https://play.picoctf.org/practice/challenge/61?category=2&page=2

first question:

```
q : 60413
p : 76753
##### PRODUCE THE FOLLOWING ####
n
IS THIS POSSIBLE and FEASIBLE? (Y/N):Y
#### TIME TO SHOW ME WHAT YOU GOT! ###
n:
```

python code:

```
p=60413
q= 76753
n=p*q
print(n)
```

->4636878989

Third question is insane :

```
#### NEW PROBLEM ####
e : 3
n : 127381628029105465038219208869053933163863627595674808394284565252242264451730316353066837261825224949108085189204090194140348144093300942458257496809132045668323377047001659931988970297957869691242321388697846262025013661359752238272878123262505771486253608876989306255043343258045873299056179365811163927846843346642043097714308144496061472213498883204034516378824477097962217064702396252922979887664937462096848808431111381706000398881124044113109747585326039986080570088118363845975791472447376060887562999396542650868990963590706672661677549445879486958421719150486198462828737694134890722434777643500717873279 13
##### PRODUCE THE FOLLOWING ####
q
p
IS THIS POSSIBLE and FEASIBLE? (Y/N):N
Outstanding move!!!
```

the size of n is crazy :v

```
#### NEW PROBLEM ####
q : 66347
p : 12611
##### PRODUCE THE FOLLOWING ####
totient(n)
IS THIS POSSIBLE and FEASIBLE? (Y/N):
```

just compute (p-1)*(q-1) -> 836623060

next problem:

```
#### NEW PROBLEM ####
plaintext : 635729417148931154719098761554457513358196788649948409135266140641404444047520534288284123
3635766597343146249135508941371039227338020303879324156430477427152910872971 7
e : 3
n : 291294636093263225595211231362220787805854512081491385477991210836223332506466787677691262481822
7478527881025116332742616201890576280859777513414460842754045651093593251726785499360828237897586278
6841987551754301354536987170415971810535469080272664571069902993675426565438192965049438362258317407
8057977666851923258599827977960603912718175780874729482056262577174798583697545026151737735140874375
4532994142632207906501079835037052797306690891600559321673928943158514646572885986881016569647357891
9854588030423614554805952089813314208754536917987606565721422582699767684400005432714166632055308212
424707948750331
##### PRODUCE THE FOLLOWING ####
ciphertext
IS THIS POSSIBLE and FEASIBLE? (Y/N):Y
#### TIME TO SHOW ME WHAT YOU GOT! ###
```

remember that c =plt^e (mod n)

I tried this :

```
n=291294636093263225595211231362220787805854512081491385477991210836223332506466787677691262481822074785278810251163327426162018905762808597775134144608427540456510935932517267854993608282378975862780684198755175430135453698717041597181053546908027266457106990299367542656543819296504943836225831740758057977666851923258599827977960603912718175780874729482056262577174798583697545026151737735140874375045329941426322079065010798350370527973066908916005593216739289431585146465728859868810165696473578915985458803042361455480595208981331420875453691798760656572142258269976768440000543271416663205530821284247079487503314
```

Admittedly, I don’t know that python can handle such a big size-number like this.

output:

256931246631782714357241556582441991993437399854161372646318659020994329843524306570818293602492485385337029697819837182169818816821461486018802894936801257629375428544752970630870631166355711254848465862207765051226282541748174535990314552471546936536330397892907207943448897073772015986097770443616540466471245438117157152783246654401668267323136450122287983612851171545784168132230208726238881861407976917850248110805724300421712827401063963117423718797887144760360749619552577176382615108244813

Next one,

```
#### NEW PROBLEM ####
ciphertext : 1075240134510793485399445107561436042039257172621850337993284450117927605455289449937197
8339254216342863717232351225262456711111066616866474311520379151098570994236660962643699588778167465
2722335663038149796775071011685877393756990097345889854823697026344995448915092284401946153763395736
528512573028662332 3
e : 3
n : 275669962915082139324193713851415228593432265600509211962947618705008461401323850809946309461076
7533018960602116526059014706878582020360088209246779781351943465263212606135358312406394437333665424
6386074125394368479677295167494332556053947231141336142392086767742035970752738056297057898704112912
6165652994513597915485368460258543783474235201049479073344510563394397066230695030889163163698134997
0507357377757716939240141170892061557490859378428254615448644677924679029439819885454706959398722457
8333683144886242572837465834139561122101527973799583927411936200068176539747586449939559180772690007
261562703222558103359
##### PRODUCE THE FOLLOWING ####
plaintext
```

n,e and c -> impossible to find p

another “big” problem

```
#### NEW PROBLEM ####
q : 9209207680589253373972472260266867584067109300852024154819191421539982402037207618646076820681491
44238022303984109802187419069605271045689702258043744046126177365792869598652872265386929113765079342
5684445633323636266987934707375623889478495159721110573417938830005157999425356545930474305953364675
003894559
p : 97846775312392801037224396977012615848433199640105786119757047098757998273009741128821931277074
57318132894238913899118012503262993240185570727270517655471155147913375787588598038901731532772523264
96062476389498019821358465433398338364421624871010292162533041884897182597065662521825095949253625730
631876637
e : 65537
##### PRODUCE THE FOLLOWING ####
d
```

