EXP NO:6 **DATE: 02/03/24**

DSA

Aim: To implement Digital Signature Algorithm (DSA) using C.

Algorithm:

- Step 1: Include the necessary header files #include <stdio.h> and #include <math.h>.
- Step 2: Declare the required variables for the program, including integers for prime numbers, private keys, hash value, and computed values like g, r, and s.
- Step 3: Prompt the user to enter the prime number p and the prime divisor q of (−1) (p−1). Also, prompt the user to enter *h*h such that it's greater than 1 and less than (−1)(p−1).
- Step 4: Calculate g using the function power(h,t,p).
- Step 5: Prompt the user to enter their private key x and per-message secret key k. Also, prompt the user to enter the hash value M.
- Step 6: Compute r and s values for the signature using the provided formulas.
- Step 7: Print the computed values of g, y, r, and s.
- Step 8: Define the power function to calculate the power of a number modulo p.
- Step 9: Define the multiplicativeInverse function to find the multiplicative inverse of a number modulo n.

Program:

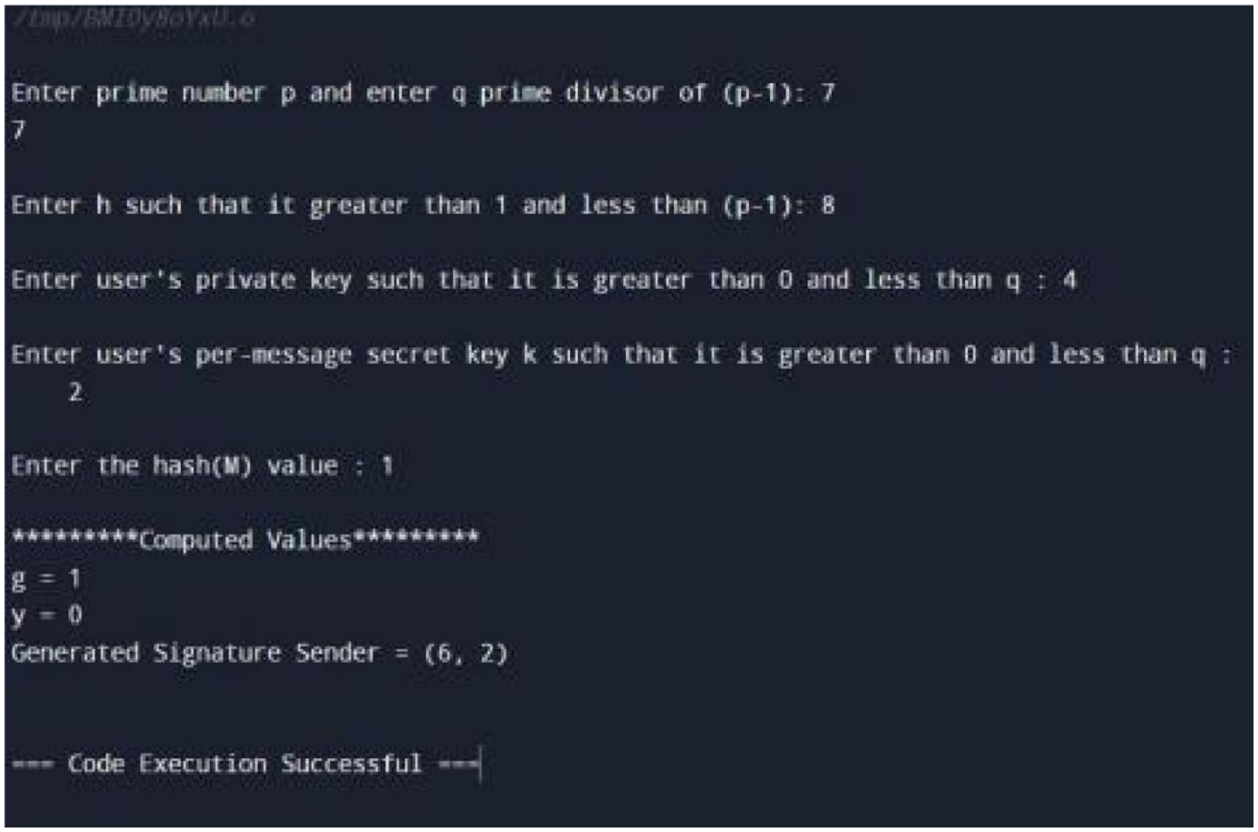
```
#include <stdio.h>
 #include <math.h>
int   power(int,unsigned int,int);
int
multiplicativeInverse(int,int,int);
int main() {
int p,q,h,g,r,s,t,x,y,z,k,inv,hash;
```

```
printf("\nEnter prime number p and enter q prime divisor of (p-1): "); scanf("%d
%d",&p,&q);   printf("\nEnter h such that it greater than 1 and
less than (p-1): ");
scanf("%d",&h); g = power(h,t,p);

printf("\nEnter user's private key such that it is greater than 0 and less than q : ");
scanf("%d",&x);
printf("\nEnter user's per-message secret key k such that it is greater than 0 and
less
than q : ");
scanf("%d",&k);
printf("\nEnter the hash(M) value : ");
scanf("%d",&hash);   r = z % q; inv =
multiplicativeInverse(k,q,p);
s = inv * (hash + x * r) % q;
printf("\n*********Computed Values*********");
printf("\ng = %d",g); printf("\ny = %d",y);
printf("\nGenerated Signature Sender = (%d, %d) \n",r,s);
}   int power(int x, unsigned int y,
int p)
{ int res =
1; x = x
% p;
{   res = (res * x)
% p;
} return   res; }   int
multiplicativeInverse(int a, int b, int n)
{
int sum,x,y; for(y=0;y<n;y++)
{
for(x=0;x<n;x++)
{   sum = a * x + b *
(-y);
if(sum == 1) return
x;
}
}
```

}

Output:



```
/tmp/BNIOy8oYxU.o

Enter prime number p and enter q prime divisor of (p-1): 7
7

Enter h such that it greater than 1 and less than (p-1): 8

Enter user's private key such that it is greater than 0 and less than q : 4

Enter user's per-message secret key k such that it is greater than 0 and less than q :
    2

Enter the hash(M) value : 1

*********Computed Values*********
g = 1
y = 0
Generated Signature Sender = (6, 2)


=== Code Execution Successful ===
```

Result:

Thus the Diffie-Hellman key exchange using C is implemented successfully.