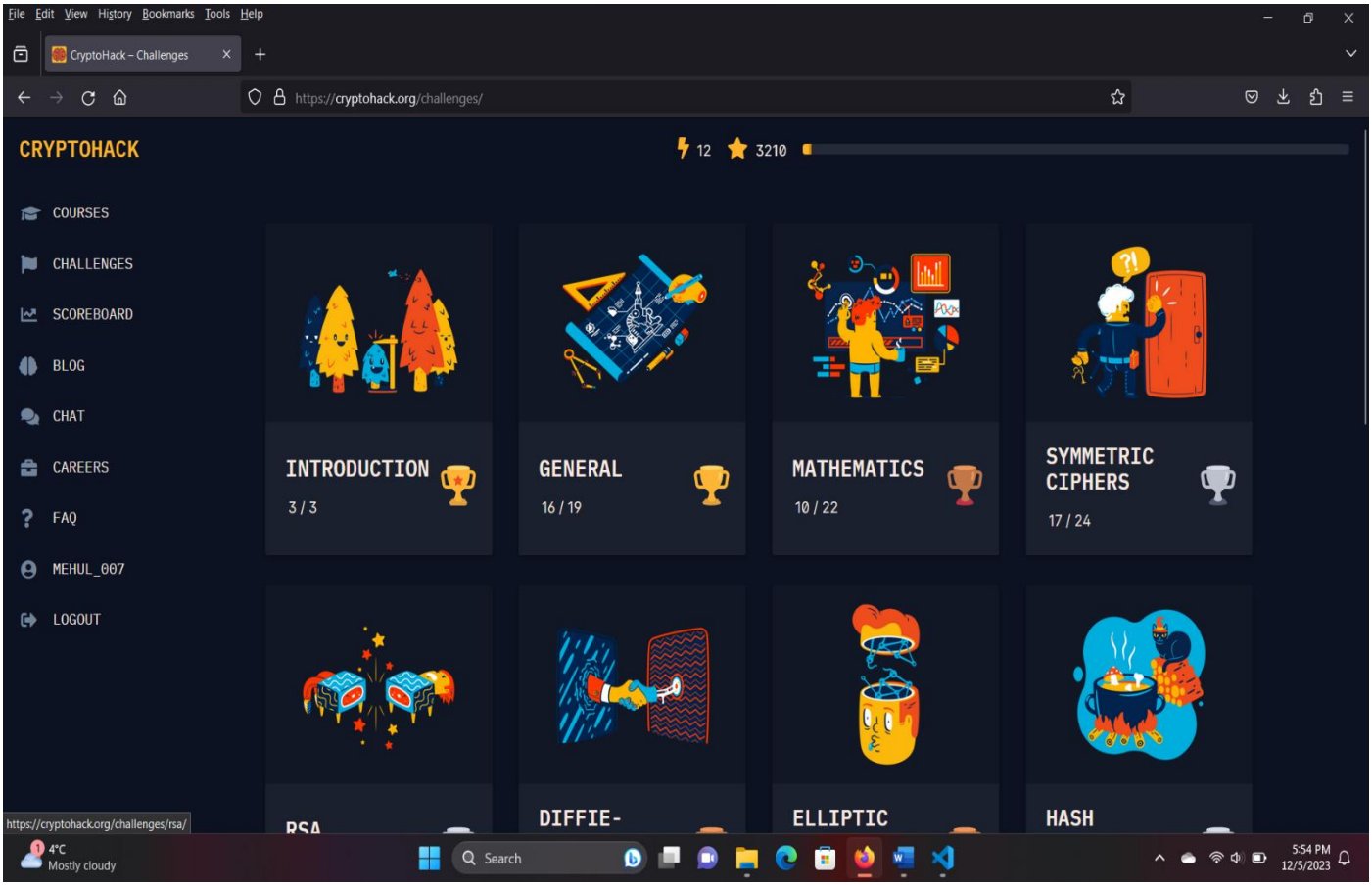
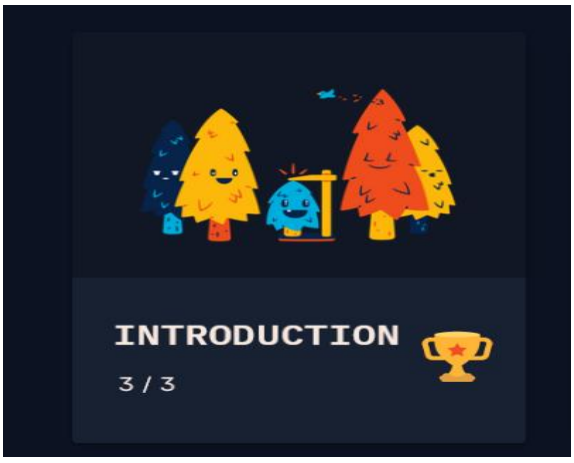
Cryptohack solutions:

