Candidate Name: Dheeraj Nahariya

Reg. No: 11704064

Candidate Mail: nahariyadheeraj@gmail.com

Cypherock Assignment

- **Shamir's Secret Sharing Algorithm: -**

- **Problem Link:** https://www.codechef.com/IPTS2019/problems/CTS1

- **Solution**: -

  - **Explanation/Summary: -**

In this Problem We've Asked to Write a Program on **Shamir's Secret Sharing** Algorithm. So, we will Take Input of a Byte Array and Each Byte of the Array should form 4 Shares and That Respective Byte Can be Generated by Using any 2 Shares out of the 4.

1. **Generating 4 Shares for Each of the Byte (Run a For Loop for Traversing in Byte Array): -**

   i. Choose any Random Integer ***r1*** between 1 to 100.
   ii. Extract the ith Digit from the Byte Array. Store it in Variable ***h***.
   iii. Run a For Loop 4 Times
      a) Integer ***x*** = Any Random Integer
      b) Integer ***y*** = ***h*** + ***x*** * r1
      c) So, this Pair of **(x, y)** is our **Share**.
      d) Print the Share, Now move to the Second Part of Algorithm.

2. Extracting Each Byte Again from Any Two Shares (out of Four) and Building the Exact same Byte Array: -

i. We will Choose Any of the Two Shares (out of the 4 Pairs we have generated in First Part).
   a) Suppose You have Chosen 1st and 2nd Share, **(xi0, yi0)** and **(xi1, yi1)** Respectively.

ii. Now we will Run the For Loop to Traverse the Byte Array.
   a) Formula for Decrypting / Extracting the Original Byte
   Byte = ((yi1 * xi0) – (yi0 * xi1)) / (xi0 – xi1) <- **important**
   b) We will take a new Byte Array and append the ith Byte One by One.
   c) Print the Extracted Byte Array and You are Done.

- **C++ Implementation: -**

Link for the Source Code: https://ide.codingblocks.com/s/245588

I am Also Pasting the Source Code in this Document.

Dheeraj Cypherock.cpp

```
// NO External Libraries has been Used for implementation.
#include <iostream>
#include <vector>
#include <cstdlib>
using namespace std;
#define int long long

int Calculate_Y(int w , int s0 , int s1)
{
    return s0 + s1 * w;
```

```
}

int32_t main()
{
    int testcases;
    cin >> testcases; // Test Cases
    while(testcases--)
    {
        string input;
        cin>>input;

        vector<vector<pair<int,int>>>v(4,vector<pair<int,int>>(input.size()));

        for(int i=0;i<input.size();i++)
        {
            cout << "Choose Any Random Integer between 1 to 100 " << endl;
            int r1 = rand()%100;
            cout << r1 << endl;
            cout << endl;
            cout << "Extracting the " << i+1 << " Byte " << endl;
            int h = input[i];
            cout << h - '0' << endl;
            cout << endl;

            cout << "Generating 4 Shares for Each Byte " << endl;;
            for(int j=0;j<4;j++)
                {
                    int x=rand()%100;
                    int y = Calculate_Y(x,h,r1);
                    v[j][i] = {x,y};
                    cout << x << " " << y << endl;
                }
            cout << endl;
        }

        string output;
        cout << "Now we will Apply Lagranges Polynomial for Decrypting " << endl;
        cout << "Generating the Original Byte Array By 1st and 2nd Share " << endl;
        cout << endl;
        for(int i=0;i<input.size();i++)
            {
                cout << "Restoring " << i+1 << " digit by 1st and 2nd Share" << endl;
                int xi0 = v[1][i].first;
                int xi1 = v[2][i].first;
                int yi0 = v[1][i].second;
                int yi1 = v[2][i].second;
                cout << xi0 << " " << xi1 << " " << yi0 << " " << yi1 << endl;
                cout << endl;
                char apnd = char(((yi1*xi0)-(yi0*xi1))/(xi0-xi1));
                cout << "Restored " << i+1 << " Byte " << apnd << endl;
```

```
                output+=apnd;
                cout << endl;
            }
        cout << endl;
        cout << "Original Byte Array Restored Completely By SSS Algorithm " << endl;
        cout << output;
    }

}
```

- **Output (Dry Run on Test Cases): -**

INPUT:

FIRST LINE: NUM OF TEST CASES

SECOND LINE: BYTE ARRAY

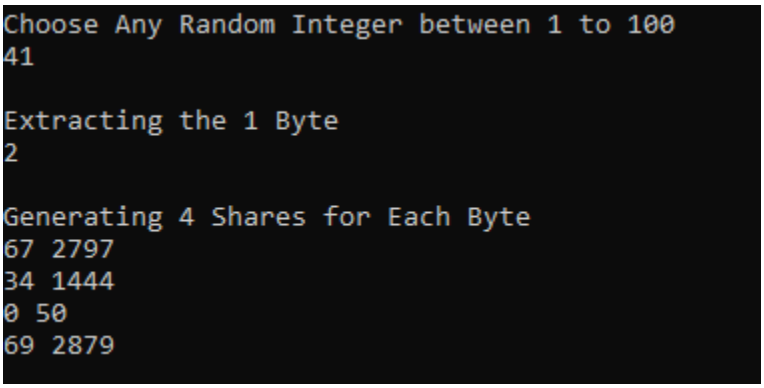
1

247

OUTPUT:

So, Our Input Byte Array Length is 3, So we have to generate 4 Shares for each of these 3 Byte Separately.

a) Extracting the First Byte i.e. “2” and generating 4 Shares for this.



b) Extracting the Second Byte i.e. “4” and generating 4 Shares for this.

```
Choose Any Random Integer between 1 to 100
24

Extracting the 2 Byte
4

Generating 4 Shares for Each Byte
78 1924
58 1444
62 1540
64 1588
```

c) Extracting the Third Byte i.e. “7” and generating 4 Shares for this.

```
Choose Any Random Integer between 1 to 100
5

Extracting the 3 Byte
7

Generating 4 Shares for Each Byte
45 280
81 460
27 190
61 360
```

d) Applying SSS Algorithm and Extracting / Decrypting Each Byte one by one:

```
Now we will Apply Lagranges Polynomial for Decrypting
Generating the Original Byte Array By 1st and 2nd Share

Restoring 1 digit by 1st and 2nd Share
34 0 1444 50
Restored 1 Byte 2

Restoring 2 digit by 1st and 2nd Share
58 62 1444 1540
Restored 2 Byte 4

Restoring 3 digit by 1st and 2nd Share
81 27 460 190
Restored 3 Byte 7
```

e) Now We can see that we have Decrypted / Restored Each Byte Correctly. So now we will append it in Output Byte Array and Print the Array.