给定一个n，系统随机选定了一个数x，(1<=x<=n)。

你可以询问系统x是否能被y整除，系统会回答"Yes"or“No"。

问：至少询问多少次可以唯一确定x，并输出询问序列。(special judge).

其实依据还是唯一分解定理

题目最后要求确定这个数x，用质数确定

这样的分解可以唯一确定一个数，所以你必须要把<n的质数和质数的次幂都询问一遍，才能保证一定可以确定出x

注意不能光看质数，一个质数小于n的次幂也要看

如果质数p[i]能够整除，那么x有可能是p[i]^2,p[i]^3....。

所以p[i]^2,p[i]^3你都要询问

做法：求质数的整数次幂（不大于n）。

思路：首先我们肯定是用质数来判断，因为质数排除的是最多的。

如果质数p[i]能够整除，那么x有可能是p[i]^2,p[i]^3....。

#include <bits/stdc++.h>

#define ll long long

using namespace std;

const int maxn=1010;

vector<int> v;

bool is\_prime[maxn]; //初始化，=false表示是素数

int prime[maxn]; //保存素数

int k;

void Euler(int n)

{

int i, j;

k = 0;

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

{

if(is\_prime[i] == false)

prime[k++] = i;

for(j = 0; j<k && i\*prime[j]<n; j++)

{

is\_prime[ i\*prime[j] ] = true;

if(i%prime[j] == 0)

break;

}

}

}

int main()

{

// freopen("input.txt","r",stdin);

int n,x;

Euler(maxn);

int ans(0);

cin>>n;

for(int i=0;i<k;i++)

{

if(prime[i]<=n)

{

int p=prime[i];

while(p<=n)

{

v.push\_back(p);

p\*=prime[i];

}

}

}

cout<<v.size()<<endl;

if(v.size()>0)

{

for(int i=0;i<v.size()-1;i++)

cout<<v[i]<<' ';

cout<<v[v.size()-1]<<endl;

}

return 0;

}