

## Do Practice HW Problems!

### Algorithm Analysis

- Be able to look at code and determine if it's  $O(1)$ ,  $O(\log(n))$ ,  $O(n)$ , etc.
- Given  $N_1$ ,  $T_1$ ,  $O(N)$  – predict  $T_2$  from  $N_2$  (and vice versa)
  - In other words, given the time it takes to process  $N_1$  items, be able to determine the time it would take to process  $N_2$  items, or determine the number of items that can be processed in a different amount of time.

### For each of the Data Structures covered so far (see below), know:

- All relevant Operations
  - $O(???)$  for each
    - Worst case, Best case Average case
  - Theoretical (non-Python) Algorithms for each (including recursive ones), and including drawing pictures of linked-lists and arrays.
  - Be able to Read/Write/debug Python code for each operation
- Data structures:
  - Stack
  - Queue
  - Circular Queue
  - Link-based list
  - Array-based list

### Data Structures covered so far:

#### Stack

- Both Array and Linked List implementations, as implemented in Lab 2
- Operations:
  - push
  - pop
  - size
  - is\_empty
  - is\_full

#### Circular Queue, as implemented in Lab 3

- Both Array and Linked List implementations
- Operations:
  - enqueue
  - dequeue

### List

- Both Array based and Link based
- Operations:
  - add (beginning, end, middle)
  - remove (beginning, end, middle)
  - find/get/contains (beginning, end, middle)
  - set (beginning, end, middle)

### Binary Trees

Traversals - inorder, postorder, preorder, level order

### Binary Search Tree

- Understand what a BST is
- Operations:
  - Insert/Replace
  - Search/Find
  - Find min/max
  - Height
  - Delete

### Code Reading/Writing

- Be able to read/write Python code for operations of the covered data structures, especially for the operations implemented in labs/projects.
- Be able to read/write recursive functions such as those implemented in Lab 1 and Project 1
- Be able to read/write code that evaluates postfix expressions, and be able to evaluate them by hand.
- BST operations - insert/replace, search/find, find min/max, height - no delete