```
def rsa(bits):
    proof = (bits <= 1024)
    p = next_prime(ZZ.random_element(2**(bits//2 +1)), proof=proof)
    q = next_prime(ZZ.random_element(2**(bits//2 +1)), proof=proof)
    n = p * q
    phi_n = (p-1) * (q-1)
    while True:
        e = ZZ.random_element(1,phi_n)
        if gcd(e,phi_n) == 1: break
    d = lift(Mod(e,phi_n)^(-1))
    return e, d, n
def encrypt(m,e,n):
    return lift(Mod(m,n)^e)
def decrypt(c,d,n):
    return lift(Mod(c,n)^d)

time e,d,n = rsa(1024); e, d, n
       (631241565900199513268767904037921496219457548531544752548345494060
       817169013986707768544273240080569836943410066543839239641054931063
       191808550484201142739880038363854857350511918297842484571414505845
       099913736069348082357705777157999793076905664512086822966223270507
       634180050077957909415627340129964381230068634858717323291259574370
       227764752028617666852991869313441258260600079670884454394679596764
       970888592313358707678614754368587980774762785427461179594549589274
       168496083810078268688594011504973533793619171144981771155740001204
       845507422030977109941218875496529968952107836300534202452314364876
       634185028251962245165798810527323686317809729172004321580996010821
       620671134600833200404456069014896673340633923272476398863502225356
       334198731041353176870689864208531240741550480992808236203206254753
       CPU time: 2.60 s,  Wall time: 6.15 s
time c = encrypt(123456, e, n)
       CPU time: 0.00 s,  Wall time: 0.00 s
time decrypt(c, d, n)
       123456
       CPU time: 0.00 s,  Wall time: 0.01 s
time e,d,n = rsa(2048)
       CPU time: 0.46 s,  Wall time: 0.74 s
time e,d,n = rsa(4096)
       CPU time: 19.43 s,  Wall time: 30.86 s
def encode(s):
    s = str(s)
    return sum(ord(s[i])*256^i for i in range(len(s)))

def decode(n):
    n = Integer(n)
    v = []
    while n != 0:
        v.append(chr(n % 256))
        n //= 256
    return ''.join(v)

m = encode('Run Nikita!'); m
       403547690147146494219687 22
decode(m)
       'Run Nikita!'
```

```
def crack_rsa(n, phi_n):
    R.<x> = PolynomialRing(QQ)
    f = x^2 - (n+1 -phi_n)*x + n
    return [b for b, _ in f.roots()]

crack_rsa(31615577110997599711, 31615577098574867424)
        [8850588049, 3572144239]
def crack_when_pq_close(n):
    t = Integer(ceil(sqrt(n)))
    while True:
       a = (abs(t^2 - n)).sqrt_approx()
       s = Integer(int(a))
       if s^2 + n == t^2:
          return t+s, t-s
       t += 1

crack_when_pq_close(23360947609)
        (153649, 152041)
p = next_prime(2^128); p
        340282366920938463463374607431768211507
q = next_prime(p)

crack_when_pq_close(p*q)
        (340282366920938463463374607431768211537,
        340282366920938463463374607431768211507)
def crack_given_decrypt(n, m):
    n = Integer(n); m = Integer(m);  # some type checking
    # Step 1: divide out powers of 2
    while True:
        if is_odd(m): break
        divide_out = True
        for i in range(5):
           a = randrange(1,n)
           if gcd(a,n) == 1:
              if Mod(a,n)^(m//2) != 1:
                  divide_out = False
                  break
        if divide_out:
            m = m//2
        else:
            break
    # Step 2: Compute GCD
    while True:
        a = randrange(1,n)
        g = gcd(lift(Mod(a, n)^(m//2)) - 1, n)
        if g != 1 and g != n:
            return g

n=32295194023343; e=29468811804857; d=11127763319273

crack_given_decrypt(n, e*d - 1)
        737531
factor(n)
        737531 * 43788253
e,d,n=rsa(256)
```

```
e
        208419239869510953040749673175978554404565813749839646529258913373
d
        760666308539660470400455105378488612796042185030000035976781471954
n
        188218598425585157023155473424475715600563298821418224755019084102
len(n.str(2))
        251
p = crack_given_decrypt(n, e*d - 1); p
        1220258534850384284032604317860091450 3
n % p
        0
192/2
        96
n = 427614569095223689307737850060005054546598668 9

n = next_prime(randrange(2^96))*next_prime(randrange(2^97))

len(n.str(2))
        190
time qsieve(n)
        ([305420967330157491016304123 63, 3321084641677740025224112129 7], '
        CPU time: 0.03 s,  Wall time: 15.79 s
len(n.str(2))
        190
time factor(n)
        305420967330157491016304123 63 * 3321084641677740025224112129 7
        CPU time: 13.07 s,  Wall time: 22.65 s
```