HW4

100分標準__檔名：

rsa_1024.py

建置環境

mac + vscode + python3

操作方法：

python3 rsa_1024.py [plaintext]

會輸出：

p,q,n,e,d,ciphertext 和plaintext

輸出結果圖：

Code

Miller test:

```python
def isprime(n):
    n = int(n,2)
    if (n == 2 ):
        return True
    if (n < 2):
        return False

    a = random.randint(1,n) % (n-2) + 2

    u = n-1
    t = 0
    while (u % 2 == 0):
        u >>= 1
        t+=1

    x = exp_mod(a, u, n)  # x = a ^ u % n
    if (x == 1 or x == n-1):
        return True

    for i in range(t-1):

        x = exp_mod(x, 2, n);   # x = x * x % n;
        if (x == 1):
            return False
        if (x == n-1):
            return True

    return False
```

Extend gcd:

```python
def xgcd(a, b):
    """return (g, x, y) such that a*x + b*y = g = gcd(a, b)"""
    x0, x1, y0, y1 = 0, 1, 1, 0
    while a != 0:
        q, b, a = b // a, a, b % a
        y0, y1 = y1, y0 - q * y1
        x0, x1 = x1, x0 - q * x1
    return b, x0, y0
```

Modinv:

```
def modinv(a, b):
    """return x such that (x * a) % b == 1"""
    g, x, _ = xgcd(a, b)

    if g == 1:
        return x % b
    else:
        raise Exception('modular inverse does not exist')
```

如果找不到會丟exception

Square and multiply:

```
def exp_mod( a, b, n):
    """return (a ** b )%n"""
    r = int(1)
    while(b):
        if(b&1):
            r=(r*a)%n
        a=(a*a)%n
        b>>=1        # b = b>>1

    return r
```

gen_prime:

```
def gen_prime():
    prime = ''
    for i in range(1024):
        if i == 1023 or i == 0:
            prime += '1'
        else:
            prime += random.choice(['0','1'])


    while isprime(prime) == False:
        prime = ''
        for i in range(1024):
            if i == 1023 or i == 0:
                prime += '1'
            else:
                prime += random.choice(['0','1'])
    return int(prime,2)
```

key_generator:

```python
def key_generator():
    p = gen_prime()
    q = gen_prime()
    n = p * q
    # print('p = ',p)
    # print('q = ',q)
    # print('n = ',n)
    phi_n = (p - 1) * (q - 1)
    pe = [e for e in range(10000) if hcfnaive(e,phi_n) == 1]
    e = random.choice(pe)
    while (1):
        try:
            d = modinv(e, phi_n)
            break
        except:
            p = gen_prime()
            q = gen_prime()
            n = p * q
            phi_n = (p - 1) * (q - 1)
            continue

    # print('e = ',e)
    # print('d = ',d)
    return p,q,n,e,d
```

會確保e 能產生乘法反元素d

心得：

花比較多時間在了解數學定理，產生質數後只要確保 e 能生成 d 並且e * d % phi_n = 1
keygenerator就沒什麼問題了，後面只要用square and multiply 做加解密就好了。
基本上只要理解數學公式程式就這沒有那麼難了。