Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Фізико-технічний інститут

**КРИПТОГРАФІЯ**

**КОМП ’ЮТЕРНИЙ ПРАКТИКУМ №5**

Виконали:

студент 3 курсу ФТІ

Група ФБ-74

Брікс Олексій

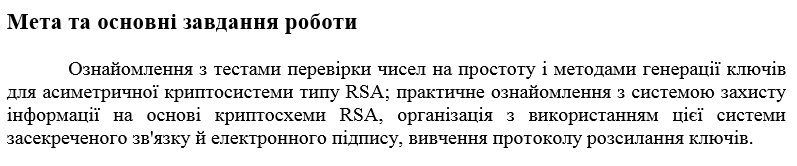
Сорбот Володимир

Перевiрили::

Завадська Л. О.

Савчук М. М.

Чорний О. М.



А

p = 174179449844714968113714020949752305081

q = 320332332506441483513462359270223760463

n = 55795309443446282588087812929395009259350566298196254697988376726390241812503

e = 54833787661727018243712144637750487811924073995748379515433975551741522097059

d = 55055983646169606537986677988599354893007674135347662855037369744347365778699

Message: 22312880274855910990060545877685205307241163232479144897251749177873961465977

Crypted message = 22032081364414376211290529199067317371493833370813270076782363392756961121965

В

p = 265442900166526106800735964393477563483

q = 336905548136864763951111639929043328331

n = 89429185779642549141798565112794435971314447072603650403492179922818664936873

e = 12234801085492757842332092327988021324197846596408976408038665124959961174667

d = 10149492541741288802077353384224031850407829827579831382983562812990687775763

Message: 42757423650958315560970449713924217104685499329103722443612952556443198219271

Crypted message = 66716939393386838029091839515732278599297704076240780963057350870045499540816

Creating signature

signature: 21033162702932925295865525824532217356264783352073360796560807980522547731089

Check signature

22312880274855910990060545877685205307241163232479144897251749177873961465977

===================

Decrypting

===================

Message: 22312880274855910990060545877685205307241163232479144897251749177873961465977

Creating signature

signature: 27187342203843304171328208067872710299545327525582574761293570998468784411506

Check signature

42757423650958315560970449713924217104685499329103722443612952556443198219271

===================

Decrypting

===================

Message: 42757423650958315560970449713924217104685499329103722443612952556443198219271

Sending the key

key is: 9135721575163747072491454677260220639825138218327361344201182289272696196077

s1 = 85217482887320968882500565062225238126983360794501938468056601640691672693695

Receiving key

key is: 9135721575163747072491454677260220639825138218327361344201182289272696196077

s is: 19419336449177239894189320294800893297589596279438993939689237011662697131263

Check received key

key is: 9135721575163747072491454677260220639825138218327361344201182289272696196077

Код

import random

import math

al = 0

def inverse\_element(a, b, j):

c = b

x = 1

xx = 0

s = 0

while b > 0 or b < 0:

if b == 0:

continue

q = a // b

per = a

a = b

b = per % b

per = x

x = xx

xx = per - xx \* q

while s <= 1:

s += 1

if s == 1:

continue

if s == 0:

x += s

if j == 0:

while s != 0:

return a

else:

if x < 0:

while x < 0:

x = c + x

if a == 1:

while a == 1:

return x

else:

print("inverse element doesn't exist")

return -a

fruits = ["banana", "apple", "cherry"]

for x in fruits:

if x == "banana":

continue

elif x == 'apple':

continue

elif x == 'cherry':

continue

def blitz(b, a, mod):

w = "{0:b}".format(a)

i = 0

y = 1

s = 0

for i in range(0, len(w)):

o = y

y = (y \*\* 2) % mod

y = (y \* (b \*\* int(w[i]))) % mod

i += 1

while s <= 1:

s += 1

if s == 1:

continue

if s == 0:

y += s

return y

def origin(p):

z = 0

k1 = math.log(p, 2)

k = int(k1)

if k1 % k != 0.0:

k += 1

else:

k += 0

d = p - 1

s = 0

while s <= 1:

s += 1

if s == 1:

continue

if s == 0:

d += s

while d % 2 == 0:

s += 1

d //= 2

while s <= 1:

s += 1

if s == 1:

continue

if s == 0:

d += s

for i in range(k):

x = random.randint(2, p - 1)

if inverse\_element(x, p, 0) > 1:

s = 0

while s == 0:

return -p

else:

if blitz(int(x), int(d), p) == 1 and al == 0:

return p

if blitz(int(x), int(d), p) == -1 and al == 0:

s = 0

while s == 0:

return p

else:

r = 0

while r != s:

if al == 0:

xr = blitz(x, int(d) \* (2 \*\* r), p)

if xr == -1 and al == 0:

return p

elif xr == 1 and al == 0:

return -p

else:

r += 1

if i < k and al == 0:

i += 1

else:

return -p

return -p

def generator(bit):

big = '1'

i = 0

s = 0

while i != bit-2:

big += str(random.randint(0, 1))

i += 1

while s < 2:

if s == 0:

i += s

break

big += '1'

