Name: Example Section: _____

CSC 210: Data Structures

Fall 2025 Midterm Exam

All Sections

Self-scheduled exam, Oct. 17-19 2025

Section 01: Nick Howe (nhowe@smith.edu)
Section 02: Halie Rando (hrando@smith.edu)

- You have 2 hours (120 minutes) to complete this exam unless you have a documented accommodation for additional time.
- The exam is closed book. You are allowed one 8.5"x11" page of notes (double-sided is ok).
- Please write all your answers in this exam booklet. You may use the backs of pages as scratch paper.
- If you think there is a problem with the exam, do your best to interpret what is requested, and document your choice with a note.
- You may not communicate or consult about the exam with anyone other than the professors until after the weekend is over.
- If you are unable to make progress on part of the exam, write down what you know and what you tried, describing your thought process. Partial credit may be possible.

Please copy, sign, and date the statement below: "I have read the instructions above and certify that I have adhered to the Smith Honor Code while completing this exam."

_____

_____

_____

_____

_____

Signature: _____

1. **SORTING**

Sophia is practicing sorting using magnets.

She begins by taking the following moves:

1. Moves the **orange A** to the front position (the smallest magnet).
2. Then moves the **blue A** right after it (the next smallest magnet).
3. On the third pass, she thinks carefully but doesn't end up moving anything.

Now her magnets look like this:

A. Which sorting algorithm do you think Sophia is using? Please briefly justify your answer.

- ☐ Insertion Sort
- ☑ Selection Sort
- ☐ Quicksort
- ☐ Merge Sort

Also ok: quicksort w/ B as pivot

Justification: on each pass, the current min (unsorted) is being moved to the end of the sorted section

B. Which magnet will Sophia move next?

C

C. Is Sophia treating the magnets more like an **array** or a **linked list,** and why? Think about how Sophia is rearranging the magnets.

☐ Array ☑ Linked List

Justification: She is not swapping but rather inserting/removing (w/o any copying)

D. Sophia's friend Otelia has been looking for a working whiteboard marker since they got the magnets, and she comes back at the stage shown above (AABJKC). Will Otelia be able to tell which algorithm Sophia has been using if she didn't see the intermediate steps, or could more than one sorting algorithm reach this same arrangement during execution?

☐ Only one algorithm could produce this arrangement

☑ Multiple algorithms could produce this arrangement, even if they take different intermediate paths

Justification: Quicksort (w/ B as pivot) would reach the same state

E. Their friend Julia is practicing insertion sort. Her magnets currently look like this, with the caret (^) marking the boundary between the sorted and unsorted sections.

Which of the following invariants MUST hold at this point in Julia's execution of insertion sort? (Check all that apply.)

☑ The magnets to the left of the caret are in sorted order

☑ The magnets to the right of the caret may be in any order → alt: in starting order

☐ All magnets to the left of the caret are in their final sorted positions ✗ will get moved as more el. are added

☑ The next magnet Julia handles will be the one immediately to the right of the caret

☐ The magnets to the left of the caret are the smallest magnets in the overall collection ✗ not necessarily (and not here)

☐ No magnet to the left of the caret will need to move again ✗ ← will move to accommodate new elements moving to "sorted"

