# CS210 Data Structures
# (212) Final Exam

Name:_______________________________________________________ ID__________________________

Check your section:

□ Dr. Basit Qureshi
□ Dr. Syed Umar Amin
□ Dr. Skander Alturki

Instructions:

- This exam contains four questions with multiple parts, on 6 sheets of papers (2 sided).
- Time allowed: 180 minutes
- Closed Book, Closed Notes.
- Use of Calculators is ALLOWED. Use of other computing devices / smartphones etc is strictly prohibited.
- Answer the problems on the exam sheets only. No additional attachments would be accepted.
- When the “time is over” is called, it is the students’ responsibility to submit his exam to the invigilator. Submitting completed exam 3 minutes after the “time is over” will incur a penalty of **5 points**.

Few gentle reminders:

- If you get stuck on some problem for a long time, move on to the next one.
- You should be better off by first reading all questions and answering them in the order of what you think is the easiest to the hardest problem.
- Keep the points distribution in mind when deciding how much time to spend on each problem.

| Question No. | Part a | Part b | Part c | Part d | Part e | Student’s Score |
|---|---|---|---|---|---|---|
| Question 1 (CLO 1) | /3 | /3 | /2 | /2 | | /10 |
| Question 2 (CLO 2) | /3 | /3 | /4 | | | /10 |
| Question 3 (CLO 3) | /2 | /2 | /4 | /4 | /4 | /16 |
| Question 4 (CLO 4) | /4 | | | | | /4 |
| Total | | | | | | /40 |

**Question 1. [3 + 3 + 2 + 2 = 10 points] [CLO 1]**

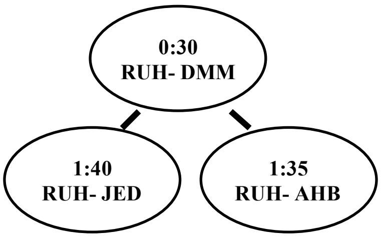
| | Riyadh (RUH) | JEDDAH (JED) | DAMMAM (DMM) | ABHA (AHB) | Tabuk (TUU) |
|---|---|---|---|---|---|
| Riyadh (RUH) | - | 1:40 | 0:30 | 1:35 | - |
| JEDDAH (JED) | 1:30 | - | 1:50 | - | - |
| DAMMAM (DMM) | 0:35 | 2:00 | - | - | 2:10 |
| ABHA (AHB) | - | 1:00 | - | - | 2:30 |
| Tabuk (TUU) | 1:50 | 1:00 | - | - | - |

The above adjacency-matrix represents the scheduled flight duration between selected Saudi airports. For example, there is a flight from Riyadh to Jeddah for one hour and forty minutes. Similarly, another flight from Jeddah to Riyadh with a duration of one hour and thirty minutes. However, there is no scheduled flight between Riyadh and Tabuk. Using the information provided, do the following:

**Part a. [3 points]** Draw a directed graph representation of the above table where cities are vertices and flight-duration (time) as edge-weights. Also give an adjacency list for this graph. Each node in the list must store the duration (time). For easier writing, you may use the three letters code for each airport; for example: RUH to represent Riyadh Airport.

**Part b. [3 points]** Show a DFS **and** BFS traversal of this graph starting at Riyadh. Assume lowest flight-duration to determine the next path.

**Part c. [2 points]** Assume that we would like to list flights by duration in a sorted manner (shortest time first). Show how you would insert/remove flight durations using a **min-heap**. Here is a sample for you to start with. Show the insertion of all flights in the heap with appropriate sink/downheap and swim/upheap operations. Insert the nodes in the heap by going through the adjacency-matrix row by row.



**Part d. [2 points]** Consider the following method which uses recursion to add all of the numbers between the start and end specified numbers. Answer the following:

1. Describe the worst-case running time T(n) of the function `sum`
2. Provide the Big-Oh notation.
3. Show/draw the recursion trace as necessary for this call.
4. What is the returned value?

**CALL:** `sum (5, 10)`

```
public static int sum(int start, int end) {
if (end > start) {
  return end + sum(start, end - 1);
} else {
  return end;
}
```

**Question 2. [3 + 3 + 4 = 10 points] [CLO 2]**

**Part a. [3 points]** Suppose you are given the following hash table, implemented using linear probing. The hash function used was the identity function, h(x) = x mod 9. Assume that the hash table has never been resized, and no elements have been deleted yet.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| **9** | **18** |  | **12** | **3** | **14** | **4** | **21** |  |

Given the following order of items, in which order could the elements have been added to the hash table? There are several correct orders, and you should give all of them.

