Carleton UNIVERSITY

Sample Solution to FINAL
EXAMINATION
Winter 2015

DURATION: 3 HOURS **No. Of Students: 99**

Department Name & Course Number: **Systems and Computer Engineering**
SYSC 2100B

Course Instructor (s): Thomas Kunz

AUTHORIZED MEMORANDA
Course textbook: *Data Abstraction and Problem Solving with Java: Walls and Mirrors, 3rd edition,* by Frank M. Carrano and Janet J. Prichard. Prentice Hall 2011 (either the original textbook, and earlier version, or the custom-tailored textbook from the publisher and sold through the bookstore for the 2013 version of the course)

Students MUST count the number of pages in this examination question paper before beginning to write, and report any discrepancy to a proctor. This question paper has ____11____ pages + cover page = _12_ pages in all.

This examination question paper may not be taken from the examination room.

In addition to this question paper, students require: an examination booklet no
Scantron Sheet no

Name: _____

Student Number: _____

NOTE: Questions 4 and 7 may be more difficult, so you may want to address them last.

Exam questions will not be explained, and no hints will be given. If you think that something is unclear or ambiguous, make a reasonable assumption (one that does not contradict the question), write it at the start of the solution, and answer the question. Do not ask questions unless you believe you have found a mistake in the exam paper. If there is a mistake, the correction will be announced to the entire class. If there is no mistake, this will be confirmed, but no additional explanation of the question will be provided.

Question 1: Linked Lists (10 marks)

Assume we had a single linked list with no dummy entry. The list is pointed to by head and is maintained in sorted order. Right now, the list has the following content:

```
head -> J -> K -> M -> P.
```

For each of the following, write the Java statements that perform the requested operation on the list and draw a picture of the status of the list after each operation is complete. When you delete a node from the list, make sure it will eventually be returned to the system. All insertions into the list should maintain the list's sorted order. Do not simple invoke any of the list methods presented in class/the course textbook, but write the statements that work directly with the underlying list representation.

1. Assume that `prev` references the first node and `curr` references the second node. Insert `L` into the list.

   Answer (4 marks):

```
// Need to advance prev and curr because they now point to J and K,
// and we need to put L after K.  In other words, we want L between
// K and M, not between J and K.
curr = curr.next;
prev = prev.next;

// To insert L in the list means that it will be followed by curr.
newNode = new Node('L', curr);
prev.setNext(newNode);

// Resulting list:  head --> J --> K --> L --> M --> P
```

2. Assume that `prev` references the second node and that `curr` references the third node of the list after you revised it in Part 1). Delete the last node of the list.

   Answer (3 marks):

```
// curr now points to L.  We want to delete P.  We do this by setting
// M's next to null.  And note that M is the next of L (i.e. M is curr.next).
curr.next.setNext(null);

// Resulting list:  head --> J --> K --> L --> M
```

3. Assume that `prev` references the last node of the list after you revised it in Part 2), and assume that `curr` is `null`. Insert `Q` into the list.

   Answer (3 marks):

```
// prev points to M and next points to null.  Insert Q at the end.
newNode = new Node('Q', null);
prev.setNext(newNode);

// Resulting list:  head --> J --> K --> L --> M --> Q
```

Question 2: Recursion and Grammars (10 marks)

1. Is +A/-*BC+DEF a prefix expression? Explain in terms of the grammar for a prefix expression (Section 6.2 in the 3rd edition of the textbook).

Answer (5 marks):
Let S = "+A/-*BC+DEF". There are only two rules for creating a prefix expression: <identifier> and <operator><prefix><prefix>. Since S is clearly not just an identifier, we must investigate the use of the recursive rule one or more times. Let's assume S is a prefix expression. We will work backwards to see how it came about using the grammar rules, or we will arrive at a contradiction.

S begins with an operator, so the string is of the form $+E_1E_2$ where E_1 and E_2 must also be prefix expressions. We continue by decomposing E_1 and E_2.

E_1 begins with A. This is a base case because A is an identifier, not an operator. Therefore, the rest of the string, /-*BC+DEF, corresponds to E_2.

E_2 begins with the operator /.
$E_2 = /E_3E_4$

E_3 begins with the operator -.
$E_3 = -E_5E_6$

E_5 begins with the operator *.
$E_5 = *E_7E_8$

At this point we have resolved E_5 = *BC. Let's continue determining what E_6 is. E_6 begins with the operator +.

$E_6 = +E_9E_{10}$

E_9 begins with the identifier D. This must be followed by E_{10}, which is the next symbol: the identifier E.

At this point we have resolved E_6 = +DE, and therefore E_3 = -*BC+DE.

