/\*

Count the number of prime numbers less than a non-negative number, n.

求素数

方法一：硬解,结果是TLE.

方法二：空间换时间,用map存非素数，也TLE。 最后是改为vector来存才能过！

方法三：还是两个循环，第二个循环只除已经找到的素数！

\*/

class Solution {

public:

int countPrimes(int n)

{

//way-1

/\*

bool flag=1;

int ret=0;

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

{

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

{

if(i%j==0)

{

flag=0;

break;

}

}

if(flag==1)

ret++;

flag=1;

}

\*/

//way-2-1

/\*

if(n<=2)

return 0;

else if(n==3)

return 1;

map<int,int> p;

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

{

if(p.find(i)!=p.end())

continue;

for(int j=i;j<=(n-1)/i;j++)

{

if(p[j\*i]==0)

p[j\*i]++;

}

}

return n-p.size()-2;

\*/

//way-2-2 map改为数组 也是超时

/\*

int a[n+1]={0};

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

{

if(a[i]==1)

continue;

for(int j=i;j<=(n-1)/i;j++)

a[i\*j]=1;

}

int ret=0;

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

{

if(a[i]==0)

ret++;

}

return ret;

\*/

//way-2-3 改为vector<bool> 过！

if(n<=2)

return 0;

vector<bool> a(n+1,false);

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

{

if(a[i]==true)

continue;

for(int j=i;j<=(n-1)/i;j++)

a[i\*j]=true;

}

int ret=0;

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

{

if(a[i]==false)

ret++;

}

return ret;

//way-3

/\*

if(n<=2)

return 0;

int ret=0;

map<int,int> p;

p[2]=1;

bool flag=1;

for(int i=3;i<n;i++)

{

for(map<int,int>::iterator it=p.begin();it!=p.end();it++)

{

if(i%it->first==0)

{

flag=0;

break;

}

}

if(flag==1)

{

ret++;

p[i]++;

}

flag=1;

}

return ret+1;

\*/

}

};