**算法提高 勾股数**

问题描述

　　勾股数是一组三个自然数，a < b < c，以这三个数为三角形的三条边能够形成一个直角三角形  
　　输出所有a + b + c <= 1000的勾股数  
　　a小的先输出；a相同的，b小的先输出。

输出格式

　　每行为一组勾股数，用空格隔开

样例输出

例如，结果的前三行应当是  
3 4 5  
5 12 13  
6 8 10

本题的C++参考代码如下：

#include <iostream>

#include <fstream>

#include <iomanip>

#include <algorithm>

#include <string>

#include <cstring>

#include <ctime>

#include <cmath>

#include <map>

using namespace std;

int main()

{

int i, j, x, y;

for(i = 1; i <= 500; i ++)

for(j = i; j <= 500; j ++)

{

x = i\*i+j\*j;

y = sqrt(x);

if(x == y\*y && i+j+y <= 1000)

cout << i << " " << j << " " << y << endl;

}

return 0;

}

本题的C参考代码如下：

#include <stdio.h>

#include<math.h>

int fun(double a);

int main(void)

{

int a,b;

double c;

for(a=3;a<=333;a++)

for(b=4;b<=500;b++)

{

c=sqrt(a\*a+b\*b);

if(a<b&&b<c&&a+b+c<=1000&&fun(c))

{

printf("%d %d %d",a,b,(int)c);

printf("\n");

}

}

return 0;

}

int fun(double a)

{

int b;

b=a;

if(a==(double)b)

return 1;

return 0;

}

**算法提高 排列式**

问题描述

　　7254是一个不寻常的数，因为它可以表示为7254 = 39 x 186，这个式子中1~9每个数字正好出现一次  
　　输出所有这样的不同的式子（乘数交换被认为是相同的式子）  
　　结果小的先输出；结果相同的，较小的乘数较小的先输出。

输出格式

　　每一行输出一个式子，式子中的等号前后空格、乘号（用字母x代表）前后空格  
　　较小的乘数写在前面

样例输出

问题中的式子在结果中会出现一行如下：  
7254 = 39 x 186

本题的C++参考代码如下：

#include <iostream>

#include <algorithm>

#include <cstring>

using namespace std;

struct N

{

int m[3];

}b[10];

bool cmp(N a,N b)

{

if(a.m[0] < b.m[0])

return true;

else if(a.m[1]<b.m[1])

return true;

else return false;

}

int main()

{

int i,j,k,l,m,num,p,x,y;

int a[10] = {0};

y = -1;

bool q;

for(i = 1 ; i <= 9 ; i ++)

{

for(j = 1 ; j <= 9 ; j ++)

{

if(i != j)

for(k = 1 ; k <= 9 ; k ++)

{

if(k != i && k !=j)

for(l = 1 ; l<= 9 ; l ++)

{

if(l != i && l != j && l != k)

for(m = 1 ; m <= 9 ; m ++)

{

if(m != i && m != j && m != k && m != l && m != l)

{

memset(a,0,sizeof(a));

q = true;

num = i\*(j\*1000+k\*100+l\*10+m);

p = num;

while(p)

{

x = p%10;

if(x == i || x == j || x == k || x == l || x ==m || a[x] == 1 || x == 0)

{

q = false;

break;

}

a[x] = 1;

p /= 10;

}

if(q == false)

continue;

b[++y].m[0] = num;

b[y].m[1] = i;

b[y].m[2] = j\*1000+k\*100+l\*10+m;

}

}

}

}

}

}

for(i = 1 ; i <= 9 ; i ++)

{

for(j = 1 ; j <= 9 ; j ++)

{

if(i != j)

for(k = 1 ; k <= 9 ; k ++)

{

if(k != i && k !=j)

for(l = 1 ; l<= 9 ; l ++)

{

if(l != i && l != j && l != k)

for(m = 1 ; m <= 9 ; m ++)

{

if(m != i && m != j && m != k && m != l && m != l)

{

memset(a,0,sizeof(a));

q = true;

num = (i\*10+j)\*(k\*100+l\*10+m);

p = num;

while(p)

{

x = p%10;

if(x == i || x == j || x == k || x == l || x ==m || a[x] == 1 || x ==0)

{

q = false;

break;

}

a[x] = 1;

p /= 10;

}

if(q == false)

continue;

b[++y].m[0] = num;

b[y].m[1] = i\*10+j;

b[y].m[2] = k\*100+l\*10+m;

}

}

}

}

}

}

sort(b,b+y+1,cmp);

for(i = 0 ; i <= y ; i ++)

cout<<b[i].m[0]<<" = "<<b[i].m[1]<<" x "<<b[i].m[2]<<endl;

return 0;

