EX.NO: 2.b) ROLL.NO: 210701275

DATE:

Diffie Hellman Algorithm

AIM:

To Implement Diffie Hellman Algorithm to find the secret key

ALGORITHM:

1. Define large prime number *p* and a primitive root modulo *p*, denoted as *g*.
2. Party A selects a random private key *a*.Party B selects a random private key *b*.
3. Party A computes =*g^a* mod *p* Party B computes *B*=*g^b* mod *p*.
4. Parties A and B exchange their calculated public keys *A* and*B* with each other.
5. Party A computes *s*=*B^a* mod *p*. Party B computes *s*=*A^b* mod *p*.
6. Both parties now have the same shared secret *s*, which they can use as a symmetric encryption key for further communication.

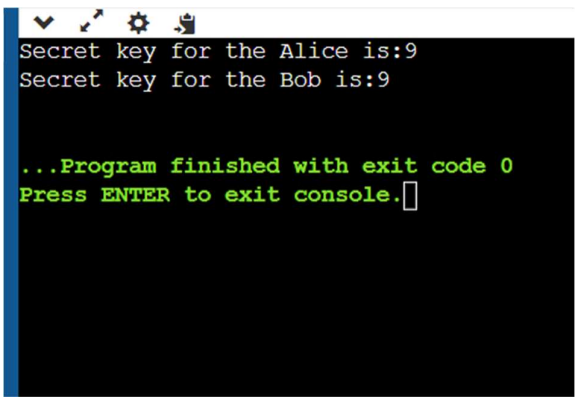
PROGRAM:

```
class Main {
    private static long power(long a, long b, long p)
    {
        if (b == 1)
            return a;
        else
            return (((long)Math.pow(a, b)) % p);
    }
    public static void main(String[] args)
    {
        long P, G, x, a, y, b, ka, kb;
        P = 23;
        G = 9;
        a = 4;
        x = power(G, a, P);
        b = 3;
        y = power(G, b, P);
```

```
        ka = power(y, a, P); // Secret key for Alice
        kb = power(x, b, P); // Secret key for Bob

        System.out.println("Secret key for the Alice is:"
                                    + ka);
        System.out.println("Secret key for the Bob is:"
                                    + kb);
    }
}
```

OUTPUT:



RESULT: