EX NO:6 **REGISTER NO:210701509**

DATE:

DSA

AIM:

To implement Digital Signature Algorithm (DSA) using C.

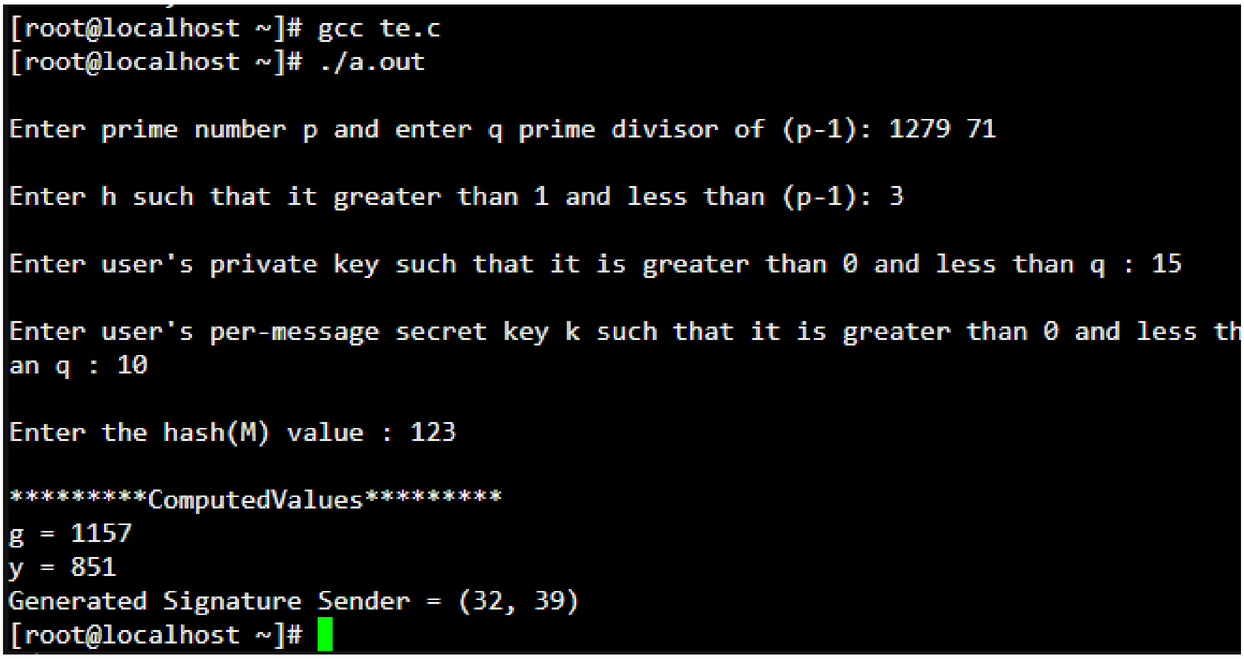
ALGORITHM:

- Get the prime number p and its divisor q from the user.
- Get the value of h from the user.
- Compute the value of g.
- Get the private key xa from the user.
- Compute the user's public key y.
- Get the per-message secret key k and hash value of message M.
- Compute the value of z using g, k & p
- Compute z % q to get the value of r
- Compute the multiplicative inverse.
- Compute the value of s.
- Print the signature (r, s).

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);
//Compute g
t = (p-1)/q;
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);
//Computer user's public key
y = power(g,x,p);
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);
//Signing. Compute r and s pair
z = power(g,k,p);
r = z % q;
inv = multiplicativeInverse(k,q,p);
s = inv * (hash + x * r) % q;
//Display
printf("\n*********ComputedValues*********");
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; // 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;
}
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:



```
[root@localhost ~]# gcc te.c
[root@localhost ~]# ./a.out

Enter prime number p and enter q prime divisor of (p-1): 1279 71

Enter h such that it greater than 1 and less than (p-1): 3

Enter user's private key such that it is greater than 0 and less than q : 15

Enter user's per-message secret key k such that it is greater than 0 and less th
an q : 10

Enter the hash(M) value : 123

*********ComputedValues*********
g = 1157
y = 851
Generated Signature Sender = (32, 39)
[root@localhost ~]# 
```

RESULT: