Student Number:		Finish Time:	
Signature:			

University of Cape Town ~ Department of Computer Science
Computer Science 3003S Theory of Algorithms ~ 2014

Procedure

You may consult the electronic Java and C/C++ API documentation (docs.cs.uct.ac.za), and submit to the automatic marker via Vula (vula.uct.ac.za), but nothing else! You may NOT use your class notes, textbooks, internet or files on your flash disk, hard drive, etc.

You may use only the computers in the lab. No tablets, headphones or other personal devices are permitted.

Please sign and return this sheet when you have finished.

Submission

The automatic marker contains a submission entry for each question. Entry and question bear the same name.

Submit your source file within a single compressed, '.ZIP', archive.

Make sure you create a '.ZIP' archive, not a gzipped, '.gz', or tar-gzipped, '.tgz', or other kind of file.

Make sure your source file is the only item within the archive. Especially, avoid submitting an archive containing a folder containing the file.

When submitting a Java source file copied from an editor like Eclipse or Netbeans, please remove any package line that may appear at the beginning of the code.

Practical Test 3 Session 1 Question 1

100 Marks

Each test case that is answered correctly will earn 10 marks.

File names

- Use cycling.c if you are writing your program in C.
- Use cycling.cpp if you are writing your program in C++.
- Use Cycling.java if you are writing your program in Java.

Note that case matters.

Problem Description

The scenario:

- N signs have been set up along a straight road, each displaying a number (which can be positive or negative).
- A cyclist must choose a position on the road at which to start cycling and a position at which to stop.
- The cyclist starts with 0 points. As they cycle from the chosen start position to the end position, they must add the number on any sign that they pass (whether positive or negative) to their score.
- The challenge is to choose a start and end position such that the point score is maximised.

Question continues on next page

Write a program that, given the numbers from a series of N road signs (in the order in which the signs appear), calculates the maximum point score that can be achieved.
Example:

Assume a road with N = 6 consecutive signs with the following values:

Sign	*S_1*	*S_2*	*S_3*	*S_4*	*S_5*	*S_6*
Value	1	-2	4	-1	5	-3

Choosing to start cycling before the first sign and to stop after the fifth sign, would give a score of 1 + (-2) + 4 + (-1) + 5 = 7 points.

This is not, however, the maximum achievable score. Starting before the third sign and stopping after the fifth would render 4 + (-1) + 5 = 8 points. Thus the correct answer for this case would be 8.

Note that it is possible to start AND stop before the first sign, (or any other sign), giving a score of 0. While not true for this example, there may be situations in which that is the best choice.

Input and Output

Input consists of a series of integer values, each on a separate line. The first value is N, the number of signs on the road, followed by the point values P_1, …, P_N, for those signs. The values of the signs are given in the order they appear on the road.

Constraints:

$1 \leq N \leq 2{,}000$

$-1{,}000{,}000 \leq P_i \leq 1{,}000{,}000$ (for $1 \leq i \leq N$)

Output consists of a single integer, K (the maximum point score that can be achieved), followed by a line break --- in Java, for example, use `System.out.println`, not `System.out.print`. The automatic marker expects output in this precise form.

The maximum achievable point score will fit within a 32 bit signed integer type.

Sample Input:

```
6
1
-2
4
-1
5
-3
```

Sample output:

```
8
```

END

Question continues on next page