```
import time





#Variable Initalization
def attack1(e, n, C):                                           # This attack 1 
function will begin by taking three parameters e n and C it will then progress to 
compute i such that i^e mod n =C'
    time_start = time.time()                                    # Imported from the 
time libairy this will keep track of how long the attack has taken to create these 
lists

    found = False                                               #Initalizes found 
to false and i to 0 to begin from couting i from 0
    i = 0

    while not found:                                            #This while loop 
will remain true until found is equal to true, found will equal true once the value 
of i is computed with M = i^e mod n to be queal to C (M == C) this will ensure you 
have found the correct i within the equation
        i += 1
        M = pow(i, e, n)                                        #pow function will 
return the value of i to the power of e and since there is a third value present it 
will return i to the power e with modulus n
        if M == C:
            found = True

    time_end = time.time()


    return i, time_end - time_start


def attack2(e, n, C):                                            # Attack2  too 
will take in same parameters as 1 but it will utalize the facrotize funtion to find 
factors p and q of n it will then compute phi = (p-1) * (q-1)
    time_start = time.time()                                     # Next this 
function will call find_d in order to find the value d which satisfies d*e mod phi 
= 1. After success it will compute M = C^d mod n
                                                                 # Using the naive 
and inefficient modular exponentiation algorithm it i will brute force through and 
return p q d and M
    p, q = factorize(n)
    phi = (p-1) * (q -1)
    d = find_d(e, phi)

    M = pow(C , d, n)                                            # Naive and 
inefficient modular exponentiation

    time_end = time.time()
    return f"{p, q, d, M,}, {time_end - time_start}"

def factorize(n):                                                # Takes in integer 
parameter n and will return the two factors of i and n//i where i is the smallest 
factor of n > 1
    for i in range(2, n):
        if n % i == 0:
            return i, n//i                                       # Returns the
current value of i and then the integral result with the discarded remainder

def find_d(e, phi):                                              # Takes in two
parameters e and phu returns an integer value d such that d * e mod phi = 1, by
iterating over values of d from 2 to phi and checking if (d* e) mod phi = 1 and if
so return d
    for d in range(2, phi):
        if (d*e) % phi == 1:
            return d



# #a and b with attack1
a1_1 = attack1(e = 3,n = 15,C = 8)
a2_1 = attack1(e = 13, n = 527, C = 152)

# #a and b with attack2
a_1_2 = attack2(e = 3, n = 15, C = 8)
a_2_2 = attack2(e = 13, n = 527, C = 152)

#Guess and check c for attack1 and attack2
#Attack 1
a3_1 = attack1(e = 445, n = 51500, C = 43)

#Attack 2
a3_2 = attack2(e = 445, n = 51500, C = 300)



print(" Attack 1: ","e = 3, n = 15, C = 8 :", a1_1, "\n ",              "e = 13, n
= 527 , C = 152 : ", a2_1, "\n\n", " Attack 2: ","e = 3, n = 15, C = 8 : ", a_1_2,
"\n" ,
                     "e = 13, n = 527 , C = 152 : ", a_2_2)
print("----------------------------------------------------------------------")
print(" Part C Guess Attack 1:  ", "e = 445, n = 51500, C = 43 : ", a3_1, "\n", "
Part C Guess Attack 2: ", "e = 445, n = 51500, C = 300 : ", a3_2  )






#============================== Guess and Check Part C Work
=============================================#
#Values with resonable results
# e = 12 , n = 534, C = 4 // Results: (4,0.0)
# e = 445, n = 51500, C = 43 // Attack 1:  (4823, 0.0039899349212646484)
# Attack 1:  (123, 0.0)
# Attack 1:  (43, 0.0)
# Attack 1:  (123, 0.0)
#




#Values that did not work
```

```
#Attack 1
# e = 18, n = 661, C = 252
# e = 12, n = 528, C = 100
# e = 12, n = 20, C = 100
# e = 12, n = 54, C = 56
# e = 12, n = 534, C = 4
# e = 12, n = 528, C = 173
# e = 12, n = 3, C = 321
# e = 12, n = 534, C = 4
# e = 12, n = 528, C = 173
# e = 12, n = 54, C = 56
# e = 12, n = 534, C = 4
# e = 12, n = 528, C = 173
# e = 445, n = 51500, C = 42
# e = 445, n = 323, C = 43
#e =


#Attack 2
# e = 24, n = 771, C = 420
# e = 16, n = 771, C = 300
# e = 12, n = 54, C = 56
# e = 12, n = 334, C = 4
# e = 12, n = 451, C = 173
# e = 12, n = 543, C = 321
# e = 12, n = 5321, C = 4
# e = 12, n = 528, C = 173
# e = 12, n = 54, C = 56
# e = 12, n = 534, C = 4
# e = 12, n = 528, C = 173
# e = 445, n = 51500, C = 42
# e = 445, n = 323, C = 43
```