SCORE:3210

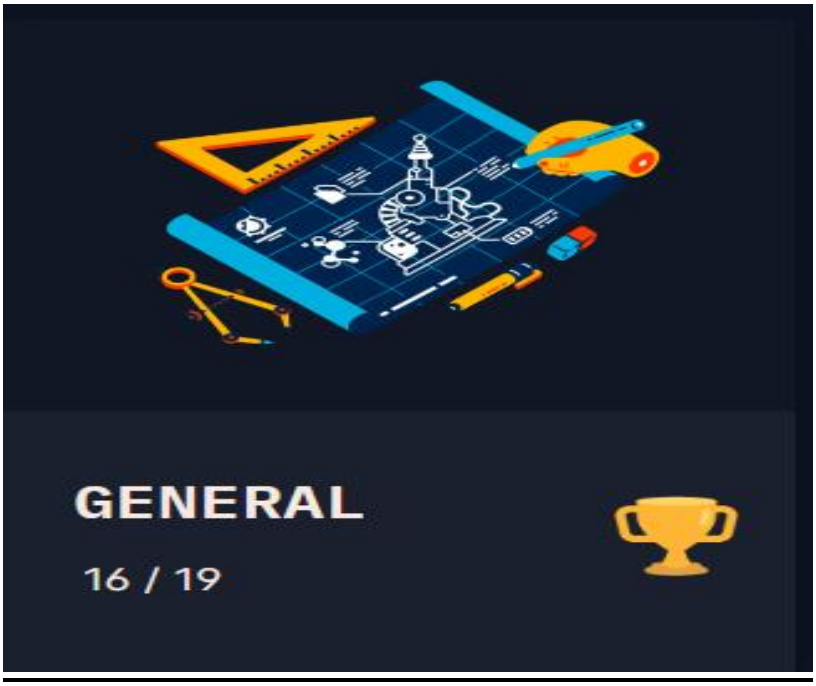


Challenges:

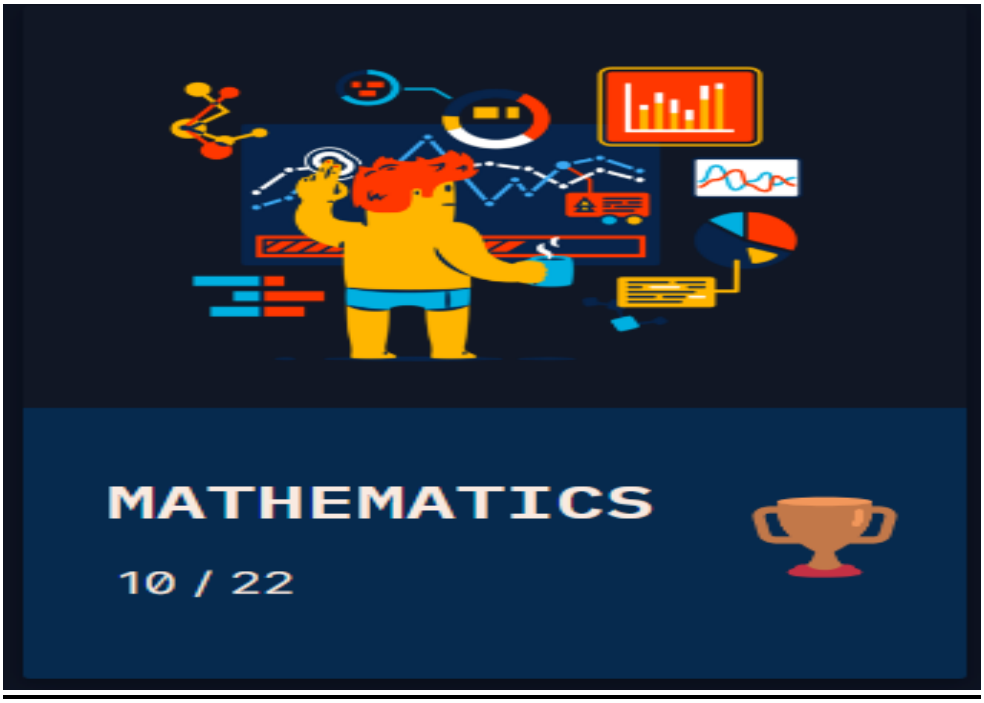
Introduction: Completed 3/3 challenges:



