CS 240 Practice Exam

Fall 2022

Printed Name: _____

Please sign your name to indicate that you agree with the following honor code statement:

This work complies with the JMU Honor Code. I have neither given nor received unauthorized assistance, and I will not discuss the exam contents with anyone who has not taken it for credit.

_____

1. You decide to use a hash table to store student information using the student's ID number as the hash key (guaranteed to be unique). Answer the following:

   (a) Given 30 students, what size hash table would you select to use, justify your answer?

   (b) Given a hash table of size 1,000,000, is it still possible to have a collision? If so, explain why. If it is not, explain why it is not possible.

   (c) You are given the task of printing all of the students in sorted order by their ID numbers. You are told that the application that utilizes this data structure will need to print this list in sorted order frequently. Justify either the continued use of a hash table or propose a new data structure to use and justify it. If you select to continue to use a hash table, discuss the Big-Oh complexity of performing the sort. If you select a different data structure, discuss **BOTH** the Big-Oh complexity of finding a student and the sorted printing.

2. Linear Probe Hashing. You are given the following hash function, which accepts an integer x and then compresses it so that the resulting values will work in a table of size 7.

   $H(x) = x \ \% \ 7$

   (a) (5 points) Insert keys 5, 100, 22, 18, 1, 8 (in that order). The collision resolution strategy is *linear probing*.

0	1	2	3	4	5	6

   (b) (2 points) What is the load factor after all of the items have been inserted? You can express this as a fraction if you wish.

   (c) (2 points) True/False. Can items be removed from the hash table/array when deleted? Explain why or why not this is OK.

3. Hashing w/Chaining You are given the following hash function, which accepts an integer x and then compresses it so that the resulting values will work in a table of size 7.

   Utilize the hash function $H(x) = x \ \% \ 7$.

   (a) (2 points) Draw a picture of how the hash table/structure looks after inserting the following keys: 5, 100, 22, 18, 1, 8 (in that order).

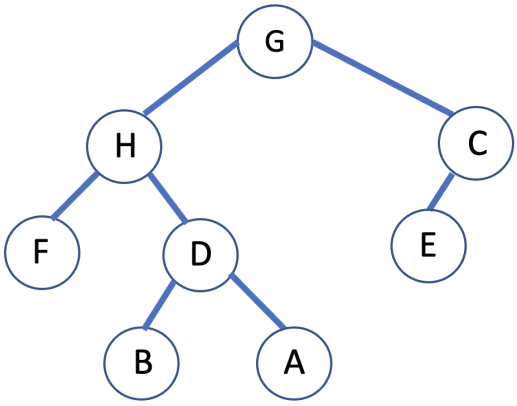


Figure 1: A tree

4. Binary Tree. For the tree in Figure 1, specify the following properties:

   (a) (2 points) Height: _____

   (b) (2 points) Consider the node with key G. Select **ALL** of the following that are true.
   ☐ a leaf node ☐ a root node ☐ an internal node

   (c) (2 points) Consider the node with key C. Select **ALL** of the following that are true.
   ☐ a leaf node ☐ a root node ☐ an internal node

   (d) (2 points) Consider the node with key F. Select **ALL** of the following that are true.
   ☐ a leaf node ☐ a root node ☐ an internal node :

   (e) (2 points) Select **ALL** of the following that is true for the tree in Figure 1:
   ☐ A complete tree and a full tree ☐ A complete tree ☐ A perfect tree ☐ A full tree
   ☐ None of the above

   (f) (2 points) Provide the in-order traversals of this tree: _____

   (g) (2 points) Provide the pre-order traversals of this tree: _____

   (h) (2 points) Provide the post-order traversals of this tree: _____

   (i) (4 points) A binary tree of height 5 must contain a minimum of _____ nodes and a maximum of _____ nodes.

5. Binary Search Tree

(a) (4 points) Draw a BST that results from inserting the following nodes in the given order: 19, 20, 17, 4, 1, 5, 24, 23, 22

(b) (3 points) Assume that remove(24) is called on the tree you built above, What value will be stored at the position where the node with key 24 after the remove operation completes?

(c) (2 points) List the worst-case $\mathcal{O}$ running times for a BST and an AVL tree for a "find" operation. Justify/explain your answer (in other words, explain why they are the same, or if they are different why they are different).

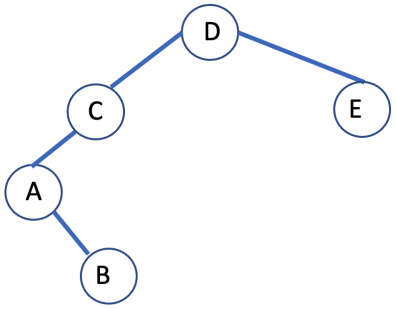


Figure 2: An AVL tree with a new node added.

