**WEEK -7**

**7**. Implement the Diffie-Hellman Key Exchange algorithm for a given problem using JAVA.

**AIM:** To apply Diffie-Hellman Key Exchange algorithm for secrete key generation

**OBJECTIVE:** To understand the secret key generation using Diffie-Hellman Key Exchange algorithm

**THEORY:**

Whitefield Diffie and Martin Hellman develop Diffie Hellman key exchange Algorithms in 1976 to overcome the problem of key agreement and exchange. It enables the two parties who want to communicate with each other to agree on a symmetric key, a key that can be used for encrypting and decryption; note that Diffie Hellman key exchange algorithm can be used for only key exchange, not for encryption and decryption process. The algorithm is based on mathematical principles.

**ALGORITHM:**

**STEP-1:** Sender and receiver publicly agree to use a modulus p and base g which is a primitive root modulo p.

**STEP-2:** Sender chooses a secret integer x then sends Bob **R1 = g x mod p**

**STEP-3:** Receiver chooses a secret integer y, then sends Alice **R2 = g y mod p**

**STEP-4:** Sender computes **k1 = B x mod p**

**STEP-5:** Receiver computes **k2 = A y mod p** **STEP-6:** Sender and Receiver now share a secret key

**PROGRAM:**

**import** java.io.\*;

**import**java.math.BigInteger;

**class** DiffieHellman

{ **public** **static** **void** main(String[]args)**throws** IOException

{

BufferedReader br=**new** BufferedReader(**new**

InputStreamReader(System.***in***));

System.***out***.println("Enter prime number:");

BigInteger p=**new** BigInteger(br.readLine());

System.***out***.print("Enter primitive root of "+p+":");

BigInteger g=**new** BigInteger(br.readLine());

System.***out***.println("Enter value for x less than

"+p+":");

BigInteger x=**new** BigInteger(br.readLine());

BigInteger R1=g.modPow(x,p);

System.***out***.println("R1="+R1);

System.***out***.print("Enter value for y less than

"+p+":");

BigInteger y=**new** BigInteger(br.readLine());

BigInteger R2=g.modPow(y,p);

System.***out***.println("R2="+R2);

BigInteger k1=R2.modPow(x,p);

System.***out***.println("Key calculated at Sender's

side:"+k1);

BigInteger k2=R1.modPow(y,p);

System.***out***.println("Key calculated at Receiver's

side:"+k2);

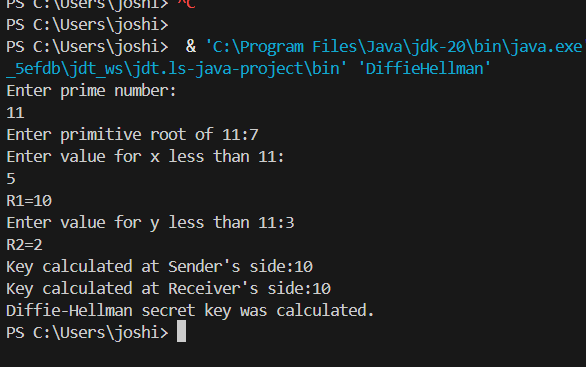
System.***out***.println("Diffie-Hellman secret key was

calculated.");

}

}

**Output :**

****