Ex. No.: 5 **Date:**

DIFFIE-HELLMAN KEY EXCHANGE

The simplest and the original implementation of the protocol uses the multiplicative group of integers modulo *p*, where *p* is prime, and *g* is a primitive root modulo *p*. Here is an example of the protocol, with non-secret values in blue, and secret values in **red**.

1. Alice and Bob agree to use a prime number $p = 23$ and base $g = 5$ (which is a primitive root modulo 23).
2. Alice chooses a secret integer $\boldsymbol{a} = \mathbf{6}$, then sends Bob $A = g^{a} \bmod p$
   - $A = 5^{6} \bmod 23 = 8$
3. Bob chooses a secret integer $\boldsymbol{b} = \mathbf{15}$, then sends Alice $B = g^{b} \bmod p$
   - $B = 5^{15} \bmod 23 = 19$
4. Alice computes $\boldsymbol{s} = B^{a} \bmod p$
   - $\boldsymbol{s} = 19^{6} \bmod 23 = \mathbf{2}$
5. Bob computes $\boldsymbol{s} = A^{b} \bmod p$
   - $\boldsymbol{s} = 8^{15} \bmod 23 = \mathbf{2}$ 6. Alice and Bob now share a secret (the number **2**).

Aim:

To implement Diffie-Hellman key exchange using C.

Algorithm:

1. Get a prime number q as input from the user.
2. Get a value xa and xb which is less than q.
3. Calculate primitive root α
4. For each user A, generate a key Xa < q
5. Compute public key, α pow(Xa) mod q
6. Each user computes Ya
7. Print the values of exchanged keys.

Program Code:

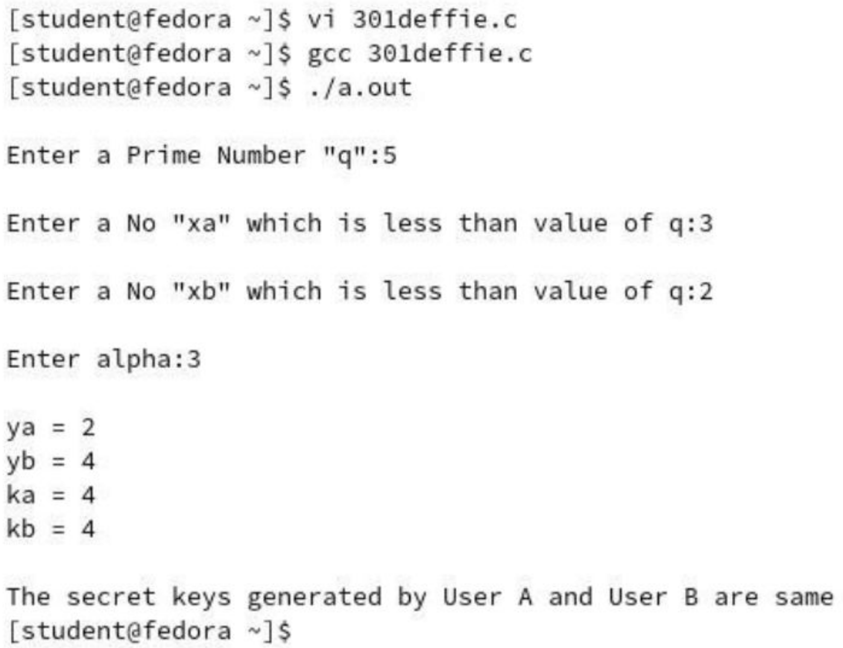
```
//This program uses fast exponentiation function power instead of pow library function
#include <stdio.h> #include <math.h>
int power( int,unsigned int,int); int
main()
{ int x,y,z,count,ai[20][20]; int
 alpha,xa,xb,ya,yb,ka,kb,q;
 printf("\nEnter a Prime Number \"q\":");
 scanf("%d",&q);
 printf("\nEnter a No \"xa\" which is less than value of q:");
 scanf("%d",&xa);
 printf("\nEnter a No \"xb\" which is less than value of
 q:"); scanf("%d",&xb); printf("\nEnter alpha:");
 scanf("%d",&alpha); ya
 = power(alpha,xa,q); yb
 = power(alpha,xb,q); ka
 = power(yb,xa,q); kb =
 power(ya,xb,q);
 printf("\nya = %d \nyb = %d \nka = %d \nkb = %d \n",ya,yb,ka,kb); if(ka ==
 kb) printf("\nThe secret keys generated by User A and User B are same\n");
 else printf("\nThe secret keys generated by User A and User B are not
     same\n");
 return 0;
}

int power(int x, unsigned int y, int p)
{ int res = 1;          // Initialize result x = x % p; //
   Update x if it is more than or equal to p

   while (y > 0)
   {
      // If y is odd, multiply x with result
      if (y & 1)
         res = (res*x) % p;

      // y must be even now y
      = y>>1;   // y = y/2 x
      = (x*x) % p;
   } return
   res;
}
```

Output:



```
[student@fedora ~]$ vi 301deffie.c
[student@fedora ~]$ gcc 301deffie.c
[student@fedora ~]$ ./a.out

Enter a Prime Number "q":5

Enter a No "xa" which is less than value of q:3

Enter a No "xb" which is less than value of q:2

Enter alpha:3

ya = 2
yb = 4
ka = 4
kb = 4

The secret keys generated by User A and User B are same
[student@fedora ~]$
```

Result: