

# CS303 Data Structures Assignment #3

attachments and source available at <https://github.com/alexskc/cs303>

Aleksander Charatonik

September 30, 2018

## 1

A `const_iterator` is useful in preventing modifying the referenced value. It's simply about const correctness, and informs what the programmer should be able to do. `iterator`, by contrast, has read-write access, and is useful in scenarios where that is necessary.

## 2

### a

An `iterator`. Regardless of whether you have an array-based structure, or a linked-list structure, you need to be able to change either the value of the next item, or the pointer to the next item.

### b

`iterator` as well. You are modifying data, so you cannot be read-only.

### c

For this one, a `const_iterator` will suffice. We are not changing any data.

### d

`iterator` as well. We can avoid changing the element pointed to if we're using a linked-list structure, but we still need to change the element before it to point to the new item. And of course, in an array-based structure, we're going to be moving elements around in the array to make space for the new element.

## 3

See attached `reverser.cpp`

# 4

Expression	Action	Stack
10 2 * 5 / 6 2 5 * + - ↑	Push 10	10
10 2 * 5 / 6 2 5 * + - ↑	Push 2	2 10
10 2 * 5 / 6 2 5 * + - ↑	Eval *	20
10 2 * 5 / 6 2 5 * + - ↑	Push 5	5 20
10 2 * 5 / 6 2 5 * + - ↑	Eval /	4
10 2 * 5 / 6 2 5 * + - ↑	Push 6	6 4
10 2 * 5 / 6 2 5 * + - ↑	Push 2	2 6 4
10 2 * 5 / 6 2 5 * + - ↑	Push 5	5 2 6 4
10 2 * 5 / 6 2 5 * + - ↑	Eval *	10 6 4
10 2 * 5 / 6 2 5 * + - ↑	Eval +	16 4
10 2 * 5 / 6 2 5 * + - ↑	Eval -	-12