**Chapter 11 Graphs: Instructor's CD questions**

1. Which is not the name for a standard graph traversal?

\*a) Preorder.

b) Depth first.

c) Breadth first.

2. Depth-first search is best implemented using:

\*a) A stack or recursion.

b) A queue.

c) A tree.

3. Breadth-first search is best implemented using:

a) A stack or recursion.

\*b) A queue.

c) A tree.

4. The goal of a topological sort is to:

a) Sort all of the graph vertices by value.

\*b) Sort all of the graph vertices so that each vertex is listed prior to

any others that depend on it.

c) Sort all of the graph vertices by distance from the source vertex.

5. A topological sort requires all of the following except:

a) The graph be directed.

b) The graph contain no cycles.

\*c) The graph contain weights on the edges.

6. The single-source shortest path problem can be used to:

a) Sort all of the graph vertices by value.

b) Sort all of the graph vertices so that each vertex is listed prior to

any others that depend on it.

\*c) Sort all of the graph vertices by distance from the source vertex.

7. Dijkstra's algorithm requires that vertices be visited in:

a) Depth-first order.

b) Breadth-first order.

\*c) Order of distance from the source vertex.

d) No particular order.

8. In the all-pairs shortest paths problem, a k-path is:

a) The shortest path to vertex k.

b) The sorest path that goes through vertex k.

\*c) A path such that all intermediate vertices have index less than k.

9. ！！For a graph of n nodes, no algorithm to solve the all-pairs shortest paths problem could possibly have a cost less than:

a) Omega(log n)

b) Omega(n)

c) Omega(n log n)

\*d) Omega(n^2)

e) Omega(2^n)

10. Which is a good example of a greedy algorithm?

a) Floyd's all-pairs shortest path algorithm.

\*b) Prim's minimal-cost spanning tree algorithm.

c) Depth-first search.

d) Topological sorting.

**Chapter 12 Lists and Array Revisited: Instructor's CD questions**

1. A Skip List will typically have better performance than a BSTbecause:

a) It is always balanced.

b) The amount of balance in the Skip List is driven by the data

values.

\*c) The amount of balance in the Skip List is driven by random chance independent of the data values.

2. The Skip List is an example of a:

a) Greedy data structure.

\*b) Probabilistic data structure.

c) Direct access data structure.

3. The average access time for the Skip List is:

a) O(1).

\*b) O(log n).

c) O(n).

d) O(n log n).

e) O(n^2).

4. The average number of links store with each Skip List node is:

\*a) O(1).

b) O(log n).

c) O(n).

d) O(n log n).

e) O(n^2).

5. A "pure list" form of a multilist is, structurally, most like a:

a) Linked list.

\*b) Binary tree.

c) Heap.

d) Graph.

e) Directed acyclic graph.

6. A "reentrant list" form of a multilist is, structurally, most like a:

a) Linked list.

b) Binary tree.

c) Heap.

d) Graph.

\*e) Directed acyclic graph.

7. A "cyclic list" form of a multilist is, structurally, most like a:

a) Linked list.

b) Binary tree.

c) Heap.

\*d) Graph.

e) Directed acyclic graph.

8. If a data element requires 4 bytes and a pointer requires 4 bytes,

then an orthogonal list representation will be more space efficient

than a standard array representation when the fraction of non-zero

elements is less than about:

a) 100%

b) 75%

c) 25%

\*d) 15%

e) 5%

f) never

9. External fragmentation occurs when:

\*a) The memory requests create lots of small free blocks, no one of

which is useful for servicing typical requests.

b) More space is allocated to a request than required.

c) The disk has no room for the file.

10. Which of the following is not the name for a standard

sequential-fit technique for allocating space from a memory pool:

a) first fit.

b) best fit.

c) worst fit.

\*d) priority fit.

11. The buddy method creates which form of fragmentation?

\*a) Internal.

b) External.

12. The best-fit method creates which form of fragmentation?

a) Internal.

\*b) External.

**Chapter 13 Advanced Tree Structures: Instructor's CD questions**

1. A trie is different from a BST in that the trie uses:

a) an object space decomposition.

\*b) a key space decomposition.

2. In a trie,

a) The internal nodes and the leaf nodes store data.

b) The internal nodes store data and the leaf nodes are placeholders.

\*c) The leaf nodes store data and the internal nodes direct the search.

3. Which of the following is NOT guaranteed to have height O(log n)

when storing n nodes?

a) AVL tree.

\*b) Splay tree.

c) 2-3 tree.

4. Which of the following is NOT a form of BST?

a) AVL tree.

b) Splay tree.

\*c) 2-3 tree.

d) b and c.

5. Combining multidimensional coordinates into a single key value to

store in a BST would make which operation inefficient?

a) insert.

b) delete.

c) exact-match queries.

