)步，否则结束do循环的执行过程，继续执行其后面的语句。

在do语句的循环体中也可以使用break语句，用它来非正常结束循环的执行。

第三节 do-while语句

三、实例

例4.9 对于求两个正整数m，n的最大公约数可以用do—while实现。

代码如下，请完善：

```
#include <iostream>
using namespace std;
int main ()
{
  int m，n，r;
  cin>>m>>n;
  do                                    //辗转相除法
  {
    r =m % n;
    m=____;
    n=_____;
  }
  while ( _______ );
  cout<<"the greatest common divisor is:"<<_____;
  return 0;
}
```

第三节 do-while语句

例4.10　求1992个1992的乘积的末两位数是多少？

【分析】积的个位与十位数只与被乘数与乘数的个位与十位数字有关，所以本题相当于求1992个92相乘，而且本次的乘积是下一次相乘的被乘数，因此也只需取末两位参与运算就可以了。

```
#include<iostream>
using namespace std;
int main()
{
   int a=1,t=0;
    do
    {
        ++t;
        a=(a*92)%100;
    }while (t!=1992);
    cout<<a<<endl;
    return 0;
}
```

第三节 do-while语句

例4.11　校体操队到操场集合,排成每行2人,最后多出1人;排成每行3人,也多出1人;分别按每行排4,5,6人,都多出1人;当排成每行7人时,正好不多。求校体操队至少多少人?

【分析】①设校体操队为x人,根据题意x应是7的倍数,因此x的初值为7,以后用x+=7)改变x值;　②为了控制循环,　用逻辑变量yes为真(true)　使循环结束;

③如果诸条件中有一个不满足,　yes　的值就会为假(false),　就继续循环。

```
#include<iostream>
using namespace std;
int main()
{
    bool yes;
    int x=0;
    do
    {
        yes=true;
        x+=7;
        if (x%2!=1) yes=false;
        if (x%3!=1) yes=false;
        if (x%4!=1) yes=false;
        if (x%5!=1) yes=false;
        if (x%6!=1) yes=false;
    }while (yes==false);  //直到yes的值为真
    cout<<"All="<<x;
    return 0;
}
```

程序中对每个x值，都先给yes 赋真值，只有在循环体各句对x进行判断时，都得到“通过”（此处不赋假值）才能保持真值。

【上机练习】

1.球弹跳高度的计算【1.5编程基础之循环控制17】

一球从某一高度h落下(单位米)，每次落地后反跳回原来高度的一半，再落下。编程计算气球在第10次落地时，共经过多少米？第10次反弹多高？

输出包含两行，第1行：到球第10次落地时，一共经过的米数。第2行：第10次弹跳的高度。

输入:

输入一个整数h，表示球的初始高度。

输出:

第1行：到球第10次落地时，一共经过的米数。

第2行：第10次弹跳的高度。

注意：结果可能是实数，结果用double类型保存。

提示：输出时不需要对精度特殊控制，用cout << ANSWER，或者printf("%g", ANSWER)即可。

样例输入:

20

样例输出:

59.9219

0.0195313

【上机练习】

2.角谷猜想【1.5编程基础之循环控制18】

谓角谷猜想，是指对于任意一个正整数，如果是奇数，则乘3加1，如果是偶数，则除以2，得到的结果再按照上述规则重复处理，最终总能够得到1。如，假定初始整数为5，计算过程分别为16、8、4、2、1。程序要求输入一个整数，将经过处理得到1的过程输出来。

输入:

一个正整数N(N <= 2,000,000)

输出:

从输入整数到1的步骤，每一步为一行，每一部中描述计算过程。最后一行输出"End"。如果输入为1，直接输出"End"。

样例输入:

5

样例输出:

5*3+1=16

16/2=8

8/2=4

4/2=2

2/2=1

End

【上机练习】

3.级数求和【1.5编程基础之循环控制24】Noip2002普及组第1题

已知：Sn= 1＋1/2＋1/3＋...＋1/n。显然对于任意一个整数K，当n足够大的时候，Sn大于K。现给出一个整数K（1<=k<=15），要求计算出一个最小的n，使得Sn＞K。

输入:

一个整数K。

输出:

