CMSC 132: OBJECT-ORIENTED PROGRAMMING II

Linear Data Structures - Restricted Abstractions

Department of Computer Science
University of Maryland, College Park
Restricted Abstractions

- Restricting the operations an abstraction supports can be a good thing
  - Efficiently supporting only a few operations efficiently is easier
  - If limited abstraction is sufficient, easier to reason about limited abstraction than a more general one
- Restricted list abstractions
  - Stack (aka LIFO queue)
  - Queue (aka FIFO queue)
  - Dequeue (aka double ended queue)
Stack

- Properties
  - Elements removed in opposite order of insertion
  - Last-in, First-out (LIFO)
- A restricted list where:
  - Access only to elements at one end
  - Can add / remove elements only at one end
- Stack operations
  - Push $\rightarrow$ add element (to top)
  - Pop $\rightarrow$ remove element (from top)

(a) A three-element stack
(b) After a pop() operation
(c) After a push(W) operation
Stack Implementations

- Linked list
  - Add / remove from head of list

- Array
  - Increment / decrement Top pointer after push / pop
Queue

- Properties
  - Elements removed in order of insertion
  - First-in, First-out (FIFO)
- A restricted list where:
  - Access only to elements at beginning / end of list
    - Add elements only to end of list
    - Remove elements only from front of list
  - Alternatively, can add to front & remove from end
- Queue operations
  - Enqueue = add element (to back)
  - Dequeue = remove element (from front)
- Example:

  | X | Y | Z |
  | ^ | ^ |   |
  | front | back |

  | Y | Z |
  | ^ | ^ |
  | front | back |

  (a) Three-element queue  (b) After deletion of X  (c) After insertion of W
Queue Implementations

• Linked list
  • Add to tail (back) of list
  • Remove from head (front) of list

• Circular array
Queue – Circular Array Implementation

• Inherent problem for queue of size $N$
  • Only $N$ possible (Front – Back) pointer locations
  • $N+1$ possible queue configurations
    • Queue with 0, 1, … $N$ elements

• Solutions
  • Maintain additional state information
    • Use state to recognize empty / full queue
  • Examples
    • Record Size
    • Record QueueEmpty flag
  • Leave empty element in queue
  • Store marker in queue