Introduction to Time Complexity

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

This problem will not be graded for Homework 1

In algorithmic studies, we often focus on understanding how an algorithm works and analyzing its theoretical time complexity. However, in real-world applications, what really matters is the actual size of the data and the time constraints within which the algorithm must execute.

A widely accepted rule of thumb for estimating execution time, particularly on personal computers and online coding platforms, is that approximately 10^8 operations take around 1 second.

So, if an algorithm runs in $O(n)$ time, and $n \leq 10^8$, you can generally expect it to complete in under 1 second. However, this estimate can vary depending on the programming language and the specific hardware running the code.

Given an array of length n, your task is to find and output the maximum sum of any contiguous subarray within it. Note that the empty subarray is considered and has a sum of 0. Once you've implemented your solution, evaluate its time complexity to ensure it meets the real-world time constraints mentioned above.

Hint: For this problem, your grade will depend on the efficiency of your algorithm. If you implement an $O(n^3)$ algorithm, you will receive 60%. If you manage to optimize it to $O(n^2)$, you'll receive 80%. Achieving a solution with $O(n)$ time complexity will earn you 100%. However, this problem is ungraded, so feel free to experiment and try out different approaches without worrying about the final grade.

Input

The first line contains a positive integer n

The second line contains n integers $a_1, \cdots, a_n$. For all i, $-100 \leq a_i \leq 100$.

Output

Output a single integer, indicating the maximum sum. Note that empty subarrays are valid and they have a sum of 0.

Scoring

There are some subtasks in this problem, you will get the percentage of score if you pass the subtask

Subtask	Condition	Score	Additional Limitations
1	$n \leq 10$	20%	None
2	$n \leq 100$	20%	Must pass Subtask 1
3	$n \leq 1000$	20%	Must pass Subtask 1, 2
4	$n \leq 10^4$	20%	Must pass Subtask 1, 2, 3
5	$n \leq 10^6$	20%	Must pass Subtask 1, 2, 3, 4

Example

standard input	standard output
8 -1 3 -2 5 3 -5 2 2	9

Note

The subarray [3, -2, 5, 3] gives the maximum sum, which is 9.