A) 9, 14, 4, 18, 12, 3, 21<br>
B) 12, 3, 14, 18, 4, 9, 21<br>
C) 12, 14, 3, 9, 4, 18, 21<br>
D) 9, 12, 14, 3, 4, 21, 18<br>
E) 12, 9, 18, 3, 14, 21, 4

**Part b. [3 point]** Show the trace how Quicksort sorts the given array, **always taking the first element as the pivot**. Identify (Circle) the pivot in each pass/iteration.

| K | R | A | T | E | L | E | P | U | I | M | Q | C | X | O | S |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**Part c. [4 points]** Give the run-time T(n) for the following tasks

|  |  |
|---|---|
| Number of comparisons done soring the data {0, 1, 2, 0, 1, 2, 0, 1, 2} using Selection sort. |  |
| Number of comparisons done soring the data {0, 0, 0, 1, 1, 1, 2, 2, 2} using Insertion sort. |  |
| Run time of quick sort, sorting the data {A, B, C, D, E, F, G, H, I} |  |
| Run time of heap-sort, sorting a pre-sorted array |  |
| Space requirement for Merge sort algorithm |  |
| Number of swaps (element exchanges) applying Insertion sort on {0, 0, 0, 0, 0, 0, 0, 0} |  |
| Performing a lookup (search) operation on an M-entry hash table with chaining that contains N keys? |  |
| Searching for the smallest value in a Binary Search Tree |  |

**Question 3. [2 + 2 + 4 + 4 + 4 = 16 points] [CLO 3]**

**Part a. [2 points]** Give the best **Big-O** characterization for each of the following running time estimates F(n), where n is the size of the input problem.

| **F(n)** | **Big-O Characterization of F(n)** |
|---|---|
| $n^{100} + 2^n$ | |
| $n^{10} + 2^{10}$ | |
| $5999434 + 2^{10} + 3^{10}$ | |
| $9999\ n \log n + \log n$ | |

**Part b. [2 points]** Give the run-time **T(n)** for the following code snippets:

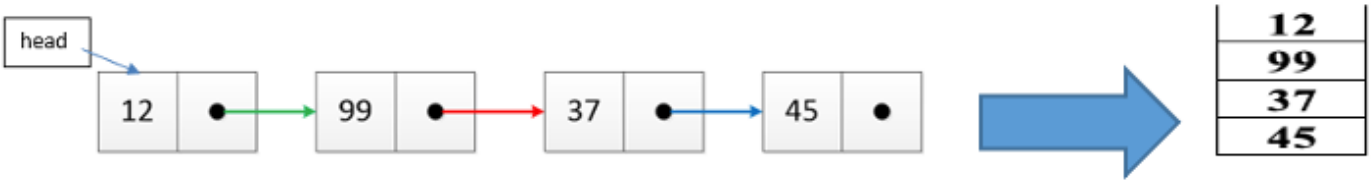
```
public countZeros(int [] a){
  int count = 0;
  for(int i = 0; i < a.length; i++)
    if(a[i] == 0)
       count++;
  return count;
}
```

```
public int returnTail (SinglyLinkedList S) {
  Node Temp = S.Head;
  while(Temp.next!=null)
    Temp = Temp.next;
  return Temp;
}
```

```
public void Myst(int arr[], int F, int L, int K){
   int m = (F + L)/2;
   while(F <= L ){
      if (arr[m] < K){
        F = mid + 1;
      }
      else if (arr[m] == K){
        System.out.println("found at:" + mid);
        break;
      }else
      {
         L = m - 1;
      }
      m = (F + L)/2;
   }
   if (F > L){
      System.out.println("not found!");
   }
 }
```

```
public int search (int [][][] a, int key){
  int p = a.length();
  int r = a[0].length();
  int c = a[0][0].length();
  boolean FLAG = false;
  for(int i=0;i<p;i++)
    for(int j=0;j<r;j++)
      for(int k=0;k<c;k++)
        if(a[i][j][k]==key)
          FLAG = true;
  return FLAG;
}
```

