Midterm Exam 2 Results
High Score:
Average Score:
Low Score:
S.D.:

Comments

Program 4 due 10 pm Sunday, April 16th

Last Week
Exam 2, Red-Black Trees: tree properties, print, lookup, insert

This Week
Read: Graphs
Return exam

Graphs
- terminology
- implementation
- edge representations
- traversals
- applications of BFS/DFS
- more terminology
- topological ordering

Next Week (more Graphs and Hashing)
Read: continue Graphs, Hashing
- topological orderings
- Dijkstra's Shortest Path algorithm
Hashing
- terminology
- designing a good hash function
- choosing table size
- expanding a hash table
- handling collisions
Java Support for Hashing: Tree Map vs Hash Map
Sorting Intro
ADTs/Data Structures

Linear (Lists, Stacks, Queues)

- predecessors: at most 1
- successors: at most 1

Hierarchical (Heaps, BSTs, Balanced Search Trees)

- predecessors: at most 1
- successors: 0 or more - general tree, at most two - binary tree

Graphical

- predecessors:
- successors:
Graph Terminology
Implementing Graphs

Graph ADT Ops

Graph Class

Graphnode Class
Representing Edges

Adjacency Matrix

Given the following graphs:

Graph 1

Graph 2

Show the adjacency matrix representation of the edges for each of the graphs:

Graph 1

Graph 2
Representing Edges

Adjacency Lists

Given the following graphs:

Graph 1

Graph 2

→ Show an adjacency list representation of the edges for each of the graphs:

<table>
<thead>
<tr>
<th>Graph 1</th>
<th>Graph 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0:</td>
<td>A:</td>
</tr>
<tr>
<td>1:</td>
<td>B:</td>
</tr>
<tr>
<td>2:</td>
<td>C:</td>
</tr>
<tr>
<td>3:</td>
<td>D:</td>
</tr>
<tr>
<td>4:</td>
<td>E:</td>
</tr>
</tbody>
</table>
Using Edge Representations

⇒ Write the code to be added to a Graph class that computes the degree of a given node in an undirected graph.

1. Adjacency list:

   ```java
   public int degree(Graphnode<T> n) {
   ```

2. Adjacency matrix:

   ```java
   public int degree(Graphnode<T> n) {
   ```
Comparison of Edge Representations

Ease of Implementation

**Space** (memory)

AM

AL

**Time** (complexity of ops)

node’s degree?

AM

AL

edge exit between two given nodes?

AM

AL
Searches and Traversals

Search

Traversal

→ Which connected component in the graph above can produce the longest path?
Depth-First Search (DFS)

- 
- 
- 

Algorithm

*
DFS Practice

Graph 1

Graph 2

→ Give the order that vertexes are visited for depth-first search (DFS) starting at A.

Graph 1:

Graph 2:

→ Give the DFS spanning tree starting at A.

Graph 1:          Graph 2:
Breadth-First Search (BFS)

Algorithm
BFS Practice

→ Give the order that vertexes are visited for breadth-first search (BFS) starting at A.

Graph 1:

Graph 2:

Give the BFS spanning tree starting at A.

Graph 1:

Graph 2:
Applications of DFS/BFS

Path Detection

Cycle Detection
More Graph Terminology