假定一种编码的编码范围是 a~y 的 25 个字母，从 1 位到 4 位的编码，如果我们把该编码按字典序排序，形成一个数组如下：a,aa,aaa,aaaa,aaab,aaac,……,b,ba,baa,baaa,baab,baac……,yyyw,yyyx,yyyy 其中 a 的 Index 为 0，aa 的 Index 为 1，aaa 的 Index 为 2，以此类推。编写一个函数，输入是任意一个编码，输出这个编码对应的 Index.

```
#include<stdio.h>
#include<string.h>

#defineN11
#defineN225
#defineN3(25*25)
#defineN4(25*25*25)

#defineC1N1
#defineC2(N1+N2)
#defineC3(N1+N2+N3)
#defineC4(N1+N2+N3+N4)

intmain()
{
charcode[5]={0};
scanf("%s",code);
intindex=0;
switch(strlen(code)){
case4:index+=C1*(code[3]-'a')+1;
case3:index+=C2*(code[2]-'a')+1;
case2:index+=C3*(code[1]-'a')+1;
case1:index+=C4*(code[0]-'a');
default:break;
}
printf("%d\n",index);
return0;
}
```

游戏里面有很多各式各样的任务，其中有一种任务玩家只能做一次，这类任务一共有 1024 个，任务 ID 范围[1,1024]。请用 32 个 unsignedint 类型来记录着 1024 个任务是否已经完成。初始状态都是未完成。输入两个参数，都是任务 ID，需要设置第一个 ID 的任务为已经完成；并检查第二个 ID 的任务是否已经完成。输出一个参数，如果第二个 ID 的任务已经完成输出 1，如果未完成输出 0。如果第一或第二个 ID 不在[1,1024]范围，则输出-1。

```
#include<iostream>
#include<string>
#include<vector>
#include<algorithm>
```

```
intmain(){
usingnamespacestd;
unsignedintn,m;
while(cin>>n>>m){
if(n<1||n>1024||m<1||m>1024){
cout<<-1<<endl;
}
else{
if(n==m){
cout<<1<<endl;
}
else{
cout<<0<<endl;
}
}
}
return0;
}
```

给定一个正整数，编写程序计算有多少对质数的和等于输入的这个正整数，并输出结果。输入值小于 1000。
如，输入为 10,程序应该输出结果为 2。（共有两对质数的和为 10,分别为(5,5),(3,7)）

```
#include<iostream>
#include<vector>
#include<cmath>
//判断是否为素数
boolisPrime(intn){
//注意是<=.....
for(inti=2;i<=sqrt(n);i++)
if(n%i==0)
returnfalse;
returntrue;
}

//保存 1000 以内的素数
voidPrimeIn1000(vector<int>&vec){
vec.push_back(2);
for(inti=3;i<1000;i++)
if(isPrime(i))
vec.push_back(i);
}

//用两个迭代器分别指向 vector 的头尾，遇大则尾退，遇小则头进
intSumofPrimePair(intn){
```

```
vector<int>PrimeVec;
PrimeIn1000(PrimeVec);
intresult=0;
vector<int>::iteratoriterLeft=PrimeVec.begin();
vector<int>::iteratoriterRight=PrimeVec.end()-1;
while(iterLeft<=iterRight){
inttempSum=*iterLeft+*iterRight;
if(tempSum==n){
result++;
iterLeft++;
iterRight--;
}
elseif(tempSum<n)
iterLeft++;
else
iterRight--;
}

returnresult;
}

intmain(){
intn;
cin>>n;
cout<<SumofPrimePair(n)<<endl;
system("pause");
return0;
}
```

geohash 编码：geohash 常用于将二维的经纬度转换为字符串，分为两步：第一步是经纬度的二进制编码，第二步是 base32 转码。
此题考察纬度的二进制编码：算法对纬度[-90,90]通过二分法进行无限逼近（取决于所需精度，本题精度为 6）。注意，本题进行二分法逼近过程中只采用向下取整来进行二分，针对二分中间值属于右区间。算法举例如下：针对纬度为 80 进行二进制编码过程：
1)区间[-90,90]进行二分为[-90,0),[0,90]，成为左右区间，可以确定 80 为右区间，标记为 1；
2)针对上一步的右区间[0,90]进行二分为[0,45),[45,90]，可以确定 80 是右区间，标记为 1；
3)针对[45,90]进行二分为[45,67),[67,90],可以确定 80 为右区间，标记为 1；
4)针对[67,90]进行二分为[67,78),[78,90]，可以确定 80 为右区间，标记为 1；
5)针对[78,90]进行二分为[78,84),[84,90]，可以确定 80 为左区间，标记为 0；
6)针对[78,84)进行二分为[78,81),[81,84)，可以确定 80 为左区间，标记为 0；

```
#include"iostream"
#include"string"
```

```
usingnamespacestd;

intmain(){
intinput(0);
cin>>input;

intright(-90),left(90),mid(0),cnt(6);//精度指字符串的长度
stringresult;

while(cnt--){
mid=(right+left)/2;
if(input<mid){
left=mid;
result+="0";
}
else{
right=mid;
result+="1";
}

}
cout<<result;
return0;
}
```