一个整数n。

样例输入:

1

样例输出:

2

【上机练习】

4.分离整数的各个数【1.5编程基础之循环控制25】

给定一个整数n(1<=n<=100000000)，要求从个位开始分离出它的每一位数字。从个位开始按照从低位到高位的顺序依次输出每一位数字。

输入:

输入一个整数，整数在1到100000000之间。

输出:

从个位开始按照从低位到高位的顺序依次输出每一位数字。数字之间以一个空格分开。

样例输入:

123

样例输出:

3 2 1

【上机练习】

5.数字反转【1.5编程基础之循环控制26】Noip2011普及组第1题

给定一个整数，请将该数各个位上数字反转得到一个新数。新数也应满足整数的常见形式，即除非给定的原数为零，否则反转后得到的新数的最高位数字不应为零，例如输入-380，反转后得到的新数为-83。

输入:

输入共 1 行，一个整数N。

-1,000,000,000 ≤ N≤ 1,000,000,000。

输出:

输出共 1 行，一个整数，表示反转后的新数。

样例输入:

样例 #1:

123

样例 #2:

-380

样例输出

样例 #1:

321

样例 #2:

-83

【上机练习】

6.含k个3的数【1.5编程基础之循环控制27】

输入两个正整数m和k，其中1<m<100000，1<k<5 ，判断m 能否被19整除，且恰好含有k个3，如果满足条件，则输出YES，否则，输出NO。 例如，输入：43833 3，满足条件，输出YES。如果输入：39331 3，尽管有3个3，但不能被19整除，也不满足条件，应输出NO。

输入:

m 和 k 的值，中间用单个空格间隔。

输出:

满足条件时输出 YES，不满足时输出 NO。

样例输入:

43833 3

样例输出:

YES

第四节 循环嵌套

例4.12 求 S=1!+2!+3!+....+10!

分析：这个问题是求10以内自然数的阶乘之和，可以用for循环来实现。程序结构如下：

```
for(i=1;i<=10;++i)
{
  (1)i阶乘的值存到t;        //t=i!
   (2)累加t到s中;           //s+=t
}
```

显然根据以上结构，通过10次的循环可以求出1！，2！，…10!,并不断累加起来，求出s。而求t=i!,又可以用一个for循环来实现：

```
t=1;
for (j=1;j<=i;++j)
    t*=j;
```

第四节 循环嵌套

因此整个程序为：

```
#include <iostream>
using namespace std;
int main (){
  int t,s;
  s=0;
  for(int i=1;i<=10;++i)
    {
    t=1;
    for (int j=1;j<=i;++j)                    //求i!
      t*=j;
      s+=t;                                   //累加i!
    }
  cout<<s;
  return 0;
}
```

以上程序是一个for循环的嵌套。这种方法是比较容易想到的，但实际上对于求i！，我们可以根据求出的（i-1）！乘上i即可得到，而无需重新从1再累乘到i。

第四节 循环嵌套

因此程序可改为:

```
#include <iostream>
using namespace std;
int main ()
{
  int t=1,s=0;
  for(int i=1;i<=10;++i)
  {
    t*=i;                    //t为上一个数的i-1的阶乘值，再乘以i即为i!
    s+=t;                    //累加i!
  }
  cout<<s;
  return 0;
}
```

显然第二个程序的效率要比第一个高得多。第一个程序要进行1+2+3+…+10=55次循环，而第二程序进行10次循环。若题目中求的是1！+2！+…+1000！，则两个程序的效率区别更明显。

第四节 循环嵌套

例4.13 一个炊事员上街采购，用500元钱买了90只鸡，其中母鸡一只15元,公鸡一只10元，小鸡一只5元，正好把钱买完。问母鸡，公鸡，小鸡各买了多少只？

【分析】设母鸡i只,公鸡j只,则小鸡为90-i-j只,则15*i+ 10* j+(90-i-j)*5=500,显然一个方程求两个未知数是不能直接求解。必须组合出所有可能的i,j值，看是否满足条件。这里i的值可以是0到33，j的值可以0到50。源程序如下：

```
#include <iostream>
using namespace std;
int main ()
{
  int k;
  for (int i=0;i<=33;++i)                    //枚举母鸡的数量
  for (int j=0;j<=50;++j)                    //枚举公鸡的数量
  {
    k=90-i-j;
    if (15*i+10*j+k*5==500)
    {
      cout<<"母鸡有"<<i<<"只,"<<"公鸡有"<<j<<"只,"<<"小鸡有"<<k<<"只" <<endl;
    }
  }
  return 0;
}
```