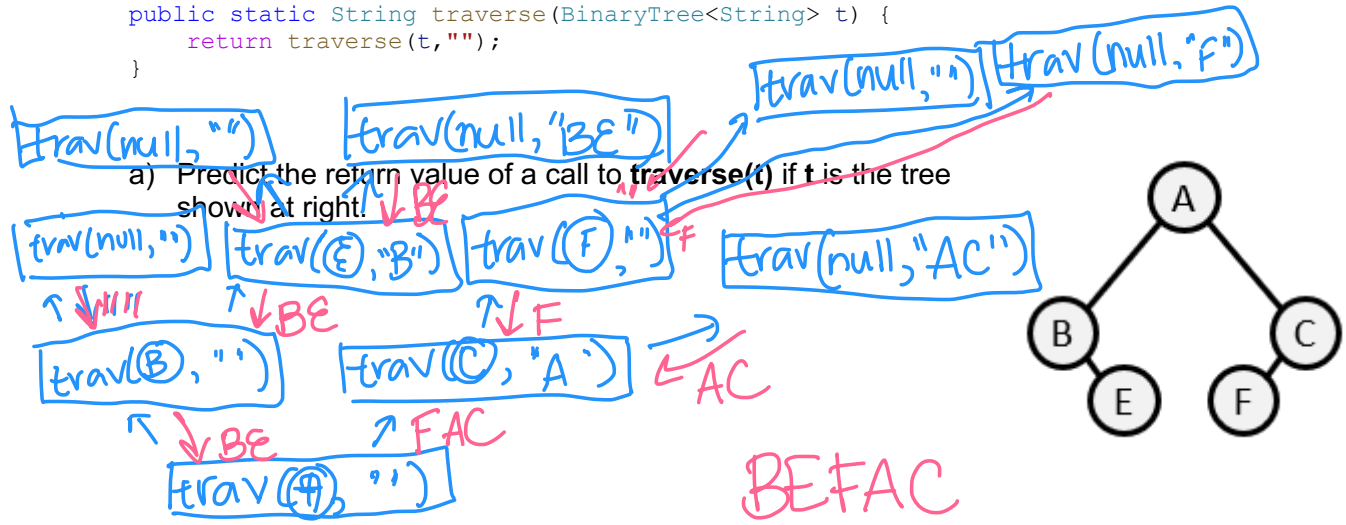
2. BINARY TREES

Consider the code below, which is designed to work with the BinaryTree class we used in the homework.

```
public static String traverse(BinaryTree<String> t, String s) {
  if (t==null) {
    return s;
  } else {
    return traverse(t.getLeft(),"")+traverse(t.getRight(),s+t.getData());
  }
}

public static String traverse(BinaryTree<String> t) {
    return traverse(t,"");
}
```

a) Predict the return value of a call to **traverse(t)** if **t** is the tree shown at right!



b) Is this one of the standard traversal orders we have studied (i.e., preorder, inorder or postorder)? If so, which one? If not, why not?

☐ Yes ☑ No

Which One / Why Not?:

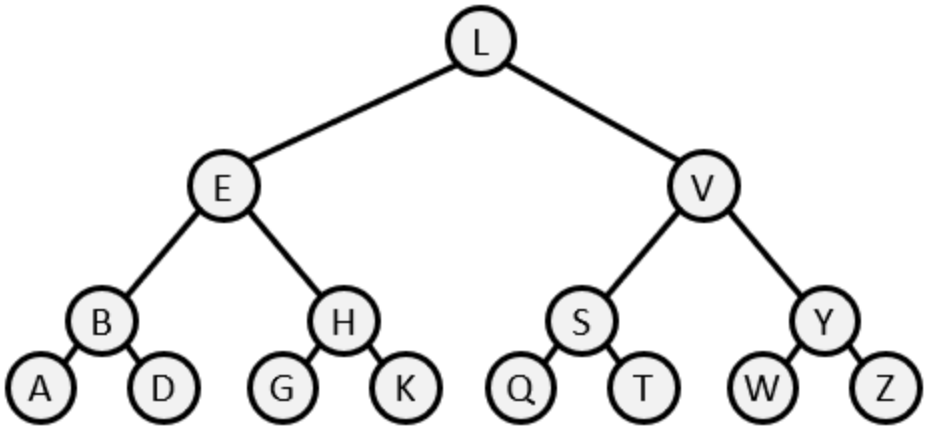
preorder: A B E C F
inorder: B E A F C
postorder: E B F C A

in a standard traversal, A (root) would not be in the middle of subtree (C / F)

you can see the subtrees remain intact in standard traversal

3. **BINARY SEARCH TREES**

For the tree shown, answer the questions in each item below.



a) If a new node J is inserted, where will it end up? Fill in the blanks: look for J: LRRL

left child of K

b) Would this still be a valid binary search tree if we delete the subtree consisting of H and all of its descendants?

