CRACK A HACK

Winning Hand of Cards

(From RookieRank 4)

By :
Vishal M N
USN : 01FE15BCS233

Problem Statement

You are given a number of cards and try to create as many combinations from those cards as possible that result in a *winning hand*. A winning hand is the one where the product of the numbers on the cards modulo a given value, the *modulo divisor* is equal to another given value, the *target value*.

Solution

```
#include <iostream>
using namespace std;
typedef long long ll;

int main() {

    int n,m,x;
    cin>>n>>m>>x;
    ll a[n];
    for(int i=0;i<n;i++)   cin>>a[i];

    ll count[m]={0},count1[m];

    for(int i=0;i<n;i++){          //loop 1
        for(int j=0;j<m;j++)    count1[j] = count[j];        //loop 2

        for(int j=0;j<m;j++)    count[((ll)j*a[i])%m] += count1[j];
        //loop 3

        count[(a[i])%m]+=1;
    }
    cout<<count[x];
    return 0;
}
```

Explaination

Using Brute force approach to generate all the combinations requires time complexity of O(2^n).
Generally problems having exponential growth in terms of time complexity are solved using dynamic programming which has time complexity in terms of a polynomial equation.

The above problem can be solved using dynamic programming with a time complexity of O(m*n).

N is the size of array elements.
M is the modulo divisor.
X is the target value.

The array count[m] stores the count of combinations that have remainder equal to ‘j’ where 0<= j < m.

The array count1[m] stores count of combinations that remainder equal to ‘j’ where 0<= j < m of the previous state.

Loop1 is to select a value from the array.

Loop2 is to store the previous state of the count[m] array into count1[m] array.

Loop3 : In this array 0<=j<m and j represents remainder which used as index in this array.

Considering i’th value from the array, the result of (j*A[i])%m is obtained and count array is updated as count[(j*A[i])%m] +=count1[j].
The operation j*A[i] indicates obtaining combinations of all the values in the array A[n] that give j as remainder after modulo operation.

Addition of count1[j] indicates inclusion of all the previous combinations that resulted into remainder = j.

Finally, the result that should be outputted is count of all combinations that resulted in remainder x after modulo operation.

Hence the final output is value in count[x].