第四节 循环嵌套

例4.14 利用for循环语句输出图4-1中的三角形。

```
*
**
***
****
*****
```

图4-1

```
#include <iostream>
using namespace std;
int main ()
{
  for (int i=1; i<=5; ++i)                    //控制行数
  {
    for (int j=1; j<=i; ++j)                  //输出一行中的*数
    cout<<"*";
    cout<<endl;                               //换行
  }
  return 0;
}
```

第四节 循环嵌套

例4.15　求100－999中的水仙花数。若三位数ABC，ABC=$A^3+B^3+C^3$，则称ABC为水仙花数。例如153，$1^3+5^3+3^3=1+125+27=153$，则153是水仙花数。

【分析】 根据题意，采用三重循环来求解。由于循环次数一定，用for循环最为简单。程序如下：

```
#include<iostream>
#include<iomanip>                                    //调用setw函数需注明使用该库
using namespace std;
int main()
{
    for (int a=1; a<=9; ++a)
      for (int b=0; b<=9; ++b)
         for (int c=0; c<=9; ++c)
           {
             if (a*a*a+b*b*b+c*c*c==a*100+b*10+c)
               cout<<setw(6)<<a*100+b*10+c;   //setw函数控制输出场宽
           }
    return 0;
}
```

运行结果：

153　370　371　407

第四节 循环嵌套

同时也可以采用一个for循环来求解，表面上看好像优于三重循环，实际上却比上面的程序效率低，请同学们自己分析。
程序如下：

```
#include<iostream>
#include<iomanip>
using namespace std;
int main()
{
   int a,b,c;
   for (int m=100; m<=999; ++m)
   {
       a=m/100;                          //m的百位
       b=(m%100)/10;                     //m的十位
       c=m%10;                           //m的个位
       if (a*a*a+b*b*b+c*c*c==m)
         cout<<setw(6)<<m;
   }
   return 0;
}
```

第四节 循环嵌套

例4.16　输出100—200中所有的素数。

分析：我们可对100-200之间的每一个整数进行判断，若它是为素数，则输出。而对于任意整数i，根据素数定义，我们从2开始，到sqrt（i），找i的第一个约数，若找到第一个约数，则i必然不是素数。

程序如下：

```
#include <iostream>
#include<cmath>                                    //在Dev C++中可调用数学函数库cmath
using namespace std;
int main ()
{
  int x;
  for (int i=100;i<=200;++i)
  {
     x=2;
     while(x<=floor(sqrt(i))&&(i%x!=0))     //floor为取整函数，需调用math.h库
     x=x+1;                                 //在枚举的范围内并且没有出现约数则继续枚举
     if ( x>floor(sqrt(i)))
       cout<<i<<"\t";
  }
  return 0;
}
```

第四节 循环嵌套

例4.17 输出所有形如aabb的四位完全平方数（即前两位数字相等，后两位数字也相等）。

【分析】分支和循环结合在一起时威力特别强大：我们枚举所有可能的aabb，然后判断它们是否为完全平方数。注意，a的范围是1～9，b可以是0。主程序如下：

```
for (a=1; a<=9; a++)
  for (b=0; b<=9; b++)
    if (aabb是完全平方数) printf("%d\n",aabb);
```

另一个思路是枚举平方根x，参考程序如下：

```
#include<cstdio>
int main()
{
        int n=0,hi,lo;
        for (int x=1 ;  ; ++x)                    //可以直接从x=32开始枚举
        {
                n=x*x;
                if (n<1000) continue;
                if (n>9999) break;
                hi = n/100;
                lo = n%100;
                if (hi/10 == hi%10 && lo/10 == lo%10) printf("%d\n",n);
        }
        return 0;
}
```