\*d) multidimensional range queries.

6. Which of the following is an example of a trie?

a) K-d tree.

b) Point quadtree.

\*c) PR quadtree.

7. Which of the following is a simple variation on a BST?

\*a) K-d tree.

b) Point quadtree.

c) PR quadtree.

d) Bintree.

8. In general, a flyweight is:

a) An empty leaf node.

\*b) An object that is created once and used at many places in the data

structure.

c) An internal node.

9. Which is not a splay tree rotation?

a) single rotation.

b) zigzag rotation.

\*c) clockwise rotation.

10. Which best characterizes the performance of the splay tree?

a) All operations require O(log n) time.

\*b) m operations require a total of O(m log n) time for m > n.

c) All operations require O(n) time.

**Chapter 14 Analysis Techniques: Instructor's CD questions**

1. Amortized analysis is used to:

a) analyze growth rates as the input size becomes large.

\*b) Analyze the cost of a series of operations.

c) Analyze the cost of a problem.

2. Which of the following is not a standard method for finding the

closed form solution for a summation?

a) Guess and test.

b) Shifting.

\*c) Contradiction.

3. Which of the following is not a standard method for finding the

closed form solution for a recurrence relation?

a) Guess and test.

b) Expanding.

\*c) Shifting.

4. In amortized analysis, potential refers to:

\*a) A measure of some resource that can be expended for a series of

operations.

b) The total value that can be computed for the problem.

c) The cost for the algorithm.

5. If we guess a polynomial for the closed form solution to a summation and derive the coefficients for the polynomial, why must we verify the result with an induction proof?

a) Any lower-order polynomial can be made to fit some points of a

higher-order polynomial.

b) Our calculations might be in error.

c) The closed form solution might not be a polynomial at all.

\*d) all of the above.

6. The closed form solution for the recurrence T(n) = 2T(n/2) + n is:

a) Theta(log n)

b) Theta(sqrt(n))

c) Theta(n)

\*d) Theta(n log n)

e) Theta(n^2)

f) Theta(2^n)

7. The closed form solution for the recurrence T(n) = 2T(n/2) + n^2 is:

a) Theta(log n)

b) Theta(sqrt(n))

c) Theta(n)

d) Theta(n log n)

\*e) Theta(n^2)

f) Theta(2^n)

8. The closed form solution for the recurrence T(n) = T(n/2) + sqrt(n)

is:

a) Theta(log n)

\*b) Theta(sqrt(n))

c) Theta(n)

d) Theta(n log n)

e) Theta(sqrt(n) log n)

f) Theta(n^2)

9. The average case for Quicksort is an example of:

a) A divide and conquer recurrence.

\*b) A recurrence with full history.

c) An asymptotic recurrence.

10. The sum of the cubes of the first n integers has as its closed form

solution:

a) Theta(n log n)

b) Theta(n^2)

c) Theta(n^3)

d) Theta(n^3 log n)

\*e) Theta(n^4)

f) Theta(n^4 log n)

**Chapter 15 Limits to Computation: Instructor's CD questions**

1. To find a lower bound for problem A we can:

\*a) Transform problem B with known lower bound to Problem A.

b) Transform problem A to problem B with known lower bound.

2. When doing algorithm analysis, a "hard" problem is one:

a) That we have trouble finding an algorithm for.

b) That we have trouble defining precisely enough to find an algorithm for.

c) That does not have an algorithm.

\*d) For which the best known algorithm requires exponential time.

3. An algorithm is said to be NP if:

a) A random event is involved.

\*b) It can be solved in polynomial time on a non-deterministic machine.

c) There are at most possible 2^n solutions to choose between.

4. What can be NP-complete?

a) An algorithm.

b) A program.

\*c) A problem.

d) An input.

5. Assume that we know that problem A is NP-complete. We can prove

that problem B is NP-complete by:

\*a) transforming problem A to problem B.

b) transforming problem B to problem A.

6. A practical reason to be familiar with NP-completeness theory is

that:

a) It helps you to solve hard problems in polynomial time.

b) It helps you to write better algorithms.

\*c) It helps you to justify why a given program is slow.

7. Some computer programs could be made much faster if we could prove:

\*a) That problem class P is equal to problem class NP.

b) That problem class P is not equal to problem class NP.

c) All NP-complete problems can be solved quickly on a non-deterministic

machine.

8. Which problem has not been proved to be NP-complete?

a) TRAVELING SALESMAN

b) CLIQUE

c) VERTEX COVER

\*d) TOWERS OF HANOI

9. Which set is uncountable?

a) Integers.

b) Positive integers.

\*c) Real numbers.

d) All of the above.

10. Which set is countable?

\*a) Programs.

b) Functions that take an integer as input and gives an integer as

output.

c) Functions that take an integer as input and gives a boolean value

as output.

d) a and c.