6. AVL trees The Figure 2 shows an AVL tree just after node B has been added, but before any rebalancing has occurred.
   (a) (1 point) What is the balance factor of node B: _____
   (b) (1 point) What is the balance factor of node A: _____
   (c) (1 point) What is the balance factor of node C: _____
   (d) (1 point) What is the balance factor of node E: _____
   (e) (1 point) What is the balance factor of node D: _____
   (f) (2 points) List the steps necessary to make the tree an AVL tree (if any). Draw the tree after these steps are performed.

   (g) (2 points) Using the rebalanced tree from part (f), remove node D. List any rotations that are required. You must follow the method described in the textbook.

7. Heaps

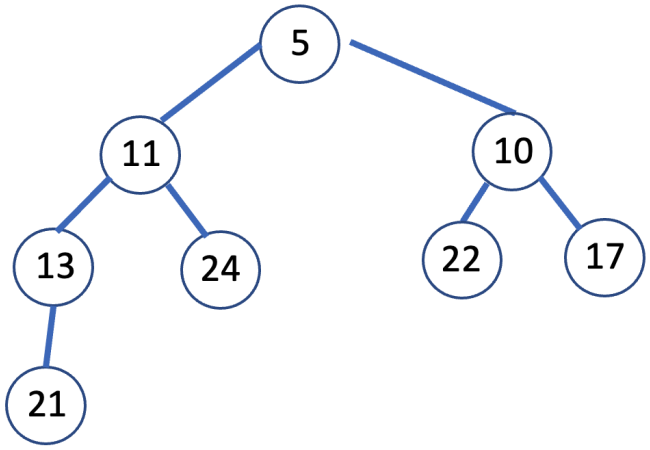


Figure 3: A min-heap

(a) (3 points) Is the tree shown in Figure 3 a legal min-heap? If true, state why it is true. If false, state what nodes are involved that violate the properties of a min-heap.

(b) (6 points) Heaps are commonly implemented with an array. Please draw an array diagram that stores the nodes in the min-heap shown in Figure 3.

(c) (4 points) A remove operation is performed on the heap shown in 3. Draw the resulting heap in tree form.

(d) (2 points) The tree corresponding to a heap is required to be a perfect binary tree. (answer true or false) _____

(e) (2 points) Starting from the tree shown in Figure 3, insert the key 9. Draw the resulting tree.

8. (10 points) Huffman code. A .txt file contains the following letters with the frequencies as shown in the table below.

Letters	A	B	C	D	E	F
Frequencies	11	4	7	1	8	5

(a) (2 points) Draw the Huffman tree that encodes the characters in the table.

(b) (3 points) For each character in the table, list the encoding bits from the tree.

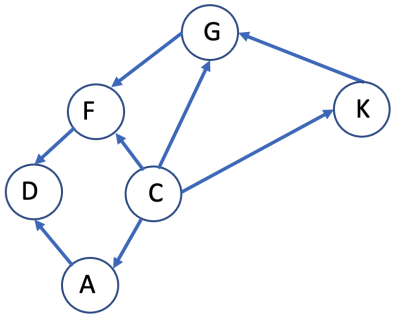
Letters	A	B	C	D	E	F
Frequencies						

(c) (3 points) How many bits are required to store a single character when compression is NOT utilized. _____

(d) (3 points) Compute the compression ratio (uncompress bit total)/(compressed bit total) for the .txt file that corresponds to the frequencies listed in this question. You can show your answer as a fraction.

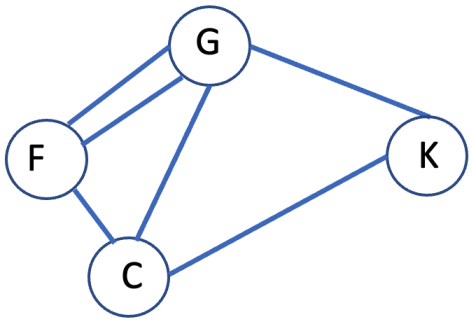
9. Graph conceptions.

(a) (2 points) Select ALL options that describe the graph.



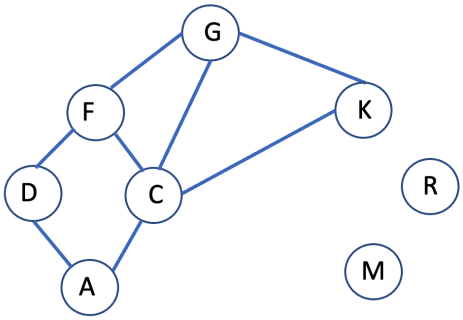
☐ is not a graph ☐ undirected ☐ directed ☐ unweighted ☐ weighted ☐ contains a cycle ☐ does not contain a cycle ☐ is a DAG

(b) (2 points) Select ALL options that describe the graph.



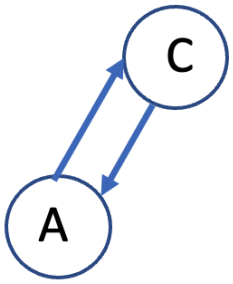
☐ is not a graph ☐ undirected ☐ directed ☐ unweighted ☐ weighted ☐ contains a cycle ☐ does not contain a cycle ☐ is a DAG

(c) (2 points) Select ALL options that describe the graph.



