Algorithms and Data Structures

String, Stack and Queue

Challenges

- How to check 2 Strings have the same identity?
- How to find duplicate characters from a string (composed by all lower [a-z])
- How to find duplicate characters from a string (composed by any chars)
- How to check if a String is palindrome?
- How to check if a String is a valid shuffle of two other Strings(i.e., ‘abcde’ is a valid shuffle of ‘ac’, ‘bde’)
- How to mask a String for certain keywords, for example ‘secret’ as ’s****t’’, without using build-in methods

Part 2: Stack and Queue

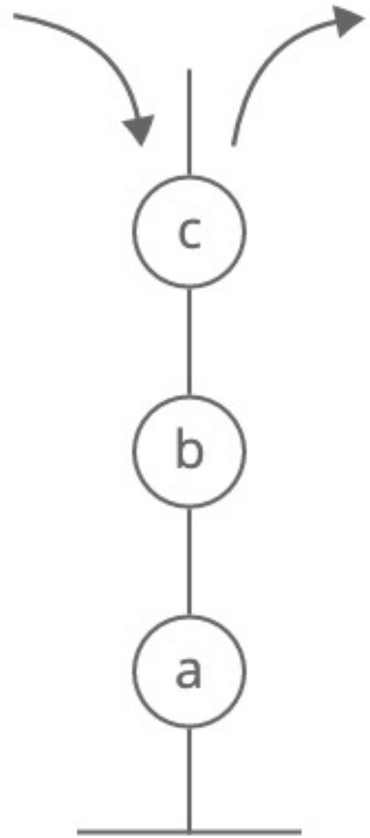
FIFO VS FILO

- In accounting, one key question to ask when you need to calculate the cost of your goods is: FIFO (First In, First Out) or FILO (First In, Last Out)?
- Example:
  - You are running a company that buy and sell Apples.
  - Your records show as:
  - for the ???:
    - the 25 apples you sold are
      - FIFO: (12 + 13) then:
        - » cost 12 * 1 + 13 * 2 = 38, profit 75 - 38 = 37.
      - FILO: (5 + 20) then:
        - » cost 5 * 1 + 20 * 2 = 45, profit 75 - 45 = 30.

	action	quantity	price	profit
time 1	buy	12	$1	n/a
time 2	buy	20	$2	n/a
time 3	sell	25	$3	???

FILO: Stack

- Stacks use First In, Last Out order.
- Stacks are naturally use for real world objects including:
  - recursion / subproblem solving
  - dirty plates to be washed
  - fill a U-haul for moving
  - Deep-First-Search (DFS) - be covered later
  - String parsing (i.e. compiler such as Eclipse)



Stack Operations

- Common operations in a stack are:
  - push()
  - pop()
  - peak()
  - contains()
- Example: create a stack, and pop it to a sentence: “Winter in Boulder will be Paradise!”

Stack Example

- Detect a valid closure of a sentence:
  - Valid: {[][(())][()]}
  - Invalid: {, {[}, {[}], ())
- That's why Stack is the natural adoption for arithmetic operations (and parser of programming languages)

Stack Implementation

- Stacks can be implemented using Linked List:
  - push <-> prepend
  - pop <-> removeHead
- Stacks can be implemented using Arrays (Dynamic Arrays) as well:
  - push <-> append
  - pop <-> remove the last element of the array
  - keep track of the last index of the array
  - A good practice project to run.

Stack Implementation

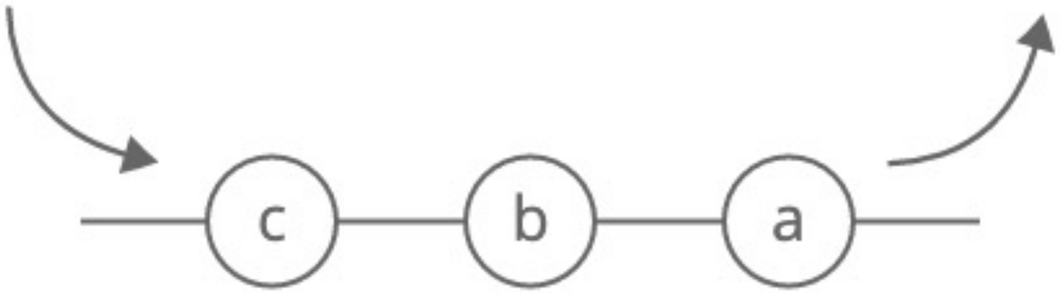
- In Python, Stack can be implemented using list
  - Use append() for push()
  - Use pop() for pop()
  - Use last index to access for peak()

Time Complexity of Stack

- All operations should have O(1) cost (a requirement for implementation):
  - push()
  - pop()
  - peak()
- These operations should/may have O(n) cost:
  - contains()
  - access
  - search

FIFO: Queue

- Queues use First In, First Out order:
- Queues are naturally used for real world objects including:
  - Waiting line
  - Restaurant's First Come, First Serve order
  - Task list (with equally importance and independent)
  - Printers
  - Breadth-First-Search (BFS): be covered later.
  - All Operations has O(1) cost:
    - enqueue
    - dequeue
    - peek



Queue Implementation

- Naturally, Queues are implemented using Linked List:
  - enqueue <-> append
  - dequeue <-> removeHead
- Queues can be implemented using Arrays:
  - keep track of the first and last index of the array, first index indicates the 1st element, and last index indicates the last element.
  - enqueue <-> increase the last index
  - dequeue <-> increase the first index
  - peak<-> return the element of the first index
  - A good practice project to run.

Challenges

- Write a program to reverse a string using stack
  - Given a string, reverse it using stack. For example “GeeksQuiz” should be converted to “ziuQskeeG”
  - Spaces on leading/tailing will be ignored.
- Follow up question: Write a program to reverse a sentence using stack
  - Given a sentence, reverse it using stack. For example “Geeks Quiz” should be converted to “Quiz Geeks”
  - Spaces on leading/tailing will be ignored.

Challenges

- Find all elements in an array that are greater than all elements present to their right
  - Given an unsorted array of integers, print all elements which are greater than all elements present to its right.
  - For example: Input : {10, 4, 6, 3, 5}
  - Output: {10, 6, 5}

Thank you!