Carleton UNIVERSITY

SAMPLE SOLUTION TO FINAL
EXAMINATION
FALL 2006

DURATION: 3 HOURS **No. Of Students: 20**

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

Course Instructor (s): Thomas Kunz

AUTHORIZED MEMORANDA
NONE

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 ____7____ pages + cover page = _8__ 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: _____

Question 1: Recursion (10 marks)

Recursively define the following types of binary trees, based on the non-recursive definitions used in class:

1) Full binary tree

   Answer (3 marks):

   Tree is empty OR

   Tree's left subtree has the same height as tree's right subtree and both subtrees are full binary trees

2) Complete binary tree

   Answer (3 marks):

   Tree is empty OR

   If Tree's left subtree has the same height as tree's right subtree

   Then left subtree has to be a full binary tree and right subtree has to be a complete binary tree

   If left subtree's height is one greater than right subtree's height

   Then left subtree has to be a complete binary tree and right subtree has to be a full binary tree

3) Balanced binary tree

   Answer (4 marks):

   Tree is empty OR

   Height of left subtree and height of right subtree differ by no more than one and both subtrees are balanced binary trees

Question 2: Linked List (10 marks)

1) The last node of a linear linked list _____.
   a) has the value `null`
   b) has a `next` reference whose value is `null`
   c) has a `next` reference which references the first node of the list
   d) cannot store any data

   Answer (1 mark): b

2) Which of the following will be true when the reference variable `curr` references the last node in a linear linked list?

```
a) curr == null
b) head == null
c) curr.getNext() == null
d) head.getNext() == null
```

   Answer (1 mark): c

3) If a linked list is empty, the statement `head.getNext()` will throw a(n) _____.

```
a) IllegalAccessException
b) ArithmeticException
c) IndexOutOfBoundsException
d) NullPointerException
```

   Answer (1 mark): d

4) What are two advantages of using a reference-based implementation of the ADT list instead of an array-based implementation?

   Answer (2 marks):
   First, a reference-based implementation does not shift items during insertion and deletion operations. Second, a reference-based implementation does not impose a fixed maximum length on the list (or alternatively: allocates just enough memory to hold all items in the current list).

5) Write the code fragment to delete the node that the reference variable `curr` references in a circular doubly linked list? Each node supports the methods `setNext(),getNext(),getPrecede()` and `setPrecede()`.

   Answer (2 marks):

```
curr.getPrecede().setNext(curr.getNext());
curr.getNext().setPrecede(curr.getPrecede());
```

6) Write the code fragment to insert a new node that the reference variable `newNode` references before the node referenced by the reference variable `curr` in a doubly linked list, using the methods listed in part 5).

   Answer (3 marks):

```
newNode.setNext(curr);
newNode.setPrecede(curr.getPrecede());
curr.setPrecede(newNode);
newNode.getPrecede().setNext(newNode);
```

Question 3: Algorithm Complexity (5 marks)

1) Assuming a linked list of n nodes, the code fragment:

```
Node curr = head;
while (curr != null) {
   System.out.println(curr.getItem());
   curr.setNext(curr.getNext());
NOTE: this is wrong, should be curr = curr.getNext() otherwise we deal with
an infinite loop (some students noticed that, and I accepted answers of
INFINITE for parts 1, 2, and 3).
}  // end while
```

requires _____ assignments.

a) n
b) n – 1
c) n + 1
d) 1

Answer (1 mark): c (don't forget to count initial assignment PLUS one to get the NULL value)

2) Assuming a linked list of n nodes, the above code fragment (in part 1))
requires _____ comparisons.

a) n
b) n – 1
c) n + 1
d) 1

Answer (1 mark): c

3) Assuming a linked list of n nodes, the above code fragment (in part 1))
requires _____ write operations.

a) n
b) n – 1
c) n + 1
d) 1

Answer (1 mark): a

4) Consider an algorithm that contains loops of the form:

```
for (x = 1 through n ) {
  for (y = 1 through x) {
    for (z = 1 through 10) {
      Task T
    }  // end for
  }  // end for
}  // end for
```

If task T requires t time units, the innermost loop on z requires _____ time units.

a) y
b) 10
c) z * t
d) 10 * t

Answer (1 mark): d

5) Consider the above algorithm (part 4) again.
If task T requires t time units, the loop on y requires _____ time units.

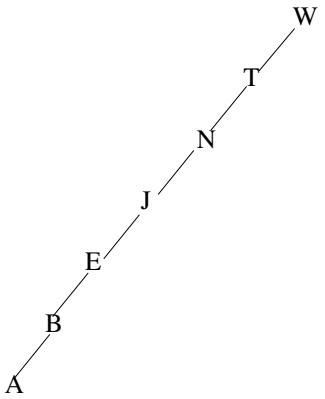
a) 10 * t
b) (10 * t) + x
c) 10 * t * x
d) t * x

Answer (1 mark): c

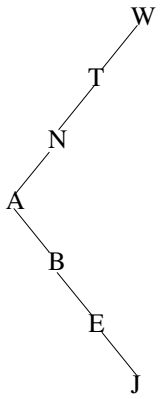
Question 4: Binary Trees (10 marks)

1) Beginning with an empty binary search tree, what binary search tree is formed when you insert the following values in the order given?
   a. W, T, N, J, E, B, A
   b. W, T, N, A B, E, J
   c. A, B, W, J, N, T, E

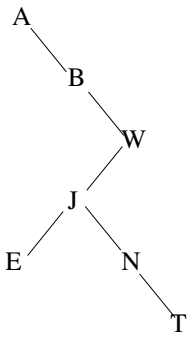
Answer (3 marks):



a) insert order
W T N J E B A



b) insert order
W T N A B E J



c) insert order
A B W J N T E

2) Write pseudocode for a method that performs a range query for a binary search tree. That is, `RangeQuery(tree, low, high)` should visit all items/nodes in `tree` that have a search key `k` with `low <= k <= high`. To indicate the a specific node `x` is being visited (printed, modified, …), simply use the pseudocode instruction `visit(x)`.

Answer (3 marks):

```
public rangeQuery(tree, low, high)
{      if (tree is not empty)
       {      if (low < tree's root item)
                     rangeQuery (tree's left subtree, low, high)
              else if ((low <= tree's root item) and
                            (tree's root item <= high))
                     visit (tree's root item);
              else if (high > tree's root item)
                     rangeQuery (tree's right subtree, low, high)
       } // end if
} // end rangeQuery
```

3) Proof by induction that a binary tree with *n* nodes has exactly *n+1* empty subtrees (or, in Java terms, *n+1 null* references in a reference-based implementation)

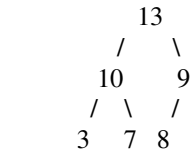
Answer (4 mark):

Base: A binary tree with one node has two empty subtrees.

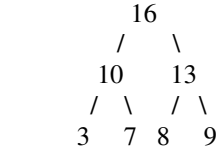
Induction: Assume that a binary tree with N–1 nodes has N empty subtrees. If we add a node to this tree, we will replace one of the N empty subtrees with a node, but we will add two empty subtrees (the subtrees of the added node). Therefore, the tree (with N nodes) will have N–1+2 = N+1 empty subtrees.

Question 5: Heaps/Priority Queues (10 marks)

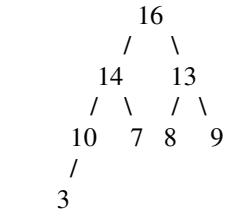
1) Given the following `maxheap h`, show what the heap would look like after each of the following pseudocode operations:



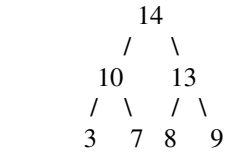
`a) h.heapInsert(16)`
Answer (2 marks):



`b) h.heapInsert(14)`
Answer (2 marks):



`c) h.heapDelete()`
Answer (2 marks):



2) Does the order in which you insert items into a heap affect the heap that results? Explain.
Answer (2 marks):

The order of item insertion does not affect the shape of the resultant binary tree; it does affect the distribution of values in the individual nodes.

3) Suppose that after you have placed several items into a priority queue, you need to adjust one of the priority values. For example, a particular task in a priority queue of tasks could become more or less urgent. How can you adjust the heap if a single priority value changes? Note: your solution should be better than "remove the item and re-insert it" as we have no operation to delete arbitrary items from a heap.

Answer (2 marks):
If the priority value increases, trickle the item up (as in insert). If the priority value decreases, trickle it down (as in adjust).

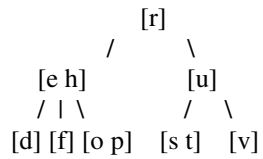
Question 6: Balanced Trees/Tables (10 marks)

1) What are the advantages of implementing the ADT table with a 2-3 tree instead of a binary search tree? Why do you not, in general, maintain a completely balanced binary search tree?

