Lecture 3: Stacks and Queues

Thanks to Kasey Champion, Ben Jones, Adam Blank, Michael Lee, Evan McCarty, Whitaker Brand, Stuart Reges, Zora Fung, Justin Hsia, and many others for sample slides and materials ...
Warm Up – Discuss with your neighbors!

From last lecture:
- What is an example of “constant time” complexity
- What is an example of “linear time” complexity
- What operations is an Array better at compared to Linked List?
- What operations is a Linked List better at compared to an Array?

From CSE 143:
- What is a “stack” and what operations is it best at?
- What is a “queue” and what operations is it best at?

Today’s Goals:
- Implementing List ADT
- Review Generics
- Implementing Stack ADT
- Review Queues
Implementing a List with a Linked List

Class Node
- string data
- Node next
Review: Generics

// a parameterized (generic) class
public class name<TypeParameter> {
  ...
}

- Forces any client that constructs your object to supply a type.
  - Don't write an actual type such as String; the client does that.
  - Instead, write a type variable name such as E (for "element") or T (for "type").
  - You can require multiple type parameters separated by commas.

- The rest of your class's code can refer to that type by name.

More details: https://docs.oracle.com/javase/tutorial/java/generics/types.html
Implementing a Generic List
**Review: What is a Stack?**

**stack:** A collection based on the principle of adding elements and retrieving them in the opposite order.
- Last-In, First-Out ("LIFO")
- Elements are stored in order of insertion.
  - We do not think of them as having indexes.
- Client can only add/remove/examine the last element added (the "top").

**basic stack operations:**
- **push(item):** Add an element to the top of stack
- **pop():** Remove the top element and returns it
- **peek():** Examine the top element without removing it
- **size():** how many items are in the stack?
- **isEmpty():** true if there are 1 or more items in stack, false otherwise
Implementing a Stack with an Array

Things to keep track of:
- index
- size of array

![Array representation of a stack with elements 1 to 6 and arrows indicating push and pop operations.](image)
Implementing a Stack with a Linked List

Discuss with your neighbors!

```plaintext
push(3)
```

```
head
```

```
stack top
```

```
stack bottom
```

```
tail
```
**Review: What is a Queue?**

**queue:** Retrieves elements in the order they were added.
- First-In, First-Out ("FIFO")
- Elements are stored in order of insertion but don't have indexes.
- Client can only add to the end of the queue, and can only examine/remove the front of the queue.

**basic queue operations:**
- **add(item):** aka “enqueue” add an element to the back.
- **remove():** aka “dequeue” Remove the front element and return.
- **peek():** Examine the front element without removing it.
- **size():** how many items are stored in the queue?
- **isEmpty():** if 1 or more items in the queue returns true, false otherwise
Implementing Queues with a Linked List

- remove(1)
- add(5)

Diagram showing a linked list with nodes labeled 1, 2, 3, and 5. The terms 'front' and 'back' indicate the locations where items are removed and added, respectively.
Implementing Queues with an Array

1) Either double the array
2) OR Reuse index 0
    memory efficient Sol^

front 0 1 2 3 4 5 back
add(5)
remove(2)
add(4)
add(6)
add(7)
TODO list

- Fill out pre-course survey!
  - Link on course webpage
  - Due today (10/1)

- Homework 1 is out!
  - Individual assignment
  - Due 10/5 11:59pm