**Practical No : 10**

**Problem Statement:**

Implement Diffie Hellman algorithm.

**Program:**

import random

#Driver Code

def main():

#Select prime q and alpha

q = 353

alpha = 3

#Generate Private key

# Xa = 97# Xb = 233

Xa = random.randint(1, q-1)

Xb = random.randint(1, q-1)

#Calculate Public key

Ya = (alpha \*\* Xa) % q

Yb = (alpha \*\* Xb) % q

#Calculate Shared Private key

K1 = (Yb \*\* Xa) % q

K2 = (Ya \*\* Xb) % q

return K1,K2,q,alpha,Xa,Xb,Ya,Yb

if \_\_name\_\_ == "\_\_main\_\_":

k1,k2,q,alpha,xa,xb,ya,yb = main()

print(f"------Publicly Shared-------")

print(f"Prime Number: {q}")

print(f"alpha: {alpha}\n")

print(f"------Alice-----")

print(f"Private Key: Xa = {xa} ")

print(f"Public Key: Ya = {ya}")

print(f"Shared Key: K = {k1}\n")

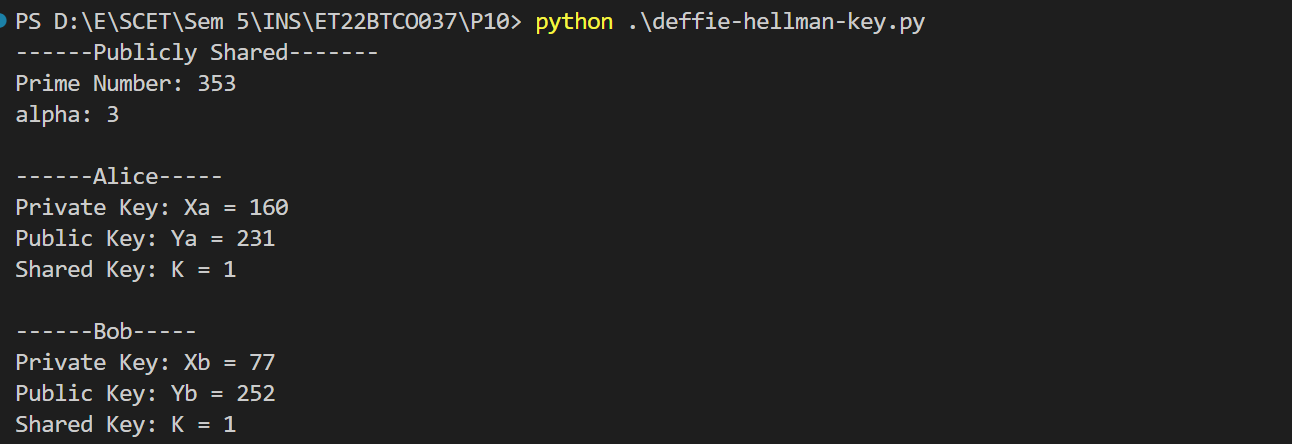
print(f"------Bob-----")

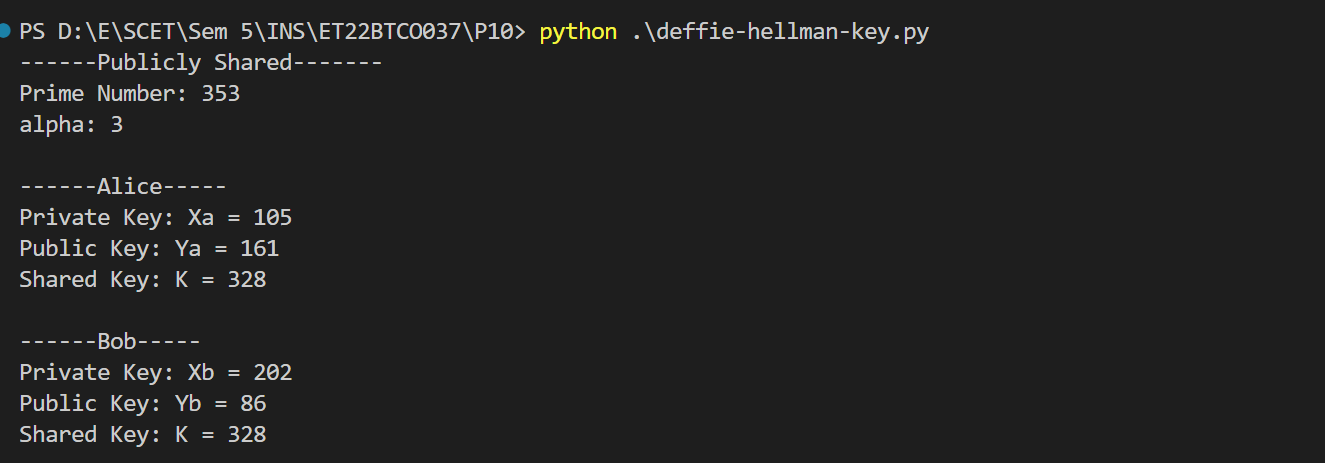
print(f"Private Key: Xb = {xb} ")

print(f"Public Key: Yb = {yb}")

print(f"Shared Key: K = {k2}\n")

**Output:**

****



**Conclusion:**

I implemented the Diffie-Hellman key exchange algorithm in Python. This algorithm allows two parties to securely establish a shared secret key over an insecure channel using prime numbers and modular arithmetic. However, it is vulnerable to man-in-the-middle attacks without proper authentication, highlighting the need for additional security measures in practical applications.