5) Substitution to affine cipher:

Code:

def egcd(a, b):

x,y, u,v = 0,1, 1,0

while a != 0:

q, r = b//a, b%a

m, n = x-u\*q, y-v\*q

b,a, x,y, u,v = a,r, u,v, m,n

gcd = b

return gcd, x, y

def modinv(a, m):

gcd, x, y = egcd(a, m)

if gcd != 1:

return None

else:

return x % m

def affine\_encrypt(text, key):

'''

C = (a\*P + b) % 26

'''

return ''.join([ chr((( key[0]\*(ord(t) - ord('A')) + key[1] ) % 26)

+ ord('A')) for t in text.upper().replace(' ', '') ])

def affine\_decrypt(cipher, key):

'''

P = (a^-1 \* (C - b)) % 26

'''

return ''.join([ chr((( modinv(key[0], 26)\*(ord(c) - ord('A') - key[1]))

% 26) + ord('A')) for c in cipher ])

def main():

text = 'AFFINE CIPHER'

key = [17, 20]

affine\_encrypted\_text = affine\_encrypt(text, key)

print('Encrypted Text: {}'.format( affine\_encrypted\_text ))

print('Decrypted Text: {}'.format

( affine\_decrypt(affine\_encrypted\_text, key) ))

if \_\_name\_\_ == '\_\_main\_\_':

main()

output:

Encrypted Text: UBBAHKCAPJKX

Decrypted Text: AFFINECIPHER