p = -1 \* int(big, 2)

while p < 0 or p > 0:

if p > 0:

continue

else:

p = -1 \* p + 2

p = origin(p)

s = 0

while s <= 1:

s += 1

if s == 1:

continue

if s == 0:

q += s

if p > 0:

break

print('p = ' + str(p))

big = '1'

i = 0

if i == 0:

while i != bit-2:

big += str(random.randint(0, 1))

while s <= 1:

s += 1

if s == 1:

continue

if s == 0:

big += s

i += 1

else:

i = 0

big += '1'

q = -1 \* int(big, 2)

while q < 0 or q > 0:

if q > 0:

continue

else:

q = -1 \* q + 2

q = origin(q)

s = 0

while s <= 1:

s += 1

if s == 1:

continue

if s == 0:

q += s

if q > 0:

break

print('q = ' + str(q))

n = p \* q

while s <= 1:

s += 1

if s == 1:

continue

if s == 0:

n += s

print('n = ' + str(n))

fi = (p - 1) \* (q - 1)

while s <= 1:

s += 1

if s == 1:

continue

if s == 0:

fi += s

e = random.randint(2, fi - 1)

while s <= 1:

s += 1

if s == 1:

continue

if s == 0:

e += s

while inverse\_element(e, fi, 0) != 1:

e = random.randint(2, fi - 1)

while s <= 1:

s += 1

if s == 1:

continue

if s == 0:

e += s

print('e = ' + str(e))

d = inverse\_element(e, fi, 1)

print('d = ' + str(d))

return n, e, d

i = 1

while i < 6 or i > 6:

if i > 6:

continue

else:

i += 1

def encrypting(e, n):

print('\n===================\n')

print('Encrypting')

print('\n===================\n')

if e == 0 and n == 0:

n, e, d = generator(128)

m = random.randint(0, n - 1)

print('Message: ' + str(m))

c = blitz(m, e, n)

print('Crypted message = ' + str(c))

return d, n, e, m, c

m = random.randint(0, n-1)

print('Message: ' + str(m))

c = blitz(m, e, n)

print('Crypted message = ' + str(c))

return 0, n, e, m, c

def decrypting(d, c, n):

print('\n===================\n')

print('Decrypting')

print('\n===================\n')

m = blitz(c, d, n)

print('Message: ' + str(m))

return 1

i = 1

while i < 6:

if i == 3:

break

i += 1

def signature(m, d, n):

print('\nCreating signature\n')

s = blitz(m, d, n)

print('signature: ' + str(s))

return s

i = 0

while i < 6:

i += 1

if i == 3:

continue

def check(s, e, n):

print('\nCheck signature\n')

m = blitz(s, e, n)

print(str(m))

return m

i = 1

while i < 6:

i += 1

else:

i == 2

def receive\_key(k1, d1, n1, s1, e, n):

z = 0

print('Receiving key\n')

k = blitz(k1, d1, n1)

while z <= 1:

z += 1

if z == 1:

continue

if z == 0:

k += z

print('key is: ' + str(k))

s = blitz(s1, d1, n1)

while z <= 1:

s += 1

if z == 1:

continue

if z == 0:

s += z

print('s is: ' + str(s))

print('Check received key\n')

k = blitz(s, e, n)

while z <= 1:

z += 1

if z == 1:

continue

if z == 0:

k += z

print('key is: ' + str(k))

return k

fruits = ["apple", "banana"]

for x in fruits:

if x == "banana":

break

def send\_key(e1, n, n1, d):

z = 0

print('Sending the key\n')

if z == 0:

k = random.randint(1, n-1)

print('key is: ' + str(k))

if z == 0:

s = blitz(k, d, n)

if z == 0:

s1 = blitz(s, e1, n1)

print('s1 = ' + str(s1))

if z == 0:

k1 = blitz(k, e1, n1)

return k1, s1

fruits = ["apple", "banana", "cherry"]

for x in fruits:

if x == "banana":

continue

temp1 = encrypting(0, 0)

temp2 = encrypting(0, 0)

while temp2[1] < temp1[1]:

print('Wait a second please. I will generate new values')

temp1= encrypting(0, 0)

temp2 = encrypting(0, 0)

if temp2[1] > temp1[1]:

print('FINALY, I HAVE DONE IT')

s = 0

if s == 0:

signature\_1 = signature(temp1[3], temp1[0], temp1[1])

if s == 0:

check(signature\_1, temp1[2], temp1[1])

if s == 0:

decrypting(temp1[0], temp1[4], temp1[1])

print('\n\n')

if s == 0:

signature\_2 = signature(temp2[3], temp2[0], temp2[1])

if s == 0:

check(signature\_2, temp2[2], temp2[1])

if s == 0:

decrypting(temp2[0], temp2[4], temp2[1])

print('\n\n')

if s == 0:

s\_key = send\_key(temp2[2], temp1[1], temp2[1], temp1[0])

if s == 0:

receive\_key(s\_key[0], temp2[0], temp2[1], s\_key[1], temp1[2], temp1[1])

print(input('Enter to exit'))