Problem setting is somewhat of a cruel task of throwing something at the contestants and having them scratch their head to derive a solution. In this problem, the setter is a bit kind and has decided to gift the contestants an algorithm which they should code and submit. The C/C++ equivalent code of the algorithm is given below:

```
long long GetDiffSum( int a[], int n )
{
    long long sum = 0;
    int i, j;
    for( i = 0; i < n; i++ )
        for( j = i + 1; j < n; j++ )
            sum += abs( a[i] - a[j] ); // abs means absolute value
    return sum;
}
```

The values of array $a[]$ are generated by the following recurrence relation:

$$a[i] = (K * a[i-1] + C)\%1000007 \text{ for } i > 0$$

where K, C and $a[0]$ are predefined values. In this problem, given the values of K, C, n and $a[0]$, you have find the result of the function

`long long GetDiffSum( int a[], int n )`

But the setter soon realizes that the straight forward implementation of the code is not efficient enough and may return the famous "TLE" and that's why he asks you to optimize the code.

Input

Input starts with an integer T (≤ 100), denoting the number of test cases.

Each case contains four integers K, C, n and $a[0]$. You can assume that ($1 \leq K, C,\ a[0] \leq 10^4$) and ($2 \leq n \leq 10^5$).

Output

For each case, print the case number and the value returned by the function as stated above.

Sample Input

```
2
1 1 2 1
10 10 10 5
```

Sample Output

```
Case 1: 1
Case 2: 7136758
```