THE UNIVERSITY OF ADELAIDE
AUSTRALIA

**Primary Examination, Semester 2, 2012**

# Algorithm and Data Structure Analysis
# COMPSCI 2201, 7201

| | |
|---|---|
| Official Reading Time: | 10 mins |
| Writing Time: | 120 mins |
| Total Duration: | 130 mins |

| Questions | Time | Marks |
|---|---|---|
| Answer all 7 questions | 120 mins | 120 marks<br>120 Total |

Instructions

- Begin each answer on a new page
- Examination material must not be removed from the examination room
- No Calculators Allowed

Materials

- 1 Blue book
- 1 Dictionary for translation purposes only

DO NOT COMMENCE WRITING UNTIL INSTRUCTED TO DO SO

**Right or Wrong?**

**Question 1**

(a) Indicate whether each of the following statements is true or false. There is one mark for each correct answer and zero marks for each incorrect answer.

| | Statement |
|---|---|
| 1 | $n^{7/8} = \omega(n)$ |
| 2 | $2.1 \cdot n^{2.1} + 120n^2 = O(n^3)$ |
| 3 | School Integer Multiplication takes time $\Theta(n^2)$. |
| 4 | The all-pairs shortest path problem is currently solvable on any graph in $O(n)$ time. |
| 5 | It is known that there are problems in $NP$ that are not in $P$ |
| 6 | The Boolean Satisfiability Problem is NP-Hard |
| 7 | Every problem that is NP-Hard is also known to be NP-Complete |
| 8 | AVL trees can degenerate to a worst case $\Theta(n)$ time for *find* operations. |
| 9 | Dijkstra's single-source shortest path algorithm cannot work on graphs containing negative-weight cycles. |
| 10 | Exponential search is useful when the length of its input list is not known. |

[10 marks]

**[Total for Question 1: 10 marks]**

**Sequences**

**Question 2**

(a) Assume you have an array-based binary heap: $a$. with the contents:

$$1, 4, 7, 8, 9, 10, 14, 12, 15, 13, 17, 12$$

Show the contents of $a$ *after* each of the following two operations. *Show your working* for each operation including the content of the list at its intermediate stages. You can assume $a$ is large enough to contain all the values inserted.

- insert 5 into $a$
- deleteMin of $a$

[6 marks]

(b) The following questions relate to implementations of unbounded arrays. Read both parts before answering either of them.

i. Describe an implementation of an unbounded array that has amortised cost of $O(1)$ on both *pushBack* and *popBack* operations. In your description make sure you say how and when the necessary allocations and deallocation operations are performed.

[5 marks]

ii. Use *amortised analysis* to demonstrate that *pushBack* and *popBack* can be performed on the structure you described in part (i) above at an amortised cost of $O(1)$.

[6 marks]

**[Total for Question 2: 17 marks]**

**Hashing and Skiplists**

**Question 3**

(a) The next two questions relate to the following hashing setup:

- table size is 11 elements.
- hash keys are lower-case alphabetic strings.
- the hash function is

$$h(k) = code(lastLetter(k)) \bmod 11$$

Where $lastLetter$ extracts the last letter from its input and $code$ returns an integer representing the position of the letter in the alphabet (starting at zero). So, for example $h(\text{anna})$ returns $0$, $h(\text{mob})$ returns $1$ and $h(\text{noon})$ returns $2$.

- collisions are resolved through chaining.

Answer the following:

i. Draw the hashtable described above after the insertion of the elements:

```
"lake" "fuss" "end" "tent" "sense" "cats" "mop"
```

[6 marks]

ii. Is $h$ described above a very good hash function? Briefly justify your answer.

[2 marks]

(b) Briefly describe the difference between open address hashing and closed address hashing.

[2 marks]

(c) Given the following sequence of coin tosses, where H stands for head and T stands for tails:
H T H T T H T T H T T T H H T H T T T T H T H T T H T T T H T H T.
Build a skip list containing the following values: 18 7 80 5 27 14. Show the full state of the skip list after each insertion. **Hint:** explicitly state the meaning you attach to heads and tails at the start of your answer.

[8 marks]

**[Total for Question 3: 18 marks]**

**Trees**

**Question 4**

(a) What is the average cost of *find* on a binary search tree generated by the insertion of the elements:

$$9, 7, 12, 10, 3, 11, 6, 4, 8$$

into an empty tree. Show your working.

