Practice Compete Jobs Rank Leaderboard bor0s

Dashboard > Functional Programming > Introduction > Reverse a List

Badge Progress ★ ★

Points: 200.00 Rank: 2986

Reverse a List

by idlecool

Problem Submissions Leaderboard Discussions

You are given a list of N elements. Reverse the list without using the *reverse* function. The input and output portions will be handled automatically. You need to write a function with the recommended method signature.

Input Format

Each element, X, of the list is displayed on a separate line.

Output Format

The output is the reverse of the input list.

Sample Input

```
19
22
3
28
26
17
18
4
28
0
```

Sample Output

```
0
28
4
18
17
26
28
3
22
19
```

Method Signature

```
Number Of Parameters: 1
Parameters: [list]
Returns: List or Vector
```

Constraints

$1 <= N <= 100$
$0 <= X <= 100$.

For Hackers Using Clojure

This will be the outline of your function body (fill in the blank portion marked by underscores):

```
(fn[lst]___________________________)
```

For Hackers Using Scala

This will be the outline of your function body (fill in the blank portion marked by underscores):

```
def f(arr:List[Int]):List[Int] = __________________
```

For Hackers Using Haskell

This will be the outline of your function body (fill in the blank portion marked by underscores):

```
    rev l = ________________
```

For Hackers Using other Languages

You have to read input from standard input and write output to standard output. Please follow the input/output format mentioned above.

NOTE: You only need to submit the code above after filling in the blanks appropriately. The input and output section will be handled by us. The focus is on writing the correct function.

Submissions: 9889
Max Score: 10
Difficulty: Easy

Rate This Challenge:
☆☆☆☆☆

More

If you cannot find your favorite language here, please check out the algorithm domain. We support over 40 popular languages.

Current Buffer (saved locally, editable)

Racket

```
#lang racket
; Enter your code here. Read input from STDIN. Print output to STDOUT
(define (print-list list)
  (if (= (length list) 1) (displayln (car list)) (begin (displayln (car list)) (print-list (cdr list)))))

; we don't use this because we read and print them in different order
(define (reverse-iter list acc)
  (if (eq? list '()) acc (reverse-iter (cdr list) (cons (car list) acc))))

(define (main list)
  (let ((x (read)))
    (if (eof-object? x) (print-list list) (main (cons x list)))))

(main '())
```

Line: 1 Col: 1

Upload Code as File □ **Test against custom input** Run Code **Submit Code**

Join us on IRC at #hackerrank on freenode for hugs or bugs.

Contest Calendar | Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature