DMPG '16 S5 - Pizza Bag

At a programming competition, there is one very large pizza which is being served for free. You brought a bag to the competition, and intend to take home the best tasting pieces that you can.

The pizza appears to be divided into N slices, where some slices have good toppings, and some have bad toppings. You quickly evaluate each section based on its deliciousness, and assign them all some value between $-100\,000\,000$ and $100\,000\,000$. It should be noted that the slices are connected in a circle — that is, slice 1 is connected to slice 2, slice 2 to 3, and so on. Slice N is additionally connected to slice 1.

Since there is only one pizza available, the organizers have limited everyone to at most K slices, and only if all K slices are connected (that is, only the two pieces at the ends are not connected to two other slices you have taken).

If you are the first person to pick, what is the most delicious sector of pizza you can take? A sector (connected series of slices) has deliciousness equal to the sum of its individual slices. There will always be at least one slice which has a positive deliciousness rating.

Scoring

For all cases, $2 \leq N \leq 100\,000$, $1 \leq K \leq N$.

For 15% of the points, $N \leq 1\,500$.

For a separate 15%, all deliciousness values given will be strictly positive.

For a separate 20%, slice N will have a deliciousness value of $-100\,000\,000$ while all other slices will have a value in the range $[-1000, 1000]$. You may assume that the optimal solution never involves taking slice N for this subtask.

For a separate 20%, $K = N$.

For the remaining 30%, no additional constraints apply.

Input Specification

On the first line, there will be two integers N and K.

On the second line, there will be N space-separated integers, where the i-th integer represents the deliciousness of piece number i.

Output Specification

Output a single integer, the maximum deliciousness which can be taken. You may assume that the solution is always positive.

It is possible for this value to exceed the limits of a 32-bit integer. It is advised to use 64-bit integers instead, meaning that Java users should use long, and C++ users should use long long.

Sample Input 1

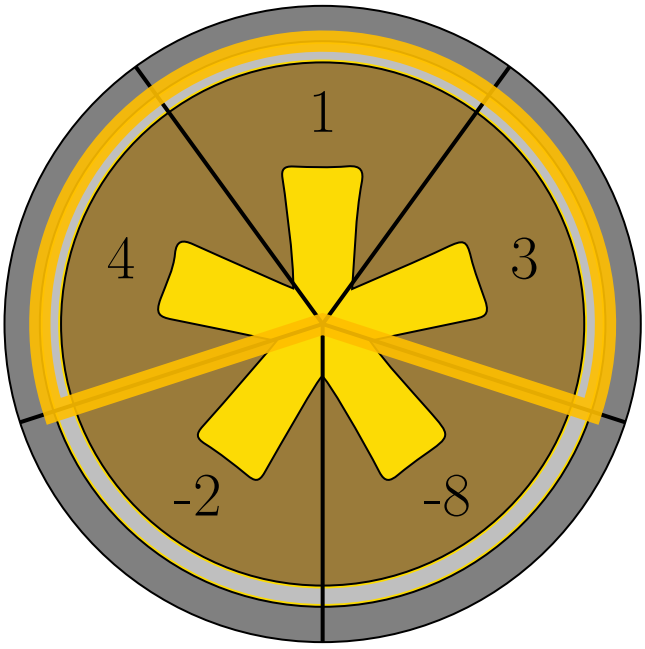
```
5 5
1 3 -8 -2 4
```

Sample Output 1

```
8
```

Illustration of Sample 1

A diagram of the pizza is given below.



Sample Input 2

```
6 2
0 1 8 1 5 5
```

Sample Output 2

```
10
```

Illustration for Sample 2