General : Completed 16/19 challenges:



Mathematics: 10/22



Symmetric Ciphers: 17/24

Code For Cryptohack Solutions:

Crossed wires:

e = 0x10001

n =
21711308225346315542706844618441565741046498277716979943478360598053144971379956916
57537034344898860190585457202963584662625948729795030523166110985585494749420913520
55892586435179615215949243684986720642932082308024410773901936829580951119220826778
13175804775628884377724377647428385841831277059274172982280545237765559969228707506
85756121526849102409706392033772178367306053018163716157740158912655855618254689678
33073705172750465227040473857861114894470647942100108027617086159072455234925858962
86374996088089317826162798278528296206977900274431829829206103227171839270887476436
89949442837132387468905569072998677 1

d =
27344116772511480307231380057161097338388665453755276020182551593196310266531907836
70493107936401603981429171880504360560494771017246468702902647370954220312452541342
85874759057627377510787045085353371711668432697626300643573338204580797189076201874
77295740210574303317780339823591848381597473312365385018499653292647749276075704103
47019418407451937875684373454982306923178403161216817237890962651214718831954215200
63765110390720934790085782472265321717954814814568718137722054486452180823012273096

745298143535533493210426548807577763860804132525677627520006754153302252796474347855494879257805770852235081215488809 7

c =
203046102795781867381727662242247931198850712624644644488634611840922257360547479769851796739054415026891262162828977045087454037990547341215839688539997916042816151541007362591314534243853643246302296711853437781728072626407093018382748246031016924856627622690212110559113743733146320188126424556221401216087517716744201095243936062339665897441390046909383679475227039952007459632905872587483408218869737759794940577903913919419606536442621320834546140703077108978752920005710574658449355472279059253047286958131011730034346120775082173784004274553087639179348403502464447553535322785132150553739888810685501274611 7

print(bytes.fromhex(hex(pow(c, pow(106979*108533*69557*97117*103231, -1, e*d-1), n))[2:]).decode())

Flag:crypto{3ncrypt_y0ur_s3cr3t_w1th_y0ur_fr1end5_publ1c_k3y}

Infinite decent:

n =
383347712330877040452238619329524841763392526146840572232926924642094891453979246383798913394114305368360426867021623649667024217266529000859703542590316063318592391925062014229671423777796679798747131250552455356061834719512365575593221216339005132464338847195248627639623487124025890693416305788160905762011825079336880567461033322240015771102929696350161937950387427696385850443727777996483584464610046380722736790790188061964311222153985614287276995741553706506834906746892708903948496564047090014307484054609862129530262108669567834726352078060081889712109412073731026030460300603417375042238220147140564137521658417493681595105881786040961919567509410783914156344722197651295616223441097698922447126684027615494121778920540512667615973306605457043172105677598287571569047784956089687857479980598574674401281560683917469196842582276828660836623452636595580668641092124572861145062284709307750927353853883162686636641390561831802380433866362540759406215437175316709958234170706660059304528363898121294620517716460484983971951574053869234468938865930486809848969898091358022768929110385880087019267292698124532268917765460376635838936254792526430425171969589902663767416765146310894664938640643161276480746096627491965459699260 51

e = 65537

c =
982804567571367662449448919870289358434415334156135925913584829060164395630761505261163698422131033334805067059936339019941072818901872484955072708686213846522076976070198991664921324083487892525551964286086613206718774127104897823582820113641277995633355629177077835636819207869944530047637554045105415745021762438967568399179918484280915949191114480239485277663683045031006503799141530581911400725280958985760188938298301043621249271405551079941141430422667587093280689026640378700757425421943180591913134686759394268109882390794248234953174640352523255219175920451981526435332 23

