reverseList

Your task is to write a function, reverseList, that reverses a given singly linked list. You should **not** change the values in any nodes or create any new nodes - instead, you should rearrange the nodes of the given list.

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
testReverseList.c	Contains the main function, which reads in a list from standard input, calls reverseList, and prints out the original and resulting list.
reverseList.c	Contains reverseList, 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:

```
$ ./testReverseList
Enter list: 2 3 5 7 11
Original list: [2] -> [3] -> [5] -> [7] -> [11] -> X
Reversed list: [11] -> [7] -> [5] -> [3] -> [2] -> X
```

```
$ ./testReverseList
Enter list:
Original list: X
Reversed list: X
```

```
$ ./testReverseList
Enter list: 1 2 7 9 2 1
Original list: [1] -> [2] -> [7] -> [9] -> [2] -> [1] -> X
Reversed list: [1] -> [2] -> [9] -> [7] -> [2] -> [1] -> X
```

Testing

You can test your program manually by compiling your code using `make`, and then running `./testReverseList`, 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.