Let’s call for python help again :3

```
>>> q = 92092076805892533739724722602668675840671093008520241548191914215399824020372076186460768206814914423802230398410
98021874190696052710456897022580437440461261773657928695986528722653869291137650793425684445633323636266987934707375623 88
9478495159721110573417938830005157999425356545930474305953364675300389455 9
>>> p = 97846775312392801037224396977012615848433199640105786119757047098757998273009741128821931277074555731813289423891
38991180125032629932401855707272705176554711551479133757875885980389017315327725232649606247638949801982135846543339833 83
6442162487101029216253304188489718259706566252182509594925362573063187663 7
>>> e=65537
>>> phi = (p - 1) * (q - 1)
>>> from Crypto.Util.number import inverse
>>> d = inverse(e, phi)
>>> d
14050462695032074691407915484036395331274164162142106949720850791717875804637768204259658981742728704860157395161257861 82
82163700660074214068255232164550374328067083981907874909273011054988189127131739645015802168825398976714557872345825276 94
65545504142139663476747479225923933192421405464414574786272963741656223941750084051228611576708609346787101088759062724 38
98741606930087833346059031425288245592235152039787079697950875066788940066282967430798862443494691318312257579268448435 54
89763878614603686957265320473565084318672273273688891878937905405012220525316570508553874365125840039058097104314464498 46
54914856729
```

answer:

1405046269503207469140791548403639533127416416214210694972085079
1717875804637768204259658981742728704860157395161257861828216370
0660074214068255232164550374328067083981907874909273011054988189
1271317396450158021688253989767145578723458252769465545504142139
6634767474792259239331924214054644145747862729637416562239417500

840512286115767086093467871010887590627243898741606930087833346059031425288245592235152039787079697950875066788940066282967430798862443494691318312257579268448435548976387861460368695726532047356508431867227327368889187893790540501222052531657050855387436512584003905809710431446449846549148567290

The last question:

```
#### NEW PROBLEM ####
p : 153143042272527868798412612417204434156935146874282990942386694020462861918068684561281763577034706600608387699148071015194725533394126069826857182428660427818277378724977554365910231524827258160904493774748749088477328204812171935987088715261127321911849092207070653272176072509933245978935455542420691737433
ciphertext : 180314885368643794960895500172725992461344351213432291642366713880386307528476457389684554130677731661152340392475400291743317437812035121086265946012932837373922403260208884172523886029140518289809134789277599348057550304938947289747208520271926698905550119698686762813722190657005740866343113838228101687566611695952746931293926696289378849403873881699852860519784750763227733530168282209363348322874740823803639617797763626570478847423136936562441423318948695084910283653593619962163665200322516949205854709192890808315604698217238383629613355109164122397545332736734824591444665706810731112586202816816647839648399
e : 65537
n : 23952937352643527451379227516428377705004894508566304313177880191662177061878993798938496818120987817049538365206671401938265663712351239785237507341311858383628932183083145614696585411921662992078376103990806989257289472590902167457302888189829313533308373450419191095323827886092315374626150075941162029986439515878350953503925971435952673892473695275975350335761493920343409207567616917911245262068773167053490606984596563345574860664906239429328996705934814320660076582002131926082705288562383068491911132413558423963252101323580466163129013379874644737990407622718763890314550516409376817454090572461904987957697239
##### PRODUCE THE FOLLOWING ####
plaintext
```

Ask python for help :3

```
>>> e=65537
>>> from Crypto.Util.number import inverse
>>> phi =(p-1)*(q-1)
>>> d= inverse (e,phi)
>>> d
220341293342511915324366310524271420220887449110874282943765333035497149477317988183256047373859040317143830114777087570174439182175949340514917314659759831297410231551876588747305040622628636770592041164730424181966554836823125710590722678915803019641670980065339844002300397895612275493365136683499145981339720917745192486767724408757690257729992782193138061320221056036021250655172762577800896954872635252333116315302708161070866635226842495609316348977064688985209868239785215655935539895442195141416588681176252861378708961487651098515192544593054272518300962701161453596676550341146423568647318012592635194260970
>>> d= inverse(e,phi)
>>> d
220341293342511915324366310524271420220887449110874282943765333035497149477317988183256047373859040317143830114777087570174439182175949340514917314659759831297410231551876588747305040622628636770592041164730424181966554836823125710590722678915803019641670980065339844002300397895612275493365136683499145981339720917745192486767724408757690257729992782193138061320221056036021250655172762577800896954872635252333116315302708161070866635226842495609316348977064688985209868239785215655935539895442195141416588681176252861378708961487651098515192544593054272518300962701161453596676550341146423568647318012592635194260970
>>> c=180314885368643794960895500172725992461344351213432291642366713880386307528476457389684554130677731661152340392475400291743317437812035121086265946012932837373922403260208884172523886029140518289809134789277599348057550304938947289747208520271926698905550119698686762813722190657005740866343113838228101687566611695952746931293926696289378849403873881699852860519784750763227733530168282209363348322874740823803639617797763626570478847423136936562441423318948695084910283653593619962163665200322516949205854709192890808315604698217238383629613355109164122397545332736734824591444665706810731112586202816816647839648399
>>> g= pow(c,d,n)
>>> g
143116639427096748671222082149019706504967881512395209716234117129771205861635358801685633 25
>>>
```

output:
143116639427096748671222082149019706504967881512395209716234117129771205861635358801685633 25

This is what you will get:

```
Outstanding move!!!


If you convert the last plaintext to a hex number, then ascii, you'll find what you need! ;)
s
```

```
>>> from Crypto.Util.number import long_to_bytes
>>> long_to_bytes(143116639427096748671222082149019706504967881512395209716234117129771205861635358801685633 25)
b'picoCTF{wA8_th4t$_ill3aGal..ode01e4bb}'
```

picoCTF{wA8_th4t$_ill3aGal..ode01e4bb}

Pheww, after this, I become familiar with RSA