Refresher

Refresher
Data Structures and Algorithms

Prof. Jigar M. Pandya

Why

- Why?
  - Third semester stage vs seventh semester stage
    - Students have learned more subjects.
      - i.e. Think of time complexity of data structures' operations w.r.t DAA
      - i.e. Think of more applications w.r.t. other subjects like OS, TAFL, CC
  - Give a try to solve below:

    What data structure do you need to judge ordered and correct count of brackets ((( ) ( )))?

...

- If you have answered stack then you need to think again...
- Remember, solution to problems in Computer Science/Engineering must follow the rule of thumb

  “No more, No less.”
- Every time you solve the problem, you need to think as if you are solving first time.
  - Bcoz you can not remember solutions to thousands of problems.
  - All you can do is work on your problem solving approach and **refresh** the existing knowledge base.

Data Structure Vs Database

- Logical processing in primary memory using programming is supported using Data Structures
- Transactions persisting with ACID support is by Database Systems
  - uses data structures to process data insertion, retrieval, recovery, etc.

Abstract Data Type

- Defined by its behavior (semantics) from the point of view of a user of the data
  - Specifically in terms of possible values, possible operations
  - Abstract because implementation details are left upto the implementer
    - i.e. stack can be implemented in different ways
    - i.e. queue can be of any required object

      int, float, books, students, jobs, etc.

Brainstorming

- For given expression

  i.e. a + b * (c - d) / e
- Find solution expression where:
  - There must not be brackets
  - Meaning must not be changed

Stack

- Last in First Out (LIFO)
- Properties
  - Insert and removal is from one common end only
  - One at a time
- Operations
  - Pushto insert on top of stack
  - Pop to retrieve from top of stack
  - Peep to view the top most element/Read only top
  - IsEmpty for underflow verification
  - IsFull for overflow verification

Thought Process

- Supported memory required?
- Does it grow and shrink?
- Are you going to need, recently stored information first and older later?
- If yes, you need stack.

Solution

- Benefits of postfix and prefix notations:
  - No precedence
    - Hence, no brackets
-

  (Postfix notation)

  a + b * (c - d) / e => a b c d - * e / +

Applications of stack

- Bracket balancer
  - [{()()}{}][]
- Palindrome validity
  - With sentinel at mid
    - You need push until sentinel
  - Generic
    - You need push all
- Infix Evaluation
  - Infix to postfix conversion
  - Postfix evaluation
- Tree Traversals
  - Depth first
- Parser
  - Xml parsing
- Undo operation
- PDA
- Solving recursive definitions based problem statements

Brainstorming

- Multiple computers use a shared printer resource, how shall printer handle this requests?

Queue

- First in First Out (FIFO)
- Properties
  - Insert into rear end and remove from front end. Different ends.
  - One at a time
- Operations
  - Insert
  - Remove
  - IsEmpty for underflow verification
  - IsFull for overflow verification
  - Size
  - peek

...

- Types
  - General queue
  - Circular queue
  - Double ended queue
  - Priority queue
- Solution:

  If request has to be served by printer based on first come first serve (FCFS) bases, use general queue. Space Efficient implementation is Circular.

Thought Process

- Supported memory required?
- Does it grow and shrink?
- Are you going to need, recently stored information last and older first (FCFS)?
  - If yes, you need queue.
- Are you going to need based on priority?
  - If yes, you need priority queue.

Examples

- Tree Traversals
  - Breadth first/ Level order
- Job scheduling/Process State Management

  i.e. ready/running/waiting/etc.
- Operation System Task Manager

Complexities

- Stack:
  - Push O(1)
  - Pop O(1)
  - Peek O(1)
  - isEmpty O(1)
  - isFull O(1)
- Queue
  - Insert O(1)
  - Remove O(1)
  - Peek O(1)
  - isEmpty O(1)
  - isFull O(1)

Priority Queues

- Used in multitasking operating system.
- Represented using “heap” data structure.
- Insertion O(n) time, deletion in O(1) time. For approach where you process while inserting, removal is straight forward.