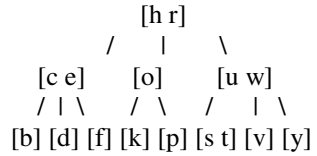
Answer (2 marks):

The height of a binary search tree depends on the order in which the items are inserted. In a tree with N items, the height can range between N (equivalent to a linked list) and $\log_2 (N+1)$ (a fully balanced tree). A 2-3 tree is always balanced and thus has height proportional to log N. Since most table operations take time proportional to the height of the tree, a 2-3 tree is a more efficient ADT table implementation than a binary search tree. Maintaining a balanced binary search tree may become very expensive in the face of frequent inserts and deletions as the entire tree must be rebuilt in the worst case

2) Given the 2-3 tree below, draw the tree that results after inserting k, b, c, y, and w into the tree.



Answer (6 marks):



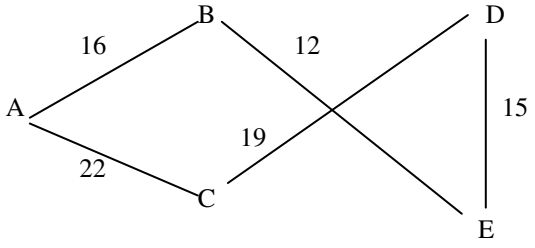
3) Write pseudocode for the *tableDelete* operation when the implementation uses hashing and linear probing to resolve collisions.

Answer (2 marks):
The following pseudocode implements operations on a table which uses hashing and linear probing to resolve collisions:

```
tableDelete(searchKey)

   i = hashIndex(searchKey)

   if (t[i].key != searchKey)
   {   do
         i = (i + 1) mod tableSize
       while (t[i].key != searchKey and i != hashIndex(searchKey))
   }

   if (t[i].key == searchKey)
   {  delete t[i]
      operation is successful   }
   else
      operation is not successful
```

Question 7: Graphs (10 marks)

1) Use both the depth-first and the breadth-first strategy to traverse the graph below, beginning with vertex A. List the vertices in the order in which each traversal visits them. If a node has multiple neighbors, assume that they are visited in alphabetical order



Answer (2 marks):

Breadth-First Strategy: A, B, C, E, D
Depth-First Strategy: A, B, E, D, C

2) By modifying the DFS traversal algorithm, write pseudocode for an algorithm that determines whether a graph contains a cycle.

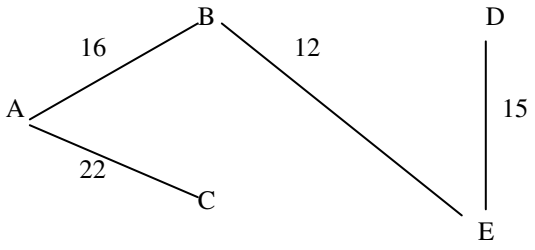
Answer (2 marks):

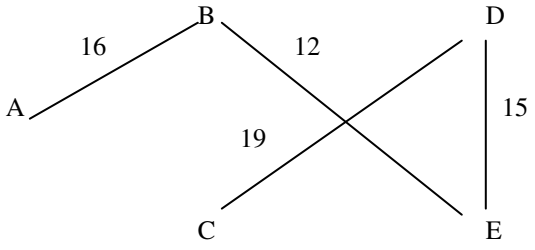
Within the while loop, insert the following just before the first if statement:

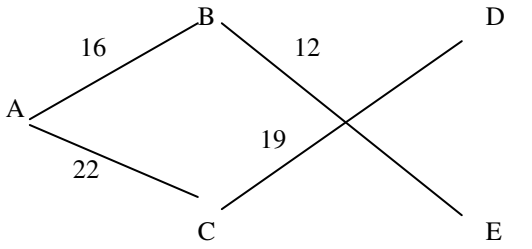
```
if (there is a visited vertex adjacent to the vertex on top of the stack)
   report a cycle
```

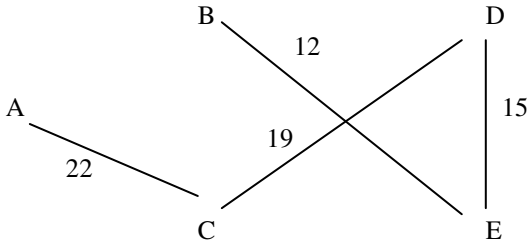
3) For the graph above, draw all possible spanning trees. Which one is the minimum spanning tree?

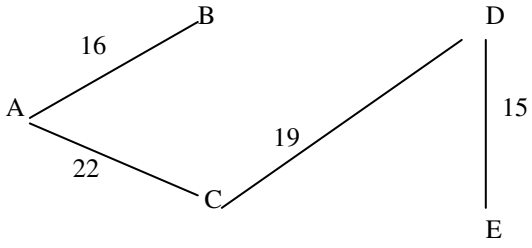
Answer (6 marks):











Minimum Spanning Tree: