70 分：

```
plaintext = input("Please input plaintext： ")
P1 = input("Please input Prime1： ")
P2 = input("Please input Prime2： ")
#while(1):
#    P1 = 1
#    while not(miller_rabin(P1,512)):
#        P1 = random.getrandbits(1024)
#    P2 = 1
#    while not(miller_rabin(P2,512)):
#        P2 = random.getrandbits(1024)
#    if(P1*P2>=int(plaintext)and P1!=P2 and P1*P2>=1.340781e+154):
#        break
E,D,CRT_D = RSA_GO(int(P1),int(P2),int(plaintext))
```

```
Please input plaintext： 2018
Please input Prime1： 71
Please input Prime2： 83
Cipher is 2221
DeCipher is 2018
CRT DeCipher is 2018
```

P1&P2：

產生兩個大質數，驗證次數為 512 次，並且兩質數相乘 N 要大於等於 1024bit

```
while(1):
    P1 = 1
    while not(miller_rabin(P1,512)):
        P1 = random.getrandbits(1024)
    P2 = 1
    while not(miller_rabin(P2,512)):
        P2 = random.getrandbits(1024)
    if(P1*P2>=int(plaintext)and P1!=P2 and P1*P2>=1.340781e+154):
        break
```

```python
def miller_rabin(N,validate_count):
    if (N<2):
        return False
    if(N==3 or N==2):
        return True
    if(N%2==0):
        return False
    u,x = 1,N-1
    r = x
    while x % pow(2,u) != 0:
        r = x / pow(2,u)
        u+=1

    for i in range(validate_count):
        candidate = random.randrange(2,N-1)
        b = pow(candidate,r,N)
        if(b==1 or b==N-1):
            continue
        for j in range(u-1):
            b = pow(b,2,N)
            if(b==N-1):
                break
            elif(b==1):
                return False
        else:
            return False
    return True
```

P3：

```python
def square_mul(x,y,N):#x^y
    output = x
    for i in y[1:]:
        output = pow(output,2) % N
        if(i=='1'):
            output = output * x % N
    return output
```

P4

```python
def CRT(D,P1,P2,Cipher):
    Dp = D % (P1-1)
    Dq = D % (P2-1)
    (Xq,_,_) = ext_GCD(P2,P1)
    return (Xq*P2*(square_mul(Cipher,bin(Dp)[2:],P1))+(1-Xq*P2)*(square_mul(Cipher,bin(Dq)[2:],P2))) % (P1*P2)
```

整個 RSA 流程：

找到兩質數後，算 N 和 PHI，再找出和 PHI 互質的數 e，透過 Extended Euclidean algorithm，得出 e_inverse d，透過 Square & multiply 加速運算，分別

使用一般的和 Chinese Remainder Theorem 進行解密，得出結果

```
def RSA_GO(P1,P2,text):
    N = P1*P2
    PHI = (P1-1)*(P2-1)
    for i in range(2,PHI):
        if(gcd(i,PHI)==1):
            e = i
            break
    (d,_,_) = ext_GCD(e,PHI)

    cipher = square_mul(text,bin(e)[2:],N)
    Decipher = square_mul(cipher,bin(d % PHI)[2:],N)
    CRT_Decipher = CRT(d % PHI,P1,P2,cipher)
    return cipher,Decipher,CRT_Decipher
```

```
plaintext = input("Please input plaintext : ")
#P1 = input("Please input Prime1 : ")
#P2 = input("Please input Prime2 : ")
while(1):
    P1 = 1
    while not(miller_rabin(P1,512)):
        P1 = random.getrandbits(1024)
    P2 = 1
    while not(miller_rabin(P2,512)):
        P2 = random.getrandbits(1024)
    if(P1*P2>=int(plaintext)and P1!=P2 and P1*P2>=1.340781e+154):
        break
E,D,CRT_D = RSA_GO(int(P1),int(P2),int(plaintext))
```

```
Please input plaintext : 2018
Cipher is 33466154331649568
DeCipher is 2018
CRT DeCipher is 2018
```