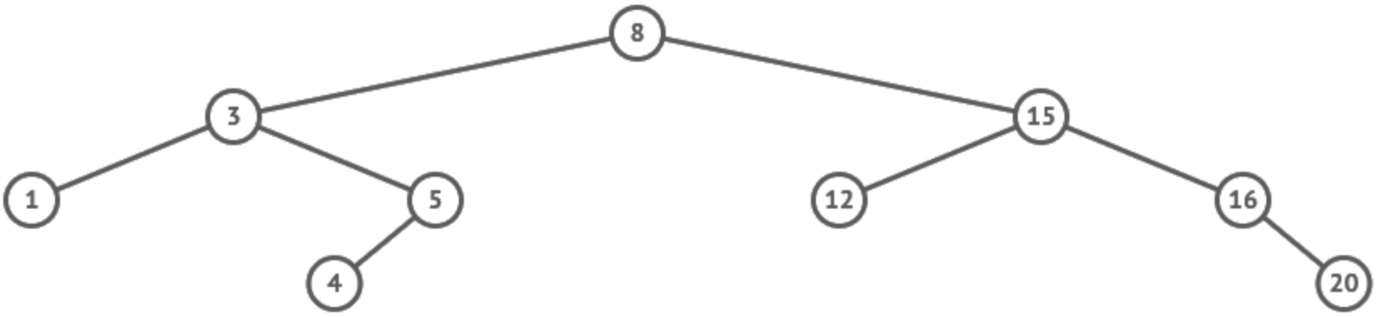
**Part c. [4 points]** Write a method called convertToStack that takes as input a singly-linked list L and returns a stack that contains all the elements of the list, in the following order: The element at the head of the list must be at the top of the stack, the tail must be the first element to enter the stack (at its bottom). The SLL L should not be destroyed.



```
public Stack ConvertToStack(SinglyLinkedList L){

}
```

**Part d. [4 points]** Given the following **AVL** Tree; for each of the following operations, show appropriate rotations and re-draw the tree:

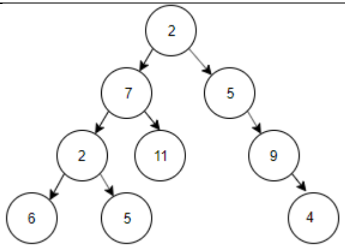


**i) Insert 18 in the above tree,**

**ii) Remove 12 from the tree resulting in i)**

**iii) Remove 8 from the tree resulting in ii)**

## Part e. [4 points] Answer the following Multiple-Choice Questions

| Problem | Answer |
|---|---|
| 1. Deleting a node in the middle of a doubly linked list requires:<br>**a.** Shifting all the proceeding nodes backward<br>**b.** Visiting all the nodes at least once.<br>**c.** Iterating until reaching the required position and updating the pointers<br>**d.** No iteration or shifting is required | |
| 2. Choose the correct answer from the following growth rate functions from the best to the worst:<br>**a.** 100 log n < 2 log n + 2 < 3 n log n + 1000<br>**b.** 2n log n + 2n < 100 log n < 3 log n + 1000<br>**c.** 2 log n + 2n < 3 n log n + 1000n < 100 log n<br>**d.** 100 log n < 3 n log n + 1000 < 2 log n + $2n^2$ | |
| 3. Which of the following runs fastest?<br>**a.** Searching a node in an AVL tree<br>**b.** Searching a node in a skewed Binary Search Tree (all items are identical)<br>**c.** Running Pre-order traversal algorithm on a Tree<br>**d.** Searching for an element in a circular linked list | |
| 4. The running time of the BFS algorithm can be described as:<br>**a.** O(V)<br>**b.** O(E)<br>**c.** O(V+E)<br>**d.** None of these | |
| 5. A normal queue, if implemented using an array of size MAX_SIZE, gets full when?<br>**a.** Rear = MAX_SIZE – 1<br>**b.** Front = (rear + 1)mod MAX_SIZE<br>**c.** Front = rear + 1<br>**d.** Rear = front | |
| 6. Given the following sequence: {2, 3, 5, 6, 9, 11, 15}. Which sorting algorithm will run in O(n) time (n comparisons)?<br>**a.** Insertion Sort<br>**b.** Selection Sort<br>**c.** Heap Sort<br>**d.** Merge Sort | |
| 7. Given the following sequence: {2, 3, 5, 6, 9, 11, 15}, looking for 15, the Binary Search algorithm will run in what time?<br>**a.** O(1)<br>**b.** O(log n)<br>**c.** O(n)<br>**d.** O(n log n) | |
|  <br>8. For the above tree, which of the following give the post-order traversal:<br>**a.** a) 6, 2, 7, 2, 5, 11, 9, 5, 4<br>**b.** b) 6, 5, 11, 2, 7, 5, 9, 4, 2<br>**c.** c) 6, 5, 2, 11, 7, 4, 9, 5, 2<br>**d.** d) 6, 2, 7, 2, 11, 5, 5, 9, 4 | |

**Question 4. [4 points] [CLO 4]** Ahmed needs to insert the following data in a hash-table of size 17. He is not sure what would be a suitable hash-function. Help him decide on the hash-function that gives the **least number of collisions.** Assume the collisions are resolved using Chaining.

**Data: 14, 18, 21, 12, 23, 36, 9.**

$\boldsymbol{f(x) = x \,\%\, 7},\qquad \boldsymbol{g(x) = x \,\%\, 9},\qquad \boldsymbol{h(x) = x \,\%\, 11},\qquad \boldsymbol{k(x) = x \,\%\, 17}$

<This sheet is left blank intentionally. DO NOT detach>

<End of Exam>