

## Final Exam Coverage

Summer II 2003  
JP Vergara

## Time and Space Complexity

- Time Complexity
  - Asymptotic assessment  
 $O(1)$ ,  $O(\log n)$ ,  $O(n)$ , ...
  - Data structure operations
  - Algorithms
- Space Complexity
  - Space overhead to represent structure
  - Tradeoffs across structures/implementations
- Best-case, worst-case, average-case analysis

## Linear and Tree Structures

- Lists and arrays as they are used in more complex data structures
- Binary search, heaps as used in later topics (sorting, indexing, etc.)
- Study:
  - Huffman coding trees
  - Traversals (preorder, post order, in-order)
  - Sequential representation for binary trees

## Sorting

- $O(n^2)$  algorithms: insertion, bubble, selection
- $O(n \log n)$  algorithms: quick, merge, heap
- Lower bound analysis for comp-based sorting
- $O(n)$  algorithms: bucket/bin, radix
- Question types: step-by-step simulation, variants, comparison between algorithms, pros/cons
- \* Shell Sort not covered in exam

## Disk Access

- Computing Disk Read Time
  - Given disk, track, sector, cluster info compute the time it takes to read data from a file
- Buffer Pools
  - Message passing versus direct access (see ADTs)
  - Replacement policies (FIFO, LRU, LFU)

## Hashing

- See handout
- Open hashing: slots are bins for lists
- Closed hashing: table contains records
- Question types: determine home position, probe sequence, or landing slot; effect of changing hash function, probe function, and other parameters
- Final project and bucket hashing

## Indexing

- Linear index
- Regular binary search tree index
- 2-3 Trees / B-Trees
- B+ Trees
  
- Question types: simulation of operations, differences, pros/cons

## Graphs

- Notation and definitions: graph, directed graph, vertices, edges, paths, cycles
- Graph representation
  - Adjacency matrix
  - Adjacency list
- Graph Traversals
  - DFS and BFS

## Topics that will not be covered in Final Exam

- Shell Sort
- External Sorting
- Self-organizing lists
- Graph algorithms that we did not cover in class