E_3 is followed by E_4, and the next symbol is the identifier F, so E_4 = F. This is the last character in S.

Now we have resolved E_2 = /-*BC+DEF. We see that we can compose the original expression $+E_1E_2$.

Thus, we can conclude that S is a prefix expression.

2. Is ABC*+DEF/GH-* a postfix expression? Explain in terms of the grammar for a postfix expression.

Answer (5 marks):
Let S = "ABC*+DEF/GH-*". There are only two rules for creating a postfix expression: <identifier> and <postfix><postfix><operator>. Since S is not just an identifier, it must have been created through the use of one or more applications of the recursive rule. Let's assume S is a postfix expression. We will work backwards to see how S was formed from the rules of postfix expressions, or we will arrive at a contradiction.

Before we begin, we may notice that there are only 5 operators and 8 identifiers, so we know something will go wrong along the way as we unwind the recursion. (There should be exactly 1 more identifier, not 3.)

S ends with *.
$S = E_1E_2*$

E_2 ends with -.
$E_2 = E_3E_4-$

E_4 ends with the identifier H, so $E_4 = H$. Let's next consider E_3. It ends with G, so $E_3 = G$.

At this point, we have resolved $E_2 = GH-$. Let's next consider E_1.

E_1 ends with /.
$E_1 = E_5E_6/$

E_6 ends with the identifier F, so $E_6 = F$. Next we consider E_5, which ends with E, so $E_5 = E$.
At this point, we have have resolved E1 = EF/. And therefore, $S = E_1E_2* = EF/GH-*$.

But this last equation is a contradiction because we started with S = ABC*+DEF/GH-*. Thus, S is not a postfix expression.

Question 3: Algorithm Complexity (10 marks)

1) If a problem of size n requires time that is directly proportional to n, the problem is _____.
    a) O(1)
    b) O(n)
    c) $O(n^2)$
    d) O(2n)

    Answer (1 mark): b.

2) The value of which of the following growth-rate functions grows the fastest?
    a) O(n)
    b) $O(n^2)$
    c) O(1)
    d) $O(\log_2 n)$

    Answer (1 mark): b.

3) Which of the following growth-rate functions indicates a problem whose time requirement is independent of the size of the problem?
    a) O(n)
    b) $O(\log_2 n)$
    c) $O(2^n)$
    d) O(1)

    Answer (1 mark): d.

4) In the best case, a sequential search is _____.
    a) O(n)
    b) O(1)
    c) $O(\log_2 n)$
    d) $O(n^2)$

    Answer (1 mark): b.

5) In the worst case, a binary search is _____.
    a) O(n)
    b) O(1)
    c) $O(\log_2 n)$
    d) $O(n^2)$

    Answer (1 mark): c.

6) The selection sort is continued until _____ of the n items in an array have been swapped.
    a) n/2
    b) n – 2
    c) n – 1
    d) n

    Answer (1 mark): c.

7) The _____ compares adjacent items and exchanges them if they are out of order.
    a) selection sort
    b) binary search
    c) bubble sort
    d) quicksort

    Answer (1 mark): c.

8) A bubble sort requires at most _____ passes to sort an array of n items.
   a) n/2
   b) n – 2
   c) n – 1
   d) n

Answer (1 mark): c.

9) In the worst case, the insertion sort's comparison occurs _____ times.
   a) n
   b) n – 1
   c) (n – 1) / 2
   d) n * (n – 1)/2

Answer (1 mark): d.

10) The quicksort is _____ in the worst case.
   a) $O(n^2)$
   b) $O(n^3)$
   c) $O(n * \log_2 n)$
   d) $O(\log_2 n)$

Answer (1 mark): a.

Question 4: Radix Sort (10 marks)

The textbook discusses radix sort by sorting a hand of cards by first ordering the cards by rank and then by suit. To implement a radix sort for this example, you could use two characters to represent a card if you use T to represent a 10. For example, S2 is the 2 of spades and HT is the ten of hearts. Further, assume that numbers are always less than characters, and characters are sorted according to the following card-game conventions: T < J < Q < K and D < H < S < C

1. Show a trace of the radix sort for the following cards: S2, HT, D6, S4, C9, CJ, DQ, ST, HQ, DK

   Answer (5 marks):

S2, HT, D6, S4, C9, CJ, DQ, ST, HQ, DK	original cards
(S**2**) (S**4**) (D**6**) (C**9**) (H**T**, S**T**) (C**J**) (D**Q**, H**Q**) (D**K**)	sorted by rank
S2, S4, D6, C9, HT, ST, CJ, DQ, HQ, DK	combined
(**D**6, **D**Q, **D**K) (**H**T, **H**Q) (**S**2, **S**4, **S**T) (**C**9, **C**J)	sorted by suit
D6, DQ, DK, HT, HQ, S2, S4, ST, C9, CJ	combined (sorted)

