



LeetCamp Week2

Zixuan Jia/Yile Chen



Housekeeping

- Every Sunday 2pm
- Check discord regularly for updates
- Project Github:
https://github.com/infknight/Leetcamp_Fall_2020

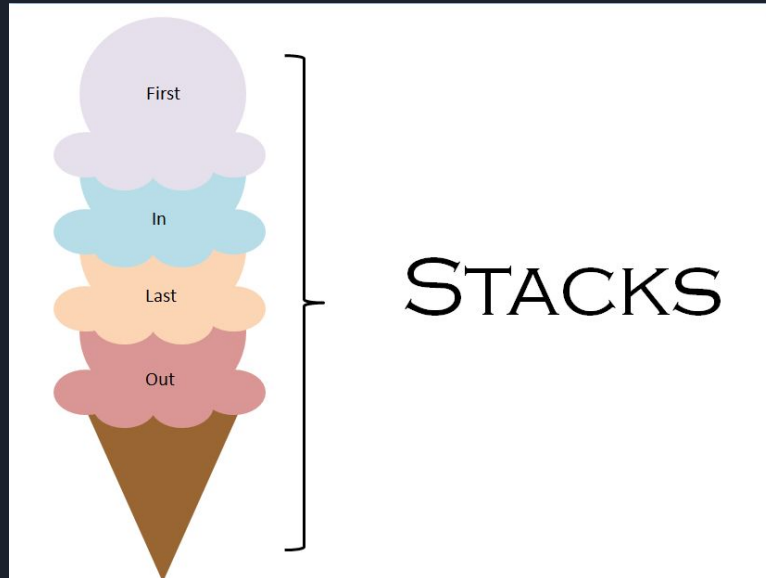


Overview

- What is a Data Structure
- Stack
- Queue
- Leetcode general question
- Leetcode challenging question

Stack

- Linear Data structure follows a particular order
- You will likely to hear people call as a LIFO (Last in First Out) or FILO (First in Last Out)





Stack

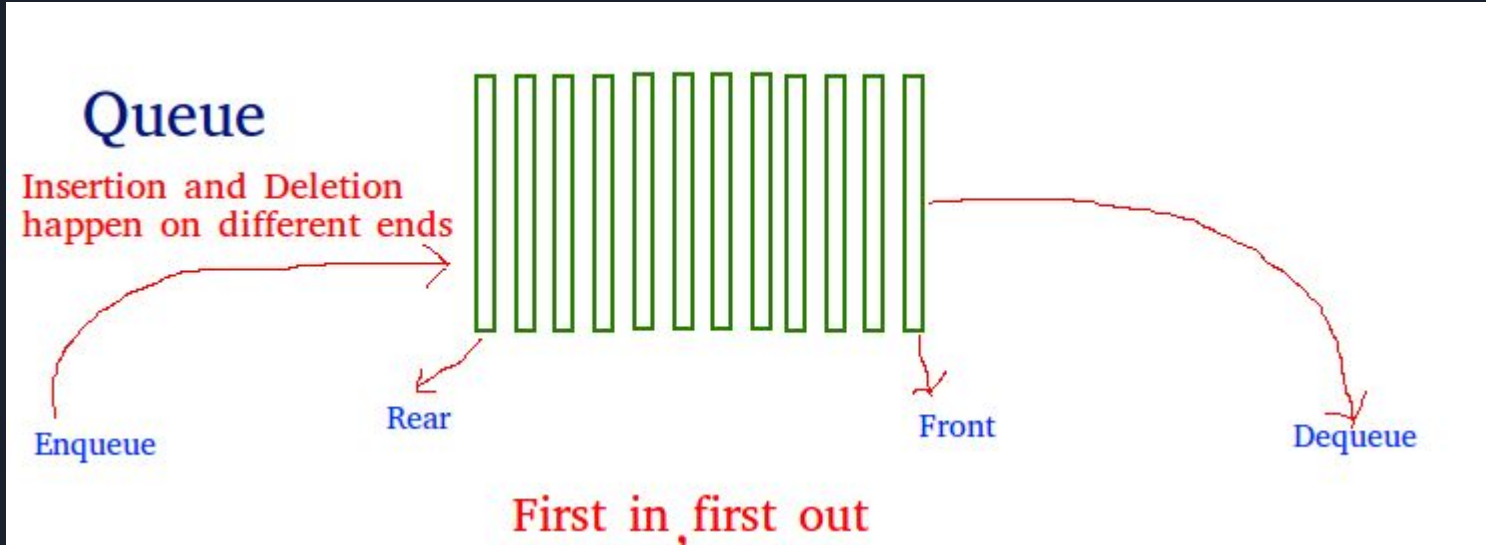
- Common Operation
 - Push(): put an element on the stack
 - pop(): removing an element from the stack
 - Which element?
 - top(): Get the top of the element from the stack
 - isFull(): check if stack is full or not
 - isEmpty(): check if the stack is empty or not
- In python, stack is just a list:
 - `Stack = []`
- In C++, you need to include the Stack Library
 - `stack<int> s;`