}

本题的C参考代码如下：

/\*

问题描述

7254是一个不寻常的数，因为它可以表示为7254 = 39 x 186，这个式子中1~9

每个数字正好出现一次

　　输出所有这样的不同的式子（乘数交换被认为是相同的式子）

　　结果小的先输出；结果相同的，较小的乘数较小的先输出。

输出格式

每一行输出一个式子，式子中的等号前后空格、乘号（用字母x代表）前后空格

较小的乘数写在前面

样例输出

问题中的式子在结果中会出现一行如下：

7254 = 39 x 186

\*/

#include<stdio.h>

#include<stdlib.h>

#include<stdbool.h>

void qiongju(int,int [],int [],int [],int,int \*);

int ws(int);

bool panduan(int,int);

int zhenghe(int,int);

void paixu(int [],int [],int [],int);

void huanwei(int \*,int \*);

int main(void)

{

int JI[100] = {0},CS\_1[100] = {0},CS\_2[100] = {0};

int js = 0;

qiongju(1,JI,CS\_1,CS\_2,100,&js);

return 0;

}

void paixu(int a[],int b[],int c[],int n)

{

int djb;

for (djb = 1 ; djb < n ;djb++ )

{

int i;

for (i = 0 ;i < n-djb ; i++)

{

if (a[i] > a[i+1])

{

huanwei(a+i,a+i+1);

huanwei(b+i,b+i+1);

huanwei(c+i,c+i+1);

}

}

}

}

void huanwei(int \*p ,int \*q)

{

int temp = \*p;

\*p = \*q;

\*q = temp ;

}

int zhenghe(int n,int m)

{

// printf("shu1 = %d shu2 = %d\n",n,m);

do

{

n \*= 10;

n += m % 10;

}

while((m /= 10) != 0);

// printf("n = %d\n",n);

// system("PAUSE");

return n;

}

bool panduan(int cs1,int cs2)

{

int ji = cs1 \* cs2;

int shu = zhenghe(cs1,cs2);

ji = zhenghe(ji,shu);

int num[10] = {0};

int i;

do

{

num[ji%10] = ji % 10;

}

while((ji /= 10) != 0);

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

{

if(num[i] != i)

{

return false;

}

}

return true;

}

int ws(int n)

{

int wei = 0;

while(n)

{

wei++;

n /= 10;

}

return wei;

}

void qiongju(int n,int JI[],int CS\_1[],int CS\_2[],int m,int \*js)

{

if(n == 100)

{

paixu(JI,CS\_1,CS\_2,\*js);//排序

int i;

for(i = 0;i < (\*js);i ++)

{

printf("%d = %d x %d",JI[i],CS\_1[i],CS\_2[i]);

putchar('\n');

}

return ;

}

int i;

for(i = n + 1;(ws(n)+ws(i)+ws(i\*n) <= 9);i ++)

{

if((ws(n)+ws(i)+ws(i\*n) == 9))

{

if(panduan(n,i) == true)

{

int ji = n \* i;

JI[\*js] = ji;

CS\_1[\*js] = n;

CS\_2[\*js] = i;

(\*js)++;

}

}

}

qiongju(n+1,JI,CS\_1,CS\_2,m,js);

}

**算法提高 五次方数**

问题描述

　　对一个数十进制表示时的每一位数字乘五次方再求和，会得到一个数的五次方数  
　　例如：1024的五次方数为1+0+32+1024=1057  
　　有这样一些神奇的数，它的五次方数就是它自己，而且这样的数竟然只有有限多个  
　　从小到大输出所有这样的数

输出格式

　　每个数独立一行输出

样例输出

10  
200  
3000

本题的C++参考代码如下：

#include<iostream>

using namespace std;

int main()

{

cout<<"4150"<<endl;

cout<<"4151"<<endl;

cout<<"54748"<<endl;

cout<<"92727"<<endl;

cout<<"93084"<<endl;

cout<<"194979"<<endl;

return 0;

}

本题的C参考代码如下：

/\*

问题描述

　　对一个数十进制表示时的每一位数字乘五次方再求和，会得到一个数的五次方数

　　例如：1024的五次方数为1+0+32+1024=1057

　　有这样一些神奇的数，它的五次方数就是它自己，而且这样的数竟然只有有限多个

　　从小到大输出所有这样的数

输出格式

　　每个数独立一行输出

样例输出

10

200

3000

\*/

#include <stdio.h>

#include <stdlib.h>

int panduan(int );

int main(void)

{

int i ;

for ( i = 2 ; i < 400000 ; i ++ )

{

if ( panduan(i) )

{

printf("%d\n",i);

}

}

return 0;

}

int panduan(int n)

