Name: _____

ID Number: _____

UNIVERSITY OF VICTORIA
EXAMINATIONS- DECEMBER 2010
CSC 225 - Algorithms and Data Structures: I
Section A01 (CRN 10849)
Instructor: Wendy Myrvold
Duration: 3 hours

TO BE ANSWERED ON THE PAPER.

Instructions:

Students **MUST** count the number of pages in this examination paper before beginning to write, and report any discrepancy immediately to the invigilator.

This exam has ten pages (the last page is blank in case you need extra space) plus the header page.

Use only space provided on exam for answering questions. Closed book. No aids permitted.

Question	Value	Mark
1	12	
2	8	
3	20	
4	10	
5	20	
6	15	
7	15	
Total	**100**	

1. Circle true or false for each question and justify your answer. **No marks will be given unless there is a correct justification.**

(a) [3 marks] For a fixed integer k, the sum $S(n) = \sum_{i=1}^{n} i^k$ is in $O(n^k)$.

True **False**

(b) [3 marks] A search for a data item in a binary search tree can take $\Omega(n)$ time in the worst case.

True **False**

(c) [3 marks] It takes at least $O(n \log_2 n)$ time in the worst case to build a heap because both bubble up and bubble down take $\Omega(\log_2 n)$ time in the worst case.

True **False**

(d) [3 marks] Suppose singly linked lists are used to implement a queue. Then, adding to the queue should be done at the front of the list, and deletions from the queue should be taken from the rear of the list.

True **False**

2. [8 marks] Prove that if $2^k \leq n < 2^{k+1}$, $k \geq 0$, then $(k+1)\,2^{k+1} \in O(n \log_2 n)$.

3. [20 marks, -2 marks for each incorrect answer] Your job is to analyse various data structures for implementing a set. The sets under consideration can contain up to n elements represented by the integers from zero up to $n - 1$. Consider the following data structures for a set S:

   (a) The elements of set S are stored in a sorted array.

   (b) The elements of set S are stored in an unsorted linked list. **Important: the linked list should not contain any duplicate values.**

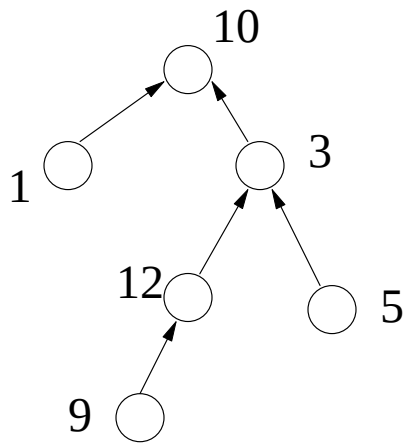
   (c) The elements of S are indicated by a *characteristic vector* which is an array of size n which has position i set to 1 if element i is in the set and 0 if it is not in the set.

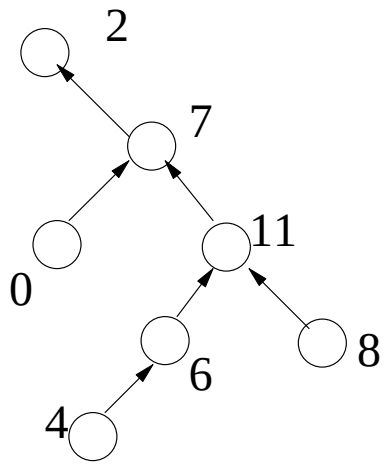
Let S be a set which has size s and let T be a set of size t. Give the **worst case** time complexities of algorithms for the following set operations as functions of n, s, and t.

If you want to use a sorting routine, use mergeSort (and not Radix sort).

Operation	(a) Sorted array	(b) Unsorted list	(c) Characteristic vector
Is x in S?			
Add x to S (do not count time to check if x in S)			
Print elements in S in sorted order			
Create union of S and T			

4. Weighted UNION/Collapsing FIND





(a) [6 marks] Show the updated data structure after UNION(9, 6) assuming that weighted UNION and collapsing FIND are used.

(b) [2 marks] What does the parent array contain before the update?

0	1	2	3	4	5	6	7	8	9	10	11	12

(b) [2 marks] What does the parent array contain after the update?

0	1	2	3	4	5	6	7	8	9	10	11	12

5. Suppose a max-heap of size n is stored in an array $A[0..n-1]$.

(a) [6 marks] Give the formulas for determining the parent, left child and right child for the node in $A[i]$.

parent	left child	right child

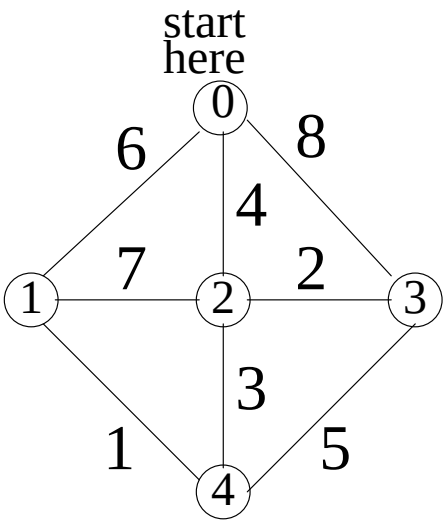
(b) [4 marks] What should you use for $f(n)$ in this loop to get the linear time heapify routine?

```
for (i= f(n); i >= 0; i--)
    bubbleDown(n, A, i);
```

(c) [10 marks] Give pseudocode for a recursive bubbleDown routine that is very close to C or Java code (syntax does not matter, but do not refer to methods/functions unless you define them).

bubbleDown(n, A, pos)
n- number of items in the heap
A- array storing the heap
pos- position in array of item to bubble down.

6. [15 marks] Show the values of the data structures *tree*, *min_wt*, and *closest* (as described in class) after each phase of the Dijkstra/Prim minimum spanning tree algorithm. The *phase* equals the number of vertices in the tree so far. The columns are numbered by the phase and the rows by the vertex numbers. Mark the edges in the MST.



tree:	0	1	2	3	4	5
0						
1						
2						
3						
4						

min_wt:	0	1	2	3	4	5
0						
1						
2						
3						
4						

closest:	0	1	2	3	4	5
0						
1						
2						
3						
4						

7. Consider this Java method:

```
public void readRear(Scanner in)
{    ListNode tmp, current; int data; int i;
     n= readInteger(in);
     start=null; rear=null;
     for (i=0; i < n; i++)
     {
          data= readInteger(in);
          tmp= new ListNode(data, null);
          if (i==0) { start=tmp; }
          else
          {
               current= start;
               while (current.next != null)
               {
                    current= current.next;  // Statement to count.
               }
               current.next= tmp;
          }
          rear= tmp;
     }
}
```

(a) [3 marks] Set up a recurrence relation which counts the number of times that the statement with the comment // **Statement to count.** is executed for a given value of *n* and justify your formula.

[Question 7 continued]

(b) [3 marks] Solve your recurrence from (a) to get a closed formula.

[Question 7 continued]

(c) [9 marks] Prove by induction that your closed formula from (b) is the number of times that the given statement is executed for a problem of size n.

Use this page if you need more space.
Clearly indicate the question you are answering.

The End.