CS32 Intro to Computer Science II

Baoxiong Jia & Muthu Palaniappan, DIS 1C Week 4
UCLA Spring 2021

About Us

- TA: Baoxiong Jia
  - Email: baoxiongjia@cs.ucla.edu
  - Office Hours: Tuesday 8:30-10:30am
  - Thursday 8:30-10:30am
  - Discussion 1C: Friday 12:00-13:50pm

- LA: Muthu Palaniappan
  - Email: muthupal@g.ucla.edu
  - Office Hours: Monday 10:30-11:30am
  - Wednesday 10:30-11:30am

Outline

- Stacks
- Queues
- Examples

Stack

http://www.cplusplus.com/reference/stack/

LIFO: Last in first out

Member functions

(constructor)	Construct stack (public member function)
empty	Test whether container is empty (public member function)
size	Return size (public member function)
top	Access next element (public member function)
push	Insert element (public member function)
emplace C++11	Construct and insert element (public member function)
pop	Remove top element (public member function)
swap C++11	Swap contents (public member function)

Queue

http://www.cplusplus.com/reference/queue/queue/

FIFO: First in first out

Member functions

(constructor)	Construct queue (public member function)
empty	Test whether container is empty (public member function)
size	Return size (public member function)
front	Access next element (public member function)
back	Access last element (public member function)
push	Insert element (public member function)
emplace C++11	Construct and insert element (public member function)
pop	Remove next element (public member function)
swap C++11	Swap contents (public member function)

Implementing Queue with Two Stacks

https://repl.it/@jiajerry/QueueWithStack

Implementing Stack with Two Queues

https://repl.it/@jiajerry/StackWithQueue

Infix to Postfix

Infix	Postfix
15 + 6	15 6 +
9 – 4	9 4 -
(15 + 6) * 5	15 6 + 5 *
7 * 6 + 5	7 6 * 5 +
3 + (4 * 5)	3 4 5 * +

Walk through: Postfix Evaluation Algorithm

Postfix Evaluation Algorithm

Inputs: postfix expression string
Output: number representing answer
Private data: a stack

7 6 * 5 +

1. Start with the left-most token.
2. If the token is a number:
   a. Push it onto the stack
3. Else if the token is an operator:
   a. Pop the top value into a variable called v2, and the second-to-top value into v1.
   b. Apply operator to v1 and v2 (e.g., v1 / v2)
   c. Push the result of the operation on the stack
4. If there are more tokens, advance to the next token and go back to step #2
5. After all tokens have been processed, the top # on the stack is the answer!