LAB 2: doctor2

A doctor heard that you are good at programming and learned something about data structures and algorithms. She contacts you and asks if you can write a small program to manage the waiting room. At any time a new patient might come in or the next patient will be treated, with the maximum priority patient first. Each patient has a name (up to 200 characters), an age (a positive integer) and a priority (an integer).

Input-Output Example 1

Altogether your program should behave exactly as follows, where the red parts of the following transcript are entered by the user.

```
Currently there are 0 patients in the queue.
What do you want to do? [N]ew patient, [Q]uit:
N
What is the name of the patient?
Alice
What is the age of the patient?
17
What is the priority of the patient?
3
Currently there are 1 patients in the queue.
What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:
N
What is the name of the patient?
Bob
What is the age of the patient?
23
What is the priority of the patient?
1
Currently there are 2 patients in the queue.
What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:
N
What is the name of the patient?
Carol
What is the age of the patient?
42
What is the priority of the patient?
2
Currently there are 3 patients in the queue.
What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:
T
Treating the patient Alice (age: 17, priority: 3).
Currently there are 2 patients in the queue.
What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:
T
Treating the patient Carol (age: 42, priority: 2).
Currently there are 1 patients in the queue.
What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:
T
Treating the patient Bob (age: 23, priority: 1).
Currently there are 0 patients in the queue.
What do you want to do? [N]ew patient, [Q]uit:
Q
Goodbye!
```

Rules and Hints

- If you have not done it already, do the `doctor` exercise from session 4b first.
- To start with the lab assignment, please download `doctor2.zip` from Canvas and extract it as usual. Write your own solution into `doctor2.c`.
- Your program should be *efficient*. In particular, the program should not iterate over all patients in the waiting room more often than necessary.
- You may use any data structures we discussed in the course so far or will discuss before the deadline: arrays, stacks, (priority) queues, linked lists, trees, search trees, heaps. You may also use any of the sorting algorithms we discussed, but this is not recommended because it will not be efficient.
- No library is included by default. You may copy and use any of the type definitions and function in the notes, slides or other exercises and provided solutions.
- Note that most of the data structures as we defined them only store integers. But for your program to work well you will need to store the name, age and priority of each patient together.

  Concretely, you might want to do any of the following: define your own `struct`s, use multiple arrays, change the definitions of `TreeNode` or `ListNode`, etc.

Input-Output Example 2

Below you can find a second example, with the input and output in separate columns. Note that the input and output lines are *not* aligned with each other. For example, the first two lines of the output already get printed before the first input line is read.

input	corresponding **output**

`N`
`Ada Lovelace`
`206`
`100`
`N`
`Margaret Hamilton`
`86`
`5`
`N`
`Donald E. Knuth`
`84`
`70`
`N`
`Agnes Meyer Driscoll`
`133`
`13`
`T`
`T`
`Q`
	`Currently there are 0 patients in the queue.`
	`What do you want to do? [N]ew patient, [Q]uit:`
	`What is the name of the patient?`
	`What is the age of the patient?`
	`What is the priority of the patient?`
	`Currently there are 1 patients in the queue.`
	`What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:`
	`What is the name of the patient?`
	`What is the age of the patient?`
	`What is the priority of the patient?`
	`Currently there are 2 patients in the queue.`
	`What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:`
	`What is the name of the patient?`
	`What is the age of the patient?`
	`What is the priority of the patient?`
	`Currently there are 3 patients in the queue.`
	`What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:`
	`What is the name of the patient?`
	`What is the age of the patient?`
	`What is the priority of the patient?`
	`Currently there are 4 patients in the queue.`
	`What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:`
	`Treating the patient Ada Lovelace (age: 206, priority: 100).`
	`Currently there are 3 patients in the queue.`
	`What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:`
	`Treating the patient Donald E. Knuth (age: 84, priority: 70).`
	`Currently there are 2 patients in the queue.`
	`What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:`
	`Goodbye!`

Input-Output Example 3

Did you manage to pass the first two test cases? The doctor is happy with your work. But she also asks you to add another feature to your program. When given the "`L`" command, your program should provide a list of all patients in the waiting room. The list should be sorted by priority, with the maximum priority first.

This feature will be tested with test case 5 on Canvas.

```
Currently there are 0 patients in the queue.
What do you want to do? [N]ew patient, [Q]uit:
N
What is the name of the patient?
Alice
What is the age of the patient?
17
What is the priority of the patient?
3
Currently there are 1 patients in the queue.
What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:
N
What is the name of the patient?
Bob
What is the age of the patient?
23
What is the priority of the patient?
1
Currently there are 2 patients in the queue.
What do you want to do? [N]ew patient, [T]reat patient, [L]ist, [Q]uit:
L
Alice (age: 17, priority: 3)
Bob (age: 23, priority: 1)
Currently there are 2 patients in the queue.
What do you want to do? [N]ew patient, [Q]uit:
Q
Goodbye!
```

Testing

You can test your solution by running `make test`. This will check if your program works for the first two input-output examples above.

For further testing, submit your solution to the online "Exercise Testing" tool on Canvas. This will use the same two test cases, and three additional test cases. For the three additional tests the input and output are hidden, so you will only see whether your program passed or failed the test. When grading your program I will use the same tests as the online tool and half of your grade (5 out of 10 points) will be given for passing the tests.

Report

Besides your program you should also submit a short report ($\leq$ 2 pages). You can find a LaTeX and an ODT template for this on Canvas between which you can choose. In any case, submit the report as a PDF file. Do not modify the template in any way. In particular, keep the same font, font size and margins.

Submitting

To submit your work, please upload `doctor2.c` and `report.pdf`.