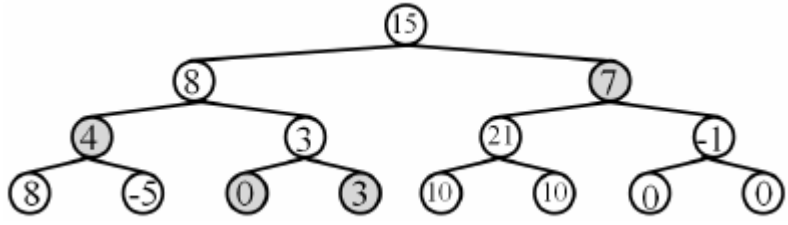
A *cut* of a binary tree is a set of tree nodes such as for each possible path from the root node to any leaf node, just one node in the cut belongs to the path. In a weighted binary tree, one can compute a *weighted* cut, because it would include weighted nodes. The total weight of such cut can be computed by adding up the weights of the nodes in it. The figure below shows a weighted binary tree. The gray nodes represent a weighted cut, and its total weight is 14.



Now, given a weighted binary tree, you have to write a program that finds the weighted cut with the maximal total weight value among all possible weighted cuts in the tree. This is hereafter called the *Optimal Cut* of the input binary tree. However, to make it a bit more interesting, your program must find the optimal cut that includes no more than K nodes, and report its weight. In the figure above, for instance, the nodes of the optimal cut sums 28 when $K = 3$, and 15 when $K = 2$.

Input

The input can contain several problems. Each problem contains the description of a complete binary tree in a single line. This description corresponds to sequence of integer numbers, separated of each other by a blank space. The description starts with an integer $0 \leq H < 20$ representing the height of the tree, followed by another integer $1 \leq K \leq 20$ representing the maximum number of nodes in the optimal cut to be found. Then, the weights $-10^3 \leq Wi \leq 10^3$ of the nodes follow, listed in pre-order. The end of input is indicated by a single line containing only an integer value of '`-1`'.

Output

For each problem in the input, the output should be a single line with a single integer number, representing the total weight of the optimal cut found.

Sample Input

```
2 3 8 6 7 -2 -1 2 1
0 1 1
3 3 -8 1 0 0 1 2 1 1 -1 1 1 1 3 1 4
-1
```

Sample Output

```
9
1
5
```