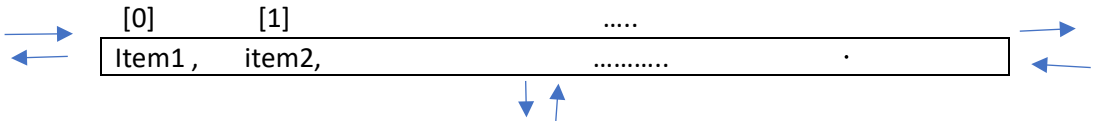
CIS2168 F22 Midterm Review and Study Guide, Part 2

Array (Fixed size)

1. Random access data structure
2. Each element is accessed by index in constant time

List

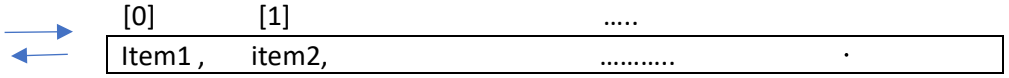
1. An ordered collection of items. Each item has an index. Can perform data operations anywhere in the linear collection (two ends, in the middle)



2. Two types: ArrayList, LinkedList
3. ArrayList
    - Use a resizable array to store items
    - Provide random access by index (position)
    - Performance (time complexity)
        - get(index): O(1)
        - indexOf(item): O(n) typical case (not first/last)
        - insert(index, item): O(n) when index != size - 1
        - remove(index, item): O(n) when index != size - 1
4. LinkedList
    - Use references to link items in the linear collection
        - Singly linked: one successor reference
        - Doubly linked: one successor reference, one predecessor reference
        - Circular linked list (doubly/singly): first is last's successor, last is first's predecessor
    - Provide sequential access in a particular order
    - Performance (time complexity)
        - Singly: O(n), except access first or from a known node: O(1); If last reference is maintained, access last: O(1)
            - get(index): O(n) typical case or index is size -1 and no tail reference maintained; O(1) special case: index is 0 or (index is size-1 & tail reference maintained)
        - Doubly/Circular: O(n), except access first or from an known node or access last: O(1)
            - get(index): O(n) typical case; O(1) special case: index is 0 or size -1
5. Iterator
    - A Java object that uses a position marker to keep track of current position while traversing items in a data collection.
    - Make the iteration over a collection easy and efficient

Stack

1. A linear collection accessed in a Last In First Out fashion. Operations are limited at one end in the linear collection. Remove most recently added item.

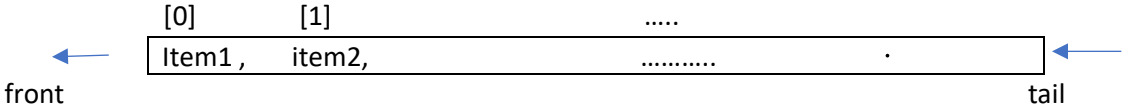


Top Bottom

2. Common operations: push, pop, peek, isEmpty
3. Two types of implementations
    - Resizable array-based stack: ArrayDeque<E> in Java API
    - Linked list based stack: LinkedList<E> in Java API
4. Resizable Array-based Stack
    - Use a resizable array to store items in the stack
    - Performance (time complexity)
        - Push: amortized O(1)
        - Pop, peek: O(1)
5. Linked List based Stack
    - Use a linked list to store items in the stack
    - Performance (time complexity)
        - Push, pop, peek: O(1)

Queue

1. A linear collection accessed in a First In First Out fashion. Operations are limited at two ends in the linear collection. Additions are at one end (tail) while removals are at the other end (front). Remove least recently added item.



2. Common operations: enqueue (offer), dequeue(poll), peek, isEmpty
3. Two types of implementations
    - Resizable circular array based queue: ArrayDeque<E> in Java API
    - Linked list based queue: LinkedList<E> in Java API
4. Resizable Circular Array based Queue
    - Use a resizable circular array to store items in the queue
    - Performance (time complexity)
        - Enqueue: amortized O(1)
        - Dequeue, peek, isEmpty: O(1)
5. Linked List based Queue
    - Use a linked list to store items in the queue
    - Performance (time complexity)
        - enqueue (offer), dequeue(poll), peek, isEmpty: O(1)

Big-O and Time Complexity

Common Growth Rates

Running time getting higher with the same input size n

Big-O	Name
$O(1)$	Constant
$O(\log n)$	Logarithmic
$O(n)$	Linear
$O(n \log n)$	Log-linear
$O(n^2)$	Quadratic
$O(n^3)$	Cubic
$O(2^n)$	Exponential
$O(n!)$	Factorial