CSC 217 Lab 09

Stacks & Queues

Lab Overview

- Deadlines and Reminders
- Team Introductions
- Stacks & Queues Refresher
- Activities
  - Bug Fix!
  - Stacks
  - Queues
  - Runtime Efficiency
  - Add Waitlist to CourseRoll
  - Test! Deploy!
- Lab Wrap-Up

Deadlines and Reminders

- Deadlines
  - See write up for your section’s Lab 09 deadline
- Reminders
  - Project: Leave time to fix bugs, work bottom to top
  - Lab: #9 is light but confusing. Use pair programming and/or multiple days
    - Design page is missing text bc **you decide** what to use!
  - This is the beginning of the end

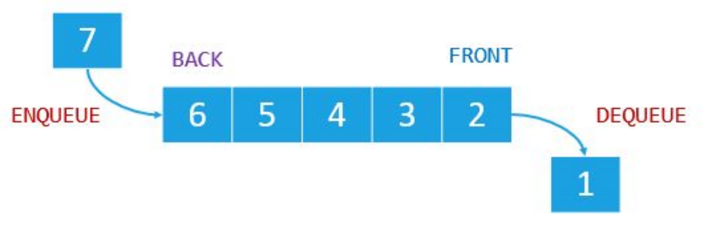
Team Introductions

Welcome to your team for CSC 217 Lab 9 through Lab 11!

Take a few minutes to get to know each other. Make sure to do the following:

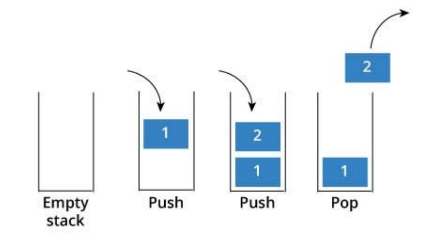
- Exchange contact information and create common chat group
- Discuss what tasks you prefer and which you want to practice
- **Move one teammate’s Lab 8 PackScheduler** into your new Lab 9 repo
  - Decide whose project to use (don’t just start pushing yours!)
  - Follow the Git guide for GP1 -> GP2 (or GP2 -> GP3)
  - Briefly inspect previous code (share tricks or solve problems)

Stacks and Queues



Both are restrictions of List.

Stack - Only add / remove on one end.
Queue - Add at one end, remove at other end.



Delegation

- You will use delegation to implement Stacks and Queues
- Implement an ArrayList version and a LinkedList version
- Use runtime efficiency and requirements to identify the best specialized data structure for implementing a waitlist in CourseRoll

Lab Activities

- Bug Fix!
  - Let’s add a capacity to LinkedAbstractList
- Stacks & Queues
  - ArrayList doesn’t have a capacity - must be in Stack/Queue classes
- Runtime Efficiency
  1. How does the data type map to the list operation? (push -> add at back)
  2. How efficient is that list operation? (add at back w/ ArrayList? O(1))
- Update LinkedAbstractList with a back reference
- Add Waitlist to CourseRoll
  - Commonly forgotten - What’s the purpose of a waitlist?
- Test! Deploy!

Wrap-Up

General Wrap-Up

- Exchange contact information with your partner
- Make a plan for finishing up the lab
- Lab 9 deadline in writeup

Participation Outside of Lab (Guess which the teaching staff prefer?)

- If you pair program/design, **note that in the commit comments so everyone gets credit!**
- If you split the work, at least one contribution by each partner

REMINDER: We are expecting a significant contribution from all team members outside of lab!

- If you pair program/design, you **MUST** note it in your commit messages or there will be deductions
- Students who don’t allow their partners to contribute will receive deductions
- Students who don’t contribute will receive deductions

Record Tasks & Owners

Tasks only get done when someone owns them!

Identify the tasks required to complete Lab 9

- Edit README.md to list the tasks required to complete Lab 9 (at top of README - should come before Lab 8 tasks)
- Add an owner to each task
- Add a deadline to each task

Deadlines should be at least 48 hours before the lab deadline so team members can help out and finish the lab if a team member runs into issues.

Notify team early if you run into problems with your tasks!