1st LGE Code Jam 2019 - Problem D (English)

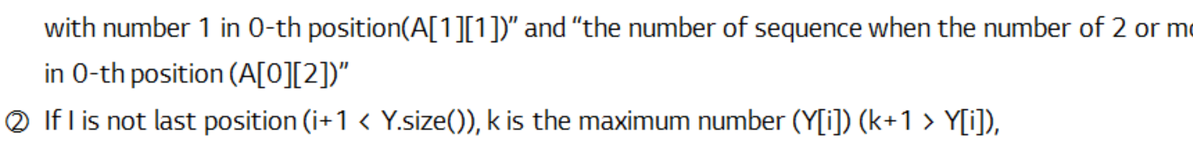
Created by 허지영 jiyoung.huh, last modified on 2019/05/14

To summarize (n: the number of children, x[i]: the maximum integer which i-th child knows),

this problem is to get the number of all possible cases when sorting selected integers by n children in non-decreasing order. (A: sequence of selected integers by n children)

For example, when Input is n = 4, X = {3, 4, 2, 3} and sorting selected integers by n children in non-decreasing order, the maximum number for each position is {2, 3, 3, 4}.

That is, this problem is to get the number of all non-decreasing sequences created with each element of array Y, which is sorted in non-decreasing order of X, as the maximum value .



Given Y, which is sorted in non-decreasing order of X, we can solve this problem with dynamic programming.

A[i][k]: the number of all possible sequences when the number of k or more is present in the i-th position

A[0][1] (the number of sequences when 0 or more is present in 0-th position) is the answer of this problem.

The following is recurrence formula for this problem.

$$A[i][k] = \begin{cases} A[i+1][k] + A[i][k+1], \text{ if } (k+1 <= Y[i]) \text{ and } (i+1 < Y.size()) & \cdots\cdots ① \\ A[i+1][k], \text{ if } (k+1 > Y[i]) \text{ and } (i+1 < Y.size()) & \cdots\cdots ② \\ Y[i] - k + 1, \text{ if } (i+1 >= Y.size()) & \cdots\cdots ③ \end{cases}$$

① If i is not last position (i+1 < Y.size()), k is not the maximum number (Y[i]) (k+1 <= Y[i]),
A[i][k] is sum "the number of sequence when the number of k or more is present in (i+1)-th position (A[i+1][k])" and "the number of sequence when the number of k+1 or more is present in i-th position (A[i][k+1])".
For example, A[0][1] is sum "the number of sequence when the number of 1 or more is present in 1-th position with number 1 in 0-th position(A[1][1])" and "the number of sequence when the number of 2 or more in present in 0-th position (A[0][2])"

② If I is not last position (i+1 < Y.size()), k is the maximum number (Y[i]) (k+1 > Y[i]),
A[i][k] is the number of sequence when the number of k or more is present in (i+1)-th position (A[i+1][k]).
For example, A[0][2] is the number of sequence when the number of 2 or more is present in 1-th position because 0-th position can have just number 2.

③ If I is last position (i+1 >= Y.size()), A[i][k] is k to Y[i].
For example, A[3][1] is xxx1, xxx2, xxx3, xxx4, which is all cases (Y[3] – 1 + 1)

The following is code for this recurrence formula

LGE Code Jam 2019 1st, Problem D(C++)

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

#define FLAG 1000000007

int N;
vector<int> X;
vector<vector<int>> A;

// Calculate the number of sequences when the number of 'start' or more is present in the 'pos'-th position
int sumCount(int pos, int start)
{
    if (start > X[pos]) return 0;

    int &ret = A[pos][start];
    if (ret > 0) return ret;

    if (pos + 1 == N) ret = X[pos] - start + 1;
    else {
        if (start + 1 <= X[pos]) ret = sumCount(pos, start + 1);
        ret += sumCount(pos + 1, start);
        ret = ret % FLAG;
    }
    return ret;
}

int main()
{
    int T;
    scanf("%d", &T);

    while (T > 0) {
       scanf("%d", &N);

       X = vector<int>(N);
       A = vector<vector<int>>(N, vector<int>(201, 0));

       for (int i = 0; i < N; i++) scanf("%d", &X[i]);
       sort(X.begin(), X.end());

        int result = (int)((long long)((long long)sumCount(0, 1) * (long long)N) % FLAG);
        printf("%d\n", result);

        T -= 1;
    }
    return 0;
}
```

Like Be the first to like this

No labels

Write a comment…