

C语言题目

一、数值计算问题

得分	
----	--

用 $\pi/4=1-1/3+1/5-1/7+\dots$ 公式求 π 的近似值，直到某一项的绝对值小于 10^{-4} 为止

```
1  #include <stdio.h>
2
3  void main()
4  {
5      double s=0,t=1;
6      int n=1;
7
8      while(t<=10000)
9      {
10         s+=n*1/t;
11         n=-n;
12         t=t+2;
13     }
14     printf("π 近似值为%.15lf\n",4*s);
15 }
```

数列——求 π 值

2. 计算公式：求 π 的值（分值：100.00）

【问题描述】

已知可以用下列公式计算 π 的近似值。给定一个 π 的近似值 e ，编程利用下列公式求得最接近 e 并且小于 e 的近似值 π ，以及迭代次数 n 。
$$\pi/2 = 1 + 1/3 + 2!/(3*5) + 3!/(3*5*7) + \dots + (n-1)!/(3*5*7* \dots * (2n-1))$$

【输入形式】

从控制台输入 e （ e 大于2，小于3.1415925）的值。

【输出形式】

输出最接近 e 并且小于 e 的近似值 π ，以及迭代次数 n （以一个空格分隔，并且输出 π 时要求小数点后保留7位有效数字）。

【样例输入】

```
3.14159
```

【样例输出】

```
3.1415896 18
```

【样例说明】

输入的 π 的近似值 e 为3.14159，当 n 为18时计算的 π 值为3.1415896，小于给定的精度值。当 n 为19时，计算的 π 值为3.1415912，大于给定的精度值，所以利用上述计算公式求得的最接近 e 并且小于 e 的 π 的近似值为3.1415896，对应的迭代次数为18。

注意：

为保证计算精度，请使用double数据类型保存计算数据。

【评分标准】

共有5个测试点。

```
1  #include <stdio.h>
2  int main()
3  {
4      double e;
5      scanf("%lf",&e);
6      double i,j=3.0,t=1.0,k=1.0,sum=1.0;
7      int count=0;
8      for(i=1.0;sum*2<e;i++){
9          t*=i;
10         k*=j;
11         sum+=t/k;
12         j+=2.0;
13         count++;
14     }
15     sum-=t/k;
16     printf("%.7lf %d",sum*2,count);
17     return 0;
18 }
```

特殊数

1. 相亲数a (分值: 100.00)

【问题描述】

2500年前数学大师毕达哥拉斯就发现，220和284两数之间存在着奇妙的联系：

220的因数之和（除了自身之外的因数）为：1+2+4+5+10+11+20+22+44+55+110=284，284的因数之和为：1+2+4+71+142=220。

毕达哥拉斯把这样的数对称称为相亲数。输入两个正整数（大于1并且在int的表示范围之内），分别计算它们所有因数之和（除了自身之外的因数），并判断它们是否是一对相亲数。

【输入形式】

从标准输入输入两个正整数，以一个空格分隔这两个正整数。

【输出形式】

以输入的先后顺序分行输出：输入的正整数，后跟英文逗号“，”，再从1开始输出其因数相加的公式（最大的因数之后没有加号），最后输出英文等号“=”和因数之和。注意：所有输出元素间无空格。若它们是一对相亲数，则在新的一行上输出1；若不是则在新的一行上输出0。

【样例输入1】

```
220 284
```

【样例输出1】

```
220,1+2+4+5+10+11+20+22+44+55+110=284
284,1+2+4+71+142=220
1
```

【样例输入2】

```
2560 3282
```

【样例输出2】

```
2560,1+2+4+5+8+10+16+20+32+40+64+80+128+160+256+320+512+640+1280=3578
3282,1+2+3+6+547+1094+1641=3294
0
```

```
1  #include <stdio.h>
2  int main()
3  {
4      int m,n;
5      scanf("%d %d",&m,&n);
6      int i,j,sum1=0,sum2=0;
7      int a[100],t=0,b[100],k=0;
8      for(i=1;i<m;i++){
9          if(m%i==0){
10             a[t]=i;
11             t++;
12             sum1+=i;
13         }
14     }
15     for(j=1;j<n;j++){
16         if(n%j==0){
17             b[k]=j;
18             k++;
19             sum2+=j;
20         }
21     }
22     printf("%d,",m);
23
24     for(i=0;i<t;i++){
25         if(i==t-1) printf("%d=",a[i]);
26         else printf("%d+",a[i]);
27     }
28     printf("%d\n",sum1);
29     printf("%d,",n);
30     for(i=0;i<k;i++){
31         if(i==k-1) printf("%d=",b[i]);
32         else printf("%d+",b[i]);
33     }
34     printf("%d\n",sum2);
35     if(sum1==n) printf("%d",1);
36     else printf("%d",0);
37     return 0;
}
```

二、遍试问题

得分	
----	--

显示从 3 开始小于 300 的所有双素数（双素数：如果 p 和 q 都是素数且 $q=p+2$ ，则 p/q 称为双素数，例如 3/5，5/7 都是双素数）。

```
1  #include <stdio.h>
2  int isPrime(int p)
3  {
4      int i;
5      for(i=2;i<=p/2;i++)
6      {
7          if(p%i==0)
8          {
9              return 0;
10         }
11     }
12     return 1;
13 }
14 void dblPrime()
15 {
16     int p,q,num;
17
18     num=0;
19     for(p=3;p<=98;p++)
20     {
21         if(isPrime(p)==1)
22         {
23             q=p+2;
24             if(isPrime(q)==1)
25             {
26                 printf("%d/%d  ",p,q);
27             }
28         }
29     }
30 }
31 void main()
32 {
33     printf("3~300的双素数:\n");
34     dblPrime();
35 }
```

字符串

使用凯撒加密算法对输入的英语单词加密，分别输入凯撒密钥 K 和一个英语单词，输出加密后的结果（凯撒加密，即移位加密，例如第 i 个字母变成第 $(i+K) \% 26$ 个字母，移位前是大写字母移位后仍为大写，移位前是小写移位后仍为小写，注意输入内容必须为英文字母）

```
1  #include <stdio.h>
2
3  void main()
4  {
5      int i,k;
6      char ch[80];
7      printf("输入英文原文: ");
8      gets(ch);
9      printf("输入密钥K: ");
10     scanf("%d",&k);
11
12     for(i=0;ch[i]!='\0';i++)
13     {
14         if(ch[i]>='A'&&ch[i]<='Z')
15             if((ch[i]+k)<=90)
16                 ch[i]=ch[i]+k;
17             else
18                 ch[i]=ch[i]+k-26;
19         else if(ch[i]>='a'&&ch[i]<='z')
20             if((ch[i]+k)<=122)
21                 ch[i]=ch[i]+k;
22             else
23                 ch[i]=ch[i]+k-26;
24     }
25     ch[i]='\0';
26     printf("输出加密密文: ");
27     puts(ch);
28 }
29
30 /*加密: y=ch[i]+K-26 or ch[i]+K (beyond 122&90 or not)*/
```