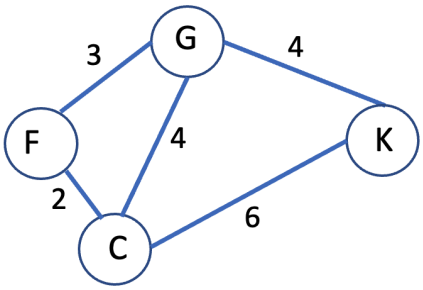
☐ is not a graph ☐ undirected ☐ directed ☐ unweighted ☐ weighted ☐ contains a cycle ☐ does not contain a cycle ☐ is a DAG

(d) (2 points) Select ALL options that describe the graph.



☐ is not a graph ☐ undirected ☐ directed ☐ unweighted ☐ weighted ☐ contains a cycle ☐ does not contain a cycle ☐ is a DAG

(e) (2 points) Select ALL options that describe the graph.



☐ is not a graph ☐ undirected ☐ directed ☐ unweighted ☐ weighted ☐ contains a cycle ☐ does not contain a cycle ☐ is a DAG

10. Graph implementations

(a) (2 points) Given a graph with N vertices, how many linked lists (or hash maps) are used to represent it in an adjacency list graph representation? _____

(b) (2 points) Consider an adjacency matrix and an adjacency list (that utilizes ArrayLists for the list of adjacent edges/vertices). The graph has many vertices (1,000) but very few edges (100). Which representation will utilize less memory? _____

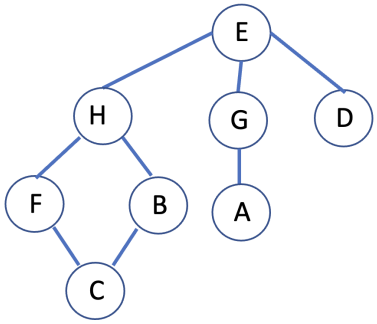


Figure 4: Graph

11. Graph algorithms

(a) (4 points) Indicate the order that the nodes will be visited during a breadth-first traversal of the graph in Figure 4. Assume that the traversal begins at vertex E, and that neighbors are accessed in increasing alphabetical order.

(b) (4 points) Indicate the order that the nodes will be visited during a breadth-first traversal of the graph in Figure 4. Assume that the traversal begins at vertex G, and that neighbors are accessed in increasing alphabetical order.

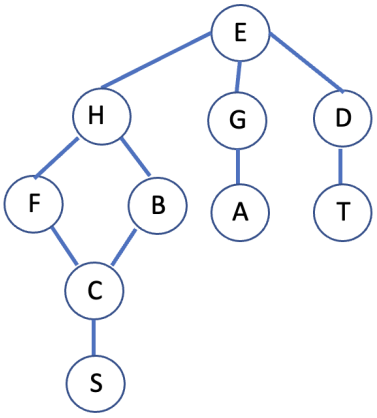


Figure 5: Graph

(c) (4 points) Indicate the order that the nodes will be visited during a depth-first traversal of the graph in Figure 5. Assume that the traversal begins at vertex E, and that neighbors are accessed in increasing alphabetic order.

12. Consider the following graph.

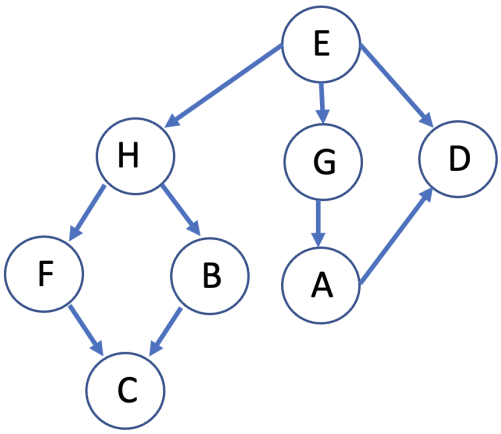


Figure 6: Graph

(a) (4 points) Indicate the order that the nodes will be visited during a breadth-first traversal of the graph in Figure 6. Assume that the traversal begins at vertex G, and that neighbors are accessed in increasing alphabetical order.

(b) (4 points) Indicate the order that the nodes will be visited during a depth-first traversal of the graph in Figure 6. Assume that the traversal begins at vertex E, and that neighbors are accessed in increasing alphabetic order.

(c) (4 points) Indicate the order that the nodes after performing the topological sorting of the graph in Figure 6

(d) (6 points) If a graph is represented as an adjacency list (without hashmaps), what is the running time of the following operations under the worst case? You can assume that the key to each vertex is its element position within the array of vertices. Assume the graph has $|V|$ vertices and $|E|$ edges.

- *public boolean hasEdge(Vertex fromVertex, Vertex toVertex)*: _____
- *public Collection<Vertex> getAllVertices()*: _____

THIS PAGE INTENTIONALLY LEFT BLANK