CSC505 Jennings Homework 1, Spring 2020 There are 16 questions. Please answer all of them.

1. Name: Shashank Shekhar (Student Id: 200262327)

2. Consider the following pseudo-code:

```
Algorithm ApproxArea
Inputs:    f (a function),
           x0, x1 (floating point numbers),
           n (a positive integer)
Output: an approximation of the area under f(x) between x0 and x1
1 sum ← 0
2 delta ← (x1 – x0) / n
3 x ← x0
4 for i = 1 to n-1
5    x ← x + delta
6    sum ← sum + f(x)
7 return (delta/2) * (f(x0) + (2 * sum) + f(x1))
```

We are interested in the asymptotic complexity of **ApproxArea** as n grows large. In particular, we want to know T(n) = the number of times we evaluate the function *f*, for a given value of n.

Give a formula for T(n) and support it using an argument that refers to specific lines (by number) in the pseudo-code above.

Note that in pseudo-code, the convention is that the bounds on *for* loops are inclusive.

Ans: $T(n) = n+1$

Supporting Argument: Function **f** is called (n-1) times in the for loop in line 4 and two more times in line 7. Therefore, total = (n-1) + 2 = (n+1) times

3. Consider the following pseudo-code:

```
Algorithm FelixHausdorff
Inputs:   S, a non-empty set of points (x, y),
          T, a non-empty set of points (x, y)
Output: a vector v=(v_x, v_y) that minimizes the maximum value of
distance(s+v, t) for any points s in S and t in T
 1 min_dist ← infinity
 2 min_vec ← null
 3 max_dist ← 0
 4 max_vec ← null
 5 for each s in S
 6    for each t in T
 7       v ← subtract(s, t)          // v+s = t
 8       max_dist ← 0
 9       for each s in S
10          d ← distance(v+s, t)
11          if d > max_dist then
12             max_dist ← d
13             max_vec ← v
14       if max_dist < min_dist then
15          min_vec = v
16          min_dist = max_dist
17 return min_vec
```

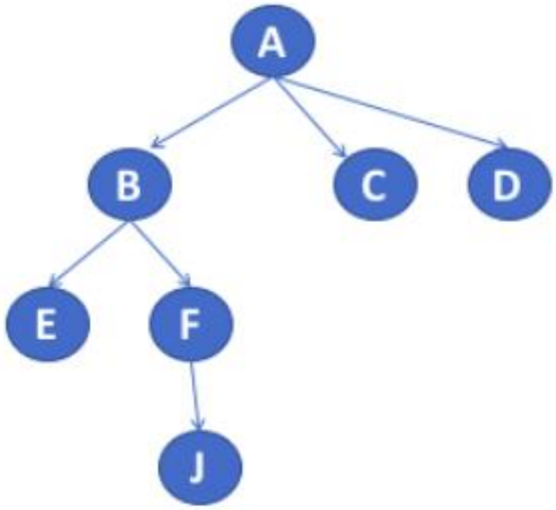
We are interested in the worst-case asymptotic complexity of **FelixHausdorff** as the sizes of S and T grow large. In particular, we want to know T(n, m) = the number of times we evaluate the function distance().

Assume S has n elements, and T has m elements, and give a formula for T(n, m). Support it by referencing (by line number) the pseudo-code above.

Ans: $T(n,m) = n^2m$

Support Argument: Function distance() is called inside 3 for loops. 1st for loop iterates n times (line 5), 2nd for loop iterates m times (line 6) and the 3rd for loop iterates n times (line 9). Since, these for loops are nested, $T(n) = n*m*n = n^2m$

Use the following tree for the next 3 questions:



4. Write the order in which the nodes of the tree above will be visited in a *pre-order traversal.*

Ans: Preorder Traversal: A->B->E->F->J->C->D

5. Write the order in which the nodes of the tree above will be visited in a *post-order traversal.*

Ans: Postorder Traversal: E->J->F->B->C->D->A

6. Write the order in which the nodes of the tree above will be visited in a *level order traversal,* also known as a *breadth-first* traversal.

Ans: Level order (Breadth First Search) Traversal: A->B->C->D->E->F->J

7. Which of these worst-case running times is better (grows more slowly) than the others?
   - ○ n^2
   - ○ n^2 *log* n
   - ✓ *log* n
   - ○ n *log* n

8. Which of these worst-case running times is worse (grows more quickly) than the others?
   - ○ n^2
   - ✓ 2^n
   - ○ n^2 *log* n
   - ○ n^3

9. Consider a singly-linked list with a head pointer. There are n items in the list. To find a given value (element) in the list requires how much time, in the worst case?
   - ○ O(*log* n)
   - ○ O(n *log* n)
   - ✓ O(n)
   - ○ O(1)

10. Consider a singly-linked list with a head pointer. There are n items in the list. To insert an item at the front of the list requires how much time, in the worst case?
   - ○ O(*log* n)
   - ○ O(n *log* n)
   - ○ O(n)
   - ✓ O(1)

11. Consider a doubly-linked list with head and tail pointers. There are n items in the list. To insert an item at the end of the list requires how much time, in the worst case?
   - ○ O(*log* n)
   - ○ O(n *log* n)
   - ○ O(n)
   - ✓ O(1)

12. Consider an array-based list with the head of the list at index 0 (zero). Inserting a new element at the head of the list requires how much time, in the worst case? (Assume the array has enough capacity for the new element.)

- ○ O(*log* n)
- ✓ O(n)
- ○ O(n *log* n)
- ○ O(1)

13. Which single answer best describes binary search in a linear data structure?
    - ○ Needs only O(n) comparisons to search *n* elements
    - ✓ Needs only O(*log* n) comparisons to search *n* elements
    - ○ The input array of elements can start off in any order
    - ○ Works for already-sorted arrays and linked lists

14. How much space is needed to store an arbitrary positive integer?
    - ○ 32 bits
    - ✓ Varies with machine architecture, e.g. 16-, 32-, or 64-bits
    - ○ *log(n)* bits needed to store the integer *n*
    - ○ *n* bits needed to store the integer *n*

15. A recursive function...
    - ✓ Calls itself
    - ○ Calls itself in an infinite loop
    - ○ Calls another function which calls another function, and so on
    - ○ Maintains its own stack

16. If T(n) counts the number of times a function performs a specific operation X, then which of these T(n) describes a recursive function that uses X?
    - ○ T(n) = 4
    - ○ T(n) = 2n + 4
    - ○ T(n) = n + 1
    - ✓ T(n) = 2T(n/2) + 1