Collections
[Chapter 4]

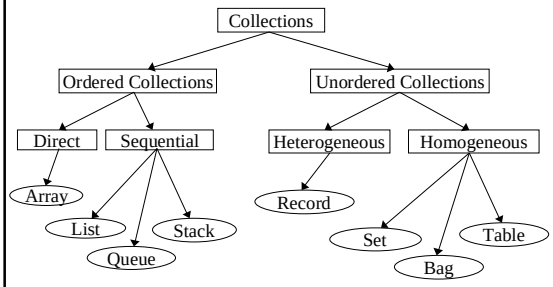
Collection ADTs

- Many standard ADTs describe collections
  - Data structures that manage groups of data
  - Operations don't depend of type of data being stored
  - Each collection effective for certain patterns of usage
  - Size of group is often not known in advance
- Extremely important in computer science
  - Used in just about every program
  - Analyses to measure performance of collections
  - Powerful tool for problem solving

Classification of Collections

- We can classify collections along several dimensions:
  - Order of data: Linear (ordered) vs. Nonlinear (unordered)
  - Method of access: Direct vs. Sequential
  - Type of data: Homogenous vs. Heterogeneous
  - Size: Fixed vs. Variable
- **Note:** these classifications apply to the abstract view of the collection, *not* the implementation

Classification of ADTs



Collections in C++

- Direct support for records (`struct`) and arrays
- Other ADTs must be implemented
  - No absolute standards
  - Many conventional operations and concepts
- STL - Standard Template Library
  - Official C++ standard as of this year
  - Contains many ADTs and operations on them
  - Radically changes approach to C++ programming
  - Not covered in this course

Our Plan

- Look at built-in types in terms of collection ADTs
- The List ADT
  - Build a specification
  - Provide a simple implementation
- Visit some of the other ADTs
  - Learn the conventional operations
  - Consider possible implementations

Arrays as ADTs [Section 4.3]

- Classification
  - Linear sequence of homogeneous elements
  - Fixed length
  - Direct access
- Operations on arrays
  - Direct indexing using [] operator (`operator []`)
  - Built in arrays provide no bounds checking
  - No searching, sorting, insertion, etc.

Records as ADTs [Section 4.4]

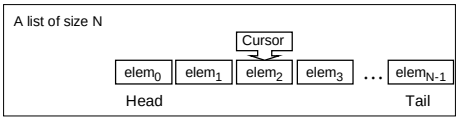
- Classification
  - Nonlinear collection of heterogeneous elements (fields)
  - Fixed number of elements
  - Direct access
- Operations on records
  - Accessing elements directly through . operator
- In C++, we'll use classes to implement records, providing encapsulation via access functions.
  - Access functions provide "read-only" variables

The List ADT [Section 4.5]

- Attributes of list type:
  - Linear sequence of homogeneous elements
  - **Varying** length
  - **Sequential** access
- Operations on lists
  - Navigation via a cursor
  - Look at element at cursor position
  - Test if full or empty
  - Insert, delete elements

List Terminology

- Head: first element in list
- Tail: last element in list
- Cursor: "current" element of list
- Size: number of elements in list



Lists in C++

- No predefined list type, so write our own
- Use the class construct
  - Member functions for list operations
  - Private data for list representation
- We'll write a list of integers
  - But only for convenience

Summary (I)

- Collections are fundamental in computer science
- Several dimensions of classification
  - Ordering, access mechanism, homogeneity, size limits
- Arrays:
  - Ordered, direct access by index, homogeneous, fixed size
- Records:
  - Unordered, direct access by name, heterogeneous, fixed size

Summary (II)

- Lists
  - Ordered, sequential access, homogeneous, variable size
  - List terminology
    - head, tail, cursor
  - List operations
  - Study the sample code!