ISI – Bangalore Center – B Math - Physics III – Mid Term Exam

Date: 28 February 2020. Duration of Exam: 3 hours

Total marks: 45

Q 1. [Total Marks:4+6+5=15]

1.a.) Describe one example each of advantages and disadvantages of using arrays vs linked lists. Explain CONCISELY the reason for the advantage or the disadvantage as the case may be.

1b.) Prove or disprove the following:

$n = O(n^2)$,

$n^2 = O(n)$

$n \log n = O(n^2)$

1c.) Suppose we have a linked list of integers as follows:

```
struct node
{ int data;
 struct node* next; };
```

and we want to add a node before the header node.

Explain why the following function WrongPush below (with obvious notation) does not work

```
void WrongPush(struct node* head, int data) {
struct node* newNode = malloc(sizeof(struct node));
newNode->data = data;
newNode->next = head;
head = newNode;
}
```

Make changes to the above code so that it works.

Q 2. [Total Marks:5+2+3=10]

2a.) Write a recursive C function that takes an array of integers and returns the minimum value by making use of the following properties:

$\text{Min}(a_1, a_2, \ldots a_n) = \text{Min}(\text{Min}(a_1, a_2, \ldots a_{n-1}), a_n)$ for n>2.

$\mathrm{Min}(a_1, a_2) = \textit{smaller of the two numnbers}$

Min(a)=a.

2b) Write a recurrence equation that expresses T(n) in terms of T(n-1) where T(n) is roughly the number of steps needed for computation of the minimum of n integers (a rough measure could be statements that may be executed)

2c.) Solve the recursion relation and show that the T(n) = O(n) in the big O notation.

Q 3. [Total Marks:4+6=10]

3a.) Given the recurrence relation T(n)=4T(n/2)+n, and T(1) = a constant,

use the substitution method explained in the class to show that $T(n) = O(n^3)$.

3b.) And then prove the stronger condition that $T(n) = O(n^2)$

Please use the proper definition of the big O functions and rigorous use of inequalities.

Q [Total Marks:10]

DO ONE OF THE FOLLOWING

Write a C function that will accept a sorted integer array
and its length as inputs (the array may have duplicate elements and the function will compact it and then return the new length of the array. Assume that the length is less than the MAXSIZE which is the maximum number of integers the array can hold.

For example suppose the input array contains: 30, 34, 39, 47, 47, 59, 69, 69, 70. When the function returns, the contents of the array should be: 30, 34, 39, 47, 59, 69, 70 with a length of 7 returned.

How will you modify the function if instead of changing the original array, it writes the compacted version to a new array and returns the new length.

OR,

Write a C function that takes a sorted or unsorted array of integers n[i], (0 =< i =< *l*-1) and the length *l* of the array as inputs and returns the location of the first local peak in the middle of the array defined by the following condition:

a[i] is a local peak if $a[i\pm1] < a[i]$ where $0 < i < lemgth-1$.

Please note that this is a different definition of the peak in which the edges cannot be peaks. Please check you function against test cases where the array has the same elements or it is sorted.

Modify the program to identify all the local peaks and their locations.