

EXP NO: 2b

DATE:

## DIFFIE HELMAN KEY EXCHANGE

AIM:

To write a C program to implement and perform Diffie Helman Key Exchange process.

ALGORITHM:

Step 1 : Start.

Step 2 : Include necessary header files.

Step 3 : Create a function for calculating primitive root of a number.

Step 4 : Define primitive\_root function.

Step 5 : Create a function for calculating public key.

Step 6 : Define calc\_public\_key function.

Step 7 : Call calc\_public\_key function inside main function.

Step 8 : End.

## PROGRAM:

```
def prime_checker(p):
    # Checks If the number entered is a Prime Number or not
    if p < 1:
        return -1
    elif p > 1:
        if p == 2:
            return 1
        for i in range(2, p):
            if p % i == 0:
                return -1
        return 1

def primitive_check(g, p, L):
    # Checks If The Entered Number Is A Primitive Root Or Not
    for i in range(1, p):
        L.append(pow(g, i) % p)
    for i in range(1, p):
        if L.count(i) > 1:
            L.clear()
            return -1
    return 1

l = []
while 1:
    P = int(input("Enter P : "))
    if prime_checker(P) == -1:
        print("Number Is Not Prime, Please Enter Again!")
        continue
    break

while 1:
    G = int(input(f"Enter The Primitive Root Of {P} : "))
    if primitive_check(G, P, l) == -1:
        print(f"Number Is Not A Primitive Root Of {P}, Please Try Again!")
        continue
    break

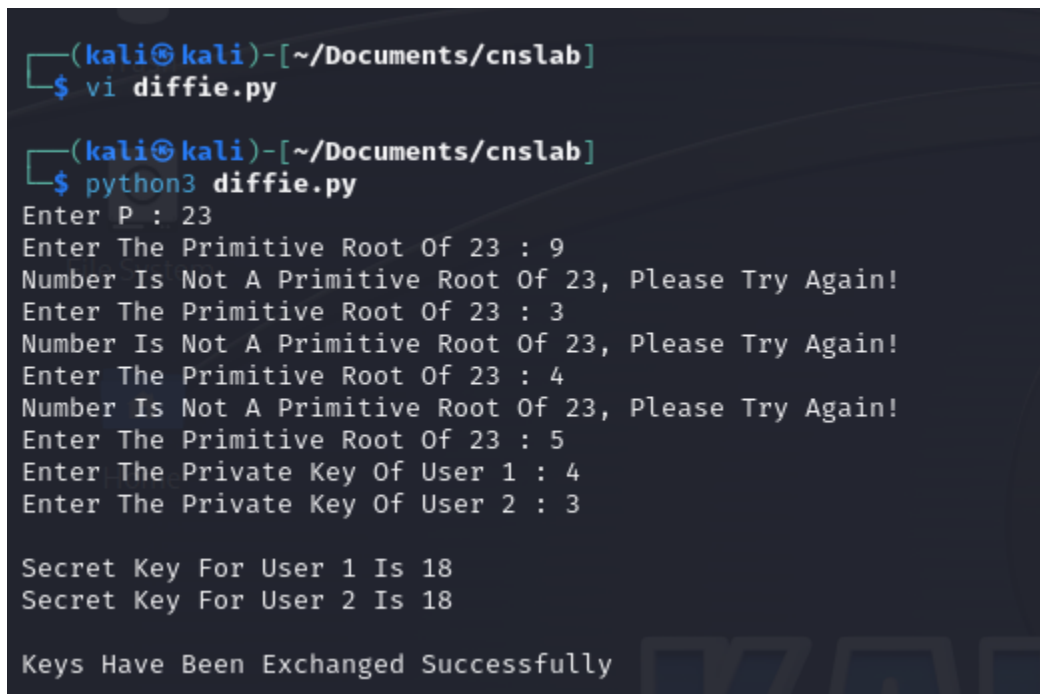
# Private Keys
x1, x2 = int(input("Enter The Private Key Of User 1 : ")), int(
    input("Enter The Private Key Of User 2 : "))
```

```

while 1:
    if x1 >= P or x2 >= P:
        print(f"Private Key Of Both The Users Should Be Less Than {P}!")
        continue
    break
# Calculate Public Keys
y1, y2 = pow(G, x1) % P, pow(G, x2) % P
# Generate Secret Keys
k1, k2 = pow(y2, x1) % P, pow(y1, x2) % P
print(f"\nSecret Key For User 1 Is {k1}\nSecret Key For User 2 Is {k2}\n")
if k1 == k2:
    print("Keys Have Been Exchanged Successfully")
else:
    print("Keys Have Not Been Exchanged Successfully")

```

OUTPUT:



```

(kali㉿kali)-[~/Documents/cnslab]
$ vi diffie.py

(kali㉿kali)-[~/Documents/cnslab]
$ python3 diffie.py
Enter P : 23
Enter The Primitive Root Of 23 : 9
Number Is Not A Primitive Root Of 23, Please Try Again!
Enter The Primitive Root Of 23 : 3
Number Is Not A Primitive Root Of 23, Please Try Again!
Enter The Primitive Root Of 23 : 4
Number Is Not A Primitive Root Of 23, Please Try Again!
Enter The Primitive Root Of 23 : 5
Enter The Private Key Of User 1 : 4
Enter The Private Key Of User 2 : 3

Secret Key For User 1 Is 18
Secret Key For User 2 Is 18

Keys Have Been Exchanged Successfully

```

RESULT:

Thus, a python program has been implemented to demonstrate Diffie Hellman Key Exchange Algorithm.