Crypto

第一题

RSA，但是n特别大。通过看代码发现p和q都符合2^n-1这一形式。我们知道指数的增长是非常快的的，那么可以通过libnum.gcd()函数得出n的一个因子。如图。

得出其中一个因子后我们就用同样的方法得到另一个因子。然后就是算逆元，得到d。再pow(c,d,n)就是明文，用一下libnum.n2s得flag。

第三题

通过“有几个朋友发送的内容还是相同的！”和一堆的n，e，c(并且每个e都是3)得出考点是中国剩余定理，公式代入并不复杂，但是这有几个是那几个呢？通过如图程序(注释内容中的)得到对应的那几个，分别是1.3.5和2.4.6，有一组数据是没用的。然后就是常规的操作了。

task-6.py

task-6.py 〉 No Selection

```
N=[n1,n2,n3,n4,n5,n6,n7]
C=[c1,c2,c3,c4,c5,c6,c7]
'''m=n1*n2*n3*n4*n5
m1=n2*n3*n4*n5
m2=n1*n3*n4*n5
m3=n2*n1*n4*n5
m4=n2*n3*n1*n5
m5=n2*n3*n4*n1

t1=gmpy2.invert(m1,n1)
t2=gmpy2.invert(m2,n2)
t3=gmpy2.invert(m3,n3)
t4=gmpy2.invert(m4,n4)
t5=gmpy2.invert(m5,n5)

x=(c1*t1*m1+c2*t2*m2+c3*t3*m3+c4*t4*m4+c5*t5*m5)%m'''
'''for i in range(7):
 for j in range(7):
  if(j<=i):
   continue
  for k in range(7):
   if(k<=j):
    continue
   m=N[i]*N[j]*N[k]
   m1=N[j]*N[k]
   m2=N[i]*N[k]
   m3=N[i]*N[j]
   t1=gmpy2.invert(m1,N[i])
   t2=gmpy2.invert(m2,N[j])
   t3=gmpy2.invert(m3,N[k])
   x=(C[i]*t1*m1+C[j]*t2*m2+C[k]*t3*m3)%m
   if(gmpy2.iroot(x,3)[1]==True):
    print(gmpy2.iroot(x,3))'''
x=5687320551454769755914702265741801348362363137679664584577713935850648870021712508210823945236536781936010346402943277442779863384017951444921287998468535322627581987600520791246790801774613389776111756063858341810137533343711764384080712638213804098446979696348450514241120554085314379257601568553207887284421679713841452453690743951599236858953004469586089802397983710698646453520406535139984311503455459738215186567668150718389315356984306066841639282959775474727315485170232447343000188235096262607720177650796746319777284125771252063723080510750668669
y=1856210014241605057368394299924381863507609207399040260175983434671613410324171101058627541163910900867688207319214864345503161048193704117590783831649790976119044134604441471303785279732373156855887068354665450765996037183830774646291030696384414382038890615830751742900755717699798110083238100524116624531384092434070088910332879135600659626039801887061643202896986776936747689892733309343669015457074999290018651322225636769983210356788135945593579085775603961958861062798974930037468634286148534401097187317901326985299351281 3
print(libnum.n2s(y))
```