[UVA - 1635](https://vjudge.net/problem/51196/origin" \t "_blank)

|  |
| --- |
| /\*  translation:  题意见lrj,p320  solution:  唯一分解定理，杨辉三角迭代公式  根据杨辉三角的迭代公式即可很容易得出最后一项的每一项系数。根据是否能够整除m，就可以得出这一项是否跟  最后的结果有关。但是问题在于最后一项的数据范围太大，必须用高精度才能保存。所以直接对m取余来求解是行  不通的。所以就必须用唯一分解定理：对m进行素因子分解，然后对于每一项m的素因子，求其在每一项Ci中的次  数（对Ci也进行了唯一分解），如果一旦有在Ci中的次数<在m中的次数，那么就可以断定这个Ci肯定不能整除m.  据此就可以得出结果。  note:  1:注意最后判断能否整除m时，当在Ci中的次数>=在m中的次数并不能说明就一定能整除m。因为还有其它的素因子  未判断。  2:这道题可以总结出一个利用唯一分解定理判断能否整除m的一个方法，这样就不需要高精度了。  \*/  #include <cstdio>  #include <cstring>  #include <cmath>  using namespace std;  const int maxn= 32000;  int prime[maxn+1];  int nprime;  void getPrime()  {  int m=sqrt(maxn+0.5);  for(int i=2; i<=m; ++i) if(!prime[i])  for(int j=i\*i; j<=maxn; j+=i) prime[j]=1;  nprime=0;  for(int i=2; i<=maxn; ++i)  {  if(!prime[i])  prime[nprime++]=i;  }  }  int n,m;  int pm[20];  int em[20];  int im;  void init()  {  memset(pm,0,sizeof(pm));  memset(em,0,sizeof(em));  im=0;  for(int i=0; i<nprime&&m>=prime[i]; i++)  {  if(m%prime[i]==0)  {  pm[im]=prime[i];  while((m%prime[i]==0)&&(m/=prime[i]))  em[im]++;  im++;  }  if(n==0||n==1)  break;  }  if(m>1)  {  pm[im]=m;  em[im]=1;  im++;  }  }  bool getFactors(int x,int y)  {  bool ff=true;  for(int i=0; i<im; ++i)  {  while((x%pm[i]==0)&&(x/=pm[i]))  em[i]--;  while(y%pm[i]==0&&(y/=pm[i]))  em[i]++;  if(em[i]>0)  ff=false;  }  return ff;  }  bool flag[100010];  int main()  {  getPrime();  while(scanf("%d%d",&n,&m)!=EOF)  {  init();  memset(flag,false,sizeof(flag));  int ans=0;  int ends=0;  for(int i=1; i<=n; ++i)  if(getFactors(n-i,i))  {  flag[i+1]=true;  ans++;  ends=i+1;  }  printf("%d\n",ans);  if(ans!=0)  {  for(int i=1; i<ends; i++)  if(flag[i])  printf("%d ",i);  printf("%d",ends);  }  printf("\n");  }  return 0;  } |