☑ Yes ☐ No

c) If node L is deleted using a **merge-left** protocol, what will be the **ancestors of Q** in the resulting tree?

E is new root, K (rightmost left) is parent of V

Q_a: S, V, K, H, E

d) If node L is deleted using a **copy-left** protocol, what will be the **ancestors of Q** in the resulting tree?

K replaces L → Q_a: S, V, K

e) If we perform a **right rotation at the root**, what will be the **ancestors of Q** in the resulting tree?

E is new root, pop H subtree onto L

Q_a: S, V, L, E

f) If we perform a **left rotation at the root**, what will be the **ancestors of Q** in the resulting tree?

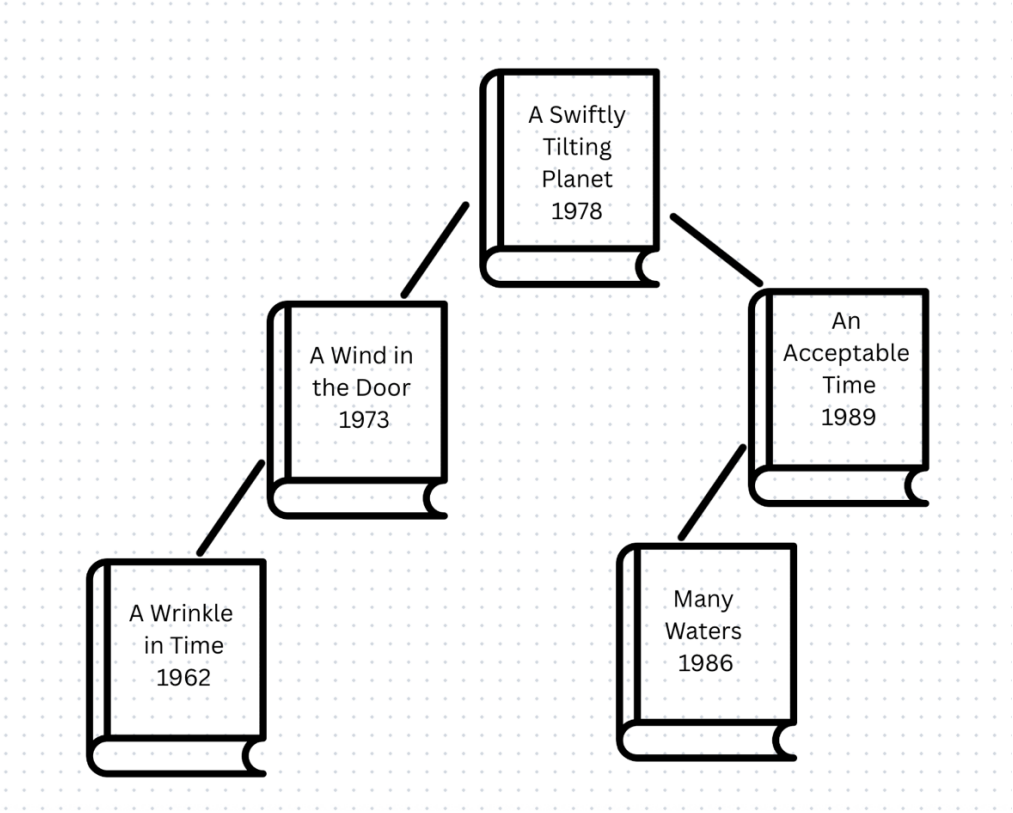
V is new root, pop S subtree onto L

Q_a: S, L, V

4. HEAPS

Some fellow Smithies are practicing Data Structures. Please answer the questions about their approaches in sections A – E below.