01595270264924949475339510097353454176628555189185964932037117856220025222877939539
39741697369985233945985171741821420074805266030255780046659368546572945413386975135
21007818552254811797566860763442604365744596444735991732790926343720102293453429936
73420624610996881715881574992706356183527463619514970231741568040198715033699458375
20625652376059531537903711559184399411934014732717530381805601297841928003516497244
65553733201451581525173536731674524145027931923204961274369826379325051601238308635
19254022348405509620329340041981602411179790344286418196595924774500682269096792095
79051884415501069307998962928352878674039796318240857900478513832943 89

p =
19579267410474709598749314750954211170621862561006233612440352022286786882372619130
07163982410978354056451242908167413233681197240456395702546503402578120646663173078
45163372102913343563964717321687427397904641098810392194525044566115891543494273038
32789968502204300316585544080003423669120186095188478480761108168299370326928127888
78681939237247706951531817975170298580902421016424340954469270868421504222693208105
28310285700603089630932176221831116433356923610198974492654022905400257905815899808
67847884281862216603571536255382298035337865885153328169634178323279004749915197270
12032334041696501413642974325276152 1

q =
19579267410474709598749314750954211170621862561006233612440352022286786882372619130
07163982410978354056451242908167413233681197240456395702546503402578120646663173078
45163372102913343563964717321687427397904641098810392194525044566115891543494273038
32789968502204300316585544080003423669120186095188478480761108168299370326928127888
78681939237247706951531817975170298580902421016424340954469270868421504222693208105
28310285700603089630932176221831116433356923626352035828685261788380189469867926568
19885261069890315500550802303622551029821058459163702751893798676443415681144429096
98966447370585061979249555372495093 1

print(bytes.fromhex(hex(pow(c, pow(e, -1, (p-1)*(q-1)), n))[2:]).decode())

Flag:crypto{f3rm47_w45_4_g3n1u5}

RSA backdoor viability:

n =
70987244318676158212574758566872450126855845855879867301467348376630096483647916724
13156600538786504217617266398720898855020049024874719464109184209276825863621111373
64814638033425428214041019139158018673749256694555341525164012369589067354955298579
13173546679591852281612739834046576140671906028409809464328939001631166831668780883
75635891240918677736550449130036685909548997053667870809237172708271842226737068561
84434629431186284270269532605221507485774898673802583974291853116198037970076073697
22504709890141463743339265850067074099600879986053003251571603144978708937140348520
58107958804169206421864510223749898916119439068911390477640420510716472030575201042
67427832746020858026150611650447823314079076243582616371718150121483335889885277291
31283408323408766039953466583529162123205647384322451590902312083437766450578832952

751793216090901341093331257281020804384952965520942005518068077571861408852101477249177665438047894859106348661502360558448333846066739726472487122113365295537102708580422395610453260411396911971648514242499625573737646483431552782256601792359862663443806672476355994344102357457516892401027426137686320259835343001087518294748510107630840606172450506588699035018518845377616231955256661421462436125146 3

c =
608484617316138126443275660524263025508135383745665175433229598517433030003704261658172582370543758277685547533834085899541036156595489206369279739210904154716464595657421948607569920498815631503197235702333017824993576326860166652845334617579798536442066184953550975487031721085105757667800838172225947001224495126390587950346822978519677673568121595427827980195332464747031577431925937314209391433407684845797171187006586455012364702160988147108989822392986966689057906884691499234298351003666019957528738094330389775054485731448274595330322976886875528525229337512909952391041280006426003300720547721072725168500104651961970292771382390647751450445892361311332074663895375544959193148114635476827855327421812307562742481487812965210406231507524830889375419045542057858679609265389869332331811218601440373121797461318931976890674336807528107115423915152709265237590358348348716543683900084640921475797266390455366908727400038393697480363793285799860812451995497444221674390372255599514578194487523882038234487872223540513004734039135243849551315065297737535112525440094171393039622992561519170849962891645196111307537341194621689797282496281302297026025131743423205544193536699103338587843100187637572006174858230467771942700918388

