

Reading: GT 2.2.0-2.2.3, p. 65–74 [up to 50 minutes]

Vectors and linked list are extremely important. The notion of a sequence abstract data type is secondary.

Ingredients: Vectors, Linked lists

Vectors

A *rank* of an element e in a linear sequence S is the number of elements before e in S . For an n element sequence, rank ranges from 0 to $n - 1$.

A *vector* is a linear sequence supporting access to elements by rank. Operations:

ELEM-AT-RANK(r) — return the element with rank r

REPLACE-AT-RANK(r, e) — replace the element at rank r with e

INSERT-AT-RANK(r, e) — insert e to have rank r (ranks of subsequent elements are shifted)

REMOVE-AT-RANK(r) — remove at rank r (ranks of the subsequent elements are shifted)

An obvious implementation: use an array.

Linked Lists

Linked list is a general data structure implemented using dynamic memory management (i.e. not inside an array, but using many objects and references between them). List elements are accessed by iteration, not directly like with arrays.

A singly linked list can easily be used to implement a stack or a queue. Uses more memory per element than the array based implementations, but does not limit the number of elements, beyond the memory limitations.

- Insertion after a given node: $O(1)$

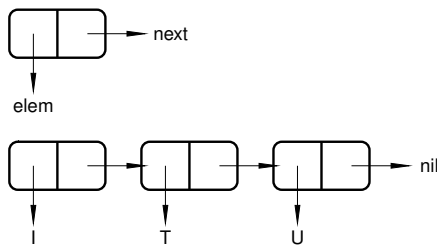


Figure 1: A singly linked list — You cannot move backwards. Always start from head, when you need to search again.

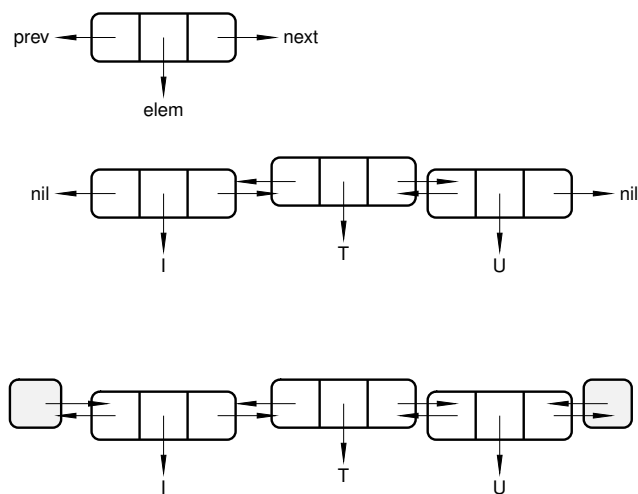


Figure 2: A doubly linked list — You can move back and forth. Top: a single node, Middle: a list, Bottom: a list with sentinels

- Search for a node: $O(n)$
- Deletion of a given node: $O(1)$
- Counting the number of elements $O(1)$

The main difference between arrays and lists:

- Arrays provide efficient random access to arbitrary cells, but are expensive to grow.
- Lists can only be accessed by iteration, but are cheap to grow.

INSERT-AFTER(n :node, o :data)

```
1  $m \leftarrow$  new node object
2  $m.elem \leftarrow o$ 
3  $m.next \leftarrow n.next$ 
4  $m.prev \leftarrow n$ 
5  $n.next \leftarrow m$ 
6  $m.next.prev \leftarrow m$ 
```

Figure 3: Inserting o after the node n into a list with sentinels. Give the head sentinel as parameter to insert in front.

Lists in Java

See the description of the *LinkedList* class in Java:

<http://java.sun.com/javase/6/docs/api/java/util/LinkedList.html>