d=5, p=1, q=-1 // P=1, Q=-1

al= (p+(d\*\*0.5))\*0.5

be= (p-(d\*\*0.5))\*0.5

Define function isSquare(x)

if x equals 1, return 1

low = 0, high = x // 2, root = high

while root \* root != x

root = (low + high) // 2

if low + 1 >= high, return 0

if root \* root > x, high = root

else low = root

return 1

Define function js(a,p) //to find the Jacobian symbol

if(a%p equals 0), return 0

else if (isSquare(a%p) equals 1), return 1

else return -1

Define function U(n) // returns the value of Un, where n is the coefficient using recursion

if(n equals 0), return 0

elif(n equals 1), return 1

else return p\*U(n-1)-q\*U(n-2)

Define u(k)

return((pow(al,k)-pow(be,k))/(al-be))

Define function isprime(n)

// The function returns true if n is prime

if n equals 2, return True

if n equals 3, return True

if n % 2 equals 0, return False

if n % 3 equals 0, return Falsse

i = 5, w = 2

while i \* i <= n:

if n % i equals 0, return False

i += w

w = 6 - w

return True

t=time.time() //stores the present time

num=3

nos=0

while(nos<10)

delta=n-js(d,n)

a=u(delta)%n

if(a equals 0):

if(prime(n) equals 0 and gcd(n,q) equals 1):

print n

nos=nos+1

num=num+2

b=time.time()

print(t-b)