EX-NO-12-ELGAMAL-ALGORITHM

AIM:

To Implement ELGAMAL ALGORITHM

ALGORITHM:

1. ElGamal Algorithm is a public-key cryptosystem based on the Diffie-Hellman key exchange and relies on the difficulty of solving the discrete logarithm problem.
2. Initialization:
   - Select a large prime (p) and a primitive root (g) modulo (p) (these are public values).
   - The receiver chooses a private key (x) (a random integer), and computes the corresponding public key (y = g^x \mod p).
3. Key Generation:
   - The public key is ((p, g, y)), and the private key is (x).
4. Encryption:
   - The sender picks a random integer (k), computes (c_1 = g^k \mod p), and (c_2 = m \times y^k \mod p), where (m) is the message.
   - The ciphertext is the pair ((c_1, c_2)).
5. Decryption:
   - The receiver computes (s = c_1^x \mod p), and then calculates the plaintext message (m = c_2 \times s^{-1} \mod p), where (s^{-1}) is the modular inverse of (s).
6. Security: The security of the ElGamal algorithm relies on the difficulty of solving the discrete logarithm problem in a large prime field, making it secure for encryption.

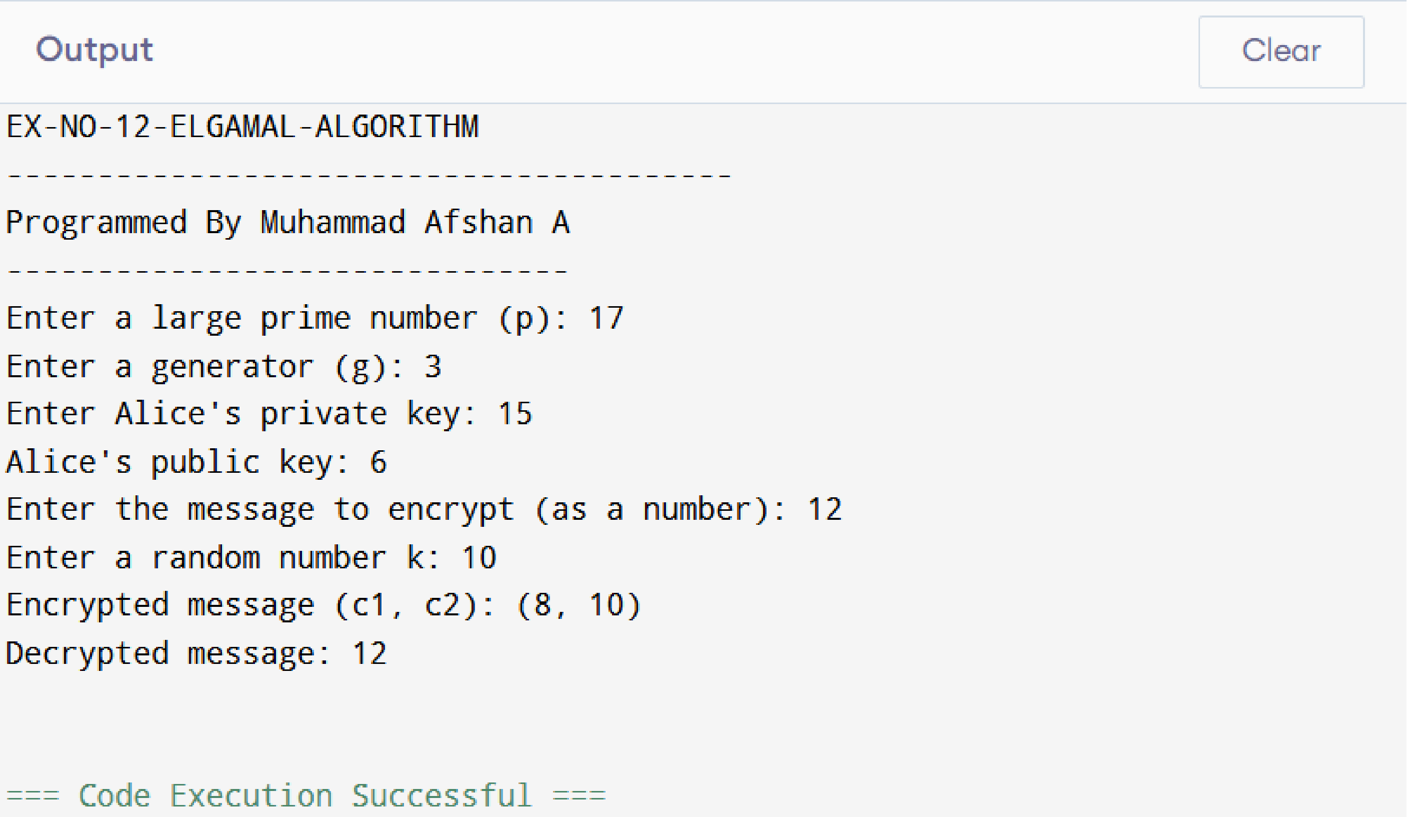
Program:

```
NAME: MUHAMMAD AFSHAN A
REG_NO: 212223100035
```

```


#include <stdio.h>
#include <math.h>

// Function to compute modular exponentiation (base^exp % mod)
```

```
long long int modExp(long long int base, long long int exp, long long int mod) {
  long long int result = 1;
  while (exp > 0) {
      if (exp % 2 == 1) {
          result = (result * base) % mod;
      }
      base = (base * base) % mod;
      exp = exp / 2;
  }
  return result;
}

int main() {
  printf("EX-NO-12-ELGAMAL-ALGORITHM\n");
  printf("----------------------------------------\n");
  printf("Programmed By Muhammad Afshan A\n");
  printf("------------------------------\n");
  long long int p, g, privateKeyA, publicKeyA;
  long long int k, message, c1, c2, decryptedMessage;

  // Step 1: Input a large prime number (p) and a generator (g)
  printf("Enter a large prime number (p): ");
  scanf("%lld", &p);
  printf("Enter a generator (g): ");
  scanf("%lld", &g);

  // Step 2: Alice inputs her private key
  printf("Enter Alice's private key: ");
  scanf("%lld", &privateKeyA);

  // Step 3: Compute Alice's public key (publicKey = g^privateKeyA mod p)
  publicKeyA = modExp(g, privateKeyA, p);
  printf("Alice's public key: %lld\n", publicKeyA);

  // Step 4: Bob inputs the message to be encrypted and selects a random k
  printf("Enter the message to encrypt (as a number): ");
  scanf("%lld", &message);
  printf("Enter a random number k: ");
  scanf("%lld", &k);

  // Step 5: Bob computes ciphertext (c1 = g^k mod p, c2 = (message * publicKeyA^k) mo
  c1 = modExp(g, k, p);
  c2 = (message * modExp(publicKeyA, k, p)) % p;
  printf("Encrypted message (c1, c2): (%lld, %lld)\n", c1, c2);

  // Step 6: Alice decrypts the message (decryptedMessage = (c2 * c1^(p-1-privateKeyA)
  decryptedMessage = (c2 * modExp(c1, p - 1 - privateKeyA, p)) % p;
  printf("Decrypted message: %lld\n", decryptedMessage);

  return 0;
}
```

Output:



```
EX-NO-12-ELGAMAL-ALGORITHM
----------------------------------------
Programmed By Muhammad Afshan A
-------------------------------
Enter a large prime number (p): 17
Enter a generator (g): 3
Enter Alice's private key: 15
Alice's public key: 6
Enter the message to encrypt (as a number): 12
Enter a random number k: 10
Encrypted message (c1, c2): (8, 10)
Decrypted message: 12


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

Result:

The program is executed successfully.