9

EASTER UNIVERSITY, SRI LANKA

SECOND EXAMINATION IN SCIENCE (2000/2001)
FIRST SEMESTER

CS 201- DATA STRUCTRE & DESIGN OF ALGORITHM

Time allowed: Two hours — Answer All Questions

Q1

a. Explain, with the aid of real world problems, involved in recursive algorithms.

b. i Explain why the following function may give the wrong value when executed.

```
long factorial ( long n )
{
   if ( n ==0 || n==1 )
       return( 1 );
   else
       return( n* factorial ( - -n ) );
}
```

ii. What is the numeric sequence generated by the recursive function f in the following code listing?

```
long f( int  n )
{
   if ( n ==0 || n==1 )
       return( 1 );
   else
       return(  3* f( n-2 ) + 2* f(n-1)  );
}
```

c. The maximum element in an n-element integer array can be computed recursively.

Define the function
    int max(int x, int y);
that returns the maximum of two integers x and y. Define the function

10

int arraymax(int a[], int n);
that uses recursion to return the maximum element of a.

stopping condition: n = = 1
recursive step: max (max(a[0], ..., a[n-1]), a[n])

d. i Write a recursive binary search algorithm to search an element in a sorted array.

ii. Compare the efficiency of Binary search algorithm with sequential search algorithm.

Q2.

a. What are the essential features of an ***abstract data types*** (**ADT**)? What are the facilities available in Java to support the implementation of Abstract Data Types.

b. A queue differs from a stack in that, items are added at one end and deleted from the other end. Design a queue with abstract data types using static memory allocation (array allocation).

c. Using the basic queue and stack operations, give an algorithm to reverse the elements in a queue.

Q3

a. What distinguishes a linked list from an array?

b. Implement the Primitive operations, ***InsertEnd, DeleteEnd, InsertFront, DeleteFront***, on linked list using java language.

c. Using the above linked list operations, write Abstract Data Types for implementing stack and queue ADTs.

Q4

a. i Write an algorithm to implement the bubble sort algorithm on an array.

Ii Run through the bubble sort algorithm by hand on the list;
4 9 2 1 5

iii develop an efficient implementation of bubble sort, with as few instructions as possible in the inner loop. Make sure that your 'improvements' do not slow down the program

//

b. i. Implement the 'Merge sort' algorithm to sort any given set of integers.

ii Describe and analyze the merge sort algorithm.

iii How many key comparisons does Merge sort do if the keys are already in sorted order, when the sort begins?

iv. Illustrate merge sort using the following data set.

4 0 9 2 7 3 1 8 5