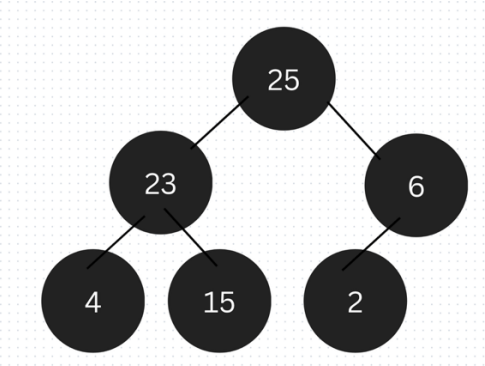
A. Madeleine wants to read these books in order of publication. What is wrong with the priority queue she assembled below? Think about the rules a priority queue must follow.



Issue #1: Neither min heap nor max heap ordering is followed

Issue #2: Not complete: must fill in Left to Right

B. Yolanda is working with the following heap:



i. Redraw this heap as an array:

25	23	6	4	15	2	

ii. Now Yolanda wants to insert a 3. Where will it go? (Fill in the blanks for "[left/right] child of [node]")

always insert at first available spot

Right child of 6

iii. How many steps will it take her to "re-heapify" (make the heap adhere to all the heap rules) after inserting 3? Why/which steps are needed?

Steps needed: 0

Justification: 3 < 6 so for a max heap, this position is fine

C. Gloria started out with an array that she wants to sort into ascending order (A, B, C, D…) using heap sort:

Y	C	X	M	N	A	Z	V

First, she heapified it (phase 1):

A	M	C	X	N	V	Z	Y

Alphabet: X Y Z

Now, she is moving to phase 2 of heapsort, where she will pop the top and swap that element into the "sorted" part of the array:

A	M	C	X	N	V	Z	Y
Y	M	C	X	N	V	Z	A
C	M	Y	X	N	V	Z	A
C	M	V	X	N	Y	Z	A

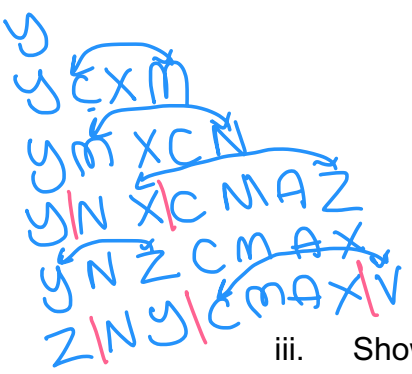
What she is doing wrong? (Hint: the full-credit answer can be articulated in three words or less).

using a min heap instead of a max heap (min heap heapsorst will cause reverse order)

D. Help Gloria fix her sort!

YCXMNAZV

i. Show what the heapified array (end of phase 1) should look like, given her sorting goal. Write your final, heapified array in the boxes.



Z	V	Y	N	M	A	X	C

ZNYVMAXC

ZVYNMAXC

*Algorithm from slides shown Floyd's algorithm gives:

ZVYMNAXC

iii. Show what the array looks like after the first element is moved to the "sorted" portion of the array and the remainder is re-heapified, just like Gloria did above. Show one swap per row. (You may not need all the rows).

sorted

C	V	Y	N	M	A	X	Z
Y	V	C	N	M	A	X	Z
Y	V	X	N	M	A	C	Z

alphabet: T U V W X Y Z

E. Sylvia also wants to practice heap sort. She remembers sorting an array in class and decides to try heap sort on a linked list. Do you think this approach will be more or less computationally efficient than heap sort on an array?

Please **circle** *more* or *less*, then **justify** your answer below.

