AIM: To implement Diffie Helman Key Exchange Algorithm

Diffie Helman key exchange is a way in by which we excahange the key securely over a public channel and was the first algorithm been implemented. The Algorithm is as follows

1. The two communicating parties Alice and Bob, agree on a two large numbers p and g.
2. Alice chooses some large random integer $x_A < p$ and keeps it secret. Likewise Bob chooses $x_B < p$ and keeps it secret. These are their "private keys".
3. Alice computes her "public key" $y_A \equiv g^{x}_{A} \pmod p$ and sends it to Bob using insecure communication. Bob computes his public key $y_B \equiv g^{x}_{B}$ and sends it to Alice. Here $0 < y_A < p$, $0 < y_B < p$.

   As already mentioned, sending these public keys with insecure communication is safe because it would be too hard for someone to compute x_A from y_A or $x_B$from y_B, just like the powers of 2 above.

4. Alice computes $z_A \equiv y_B^{x}{}_{A} \pmod p$ and Bob computes $z_B \equiv y_A^{x}{}_{B} \pmod p$. Here $z_A < p$, $z_B < p$.

Program

```
packagediffie.helmanalgo;
importjava.util.Scanner;
public class DiffieHelmanAlgo
{
public static double alice(double n,doubleg,double x)
   {
double a,a1;
      a1 = Math.pow(g,x);
      a=a1%n;
return(a);
   }
public static double bob(double n,doubleg,double y)
   {
double b,b1,k2,t2;
      b1= Math.pow(g,y);
      b=b1%n;
return(b);
   }
public static void main(String[] args)
   {
double g,x,y,a,b,k1,k2,n;
      Scanner input = new Scanner(System.in);
System.out.print("Enter value of n=>");
      n = input.nextDouble();
System.out.print("Enter value of g=>");
      g = input.nextDouble();
System.out.print("\nEnter value of x=>");
      x = input.nextDouble();
System.out.print("\nEnter value of y=>");
      y = input.nextDouble();
      a = alice(n,g,x);
System.out.println("alice end value:"+a);
      b=bob(n,g,y);
System.out.println("bob end value:"+b);
      k1=alice(n,b,x);
```

```
System.out.println("valueof k1 :"+k1);
        k2=alice(n,a,y);
System.out.println("valueof k2 :"+k2);
    }
}
```

Output