Name: ___________________________________

ID Number: _______________________________

UNIVERSITY OF VICTORIA
EXAMINATIONS- AUGUST 2012
CSC 225 - Algorithms and Data Structures: I
Section A01, CRN 30156 and Section A02, CRN 30157
Instructor: Wendy Myrvold
Duration: 3 hours

TO BE ANSWERED ON THE PAPER.

Instructions:

This question paper has twelve pages (the last page is blank in case you need extra space) plus the header page.

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

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

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

1.(a) [5 marks] Finish the definition for Omega as stated in class: Assume that T and f are functions mapping the natural numbers $\{0, 1, 2, 3, \cdots\}$ into the positive reals.

(b) [5 marks] Let $T(n) = a\,n^2 + b\,n + s$. Prove that for all constants a, b, and s where $a > 0$ that $T(n) \in \Omega(n^2)$.

2. An instructor wants to post grades for a class. Instead of including the complete student numbers, the intention is to print only enough digits so that each student has a different number. The duplication method below takes as input a linked list of integer values and returns true if the list contains duplicate values and false otherwise:

```
public class LinkedList
{
    int n;
    ListNode start, rear;
    public boolean duplication ( )
    {
        LinkedList tail;
        ListNode current;

        if (n = = 1) return(false);
        current = start.next;
        while (current   !=   null)
        {
            if (current.data = = start.data) return(true);
            current= current.next;
        }
        tail= new LinkedList(n – 1, start.next, rear);
        return( tail.duplication( ) );
    }
}
```

(a) [4 marks] Set up a recurrence relation $K(n)$ for the exact number of key comparisons that the duplication method does. Justify why your recurrence relation is correct.

(b) [4 marks] Solve your recurrence relation for $K(n)$ from part (a) by repeated substitution.

(c) [2 marks] Is the number of key comparisons a good choice for a proxy operation for the amount of work done by this duplication method? Justify your answer.

(d) [10 marks] Use induction to prove that the solution to the recurrence relation $K(n)$ that you found in part (b) is the number of key comparisons done by the duplication method.

```
public boolean duplication( ) // Copied from (a) for your convenience.
{
    LinkedList tail;
    ListNode current;
    if (n = = 1) return(false);
    current = start.next;
    while (current   !=   null)
    {
        if (current.data = = start.data) return(true);
        current= current.next;
    }
    tail= new LinkedList(n − 1, start.next, rear);
    return( tail.duplication( ) );
}
```

3. A second approach to the problem from the previous question is to first sort the values in the linked list before searching to see if any values are duplicated. The instructor decides to use a mergeSort method in the LinkedList class for this.

(a) [10 marks] Write the Java code for mergeSort:

```
public class LinkedList
{    int n;
     ListNode start, rear;
     public void mergeSort( )
     {
```

(b) [4 marks] Set up a recurrence relation that expresses the time taken by your mergeSort method. Justify where each term of the recurrence is coming from.

(c) [6 marks] Solve your recurrence from (b) using repeated substitution.

4.(a) [5 marks] Assume that the duplication method is always called with a linked list that is sorted. Rewrite the duplication method so that it is as fast as possible.

public boolean duplication( ) {

(b) [2 marks] What is a best case scenario for your duplication method, and how much time does it take in the best case?

(c) [3 marks] What is a worst case scenario for your duplication method, and how much time does it take in the worst case?

5. [10 marks] Compare the time complexities of the two strategies below for checking for duplications in the list using a Big Oh type of analysis. Justify your answers.

   Strategy 1: Apply the duplication method from Question 2.

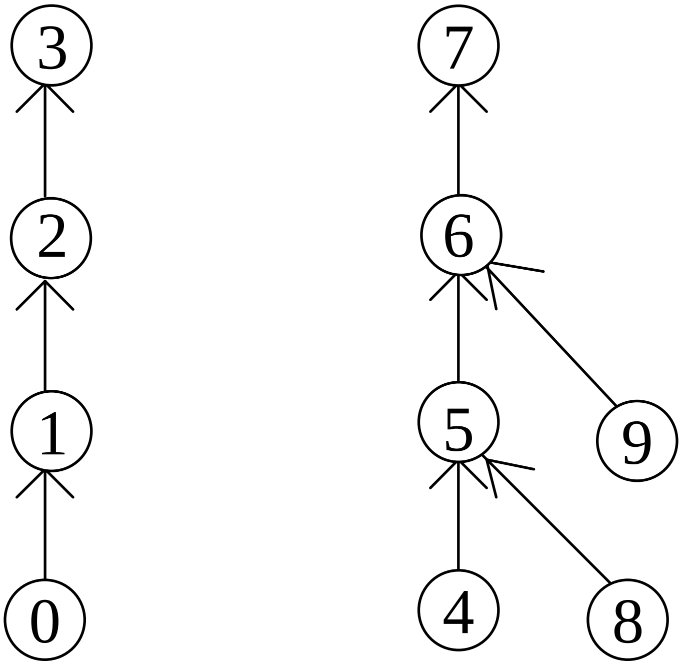
   Strategy 2: Apply mergeSort from Question 3 followed by your new duplication method from Question 4.

6. [10 marks] Compare the space complexities of the two strategies below for checking for duplications in the list using a Big Oh type of analysis. Ignore the space used to store the initial linked list of data values. Count only space used that is over and above that. Justify your answers.

   Strategy 1: Apply the duplication method from Question 2.

   Strategy 2: Apply mergeSort from Question 3 followed by your new duplication method from Question 4.

7. For this question, use **weighted union** with **collapsing find** where the *weight* is the number of nodes in a component. Consider this picture:



(a) [2 marks] What is stored in the parent array?

0	1	2	3	4	5	6	7	8	9

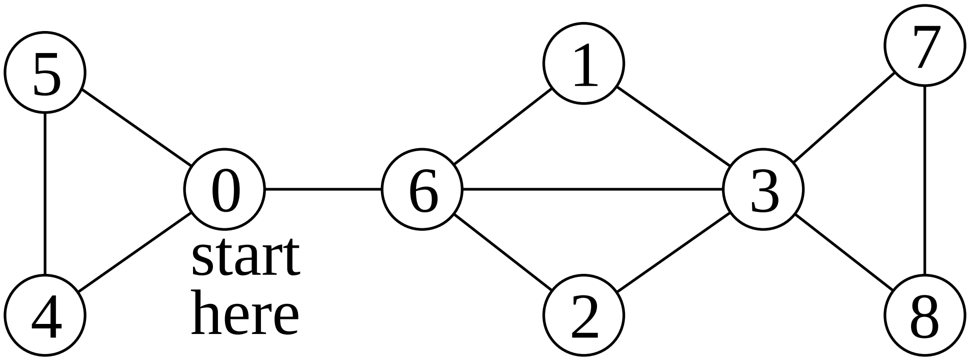
(b) [4 marks] Give the parent array that shows the result after a collapsing find, C-FIND(4) is called.

0	1	2	3	4	5	6	7	8	9

(c) [4 marks] Give the parent array that results from applying W-UNION(0,8) to your answer from part (b).

0	1	2	3	4	5	6	7	8	9

8.(a) [8 marks] Perform DFS (Depth First Search) on the graph given below. Start at vertex 0. When traversing the neighbours of a vertex, traverse them **in numerical order**. Show all your work including the contents of the stack (at each step), the parent array and the DFI array.



	0	1	2	3	4	5	6	7	8
Parent:									
DFI:									

Stack contents at each step:

(b) [2 marks] Mark the edges of the DFS tree on the picture.

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

The End.