Announcements/Reminders:

- Final tomorrow in **Psych 113** (regular class hour)

Last class:

- Graphs (cont'd)
  - Breadth-First Search
  - Dijkstra's Shortest Path Algorithm

Today:

- Graphs (finish)
  - Topological Ordering
- Course Review
- Course Evaluations
Dijkstra's Algorithm (cont'd)

Sketch of proof of key step:

Run-time analysis

Operations: insert, removeMin, decreaseEstimate

Using arrays:

Using min-heaps:

Using Fibonacci heaps:

Applications
Topological Ordering

Problem:

Example:
Topological Ordering (cont'd)

Algorithm:
Course Overview

Data Structures (DS) and Abstract Data Types (ADTs)

Algorithms

Complexity

Java Concepts
Data Structures (DS) and Abstract Data Types (ADTs)

Layout

Linear:
• 1 predecessor (all except first) and 1 successor (all except last)
• DS: array, chain of linked nodes, circular structures
• ADTs: List, Linked List, Stack, Queue, Deque, Hashtable

Hierarchical:
• 1 predecessor (except root) and 0, 1, or more successors
• DS: Tree nodes, heap
• ADTs: Trees (general, binary, search, etc.), Priority Queues

Graphical:
• 0 or more predecessors, successors. Can specify source, sink, start, end
• DS: vertices stored in list/map, edges stored in adj. matrix/list
• ADT: Graph (undirected/directed, weighted/unweighted)

Style of operation

Position-oriented: operations occur at a specific position
• List
• Stack
• Queue
• Tree (general, binary)

Value-oriented: operations occur at position determined by key value
• Heap
• BST, balanced trees
• Map, Set
• Graph

“Hybrid”:
• Priority Queue
• Hashtable
Algorithms

Operations on ADTs/data structures:

Recursion

Searching

Hashing

Sorting
Complexity

Time, Space Complexity:

Worst case, best case, average case:

Determining Complexity of Algorithms/Code:

- Non-recursive

- Recursive
Java Concepts

Reference Types

Command-line arguments

Generic Types

Exceptions

Iterators

Interfaces

Java Collections Framework