Lab-9
Crytography
Name: Smraddhi Rathore
Roll No.: 2021bcy0025

```
import random

def is_prime(n, k=5):
    if n == 2 or n == 3:
        return True
    if n <= 1 or n % 2 == 0:
        return False
    d = (n - 1) >> 1
    r = 1
    while d % 2 == 0:
        d >>= 1
        r += 1
    for _ in range(k):
        a = random.randint(2, n - 2)
        x = pow(a, d, n)
        if x != 1 and x != n - 1:
            j = 1
            while j < r and x != n - 1:
                x = pow(x, 2, n)
                if x == 1:
                    return False
                j += 1
            if x != n - 1:
                return False

    return True

def find_primitive_root(p):
    if p == 2:
        return 1
    p1 = 2
    p2 = (p - 1) // p1

    while(1):
        g = random.randint(2, p - 1)
        if not (pow(g, (p - 1) // p1, p) == 1):
            if not pow(g, (p - 1) // p2, p) == 1:
                return g
```

```python
def generate_keys():
    p = 10007
    g = find_primitive_root(p)
    x = random.randint(2, p - 2)
    y = pow(g, x, p)
    return ((p, g, y), x)

def encrypt(public_key, message):
    p, g, y = public_key
    k = random.randint(2, p - 2)
    c1 = pow(g, k, p)
    c2 = (message * pow(y, k, p)) % p
    return (c1, c2)


def decrypt(public_key, private_key, cipher):
    p, _, _ = public_key
    c1, c2 = cipher
    x = private_key
    s = pow(c1, x, p)
    s_inv = pow(s, -1, p)
    return (c2 * s_inv) % p

if __name__ == "__main__":
    public_key, private_key = generate_keys()
    message = 1234
    cipher = encrypt(public_key, message)
    decrypted_message = decrypt(public_key, private_key, cipher)
    print("Original Message:", message)
    print("Encrypted Message:", cipher)
    print("Decrypted Message:", decrypted_message)
```

```
Decrypted Message: 1234
smraddhi@smraddhis-MacBook-Air Cryptography % python -u "/Users/sm
raddhi/Documents/Cryptography/elgamal.py"
Original Message: 1234
Encrypted Message: (5694, 99)
Decrypted Message: 1234
```