listDeleteLargest

Your task is to write a function, listDeleteLargest, that deletes the largest value from a given list and returns the value that was deleted. If the largest value occurs multiple times in the list, delete only the first instance. You should **not** change the values in any nodes or create any new nodes. Your program must not have any memory leaks. You can assume that the given list is not empty.

Download

Click here to download a zip of the files.

The Files

list.c	Contains the implementation of basic list functions
list.h	Contains the definition of the list data structure and function prototypes
testListDeleteLargest.c	Contains the main function, which reads in a list from standard input, calls listDeleteLargest, and prints out the original and resulting list, and the deleted value.
listDeleteLargest.c	Contains listDeleteLargest, the function you must implement
Makefile	A makefile to compile your code
tests/	A directory containing the inputs and expected outputs for some basic tests
autotest	A script that uses the tests in the tests directory to autotest your solution. You should only run this after you have tested your solution manually.

Examples

Your program should behave like these examples:

```
$ ./testListDeleteLargest
Enter list: 2 8 4 9 5

Original list: [2] -> [8] -> [4] -> [9] -> [5] -> X
After deleting largest: [2] -> [8] -> [4] -> [5] -> X
The largest value was: 9
```

```
$ ./testListDeleteLargest
Enter list: 1 7 2 7 3

Original list: [1] -> [7] -> [2] -> [7] -> [3] -> X
After deleting largest: [1] -> [2] -> [7] -> [3] -> X
The largest value was: 7
```

```
$ ./testListDeleteLargest
Enter list: 1

Original list: [1] -> X
After deleting largest: X
The largest value was: 1
```

Testing

You can test your program manually by compiling your code using `make`, and then running `./testListDeleteLargest`, as shown above. After you are satisfied with your solution, you can autotest it by running `./autotest`. This will run some basic tests on your program, as well as check for memory leaks/errors.