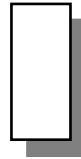


The Standard C++ Library Modifying containers

Version 1: Dr. Ofir Pele

Version 2: Dr. Erel Segal-Halevi

Iterators & Sequence Containers

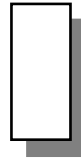


```
SeqContainerName<...> c;
```

```
SeqContainerName<...>::iterator i, j;
```

- `c.insert(i, x)` – inserts `x` before `i`
- `c.insert(i, first, last)`
 - inserts elements in `[first, last)` before `i`
- `c.erase(i)` – erases the element that `i` points to
- `c.erase(i, j)`
 - erase elements in range `[i, j)`

Iterators & Sequence Containers **c++11**



```
SeqContainerName<...> c;
```

```
SeqContainerName<...>::iterator i, j;
```

- **c.emplace(**i**, p1, ..., pn):**

Constructs and inserts before **i** an object with a constructor that gets **p1, ..., pn** parameters

Iterator validity

- When working with iterators, we have to remember that their validity can change

What is wrong with this code?

```
Container<...> c;
```

```
...
```

```
for(auto i= c.begin(); i!=c.end(); ++i )
```

```
    if( f( *i ) ) { // some test
```

```
        c.erase(i);
```

```
    }
```

Iterator validity

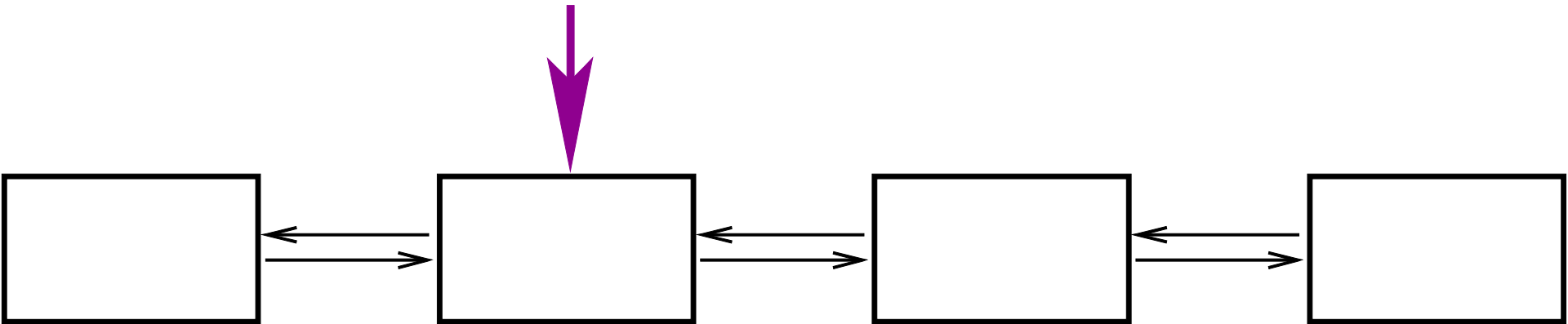
Two cases:

- list, set, map
 - **i** is not a legal iterator

Iterator validity

Two cases:

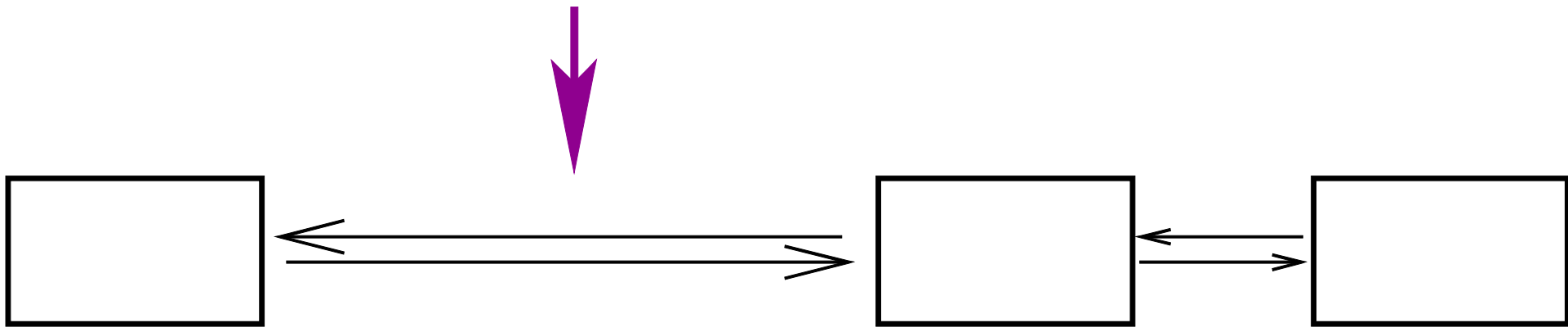
- **list**, set, map
 - **i** is not a legal iterator



Iterator validity

Two cases:

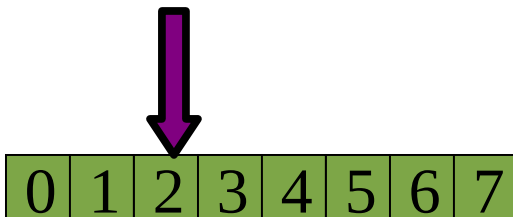
- **list**, set, map
 - **i** is not a legal iterator



Iterator validity

Two cases:

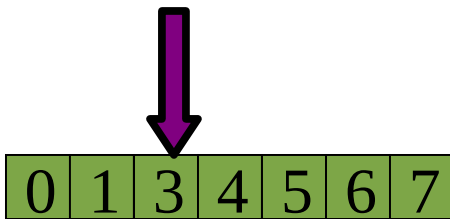
- list, set, map
 - **i** is not a legal iterator
- **vector**
 - **i** points to the element after



Iterator validity

Two cases:

- list, set, map
 - **i** is not a legal iterator
- **vector**
 - **i** points to the element after



Iterator validity

Two cases:

- list, set, map
 - `i` is not a legal iterator
- **vector**
 - `i` points to the element after

**In either case,
this is not what we want...**

Erasing during iteration (folder 3)

```
Container<...> c;
```

```
...
```

```
for(auto i= c.begin(); i!=c.end(); /*no ++i*/ )  
    if( f( *i ) ) { // some test  
        i = c.erase(i);  
    } else {  
        ++i;  
    }
```