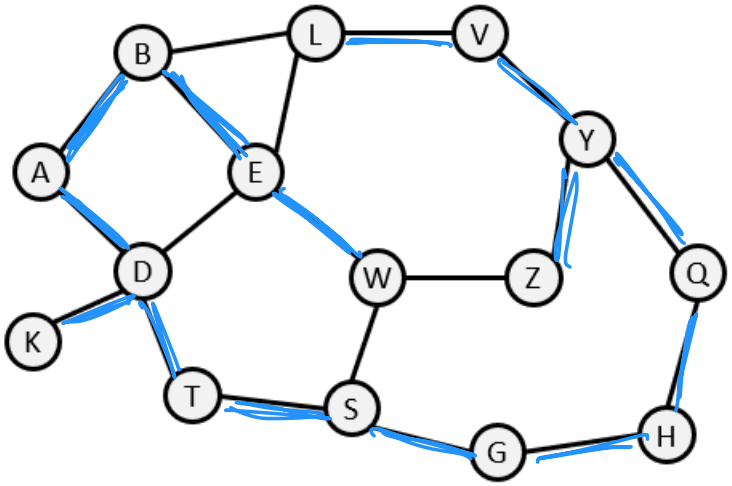
Using a linked list for heap sort will be (**more** | **less**) efficient than using an array.

Justification:

It will be expensive to locate parents/children (use index access on array)

5. GRAPH AND TREE TRAVERSAL

For each item below, give the order in which nodes will be visited using the specified traversal order. When choosing between multiple edges of the same node, go in alphabetical order.



a) Depth-first traversal, starting at W.

W E B A D K T S G H Q Y V L Z

b) Breadth-first traversal, starting at V.

Queue: V L Y B E Q Z A D W H K T S G

V L Y B E Q Z A D W H K T S G

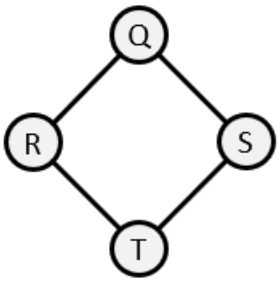
6. GRAPHS

Each diagram or code snippet below represents a graph. We would like you to identify the sets of representations that correspond to the same underlying graph.

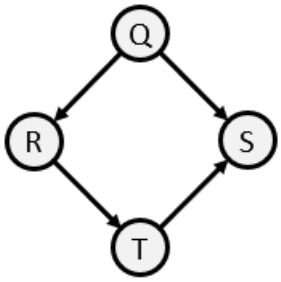
Below, for each item, identify all the other representations that show alternate representations of the same underlying graph (i.e., isomorphic and with matching data elements). There may be several groups.

Your answers may be a little redundant: if X and Y form a set of isometric representations, your answer for X should be Y and your answer for Y should be X. If there are any representations that do not match any others, write "none".

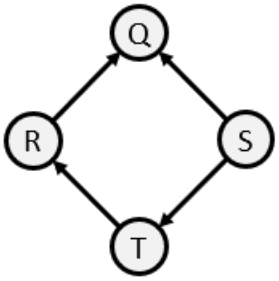
a.)



b.)



c.)



not symm → directed

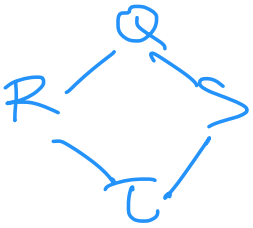
d.)

	Q	R	S	T
Q	0	1	1	0
R	0	0	0	1
S	0	0	0	0
T	0	0	1	0

symmetrical → undirected

e.)

	Q	R	S	T
Q	0	1	1	0
R	1	0	0	1
S	1	0	0	1
T	0	1	1	0



f.)

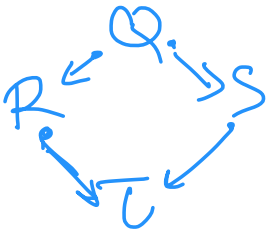
	Q	R	S	T
Q	1	1	0	0
R	1	0	1	0
S	0	1	0	1
T	0	0	1	1

self edges

Symmetrical

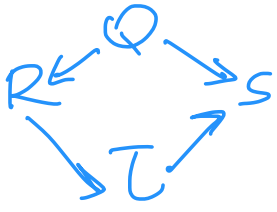
g.)

