Selection of Numbers - Hackerearth

Monday, March 9, 2020 11:31 PM

The Problem:

You are given *N* numbers placed in a line. You have to select *K* of those numbers. These numbers are priority levels and they are different.

You can select numbers from the ends of the array only. After selection, the number gets erased from the line. You want to maximize the sum of priority levels. Your task is to find the maximum sum of the priority values.

Input format
First line contains *K* and *N*.
Second line contains *N* space separated integers denoting the priority of numbers in the list.

Output format
Maximum sum of priority of *K* numbers

Constraints
1≤K≤N≤105
0≤Ai≤109

SAMPLE INPUT

```
2 4
1 2 3 4
```

SAMPLE OUTPUT

```
7
```

Explanation
Sum of priorities will be maximized if he selects 4th first , followed by the 3rd.

The Code:

```
/* It's simple: Take i from right end, K-i from left end. And vice versa. Store whichever is maximum.
*leftSum and rightSum will be helper arrays.
*leftSum[i] will give sum of all elements from 0 to i.
*rightSum[i] will give sum of all elements from i to N-1.
*/

#include<stdio.h>

int main() {
   int N, K, otherEnd, i;
   long long int maxSum, tempSum, lftTmpSum, rghtTmpSum;
   scanf("%d %d", &K, &N);
```

```
    long int* leftSum = (long int*)malloc((N+2) * sizeof(long int));
    long int* rightSum = (long int*)malloc((N+2) * sizeof(long int));
    leftSum[0] = leftSum[N+1] = 0;
    rightSum[0] = rightSum[N+1] = 0;

    for(i=1; i<=N; i++) {
        scanf("%ld", &rightSum[i]);
        leftSum[i] = rightSum[i] + leftSum[i-1];
    }

    for(i=N; i>=1; i--) {
        rightSum[i] += rightSum[i+1];
    }

    if(N != K) {
        maxSum = 0;

        // take i from left end and K-i from right. And vice versa. Store whichever is greater.
        for(i=1; i<=K; i++) {
            otherEnd = K-i;
            lftTmpSum = leftSum[i] + rightSum[N-(otherEnd-1)];
            rghtTmpSum = rightSum[N-(i-1)] + leftSum[otherEnd];

            tempSum = lftTmpSum > rghtTmpSum ? lftTmpSum : rghtTmpSum;

            if(tempSum > maxSum) {
                maxSum = tempSum;
            }
        }
    } else{
        maxSum = leftSum[N];
    }

    printf("%lld", maxSum);
}
```

The Stats:

Score
30.0

Time (sec)
0.61067

Memory (KiB)
64

Language
C