e = 65537

p =
203650292761213744862390936375180565911731535608160887049749342251376310260210062787281722630670933751277995170216426830264539418920855495964155596328371400725877433055744792186283881915870602622631704303157618903039902338715768605511661621105655750882431224118408754916145719317697891732168965276683184345711402310438418832467459974745001766719261536161687791524003063133624778882629970930361365823188816332353760262764168296528852232344113391163627325903147313917709424336259927104753940216755725750274458523714007365097727255811305376142037353501047709712838277690163245896206784321605812453814800933753033816113 23

q =
348574231621217916042354708984717615661151590845852695860078225594587747162771648825103588694762939391762876102748995097867368244617406036185985499452730294798252904590623704246574461516239056536321816780659754729682428228599269024630437306449584679218376877729069752748129055942114600949442715756980049203729057217988564298060400996988314717097740990034411115688434494524075427993274679446856302587480288751034447601525874935437991856466926840324608581509607904955759214554231857098113426891851279361119932487789622194134512585458630844037211356334284910464745404720295926131341257678640064955725042455383732079741 81

print(bytes.fromhex(hex(pow(c, pow(e, -1, (p-1)*(q-1)), n))[2:]).decode())

Flag: crypto{I_want_to_Break_Square-free_4p-1}

Everything is big:

n =
0x8da7d2ec7bf9b322a539afb9962d4d2ebeb3e3d449d709b80a51dc680a14c87ffa863edfc7b5a2a542a0f
a610febe2d967b58ae714c46a6eccb44cd5c90d1cf5e271224aa3367e5a13305f2744e2e56059b17bf520c
95d521d34fdad3b0c12e7821a3169aa900c711e6923ca1a26c71fc5ac8a9ff8c878164e2434c724b68b508a
030f86211c1307b6f90c0cd489a27fdc5e6190f6193447e0441a49edde165cf6074994ea260a21ea1fc7e2df
b038df437f02b9ddb7b5244a9620c8eca858865e83bab3413135e76a54ee718f4e431c29d3cb6e353a75d
74f831bed2cc7bdce553f25b617b3bdd9ef901e249e43545c91b0cd8798b27804d61926e317a2b745

e =
0x86d357db4e1b60a2e9f9f25e2db15204c820b6e8d8d04d29db168c890bc8a6c1e31b9316c9680174e12
8515a00256b775a1a8ccca9c6936f1b4c2298c03032cda4dd8eca1145828d31466bf56bfcf0c6a8b4a1b2fb
27de7a57fae7430048d7590734b2f05b6443ad60d89606802409d2fa4c6767ad42bffae01a8ef136441836
2e133fa7b2770af64a68ad50ad8d2bd5cebb99ceb13368fb31a6e7503e753f8638e21a96af1b6498c18578
ba89b98d70fa482ad137d28fe701b4b77baa25d5e84c81b26ee9bddf8cbb51a071c60dd57714de379cd4b
c14932809ba18524a0a18e4133665cfc46e2c4fcfbc28e0a0957e5513a7307c422b87a6182d0b6a074b4d

c =
0x6a2f2e401a54eeb5dab1e6d5d80e92a6ca189049e22844c825012b8f0578f95b269b19644c7c8af3d544
840d380ed75fdf86844aa8976622fa0501eaec0e5a1a5ab09d3d1037e55501c4e270060470c9f4019ced6c
4e67673843daf2fd71c64f3dd8939ae322f2b79d283b3382052d076ebe9bb50b0042f1f7dd7beadf0f56869
26ade9fc8370283ead781a21896e7a878d99e77c3bb1f470401062c0e0327fd85da1cf12901635f1df310e8
f8c7d87aff5a01dbbecd739cd8f36462060d0eb237af8d613e2d9cebb67d612bcfc353ef2cd44b7ac85e471
287eb04ae9b388b66ea8eb32429ae96dba5da8206894fa8c58a7440a127fceb5717a2eaa3c29f25f7

p =
11550729043680468185397251378585522909233408035687471788343423823553266444140069832
96427512646522995762986365630345661549368128949762008111163956276428241298812018796
64681775402664283913508399125714656956248098339209538838857780042711388040763913184
27661198573263589390952751421487658123810872483961480583791 9

q =
15481583883073575626683989700213231453867590913524995485254210017959019205525710060
17595234014093800494632662653237804065546922480122244854694683553256989798250727263
57567352976501667578692853621821116743497998613306224857520639271685772211046709014
87610448176690903652234922350632685294370994004783608084553 1

print(bytes.fromhex(hex(pow(c, pow(e, -1, (p-1)*(q-1)), n))[2:]).decode())

crypto{s0m3th1ng5_c4n_b3_t00_b1g}

Everything is still big:

n =
0x665166804cd78e8197073f65f58bca14e019982245fcc7cad74535e948a4e0258b2e919bf3720968a00e
5240c5e1d6b8831d8fec300d969fccec6cce11dde826d3fbe0837194f2dc64194c78379440671563c6c7526
7f0286d779e6d91d3e9037c642a860a894d8c45b7ed564d341501cedf260d3019234f2964ccc6c56b6de8a
4f66667e9672a03f6c29d95100cdf5cb363d66f2131823a953621680300ab3a2eb51c12999b6d4249dde49
9055584925399f3a8c7a4a5a21f095878e80bbc772f785d2cbf70a87c6b854eb566e1e1beb7d4ac6eb4602
3b3dc7fdf34529a40f5fc5797f9c15c54ed4cb018c072168e9c30ca3602e00ea4047d2e5686c6eb37b9

e =
0x2c998e57bc651fe4807443dbb3e794711ca22b473d7792a64b7a326538dc528a17c79c72e425bf29937
e47b2d6f6330ee5c13bfd8564b50e49132d47befd0ee2e85f4bfe2c9452d62ef838d487c099b3d7c80f14e3
62b3d97ca4774f1e4e851d38a4a834b077ded3d40cd20ddc45d57581beaa7b4d299da9dec8a1f361c8086
37238fa368e07c7d08f5654c7b2f8a90d47857e9b9c0a81a46769f6307d5a4442707afb017959d9a681fa1d
c8d97565e55f02df34b04a3d0a0bf98b7798d7084db4b3f6696fa139f83ada3dc70d0b4c57bf49f530dec93
8096071f9c4498fdef9641dfbfe516c985b27d1748cc6ce1a4beb1381fb165a3d14f61032e0f76f095d

c =
0x503d5dd3bf3d76918b868c0789c81b4a384184ddadef81142eabdcb78656632e54c9cb22ac2c4117860
7aa41adebdf89cd24ec1876365994f54f2b8fc492636b59382eb5094c46b5818cf8d9b42aed7e8051d7ca1
537202d20ef945876e94f502e048ad71c7ad89200341f8071dc73c2cc1c7688494cad0110fca4854ee6a1ba
999005a650062a5d55063693e8b018b08c4591946a3fc961dae2ba0c046f0848fbe5206d56767aae8812d
55ee9decc1587cf5905887846cd3ecc6fc069e40d36b29ee48229c0c79eceab9a95b11d15421b8585a2576
a63b9f09c56a4ca1729680410da237ac5b05850604e2af1f4ede9cf3928cbb3193a159e64482928b585ac

p =
984445496790444095062442391444438674598242279345260360529492782615058134390152974592003791087524442352326672131384640764150954869072882826305956222872372158014709401468863715156799093220908714734123848945406423999500102962145254696225057985260721701874675627659200446465744454273642315290836109557602282127012

q =
131205304707717699800023219057082007986286045823683571663112014612188606710079038751853416273709729039622908861933527111469616900188875912430487264576215232569029320804579614330240773622645122871884209068761138439268551367198798009790636662892148063583135747945604771740458352899202428704645256790931460695949

print(bytes.fromhex(hex(pow(c, pow(e, -1, (p-1)*(q-1)), n))[2:]).decode())

crypto{bon3h5_4tt4ck_i5_sr0ng3r_th4n_w13n3r5}

Marins secret:

