Lecture 12: Linked Lists

CS 62
Fall 2017
Kim Bruce & Alexandra Papoutsaki

Weekly Lab

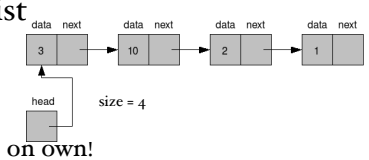
- Lab: JUnit
  - Unit testing with Java. Learn how to generate complete set of tests for each method in program.
  - Read 4 items called for in Lab handout!

Weekly Assignment

- Assignment: Compression
  - Need to define new class CurDoublyLinkedList
    - Keeps track of "current" elt.
    - Can be subclass of DoublyLinkedList from Structure5 library.
  - Get up to two points extra credit if turn in design by Thursday midnight.

Linked List

- Composed of Nodes
  - Think of as snap-lock beads
  - See code in structure5 library
    - From documentation page!
- See code in SinglyLinkedList
  - *Bailey - not std Java!*
  - keep track of head and size
  - Extends AbstractList -- look at on own!
    - Vector also extends AbstractList
- Also see SinglyLinkedListIterator



Linked List Algos

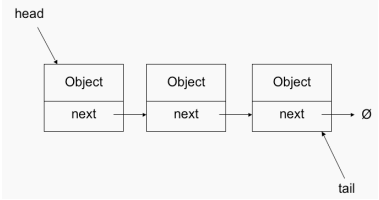
- Constructor ✔
- addFirst, removeFirst ✔
- get(i)
- indexOf(e)
- add(i,o)
- remove(e), remove(i)
- iterator

Read and understand code!!

What is worst-case complexity of each?

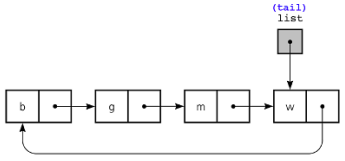
Variants of List

- If add a lot at end, add "tail" pointer
  - Makes adding at end faster
  - But bit harder to delete at end
  - More special cases -- e.g. add first when empty
  - See implementation when look at queues.



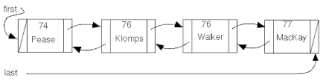
Variants of List

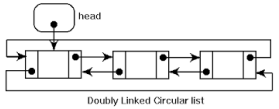
- Circular lists
  - Keep reference/pointer to end rather than beginning
    - What is the difference between adding to end & beginning?
    - getFirst vs getLast?
    - removeLast still hard!
    - How do you know when at end of list if searching?



Doubly-Linked List

- Doubly Linked Lists
  - Previous pointer as well as next
  - Useful if need to traverse in both directions
  - Provided by java.util.LinkedList (but we're using DoublyLinkedList from Bailey)
  - Must change twice as many links when adding or deleting!
  - Our class has head and tail pointers,
    - Doubly-linked lists often represented as circular!





How do you choose which to use?

Expectations

- You should be able to write any of these methods in any variant.
- Midterms always include such a question!
  - But don't try to memorize them!!!

Compact description of linked list variants:
https://wiki.cs.auckland.ac.nz/compsci105ss/index.php/Linked_Lists