CS 367 - Introduction to Data Structures
Week 6, 2017 (UPDATED)

Midterm Exam 1: Tuesday, February 28th, 5:00 pm – 7:00 pm

- Lecture 1: Room 272 of Bascom Hall
- Lecture 2: Room 125 of Agriculture Hall
- Lecture 3: Room 204 of Educational Sciences Building
- UW ID required
- See posted exam information
- Exam 1 Information and Sample Questions (available on Canvas by Tuesday)

Homework 5 due 10 pm Friday, February 25th
Program 2 due 10 pm Sunday, March 5th - GET STARTED NOW!

Last Week
Iterable (for loops); Complexity Analysis, analyzing java code, significance of scaling, comparing ListADT implementations (array vs linked nodes)

This Week
Read: Stacks, Queues, Priority Queues, Tree terms
Significance of Scaling (from last week)
Complexity Caveats (from last time)
Shadow Array - improving complexity of array resizing
Stack ADT
- concept
- array implementations
- chain of nodes implementations
Queue ADT
- concept
- chain of nodes implementations
Priority Queue
Tree Terms
Priority Queue ADT
- concept
- operations
- implementation options
Circular Array d.s.

Next Week
Read: start Priority Queues
- Exam mechanics
- Sample questions solution
Returning N Papers to N Students

problem size (N) = number of students
dominant operation?

→ What is the complexity of each algorithm below?

Algorithm 1:
call out each name,
have student come forward & pick up

   best-case:

   worst-case:

Algorithm 2:
hand pile to first student,
student linearly searches through papers & takes hers/his,
pass pile to next student who does likewise

   best-case:

   worst-case:

Algorithm 3:
sort the papers alphabetically,
hand pile to first student who does binary search,
pass to next student who does likewise
Shadow Array – Improving Array Resizing

"Naïve" Approach

![Diagram of the "Naïve" Approach]

"Shadow Array" Improvement

![Diagram of the "Shadow Array" Improvement]
Stack ADT

Concept

Operations

Implementing using an Array

→ Where should the top be located in the array?

Implementing using a Chain of Nodes

→ Where should the top be located in the chain of nodes?

Complexities
Queue ADT

Concept

Operations

Implementing a Queue using a Chain of Nodes

→ Is one option better than the other?

**Option 1:** front of queue is at head, rear of queue is at tail

**Option 2:** front of queue is at tail, rear of queue is at head
Implementing a Queue ADT using an Array

Assume a shadow array is used so that expand is $O(1)$.

**Option 1:** front of queue is at ________________, rear of queue is at ________________

```
|   |   |   |   |   |
-   -
```

**Option 2:** front of queue is at ________________, rear of queue is at ________________

```
|   |   |   |   |   |
-   -
```

**Option 3:** front of queue is at ________________, rear of queue is at ________________

```
|   |   |   |   |   |
-   -
```
Implementing a Queue ADT using Circular Array

Concept

enqueue(item)

dequeue()

expand()
1. Which is the root?

2. How many leaves are there?

3. How many nodes are in the right branch/subtree of B?

4. Which is the parent of G?

5. How many children does E have (degree of E)?

6. Which is the sibling of E?

7. How many descendants does B have?

8. What are the ancestors of C?

9. What is the length of the path from B to D?

10. What is the height of the tree?

11. What is the depth/level of J?
Priority Queue ADT

Priorities

Concept

goal:

Operations
## Options for Implementing a Priority Queue ADT

<table>
<thead>
<tr>
<th>data structure</th>
<th>insert</th>
<th>removeMax</th>
</tr>
</thead>
<tbody>
<tr>
<td>unordered array</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ordered array</td>
<td></td>
<td></td>
</tr>
<tr>
<td>unordered chain of nodes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ordered chain of nodes</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>