	Q	R	S	T
e_0	1	-1	0	0
e_1	1	0	-1	0
e_2	0	1	0	-1
e_3	0	0	-1	1



h)

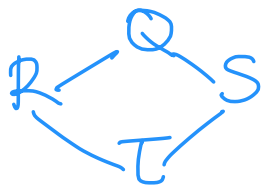
```
ImmutableGraph<String> graph = GraphBuilder
    .directed()
    .<String>immutable()
    .putEdge("Q", "R")
    .putEdge("Q", "S")
    .putEdge("R", "T")
    .putEdge("T", "S")
    .build();
```



Note: Nick & I differ on this one

Note: the version of this question printed is dif. than the one on our answer key. I need to give pts back to some of you!

```
MutableGraph<String> graph = GraphBuilder.undirected().build();
graph.putEdge("R", "Q");
graph.putEdge("S", "Q");
graph.putEdge("T", "R");
graph.putEdge("S", "T");
```



Graph A is equivalent to: E, I

Graph B is equivalent to: D, G, H

Graph C is equivalent to: None

Graph D is equivalent to: B, G, H

Graph E is equivalent to: A, I

Graph F is equivalent to: None

Graph G is equivalent to: B, D, H

Graph H is equivalent to: B, D, G

Graph I: A, E

7. HASH MAPS

You are on your House Council and put in charge of tracking the First Year's favorite colors for big sibling/little sibling activities. Your house gets between 8 and 10 first years each year (never more than 10 as there aren't enough rooms).

Two of the students have the same first name, so you decide to use their Smith **99 number** (student ID number) to store their **favorite color** in a hash map (also known as a hash table).

A. Which will your map use as keys, 99 number or favorite color? Which will you use as values? Why?

Keys: 99 #s

Values: favorite colors

Why? 99 #s will be unique, colors may not be

B. Which of the following is always going to be unique in our hash table implementation? Please check all that apply.

- ☑ Keys
- ☐ Values → can have duplicates
- ☐ Hash value of distinct keys → can have collisions
- ☑ Key-value pairs

C. We will use a simple modulo function to assign a hash to each 99 number. Based on the empty table on the next page, what equation can you infer that we are using? Fill in the box. What makes you think that?

h(k) = k % 10

Justification: 10 rows in has table → % can be 0-9

Note: (prefer prime # for scalability in real-world applications)

D. Below is the info you get back from the students, in the order you receive it. As each one comes in, add it to the hash table below. For collisions, use open addressing with linear probing, as we did in class.

% 10 : look at last digit

Email 1: Student 990123458 likes Blue
Email 2: Student 991882019 likes Green
Email 3: Student 992440928 likes Blue
Email 4: Student 993771160 likes Purple
Email 5: Student 994220544 likes Pink

Hash	Key	Value
0	9924440928	Blue
1	993771160	Purple
2		
3		
4	994220544	Pink
5		
6		
7		
8	990123458	Blue
9	991882019	Green

E. Imagine student 991882019 emails you again to say she actually like Yellow best. Which row(s) would you update and how, following the procedure we practiced in class?

put(991882019, "yellow")
updates row 9

F. Imagine Student 991882019 now decides to defer for a gap year, so you don't need her information anymore. Which row(s) would you update and how, following the procedure we practiced in class?

del 991882019
1 removes K-V pair stored in 9
2 shift k:v pair in row 0 up to 9
3 shift k:v pair in row 1 home to 0

G. If the students had replied in a different order, would your hash table look the same as what you have drawn above? Why or why not?

☐ Yes ☑ No

Justification:

Because we used open addressing, some pairs were bumped out of home row - dif. order would cause dif. bumps based on collisions

H. Could we use a lookup table instead of a hash table to track this information? If yes, what are the pros/cons of this approach compared to the one above? If no, why not?

☑ Yes ☐ No

Justification:

keys are enumerable so we can store 99 000000 - 99 999999 and have index access. Guaranteed O(1) access but Nick did the math and it WOULD take 4Gb of mostly empty space!

I. Could we use hash set instead of a hash table to track this information? If yes, what are the pros/cons of this approach compared to the one above? If no, why not?

☐ Yes ☑ No

Justification:

Hash set: check membership

No associative map property

Lossy

END OF EXAM

(BLANK PAGE)

(BLANK PAGE)