King Fahd University of Petroleum & Minerals

College of Computer Science and Engineering

Information and Computer Science Department

Second Semester 132 (2013/2014)

ICS 202 – Data Structures

Major Exam 2

Thursday, April 17th, 2014

Time: 120 minutes

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID# |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Section 01 |  | Question # | Max Marks | Marks Obtained |
| Dr. Sami |  | 1 | 20 |  |
|  |  | 2 | 10 |  |
| Section 02 |  | 3 | 20 |  |
| Dr. Ramadan |  | 4 | 15 |  |
|  |  | 5 | 20 |  |
|  |  | 6 | 15 |  |
|  |  | Total | 100 |  |

**Instructions**

1. **Write your name and ID in the respective boxes above and circle your section.**
2. **This exam consists of 7 pages, including this page, plus one reference sheet, containing 6 questions.**
3. **You have to answer all 6 questions.**
4. **The exam is closed book and closed notes. No calculators or any helping aids are allowed.**
5. **Make sure you turn off your mobile phone and keep it in your pocket if you have one.**
6. **The questions are not equally weighed.**
7. **The maximum number of points for this exam is 100.**
8. **You have exactly 120 minutes to finish the exam.**
9. **Make sure your answers are readable.**
10. **If there is no space on the front of the page, feel free to use the back of the page. Make sure you indicate this in order not to miss grading it.**

**Q.1: (20 points: 2x10):** Consider the following tree:



1. List the nodes of the left sub-tree of the root:
2. List the ancestors of node 12:
3. List the proper ancestors of node 12:
4. List the proper descendants of node 12:
5. What are the degrees of nodes 20, 16, and 10 respectively?
6. List all internal nodes of the above tree:
7. Are nodes 12 and 20 siblings?
8. What are the levels of nodes 7, 12, and 15 respectively?
9. What are the heights of nodes 13, 16, and 15 respectively?
10. Is the above tree full?

**Q.2 (10 points):** Consider the following summarized version of algorithm selection-sort:

1. Find the smallest element in the array
2. Swap the smallest element with the first element on the current array
3. Call selection-sort on the remaining array

Assuming that finding the smallest element function in the first line of the algorithm will carry out *n* – 1 element comparisons in the worst case, when Algorithm selection-sort is called to sort *n* elements.

1. (4 points) Derive the recurrence equation describing the **worst**-case time complexity of the algorithm in terms of the number of element comparisons.
2. **(**6 points) Solve the recurrence equation in part 1, and express it in terms of Big O() notation.

**Q.3 (20 points):** Consider the following directed graph:



Answer the following questions:

**Note:** In the traversals, if at any point there is more than one possible vertex to visit, visit them in the order of their labels.

* 1. (5 points) List the vertices in the order they will be visited using pre-order depth first traversal, starting from vertex 1, draw the depth-first traversal tree
	2. (5 points) List the vertices in the order they will be visited using post-order depth-first traversal, starting from vertex 1 and draw the depth-first traversal tree
	3. (5 points) List the vertices in the order they will be visited using breadth-first traversal, starting from vertex 1 and draw the breadth-first traversal tree
	4. (5 points) List the vertices in the order they will be visited using Topological order traversal. Draw the resulting graph.

**Q.4 (15 points)**

1. Apply the DSW Algorithm on the following binary search tree. Show the tree after the first step (CreateBackbone) and then in its final state (after CreatePerfectTree)

**Q.5: (20 points)** AVL Trees

Insert the following nodes into an empty AVL tree. Make sure you show each step, including rotations if any.

27, 28, 37, 10, 40, 35, 25, 60

**Q.6 (15 points)** Consider the following definition of a heap where the root is at index 1:

public class Heap {

 protected Comparable [] array;

 protected int heapSize;

 …..

}

Write a method called ***isMinHeap*** in class Heap to determine whether the current heap is a Min Heap or not.

**Quick Reference Sheet**

 ,  , , 