第四节 循环嵌套

例4.18 阶乘之和. 输入n，计算S=1! + 2! + 3! + … + n!的末6位(不含前导0)。n<=10^6，n!表示前n个正整数之积。

样例输入：10

样例输出：37913

【分析】这个任务并不难，引入累加变量S之后，核心算法只有一句话：for (i=1;i<=n;i++) S+=i!。不过C++语言并没有阶乘运算符，所以这句话只是伪代码，而不是真正的代码。事实上，我们还需要一次循环来计算i!：for (j=1;j<=i;++j) factorial*=j;。代码如下：

```
#include<cstdio>
int main()
{
        int n,s=0;
        scanf("%d",&n);
        for (int i=1;i<=n;++i)
        {
                int factorial=1;
                for (int j=1;j<=i;++j)
                  factorial*=j;
                s+=factorial;
        }
        printf("%d\n",s%1000000);
        return 0;
}
```

注意累乘器factorial(英文“阶乘”的意思)定义在循环里面。换句话说，每执行一次循环体，都要重新声明一次factorial，并初始化为1(想一想，为什么不是0)。因为只要末6位，所以输出时对106取模。

第四节 循环嵌套

当n=100时，输出-961703，直觉告诉我们：乘法溢出了。这个直觉很容易通过“输出中间变量”法得到验证，但若要解决这个问题，还需要一点数学知识。试一下$n=10^6$时输出什么？更会溢出，但是重点不在这里。事实上，它的速度太慢！让我们把程序改成“每步取模”的形式，然后加一个“计时器”，看看它到底有多慢。

```
#include<cstdio>
#include<ctime>
int main()
{
    const int MOD=1000000;
    int n,s=0;
    scanf("%d",&n);
    for (int i=1;i<=n;++i)
    {
      int factorial=1;
      for (int j=1;j<=i;++j)
      factorial=(factorial*j%MOD);
      s=(s+factorial)%MOD;
    }
    printf("%d\n",s);
    printf("Time used= %.2lf\n",(double)clock()/CLOCKS_PER_SEC);
    return 0; //输出时间包含键盘输入的时间，建议用文件输入输出，后面章节介绍文件
}
```

这个程序真正的特别之处在于计时函数clock()的使用。该函数返回程序目前为止运行的时间。这样，在程序结束之前调用它，便可获得整个程序的运行时间。这个时间除以常数CLOCKS_PER_SEC之后得到的值以“秒”为单位。

输入100000，按Enter键，系统迟迟不输出答案，原因在于程序中重复进行了多次阶乘运算，浪费了大量时间，具体优化方法请参考例4.12 。

【上机练习】

1.求阶乘的和【1.5编程基础之循环控制31】

给定正整数n，求不大于n的正整数的阶乘的和（即求1!+2!+3!+...+n!），输出阶乘的和。

输入:

输入有一行，包含一个正整数n（1 < n < 12）。

输出:

输出有一行：阶乘的和。

样例输入:

5

样例输出:

153

【上机练习】

2.求出e的值【1.5编程基础之循环控制32】

利用公式e = 1 + 1/1! + 1/2! + 1/3! + ... + 1/n! ，求e的值，要求保留小数点后10位。

输入:

输入只有一行，该行包含一个整数n（2<=n<=15），表示计算e时累加到1/n！。

输出:

输出只有一行，该行包含计算出来的e的值，要求打印小数点后10位。

样例输入:

10

样例输出:

2.7182818011

【上机练习】

3.计算多项式的值【1.5编程基础之循环控制33】

假定多项式的形式为x^n+x^(n-1)+...+x^2+x+1，请计算给定单精度浮点数x和正整数n值的情况下这个多项式的值。x在float范围内，n<=1000000。多项式的值精确到小数点后两位，保证最终结果在float范围内。

输入:

输入仅一行，包括x和n，用单个空格隔开。x在float范围内，n <= 1000000。

输出:

输出一个实数，即多项式的值，精确到小数点后两位。保证最终结果在float范围内。

样例输入:

2.0 4

样例输出:

31.00

【上机练习】

4.与7无关的数【1.5编程基础之循环控制34】

一个正整数，如果它能被7整除，或者它的十进制表示法中某一位上的数字为7，则称其为与7相关的数。现求所有小于等于n(n<100)与7无关的正整数的平方和。

输入:

