

ECE 36800

PA 1: Window Manager

Due Monday, July 1st at 11:59 PM

Goal.

For this project, you will build a simple “window manager” application that maintains an ordered list of open windows and supports the following operations:

- Open a new window
- Close an existing window
- Switch focus to an open window

and implements the following semantics:

- Newly created windows should always be in focus
- The window in focus should always be at the front of the list
- The relative ordering of windows within the list should otherwise be preserved

Input & Output.

Your program should read a sequence of operations from standard input. Each operation will consist of a command, either **open**, **switch**, or **close**, followed by a integer representing the window ID. After each operation, your program should print the name of the currently focused window to standard output if there is one, or terminate if the last window is closed.

Example.

For example, given the following input:

```
open 0
open 1
open 2
switch 1
close 2
close 1
close 0
```

your program should output:

```
0
1
2
1
1
0
```

The table below walks through this example to show the list of windows after each operation, and the expected output. Note that there is no output after “close 0”, since that operation closes the last window.

Input	State	Output
open 0	[0]	0
open 1	[1 0]	1
open 2	[2 1 0]	2
switch 1	[1 2 0]	1
close 2	[1 0]	1
close 1	[0]	0
close 0	[]	-

Grading

This programming assignment will be graded based on both the correctness and memory safety of your implementation. The breakdown is as follows:

- Each test case will be worth the same amount of points, and will be all-or-nothing (i.e. you either have the correct output or you don’t).
- A 50% penalty will be applied for any test cases on which your submission has a memory error or a memory leak.
- You will receive a 0 if your submission fails to terminate within a 10-minute time limit.

You may submit to the Gradescope autograder as often as you’d like before the deadline. Only your active submission (by default the most recent) will be counted. While the score given to you by the autograder is likely a good indicator of your final grade on the assignment, we reserve the right to add additional test cases after the submission deadline.

Submission Instructions.

Submit any source/header files with your implementation, as well as a Makefile that builds a target called `pa1`, to Gradescope. Note that to receive points, your submission must work on `eceprog`. (See the syllabus for more details.)

If you choose to do the optional writeup, submit it to the assignment on Brightspace. The writeup can be formatted however you want, but here’s a guideline of some questions you may want to think about when writing it:

- What data structures did you choose to use, and why?
- What other factors did you consider? How did they influence your final implementation?
- What, if anything, did you find most challenging about this project? Was there anything in particular you enjoyed/disliked?
- If you took a particularly creative approach to solving this problem, or ended up implementing something we didn’t ask for, tell me about it!