

Subarray Sum

Limits: 4s, 512 MB

You are given an array having **N** integers. You have to select at most **K** positions in the array and replace the values at those positions with **0** such that the sum of the subarray which has the maximum sum among all subarrays is maximized. A subarray is a non-empty sequence of consecutive elements of the array.

Input

The first line of the input contains an integer **T** denoting the number of test cases. Then the description of the **T** test cases follows. First line of each of the test cases start with two integers **N** and **K**. It is followed by a line having **N** integers denoting the array.

Constraints:

For all tasks:

$1 \leq T \leq 10$

| value of a number in the array | ≤ 100000

Easy Sub-Task:

$0 \leq K \leq 50$

$1 \leq N \leq 50$

Medium Sub-Task:

$0 \leq K \leq 1000$

$1 \leq N \leq 1000$

Hard Sub-Task:

$0 \leq K \leq 5000$

$1 \leq N \leq 5000$

Output

For every test case output one integer in a separate line in the format “**Case x: y**” where **x** is the number of the test case starting from **1** and **y** is the sum of the subarray which has the maximum sum among all subarrays that you can get after replacing at most **K** numbers with **0**.

Samples

| Input | Output |
|---------------------|------------|
| 2 | Case 1: 15 |
| 5 3 | Case 2: 3 |
| 1 2 3 4 5 | |
| 7 2 | |
| 1 -4 1 -10 -11 2 -6 | |

In the first test case we don't need to replace any number with 0.

In the second case we can replace -10 and -11 to get the following array: 1 -4 1 0 0 2 -6. The subarray [1 0 0 2] has the maximum sum which is equal to 3.

```

1 // https://toph.co/p/subarray-sum
2 #include<bits/stdc++.h>
3 using namespace std;
4 int n,k,a[5001], dp[5001][5001];
5 int fun(int pos,int cnt){
6     if(pos>n)return 0;
7     if(dp[pos][cnt]!=-1)return dp[pos][cnt];
8     int ret = -1000000000;
9     if(a[pos]<0 && cnt<k) ret = max(ret, fun(pos+1,cnt+1));
10    ret = max(ret, a[pos]+fun(pos+1,cnt));
11    return dp[pos][cnt] = max(ret,a[pos]);
12 }
13 int main(){
14     int tt; scanf("%d",&tt);
15     for(int ks=1; ks<=tt; ks++){
16         scanf("%d%d",&n,&k);
17         for(int i=1; i<=n; i++) scanf("%d",&a[i]);
18         memset(dp,-1,sizeof(dp));
19         int ans = -1000000000;
20         for(int i=1; i<=n; i++){
21             ans = max(ans, a[i]);
22             ans = max(ans, fun(i,0));
23         }
24         printf("Case %d: %d\n",ks,ans);
25     }
26     return 0;
27 }
28 -----
29 #include<bits/stdc++.h>
30 using namespace std;
31 int a[5001], dp[5001][5001];
32 int fun(int pos,int cnt){
33     if(pos==0)return 0;
34     int &ret = dp[pos][cnt];
35     if(ret!=-2000000000)return ret;
36     ret = -1000000000;
37     if(a[pos]>=0) {
38         ret = max(ret, a[pos]+fun(pos-1,cnt));
39     }else{
40         if(cnt>0) ret = max(ret, fun(pos-1,cnt-1));
41         ret = max(ret, a[pos]+fun(pos-1,cnt));
42     }
43     return ret = max(ret, a[pos]);
44 }
45 int main(){
46     int tt; scanf("%d",&tt);
47     for(int ks=1; ks<=tt; ks++){
48         int n,k; scanf("%d%d",&n,&k);
49         for(int i=1; i<=n; i++) scanf("%d",&a[i]);
50
51         for(int i=0; i<5001; i++){
52             for(int j=0; j<5001; j++){
53                 dp[i][j] = -2000000000;
54             }
55         }
56
57         int ans = -1000000000;
58         for(int i=1; i<=n; i++){
59             ans = max(ans, fun(i,k));
60         }
61         printf("Case %d: %d\n",ks,ans);
62     }
63     return 0;
64 }

```