Sequences and Iterators
Sequence ADT (§ 5.3)

- The **Sequence ADT** is the union of the Vector and List ADTs
- Elements accessed by
  - Rank, or
  - Position
- **Generic methods:**
  - size(), isEmpty()
- **Vector-based methods:**
  - elemAtRank(r), replaceAtRank(r, o), insertAtRank(r, o), removeAtRank(r)
- **List-based methods:**
  - first(), last(), prev(p), next(p), replace(p, o), insertBefore(p, o), insertAfter(p, o), insertFirst(o), insertLast(o), remove(p)
- **Bridge methods:**
  - atRank(r), rankOf(p)
Applications of Sequences

The Sequence ADT is a basic, general-purpose, data structure for storing an ordered collection of elements.

Direct applications:
- Generic replacement for stack, queue, vector, or list
- Small database (e.g., address book)

Indirect applications:
- Building block of more complex data structures
A doubly linked list provides a reasonable implementation of the Sequence ADT.

Nodes implement Position and store:
- element
- link to the previous node
- link to the next node

Special trailer and header nodes

Position-based methods run in constant time.
Rank-based methods require searching from header or trailer while keeping track of ranks; hence, run in linear time.
Array-based Implementation

- We use a circular array storing positions.
- A position object stores:
  - Element
  - Rank
- Indices $f$ and $l$ keep track of first and last positions.
## Sequence Implementations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Array</th>
<th>List</th>
</tr>
</thead>
<tbody>
<tr>
<td>size, isEmpty</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>atRank, rankOf, elemAtRank</td>
<td>1</td>
<td>$n$</td>
</tr>
<tr>
<td>first, last, prev, next</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>replace</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>replaceAtRank</td>
<td>1</td>
<td>$n$</td>
</tr>
<tr>
<td>insertAtRank, removeAtRank</td>
<td>$n$</td>
<td>$n$</td>
</tr>
<tr>
<td>insertFirst, insertLast</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>insertAfter, insertBefore</td>
<td>$n$</td>
<td>1</td>
</tr>
<tr>
<td>remove</td>
<td>$n$</td>
<td>1</td>
</tr>
</tbody>
</table>
Iterators (§ 5.4)

- An iterator abstracts the process of scanning through a collection of elements.
- Methods of the ObjectIterator ADT:
  - object object()
  - boolean hasNext()
  - object nextObject()
  - reset()
- Extends the concept of Position by adding a traversal capability.
- Implementation with an array or singly linked list.

An iterator is typically associated with an another data structure.

- We can augment the Stack, Queue, Vector, List and Sequence ADTs with method:
  - ObjectIterator elements()

Two notions of iterator:
- snapshot: freezes the contents of the data structure at a given time.
- dynamic: follows changes to the data structure.