Big O analysis

- What is the Big O for each operation in stack?

Queue



- A linear data structure that stores items in **First In First Out (FIFO)** order
- Example: Any queue of consumers where the consumer that came first is served first.



Queue

- Common Operation
 - **Enqueue**: Adds an item to the queue. If the queue is full, then it is said to be an Overflow condition – Time Complexity : $O(1)$
 - **Dequeue**: Removes an item from the queue. The items are popped in the same order in which they are pushed. If the queue is empty, then it is said to be an Underflow condition – Time Complexity : $O(1)$
 - **Front**: Get the front item from queue – Time Complexity : $O(1)$
 - **Rear**: Get the last item from queue – Time Complexity : $O(1)$
- In python, queue is just a list: `stack = []`
 - `Enqueue()` is achieved by `append()`, `Dequeue` is achieved by `pop(0)`: pop the first element
- In C++, use queue standard library
 - `queue<int> myqueue`



Stack / Queue Practice

- Write a function prints out 1, 2, 3, 4, 5 and return 2.

```
int ST(stack<int> ST){  
  
}
```

#20 Valid Parentheses

20. Valid Parentheses

Easy 5805 241 Add to List Share

Given a string `s` containing just the characters `'('`, `')'`, `'{'`, `'}'`, `'['` and `']'`, determine if the input string is valid.

An input string is valid if:

1. Open brackets must be closed by the same type of brackets.
2. Open brackets must be closed in the correct order.

Example 1:

Input: `s = "()"`
Output: `true`

Example 2:

Input: `s = "()[]{}"`
Output: `true`

Example 3:

Input: `s = "(]"`
Output: `false`

Example 4:

Input: `s = "([)]"`
Output: `false`

Example 5:

Input: `s = "{[]}"`
Output: `true`



Solution

- Push everything to the Stack
- Do condition check
- $O(N)$

#242 Valid Anagram

242. Valid Anagram

Easy



1874



150



Add to List



Share

Given two strings s and t , write a function to determine if t is an anagram of s .

Example 1:

Input: $s = \text{"anagram"}, t = \text{"nagaram"}$

Output: true

Example 2:

Input: $s = \text{"rat"}, t = \text{"car"}$

Output: false

Note:

You may assume the string contains only lowercase alphabets.


Follow up:

What if the inputs contain unicode characters? How would you adapt your solution to such case?



Method 1: Sort

- Sort the two strings into alphabetical order (a string can be turned into an array of letters)
- Compare if they are equal
- Use python `sorted()`, or turn string into list, then use `sort()`



Method 2: Hashmap

- Iterate through first string and add each occurring letter to the hashmap, increase the value by 1 if it occurs more than once
- Iterate through the second string and check the hashmap
 - if it already exists, decrease the count by 1,
 - otherwise return False



Optional Question