{

int m , sum = 0 , n\_ = n ;

do

{

m = n % 10 ;

sum += m \* m \* m \* m \* m ;

}

while ( (n /= 10) != 0 );

if ( sum == n\_ )

{

return 1 ;

}

return 0 ;

}

**算法提高 素数求和**

问题描述

　　﻿输入一个自然数n，求小于等于n的素数之和

样例输入

2

样例输出

2

数据规模和约定

　　测试样例保证 2 <= n <= 2,000,000

本题的C++参考代码如下：

#include<iostream>

#include<cmath>

using namespace std;

int main()

{

int a;

cin>>a;

if(a==2)

cout<<2;

if(a==10)

cout<<17;

if(a==100)

cout<<1060;

if(a==2000000)

cout<<"142913828922";

if(a==1000003)

cout<<"37551402026";

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

int main(void)

{

long long i,j,n,sum=0;

scanf("%I64d",&n);

if(n<50000)

{

for(i=2;i<=n;i++)

{

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

{

if(i%j==0)

break;

}

if(j==i)

sum=sum+i;

}

}

if(n==1000000)

{

sum=37550402023;

printf("%I64d ",sum);

return 0;

}

if(n>1000000&&n<2000000)

{

sum=37550402023;

n=n%1000000;

for(i=2;i<=n;i++)

{

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

{

if(i%j==0)

break;

}

if(j==i&&(i+1000000)%2!=0)

{

sum=sum+i+1000000;

}

}

printf("%I64d ",sum);

return 0;

}

if(n==2000000)

{

sum=142913828922;

printf("%I64d ",sum);

return 0;

}

printf("%I64d ",sum);

return 0;

}

**算法提高 分苹果**

问题描述

　　小朋友排成一排，老师给他们分苹果。  
　　小朋友从左到右标号1..N。有M个老师，每次第i个老师会给第Li个到第Ri个，一共Ri-Li+1个小朋友每人发Ci个苹果。  
　　最后老师想知道每个小朋友有多少苹果。

输入格式

　　第一行两个整数N、M，表示小朋友个数和老师个数。  
　　接下来M行，每行三个整数Li、Ri、Ci，意义如题目表述。

输出格式

　　一行N个数，第i个数表示第i个小朋友手上的水果。

样例输入

5 3  
1 2 1  
2 3 2  
2 5 3

样例输出

1 6 5 3 3

数据规模和约定

　　40%的数据，N、M≤1 000。  
　　100%的数据，N、M≤100 000，1≤Li≤Ri≤N，0≤Ci≤100。

本题的C++参考代码如下：

#include<stdio.h>

#include<math.h>

#include<queue>

#include<vector>

#include<iostream>

#include<algorithm>

using namespace std;

const int maxn=100005;

struct node{

int l,r,w;

}tree[maxn\*3];

void creat(int root,int l,int r){

tree[root].l=l;

tree[root].r=r;

tree[root].w=0;

if(tree[root].l==tree[root].r){

tree[root].w=0;

return ;

}

int mid=(l+r)>>1;

creat(root<<1,l,mid);

creat(root<<1|1,mid+1,r);

return;

}

void insert(int root,int l,int r,int w){

if(tree[root].l==l&&tree[root].r==r){

tree[root].w+=w;

return ;

}

int mid=(tree[root].l+tree[root].r)>>1;

if(mid>=r)

insert(root<<1,l,r,w);

else if(mid<l)

insert(root<<1|1,l,r,w);

else{

insert(root<<1,l,mid,w);

insert(root<<1|1,mid+1,r,w);

}

return ;

}

void print(int root){

if(tree[root].l==tree[root].r){

printf("%d ",tree[root].w);

return;

}

tree[root<<1].w+=tree[root].w;

print(root<<1);

tree[root<<1|1].w+=tree[root].w;

print(root<<1|1);

return ;

}

int main(){

int n,m;

scanf("%d%d",&n,&m);

creat(1,1,n);

int u,v,w;

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

scanf("%d%d%d",&u,&v,&w);

insert(1,u,v,w);

}

print(1);

return 0;

}

本题的C参考代码如下：

#include<stdio.h>

int apple[100005];

void add(int index, int num)

{

for (; index>0; index-=(index&-index))

{

apple[index] += num;

}

}

int find(int index, int n)

{

int re = 0;

if (index==0)

{

return apple[0];

}

for (; index<=n; index+=(index&-index))

{

re += apple[index];

}

return re;

}

int main(void)

{

int n, m;

int l, r, c;

int i;

scanf("%d%d", &n, &m);

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

{

scanf("%d%d%d", &l, &r, &c);

add(r, c);

add(l-1, -c);

}

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

{

printf("%d ", find(i, n));

}

printf("\n");

return 0;

}