输入为一行,正整数n(n < 100)

输出:

输出一行，包含一个整数，即小于等于n的所有与7无关的正整数的平方和。

样例输入:

21

样例输出:

2336

【上机练习】

5.数1的个数【1.5编程基础之循环控制35】

给定一个十进制正整数n(1<=n<=10000)，写下从1到n的所有整数，然后数一下其中出现的数字“1”的个数。

例如当n=2时，写下1,2。这样只出现了1个“1”；当n=12时，写下1，2，3，4，5，6，7，8，9，10，11，12。这样出现了5个“1”。

输入:

正整数n。1 <= n <= 10000。

输出:

一个正整数，即“1”的个数。

样例输入:

12

样例输出:

5

【上机练习】

6.数字统计【1.5编程基础之循环控制36】Noip2010普及组第1题

请统计某个给定范围[L, R]的所有整数中，数字2出现的次数。

比如给定范围[2, 22]，数字2在数2中出现了1次，在数12中出现1次，在数20中出现1次，在数21中出现1次，在数22中出现2次，所以数字2在该范围内一共出现了6次。

输入:

输入共 1 行，为两个正整数 L 和 R，之间用一个空格隔开。

输出:

输出共 1 行，表示数字 2 出现的次数。

样例输入:

样例 #1:

2 22

样例 #2:

2 100

样例输出:

样例 #1:

6

样例 #2:

20

【上机练习】

7.画矩形【1.5编程基础之循环控制37】

根据参数，画出矩形。输入四个参数：前两个参数为整数，依次代表矩形的高和宽（高不少于3行不多于10行，宽不少于5列不多于10列）；第三个参数是一个字符，表示用来画图的矩形符号；第四个参数为1或0，0代表空心，1代表实心。

输入:

输入一行，包括四个参数：前两个参数为整数，依次代表矩形的高和宽（高不少于3行不多于10行，宽不少于5列不多于10列）；第三个参数是一个字符，表示用来画图的矩形符号；第四个参数为1或0，0代表空心，1代表实心。

输出:

输出画出的图形。

样例输入

```
7 7 @ 0
```

样例输出

```
@@@@@@@
@     @
@     @
@     @
@     @
@     @
@@@@@@@
```

【上机练习】

8.质因数分解【1.5编程基础之循环控制38】Noip2012普及组第1题

已知正整数n是两个不同的质数的乘积，试求出较大的那个质数。

输入:

输入只有一行，包含一个正整数 n。

对于60%的数据，6 ≤ n ≤ 1000。

对于100%的数据，6 ≤ n ≤ 2*10^9。

输出:

输出只有一行，包含一个正整数 p，即较大的那个质数。

样例输入:

21

样例输出:

7

【上机练习】

9.第n小的质数【1.5编程基础之循环控制39】

输入一个正整数n，求第n小的质数。

输入:

一个不超过10000的正整数n。

输出:

第n小的质数。

样例输入:

10

样例输出:

29

【上机练习】

10. 金币【1.5编程基础之循环控制40】

国王将金币作为工资，发放给忠诚的骑士。第1天，骑士收到一枚金币；之后两天(第2天和第3天)里，每天收到两枚金币；之后三天(第4、5、6天)里，每天收到三枚金币；之后四天(第7、8、9、10天)里，每天收到四枚金币……这种工资发放模式会一直这样延续下去：当连续n天每天收到n枚金币后，骑士会在之后的连续n+1天里，每天收到n+1枚金币(n为任意正整数)。

你需要编写一个程序，确定从第一天开始的给定天数内，骑士一共获得了多少金币。

输入:

一个整数（范围1到10000），表示天数。

输出:

骑士获得的金币数。

样例输入:

6

样例输出:

14

【上机练习】

11．不定方程求解【小学奥数7650】

给定正整数a，b，c。求不定方程 ax+by=c 关于未知数x和y的所有非负整数解组数。

输入:

一行，包含三个正整数a，b，c，两个整数之间用单个空格隔开。每个数均不大于1000。

输出:

一个整数，即不定方程的非负整数解组数。

样例输入:

2 3 18

样例输出:

4