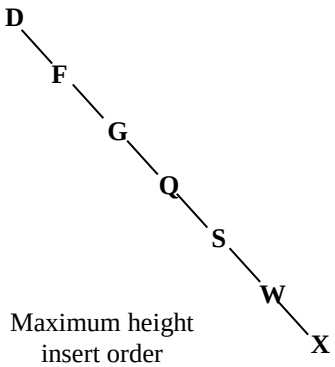
2. Suppose that we do not use T to represent a 10 – that is, suppose that H10 is now the 10 of hearts. We also padded all other two-character strings on the right with a blank to form a three-character string. How would radix sort order the entire deck of cards in this case?

   Answer (5 marks):
   All 10's would become sorted before all 2's in rank because of the 0 in the '10', then the 10's would get sorted before or after all of the suits, depending on the precedence accorded the '1' in the '10' with respect to S, H, C, and D.

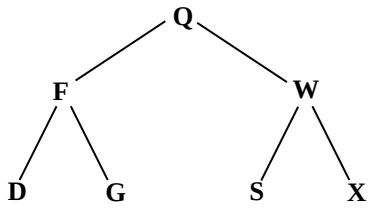
Question 5: Binary Trees (10 marks)

1. Arrange nodes that contain letters D, Q, X, F, S, G, and W into two binary search trees, one that has maximum height and one that has minimum height.

   Answer (5 marks):



Maximum height
insert order
DFGQSWX



Minimum height
insert order
QFWDGSX

2. What is the maximum number of nodes in a binary tree with 8 levels?

   Answer (1 mark):
   If a binary tree has 8 levels, then the maximum number of nodes on each level is 2^0, 2^1, 2^2, ..., 2^7. The total number of nodes in the tree is $2^8 - 1$.

3. What is the maximum and minimum number of levels of a tree with 2011 nodes?

   Answer (2 marks):
   If we have 2,011 nodes in a binary tree, then to maximize the number of levels, we could put one node in each level. This would result in 2,011 levels.

   To minimize the number of levels, we would have to place the maximum numer of nodes at each level. The question then becomes how many levels we need. For n levels, we can have up to $2^n - 1$ nodes. What is the smallest value of n for which $2^n - 1 \geq 2011$? We note that if n = 11, $2^n - 1 = 2047$. So the minimum number of levels is 11.

4. What is the maximum and minimum number of leaves in a tree with 10 levels?

   Answer (2 marks):
   If a tree has 10 levels, we could have 1 node per level: this would result in having just 1 leaf.
   To maximize the number of leaves, we would want the 10^{th} level to be completely full. It would have $2^9 = 512$ leaves.

Question 6: Binary Search Trees/Balanced Trees/Tables (10 marks)

1) In the binary search tree implementation of the ADT table, the maximum number of comparisons required by the `tableInsert` operation is equal to _____.
    a) the number of nodes in the binary search tree
    b) the height of the binary search tree
    c) the number of leaves in the binary search tree
    d) the number of internal nodes in the binary search tree

Answer (1 mark): b.

2) The maximum height of a binary search tree of n nodes is _____.
    a) n
    b) n – 1
    c) n / 2
    d) $\log_2(n + 1)$

Answer (1 mark): a.

3) A(n) _____ is a tree in which each internal node has either two or three children, and all leaves are at the same level.
    a) red-black tree
    b) 2-3 tree
    c) 2-3-4 tree
    d) AVL tree

Answer (1 mark): b.

4) In a 2-3 tree, _____.
    a) all internal nodes have 2 children
    b) all internal nodes have 3 children
    c) all leaves are at the same level
    d) all nodes have 2 data items

Answer (1 mark): c.

5) A node that contains two data items and has three children is called a _____.
    a) 2-node
    b) 3-node
    c) double node
    d) triple node

Answer (1 mark): b.

6) If a particular 2-3 tree does NOT contain 3-nodes, it is like a _____.
    a) general tree
    b) binary search tree
    c) full binary tree
    d) complete binary tree

Answer (1 mark): c.

7) A 2-3 tree of height h always has at least _____ nodes.
    a) h^2
    b) $h^2 – 2$
    c) 2^h
    d) $2^h – 1$

Answer (1 mark): d.

8) In a 3-node, _____.
   a) the left child has the largest search key
   b) the middle child has smallest search key
   c) the middle child has the largest search key
   d) the right child has the largest search key

Answer (1 mark): d.

