Muffin Selection (muffin)

You may not know that William is an expert about muffins: he is even able to immediately judge the taste T_i of a muffin without even biting it! He is now in front of a long row of N muffins, and he really wants to eat some of them.

Figure 1: Some muffins aligned in a row, ready to be tasted.

The only way to pick up a muffin is to use a wide paddle that can lift exactly K muffins at a time, precisely the number that William wants.

Help him choosing the best range of K adjacent muffins with the highest sum of tastes, to maximize his satisfaction!

> ☞ Among the attachments of this task you may find a template file `muffin.*` with a sample incomplete implementation.

Input

The first line contains two integers N and K, respectively. The second line contains N integers T_i.

Output

You need to write a single line with an integer: the maximum possible sum of tastes.

Constraints

- $1 \leq N \leq 1\,000\,000$.
- $1 \leq K \leq N$.
- $-1000 \leq T_i \leq 1000$ for each $i = 0 \ldots N-1$.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points) Examples.

– **Subtask 2** (10 points) $T_i > 0$ for each $i = 0 \ldots N-1$.

– **Subtask 3** (10 points) $K = 1$.

– **Subtask 4** (20 points) $K = 2$.

– **Subtask 5** (30 points) $N \leq 1000$.

– **Subtask 6** (30 points) No additional limitations.

Examples

input.txt	output.txt
7 3 10 -3 -1 6 4 1 -10	11
5 2 1 -2 4 -8 16	8

Explanation

In the **first sample case** the best choice is to take the muffins having taste `6 4 1` with a total taste of $6 + 4 + 1 = 11$.

In the **second sample case**, as you are forced to take exactly two muffins, the best choice is to take the last two ones, with a total taste of $-8 + 16 = 8$.