Practical 6

Aim: Write a program to implement the diffie hellman key agreement algorithm to generate symmetric keys

Code:

```
def primitive_root(n):
    primitive_no = None
    for num in range(2,n):
        freq_arr = [0 for i in range(n)]
        for num2 in range(1,n):
            if freq_arr[(num**num2)%n] == 0:
                freq_arr[(num**num2)%n] += 1
            else:
                break
        else:
            primitive_no = num
            return primitive_no


def is_prime(n):
    for i in range(2,n):
        if n % i == 0:
            return False
    else:
        return True


def key_exchange(q,x_a,x_b):
    if not is_prime(q):
        print("Enter a prime number for q")
        return
```

```
    alpha = primitive_root(q)
    print("Primitive root: ",alpha)
    y_a = alpha**x_a % q
    y_b = alpha**x_b % q

    print("key 1: ",y_b**x_a % q)
    print("key 2: ",y_a**x_b % q)
    print("The keys are equal")

q = int(input("Enter a prime number: "))
x_a = int(input("Enter value for X_a: "))
x_b = int(input("Enter value for X_b: "))
key_exchange(q,x_a,x_b)
```

Output:

```
Enter a prime number: 23
Enter value for X_a: 53
Enter value for X_b: 14
Primitive root:  5
key 1:  3
key 2:  3
The keys are equal
```