[5 marks]

(b) Write down six insertion orders of the numbers in the set:

$$1, 2, 3, 4$$

That will create a degenerated binary search tree. **Hint** a degenerated binary search tree is one of maximal depth for its number of elements.

[6 marks]

(c) Prove by induction that a tree with n nodes has n-1 edges.

[6 marks]

(d) Draw a sequence of diagrams showing the insertion of the values:

```
[13,11,9,27,19,21,3]
```

into an empty AVL tree.

You must:

- Show the resulting tree immediately after each insertion step (that is *before* any balancing has taken place).
- Indicate the node(s) at which each rotation is performed.
- Where there is a double rotation, show the tree after each single rotation.
- Show the resulting tree after balancing operation(s).

[7 marks]

**[Total for Question 4: 24 marks]**

**Graph Representations and Traversals**

**Question 5**

(a) Write pseudo-code for an algorithm that performs depth first search of a directed graph: $G = (V, E)$ and prints out all of the edges visited.

[10 marks]

(b) The following questions relate to traversals of directed graphs

i. The graph-search algorithms shown in lectures always keep track of visited nodes because failing to to do so can result in non-termination. Draw a graph that will lead to non-termination of Depth-First-Search if we fail to keep track of nodes already visited.

[3 marks]

ii. Under what conditions can the Single-Source-Shortest-Path problem be solved using Breadth-First-Search?

[2 marks]

(c) When is it advantageous in terms of memory-use to store a graph as an adjacency list rather than an adjacency matrix? Briefly justify your answer.

[2 marks]

**[Total for Question 5: 17 marks]**

**Shortest Path Algorithms**

**Question 6**

(a) Dijkstra's algorithm for single-source shortest path on directed graphs, as given in lectures, assumed a heap-based priority queue was used to store tentative unscanned node distances. However, the original version of Dijkstra's algorithm used an *array* to store these distances.

Give the time complexity of this original version of Dijkstra's algorithm taking account of the cost of array access. Give a brief explanation of your answer.

[5 marks]

(b) **Important**: read this **whole** question before starting it - it will save you time.

Write a Java method called `bellmanFord` with the signature:

```
public static double[] bellmanFord(Graph g, int s)
```

that returns an array representing the minimum distances of the nodes of `g` from the source node indexed by `s`. Your method must use the Bellman-Ford single-source shortest-path algorithm to generate its result. Note, you do **not** have to update an array of parent nodes on the path in your code, only the array of minimum distances is required by this question. Assume the following methods are available on graphs:

- `getEdges()` which returns an array of `Edge` representing the edges in the graph.
- `getNumberOfNodes()` which returns an integer representing the number of nodes in the graph.

Additionally every `Edge` object has the methods:

- `getSource()` which returns an integer representing the index of the source node of this edge in the graph.
- `getDest()` which return an integer representing the index of the target node of this edge in the graph.
- `getWeight()` which returns the weight (cost) of this edge.

**Important**: when writing your algorithm you may assume that there can be negative weight edges in `g` **but** there are **no negative weight cycles** in `g`. This means that you do **not** need to implement the stage of the algorithm that infects the graph beyond negative weight cycles with $-\infty$. In your solution you must define any helper methods (besides those mentioned above) that you use.

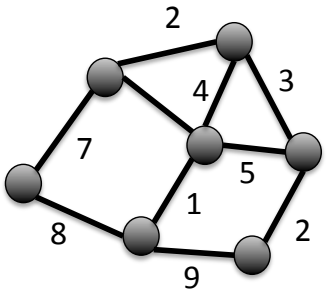
[18 marks]

**[Total for Question 6: 23 marks]**

**Minimum Spanning Trees and P vs NP**

**Question 7**

(a) Use Kruskal's algorithm to derive a minimum spanning tree for the graph below.




In your answer show your working and draw the final tree including the weights on the links of the tree.

[6 marks]

(b) What does it mean to say that a problem is not in NP? Give an example of a problem that is not in NP.

[4 marks]

(c) What was the first problem shown to be NP complete?

[1 mark]

**[Total for Question 7: 11 mark]**

**End of exam**