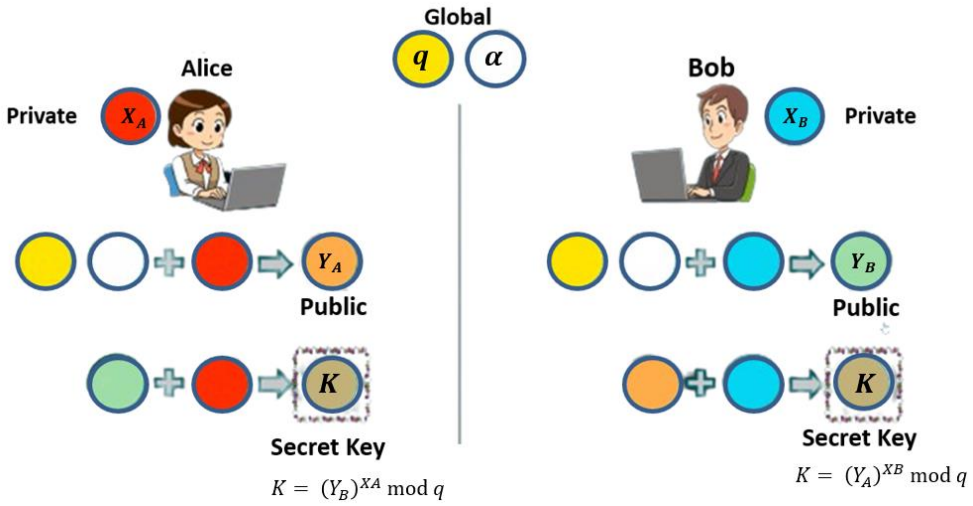
Description:

Implement the Diffie-Hellman Key Exchange algorithm for a given problem..
The Diffie-Hellman algorithm is being used to establish a shared secret that can be used for secret communications while exchanging data over a public network using the elliptic curve to generate points and get the secret key using the parameters.

For the sake of simplicity and practical implementation of the algorithm, we will consider only 4 variables, one prime **q** and **α** (a primitive root of q) and two private values **Xa** and **Xb.**

q and α are both publicly available numbers. Users (say Alice and Bob) pick private values **Xa** and **Xb** and they generate a key and exchange it publicly. The opposite person receives the key and that generates a secret key, after which they have the same secret key to encrypt.



Step-by-Step explanation is as follows:

1. Global Public Elements
   - q prime number
   - α $\alpha < q$ and α is primitive root of q
2. User A Key Generation
   - Select private X_A $X_A < q$
   - Calculate public Y_A $Y_A = \alpha^{XA} \bmod q$
3. User B Key Generation
   - Select private X_B $X_B < q$
   - Calculate public Y_B $Y_B = \alpha^{XB} \bmod q$
4. User A Key Generation
   - Select private X_A $X_A < q$
   - Calculate public Y_A $Y_A = \alpha^{XA} \bmod q$
5. User B Key Generation
   - Select private X_B $X_B < q$
   - Calculate public Y_B $Y_B = \alpha^{XB} \bmod q$
6. Calculation of Secret Key by User A
   $K = (Y_B)^{XA} \bmod q$
7. Calculation of Secret Key by User b
   $K = (Y_A)^{XB} \bmod q$

Solved Example:

- Alice and bob agrees on a prime number $q = 23$

- $\boldsymbol{\alpha = 5}$ as primitive root of q
- Alice selects a private integer $\boldsymbol{X_A = 6}$
- Alice computes $Y_A = \alpha^{XA} \bmod q \Rightarrow \boldsymbol{Y_A = 5^6 \ mod \ 23 = 8}$
- Bob selects a private integer $\boldsymbol{X_B = 15}$
- Bob computes $Y_B = \alpha^{XB} \bmod q \Rightarrow \boldsymbol{Y_B = 5^{15} \ mod \ 23 = 19}$
- Alice sends Y_A to Bob and Bob sends Y_B to Alice
- Alice computes key $K = (Y_B)^{XA} \bmod q \Rightarrow \boldsymbol{K = (19)^6 \ mod \ 23}$
- $\boldsymbol{K = 2}$
- Bob computes key $K = (Y_A)^{XB} \bmod q \Rightarrow \boldsymbol{K = (8)^{15} \ mod \ 23}$
- $\boldsymbol{K = 2}$

Instructions:

1. You need to read the value of **Xa**, **Xb** and $\boldsymbol{q}$ from the file and compute the value of **K.**
2. Create separate function to compute **primitive root($\boldsymbol{\alpha}$)** using the following formula.
   Example: p = 7 then primitive root is 3 because powers of 3 mod 7 generates all the integers from 1 to 6 **i.e. 1 to (p-1)**

$$\boldsymbol{3^1 = 3 \equiv 3 \ (mod \ 7)}$$
$$\boldsymbol{3^2 = 9 \equiv 2 \ (mod \ 7)}$$
$$\boldsymbol{3^3 = 27 \equiv 6 \ (mod \ 7)}$$
$$\boldsymbol{3^4 = 81 \equiv 4 \ (mod \ 7)}$$
$$\boldsymbol{3^5 = 243 \equiv 5 \ (mod \ 7)}$$
$$\boldsymbol{3^6 = 729 \equiv 1 \ (mod \ 7)}$$

➢ **METHODOLOGY FOLLOWED:**

```cpp
#include <bits/stdc++.h>
#include <fstream>
#include <iostream>
#include <math.h>

using namespace std;

long long int power(long long int x, long long int y, long long int mod)
{
    if (y == 0)
    {
        return 1 % mod;
    }
    if (y == 1)
    {
        return x % mod;
    }

    if (y % 2 == 0)
    {
        long long int a = power(x, y / 2, mod);
        return (a * a) % mod;
    }
```

```
    else
    {
        return (power(x, y / 2, mod) * power(x, y - y / 2, mod)) % mod;
    }
}

int main()
{

    ifstream fin;
    fin.open("input.txt");

    string qst;
    string xast;
    string xbst;

    getline(fin, qst);
    getline(fin, xast);
    getline(fin, xbst);

    long long int q = stoi(qst);
    long long int xa = stoi(xast);
    long long int xb = stoi(xbst);


    cout << q << " " << xa << " " << xb << "\n";
    long long int alpha = -1;

    for (int i = 2; i <= q - 1; i++)
    {

        long long int alp = i;
        set<int> s;

        for (int j = 1; j <= q - 1; j++)
        {

            s.insert(power(alp, j, q));
        }

        int c = 0;
        for (int j = 1; j <= q - 1; j++)
        {
            if (s.find(j) != s.end())
            {
                c++;
            }
        }
```

```
        if (c == q - 1)
        {
            alpha = alp;
            break;
        }
    }


    cout << "alpha: " << alpha << "\n";

    long long int ya = power(alpha, xa, q);
    long long int yb = power(alpha, xb, q);

    long long int key1 = power(yb, xa, q);
    long long int key2 = power(ya, xb, q);

    cout << "key1: " << key1 << "\n";
    cout << "key2: " << key2 << "\n";

    fin.close();

    return 0;
}
```

input.txt

```
practical7 > input.txt
1   179
2   6
3   15
4
```

output.txt

```
PS C:\Users\Lenovo\Desktop\INS\practical7> g++ main.cpp
PS C:\Users\Lenovo\Desktop\INS\practical7> ./a.exe
179 6 15
alpha: 2
key1: 177
key2: 177
```