n =
6584162748301845441250275199214435157898882641560747330992440401262136824977140327981163992881765024628292557845259777229030187144343096981082083886647682627543164262065157662373161788292316411757962482726124450608427437125027784935163167944117101841801849803999647254989315057718930287152031171517973071431218145624509784849166979599728983061298805852396838480882282837090019848924924339916512521924475379077976446623696513579357651619321317506140166738862222836204271705401467903295344103402150685601708106261757235119541850589938871570979599202955904211978342359732470710069406467590923871757305876411889322511160270383808061856540113990214306990111717420425287194884686443677180861643245710284453484385719873524200530907393905143379094672667223464325934953518626857162907793759783880133797309228560874420995153319986822804000443213259707339036335789237999765587885769633489221634507022764674985138120855404494044418286402651370944982348959343901736635886964816823873508759380834448436513628421972523381160533181500742458289082188726068288663254361310925286211432637207778536929257090059481448109744378126956264730367142889576422408440225960510960036309895009199988913758128395236132956672538139784348791727812172856528954691941812183430787545016947465987382152437697479565725559895945981806390983448911758794559946523821370382401663580664034754577
e = 65537

c =
4002804630889304323192803591151949775825173636105324642952106695304078707534391274554013845697054256214459439929633809830849173854286312230469088378041263993458752529170901841584403055038171932462886729864889878831773803073770250792660302626509325752051418534133025584603642423555312729674814094147836345587911758278165407675459445342381890790301928432885969349796935179646556615073467297519879281470216201650099650519332789139528991142533010447475873108304191906232825789315895875045550053615715725619168660634588129653144741604990675250674951401500921196209283630074673909201307175211691051679633641546364720550840125921385703543902467792760031561846762987107465831047005164660910345107650271679561178690519381164573703847374409651096195782274220498065660605718310176108770724842627247895710765295864274057801210965469428123223248071451370179422668635349890821151890655600118411509083809373543012431532064288963205766099043619370352639853489847942081988926158989070059554035294708471242695123161917539502037945786560293245066882934465715985060421982190803257473286362320409367617885584215289602798328021275621158523049468676283165029595622744854838674817311493382090097532294639248559301032711978313709824887034564633859148012468286622126220069473801155495298201973557385253298852321702157575856854844023444378949815551791292871649710020337597244 56

p = 2**2203-1

q = 2**2281-1

print(bytes.fromhex(hex(pow(c, pow(e, -1, (p-1)*(q-1)), n))[2:]).decode())

crypto{Th3se_Pr1m3s_4r3_t00_r4r3}

Unencryptable:

N =
0x7fe8cafec59886e9318830f33747cafd200588406e7c42741859e15994ab62410438991ab5d9fc94f3862
19e3c27d6ffc73754f791e7b2c565611f8fe5054dd132b8c4f3eadcf1180cd8f2a3cc756b06996f2d5b67c39
0adcba9d444697b13d12b2badfc3c7d5459df16a047ca25f4d18570cd6fa727aed46394576cfdb56b41

e = 0x10001

c =
0x5233da71cc1dc1c5f21039f51eb51c80657e1af217d563aa25a8104a4e84a42379040ecdfdd5afa191156
ccb40b6f188f4ad96c58922428c4c0bc17fd5384456853e139afde40c3f95988879629297f48d0efa6b3357
16a4c24bfee36f714d34a4e810a9689e93a0af8502528844ae578100b0188a2790518c695c095c9d677b

p =
8239835397208516111720362847949425401045672365829937602117480449316694558226622001
10057535873802132963548914201468383545676262090246827792522994758916609

q =
1090082435333447183000730752993735792616038646196788444616031521863068779334147107 9
17075054855470792661154201985929660518853541344779171006718643237197036 9

d = pow(e, -1, (p-1)*(q-1))

print(bytes.fromhex(hex(pow(c, d, N))[2:]).decode())

crypto{R3m3mb3r!_F1x3d_P0iNts_aR3_s3crE7s_t00}

```
from math import sqrt

# Hàm nhân vector

def dot_product(v1, v2):

    return sum(a*b for a, b in zip(v1, v2)) # tương đương với a in v1 * b in v2


# Hàm tính norm ~ 2

def vector_norm(v):

    return sqrt(dot_product(v, v))


vectors = [[4, 1, 3, -1], [2, 1, -3, 4], [1, 0, -2, 7], [6, 2, 9, -5]]
```

Gram smith

```
def gram_smith(vectors):
    u = []
    for i in range(len(vectors)):
        ui = vectors[i]
        for j in range(i):
            muj = dot_product(vectors[i], u[j]) / vector_norm(u[j])**2
            ui = [ui[k] - muj * u[j][k] for k in range(len(ui))]
        u.append(ui)
    return u


flag = round(gram_smith(vectors)[3][1],5)
print(flag)
```

gramschimedt:0.91611

Inferious prime

RSA

```
from Crypto.Util.number import getPrime, inverse, bytes_to_long, long_to_bytes, GCD


e = 3
n = 74244912912446707392154568764089512753570590245436975640133 1
ct = 392072743485784813223173406484755968073031601111338236677373


p = 752708788837165590355094155871
q = 98636968258528199393318528926 1


phi = (p-1)*(q-1)


d = inverse(e,phi)
```

a = pow(ct,d,n)

m = long_to_bytes(a)

print(m)

crypto{N33d_b1g_pR1m35}

Square Eyes:

N =
535860808044009550029177135708168016201451343147313565371014459027743491739422885443084705720731409713775527993719682583669164873806842043288439828071789970694759080842162253955259590552283047728782812946845160334801782088068154453021936721710269050985805054692096738777321796153384024897615594493453068138341203673749514094546000253631902991617197847584519694152122765406982133526594928685232381934742152195861380221224370858128736975959176861651044370378539093990198336298572944512738570839396588590096813217791191895941380464803377602779240663133834952329316862399581950590588006371221334128215409197603236942597674756728212232134056562716399155080108881105952768189193728827484667349378091100068224404684701674782399200373192433062767622841264055426035349769018117299620554803902490432339600566432246795818167460916180647394169157647245603555692735630862148715428791242764799469896924753470539857080767170052783918273180304835318388177089674231640910337743789750979216202573226794240332797892868276309400253925932223895530714169648116569013581643192341931800785254715083294526325980247219218364118877864892068185905587410977152737936310734712276956663192182487672474651103240004173381041237906849437490609652395748868434296753449

e = 65537

c =
222502885974182429500948389840563415291534726891354573907329512556439632810921927905220486727807436668035929302442754225952786602492250448020341217733646472982286222338860566076161977786095675944552232391481278782019346283900959677167026636830252067048759720251671811058647569724495547940966885025629807079171218371644528053562232396674283745310132242492367274184667845174514466834132589971388067076980563188513333661165819462428837210575342101036356974189393390097403614434491507672459254969638032776897417674577487775755539964915035731988499983726435005007850876000232292458554577437739427313453671492956668188219600633325930981748162455965093222648173134777571527681591366164711307355510889316052064146089646772869610726671696699221157985834325663661400034831442431209123478778078255846830522226390964119818784903330200488705212765569163495571851459355520398928214206285080883954881888668509262455490889283862560453598662919522224935145694435885396500780651530829377030371611921181207362217397805303962112100190783763061909945889717878397740711340114311597934724670601992737526668932871436226135393872881664511222789565256059138002651403875484920711316522536260604255269532161594824301047729082877262812899724246757871448545439896

from factor.db we get N's square root

p =
23148667521998097720857168827790771337662483716348435477360567409355026169165934446
94980966459552377085389720310375910698398511326404905741690819116672000850327595162
57389756660190291723776531706024403735795932925765306677739514076472227577564378672
16095193174201323278896027294517792607881861855264600525772460745259440301156930943
25524091568571855233419223026478035579917903781602633070542248400008654236208400695
81585503463959418623839259420337300300046063603083797762554362064405294417118592468
11586652746028418496020145441513037535475380962562108920699929022900677901988508936
50935438566073569456821663138265310 7

phi = (p)*(p-1)

from Crypto.Util.number import getPrime, inverse, bytes_to_long, long_to_bytes, GCD

d = inverse(e,phi)

m = pow(c,d,N)

print(m)

a = long_to_bytes(m)

print(a)

crypto{squar3_r00t_i5_f4st3r_th4n_f4ct0r1ng!}