9) In a 2-3 tree, a leaf may contain _____ data item(s).
   a) one
   b) one or two
   c) two
   d) two or three
   e) three

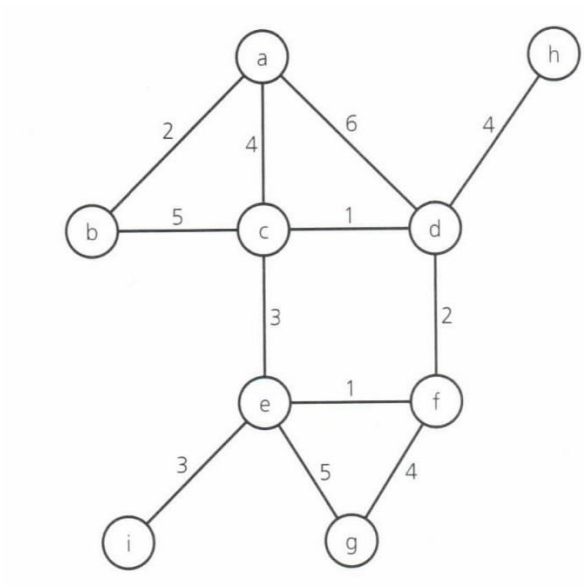
Answer (1 mark): b.

10) A 2-3 implementation of a table is _____ for all table operations.
   a) O(n)
   b) $O(\log_2 n)$
   c) $O(\log_2 n * n)$
   d) $O(n^2)$

Answer (1 mark): b.

Question 7: Graphs (10 marks)

1. Use both the depth-first and the breadth-first strategy, traverse the graph below starting with vertex b. List the vertices in the order in which each traversal visits them. As usual, when a node has multiple (unvisited) neighbors they are visited in lexicographical order (i.e., neighbor a first, neighbor z last).

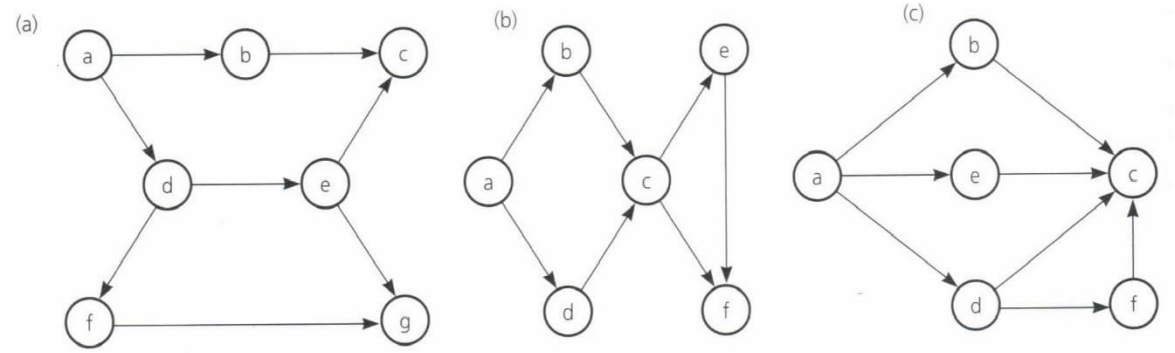


Answer (4 marks):

depth-first search: b, a, c, d, f, e, g, i, h.

breadth-first search: b, a, c, d, e, f, h, g, i.

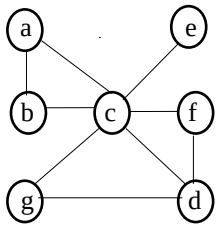
2. For the three graphs below, write the topological order of the vertices that results from algorithm `topSort1`.



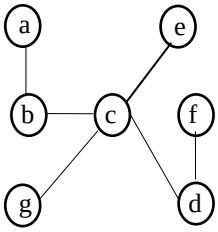
Answer (3 marks):
The strategy is to find a vertex in the graph that does not have a successor, and prepend it to the output list of vertices. A number of permutations are possible, as often there is more than one candidate node. Here I only list some one of the possible examples for each graph (here we take the lexicographically later node first):

Graph a:	a, b, c, d, e, f, g
Graph b:	a, b, d, c, e, f
Graph c:	a, b, d, e f, c

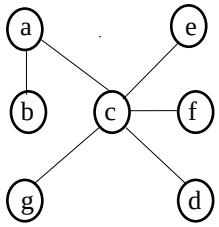
3. Draw the DFS and BFS spanning trees rooted at `a` for the graph below. Then draw the minimum spanning tree rooted at `a` for this graph.



Answer (3 marks):



A DFS spanning tree



A BFS spanning tree

Since the graph is not weighted, any spanning tree will be a minimum spanning tree.