United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

Final Assessment Fall-2020 Total Marks: **25**

Course Code: **CSE 2215** Course Title: **Data Structures and Algorithm I**

Time: *1 hour 15 minutes* for answering. *Another 15 minutes* for submitting.

Any examinee found adopting unfair means will be expelled from the trimester / program as per UIU disciplinary rules.

There are **FOUR questions**. **Answer all of them**. Figures in the right-hand margin indicate full marks.

---

1. Consider the values **13, 2, 19, 14, 8, 22, 5, 20, 11, 16, 26.**

a) Now develop a binary search tree (BST) using those values. You must insert these values **in the given order**. (Draw only the final tree) **[2]**

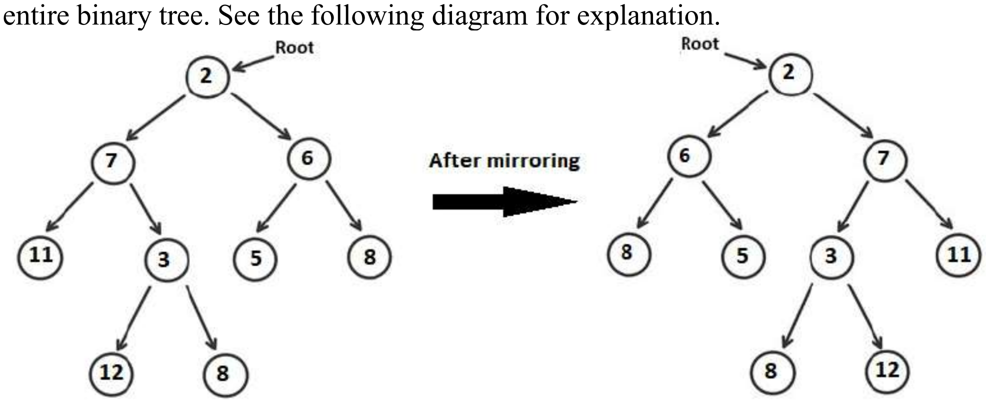
b) Apply the following operations **sequentially** on the Binary Search Tree you constructed in **1.a)** and draw the state of the Binary Search Tree after each operation. **[3]**

(i) Delete 13
(ii) Insert 7
(iii) Delete 19

c) Consider the **treenode** structure given below to create a binary tree. **[1.5]**

```
struct treenode {
    int data ;
    struct treenode *left ;
    struct treenode *right ;
};
```

Write a recursive C function void mirror_tree(struct treenode* t) that will mirror the entire binary tree. See the following diagram for explanation.



Hints: Swap the pointers using postorder traversal technique.

2. Consider the **Array *A* = [10, 8, 20, 3, 6, 15, 2, 25, 18].** Assume indexing starts from 1.

a) Convert the array ***A*** into a **Min Priority Queue**. Clearly show each step. **[3]**

b) Apply the following operations **sequentially** on the Min Priority Queue you constructed in 2.a) and draw the state of the Min Priority Queue after each operation. **[3]**

i. Insert **1**
ii. Extract Min
iii. Decrease key of **25** to **4**

3.

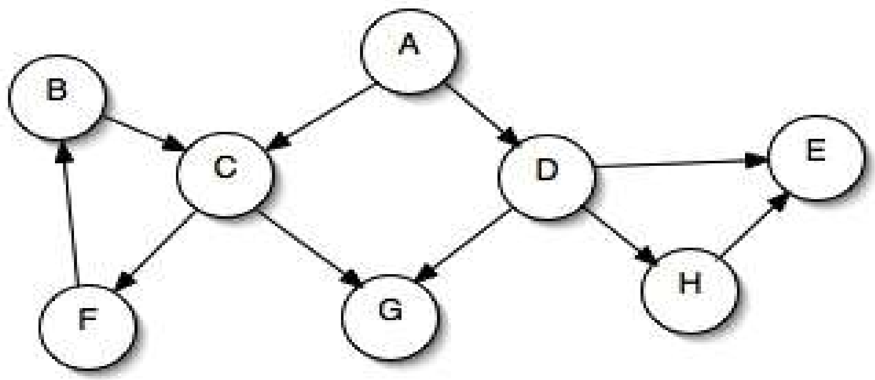
a) adj(s) = [a, c, d], **[2]**
adj(a) = [ ],
adj(c) = [e, b],
adj(b) = [d],
adj(d) = [c],
adj(e) = [s].
Convert the adjacency list representation into adjacency Matrix representation.

b) Give the visited node order for Breadth First search, starting with s, given the above adjacency lists. **[2]**

c) Give an example of a graph with 5 nodes, such that both BFS and DFS will visit the nodes in the same order. **[2]**

4.

a) Run DFS in the following directed graph. Record the start and finish time for each node. Also classify each of the edges [tree edge, cross edge, forward edge, back edge]. Select the node using alphabetical order. **[2.5]**



b) Suppose you are given a Directed Graph G. Graph G may or may not have cycle(s). Now you want to convert G into an acyclic graph by removing edges. The following table contains pseudocode of a simplified version of the DFS algorithm. Make necessary modifications to the pseudocode to remove edges to make the graph acyclic. Specifically mark the step where your algorithm is deciding to remove edges. **[2]**

DFS(G)	DFS_Visit(u)

<pre>DFS(G)
{
  for each vertex u ∈ G→V
  {
    u→color = WHITE;
  }
  for each vertex u ∈ G→V
  {
    if (u→color == WHITE)
      DFS_Visit(u);
  }
}</pre>
	<pre>DFS_Visit(u)
	{
	  u→color = GREY;
	  for each v ∈ u→Adj[ ]
	  {
	    if (v→color == WHITE)
	      DFS_Visit(v);
	  }
	  u→color = BLACK;
	}</pre>

c) Suppose you have calculated the topological ordering of a Directed acyclic graph. Now which of the following actions will invalidate the current topological ordering. Also state the reason. **[2]**
i) Adding an Edge
ii) Deleting an Edge
iii) Reversing the Direction of an Edge