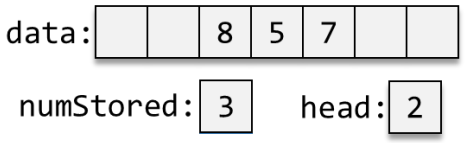
Stacks, Queues, and Deques (12 points)

Consider the data structure shown in the diagram, which represents a deque implementation like the one we studied. The corresponding class has methods **`addFirst`**, **`addLast`**, **`removeFirst`**, and **`removeLast`**.



a.) Assume that we are using the deque in the role of a stack, and we are implementing the **`push`** operation by calling **`addFirst`**. How should we implement the **`pop`** operation?

b.) Assume that we are using the deque in the role of a queue, and we are implementing the **`add`** operation by calling **`addFirst`**. How should we implement the **`remove`** operation?

c.) Starting with the configuration shown, what would be the values of **`numStored`** and **`head`** after a single call to **`addFirst`**?

numStored: ☐ head: ☐

d.) Starting with the configuration shown (and not the results of the previous question), what would be the values of **`numStored`** and **`head`** after a single call to **`removeLast`**?

numStored: ☐ head: ☐

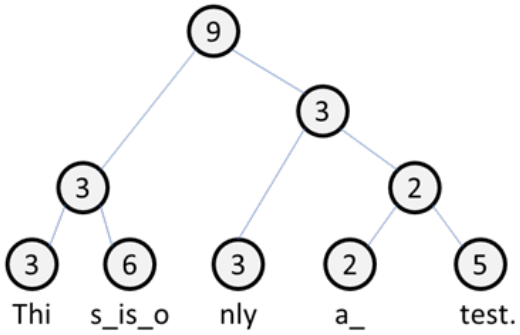
e.) Starting with the configuration shown (and not the results of the previous questions), what would be the values of **`numStored`** and **`head`** after four calls to **`addLast`** and two calls to **`removeFirst`**?

numStored: ☐ head: ☐

f.) Which elements of **`data`** will not be in use to store the current contents of the deque after the sequence described in part (e) above? (List all their index numbers.)

Recursion (12 points)

Consider the recursive method below, which is intended to look up a character in a rope data structure given its index. (Note that you do not need to remember any details about ropes to answer this question.) This implementation of ropes uses our BinaryTree<String> as its base, adding an int weight field to keep track of the total number of characters in the left subtree. Please answer the questions below, referring as necessary to the diagram of a rope data structure shown. (In this diagram, the weight of a node is shown inside, and the data is written just below it.)




```
public int charAt(int i) {
    if (isLeaf()) {
        return this.getData().charAt(i);
    } else if (i >= this.weight) {
        return this.getRight().charAt(i - this.weight);
    } else {
        return this.getLeft().charAt(i);
    }
}
```

a.) Of the three **`if`**/**`else`** branches (first, second, and third), identify the base case(s).

☐ First ☐ Second ☐ Third

b.) List in order each of the calls to **`getData`**, **`getRight`**, and **`getLeft`** that would occur as the result of the initial call to **`charAt(13)`**.

c.) During the call to **`charAt(13)`**, what will be the value of **`i`** when the base case is reached?

Selecting a Data Structure (12 points)

Now that you are finished with CSC 210, you are going to find yourself in many positions where you need to strategically select which data structure you'll use to solve a particular problem. Below, find descriptions of possible problems. First, identify an abstract data type we have studied (Array, Sequence, List, Stack, Queue, Deque, Set, Map, Tree, Graph) that seems best suited and most efficient for the problem. Then name a Java class we have seen that you could use in an implementation (there may be more than one correct answer; you may include Guava classes and ones that we wrote in class). You may optionally provide a justification for your answer.

CHOOSE SIX OUT OF EIGHT TO ANSWER. Write "SKIP" for the items you are skipping.

a.) Developing a chatbot to provide help in choosing CSC courses, where the program's recommendation is determined based on the user's responses to a series of questions that begin general and become more specific.

Abstract Data Type: Java class:

b.) A new tech company wants to start a courier service that will ship packages in unused space in the personal luggage of airline passengers. It will begin service with 20 major cities. Passengers can register flights they plan to take in return for a small payment and indicate how much extra space they have in their bag. The company will figure out how to get packages to their destinations by sending them along with a series of passengers until they reach their destinations.

Abstract Data Type: Java class:

c.) A bookstore wants to keep track of the books on its shelves. To make it easy to find things, the data on the books should be kept in the same order they found on the shelves, from left to right. The records need to be updated as new books are added or as current books are sold.

Abstract Data Type: Java class:

d.) You are making an online news site that will have an interactive glossary feature: while reading, the user can highlight any word, and if there is a glossary entry for that word it will be displayed. The news articles are posted as plaintext from a wire service (a centralized agency that distributes news) and change every day. What data structure would you use to organize the glossary entries?

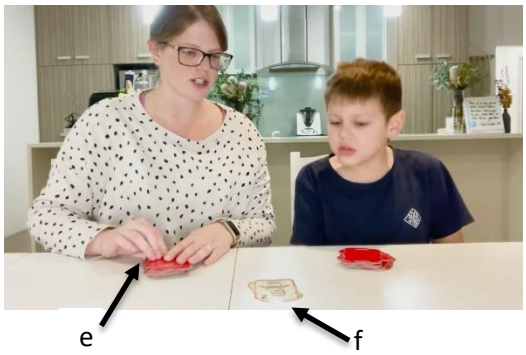
Abstract Data Type: Java class:

e.) In the children's game Slamwich, each player is dealt a hand of cards. They keep their hand of cards in a pile face-down in front of them. Players take turns playing a card face-up into a pile in the middle of the table (see photo). If certain combinations of cards are dealt, such as repeated cards (e.g. lettuce then lettuce), or "sandwiches" (e.g. tomato then bacon then tomato), the first player to slap the center pile gets to take all of the cards from the middle. They will then shuffle

them and add those cards to the bottom of their own face-down pile.

If you were making a computerized version of Slamwich, what data structure would you use for each player's hand?

Abstract Data Type:
Java class:



f.) In Slamwich (see above), what data structure would you use for the middle pile?

Abstract Data Type: Java class:

g.) A popular restaurant chain is designing a promotional campaign that involves scratching off tickets to reveal discount codes. Each code may only be used once. They need a quick way to determine whether a particular code has already been used, so that they don't give out extra discounts.

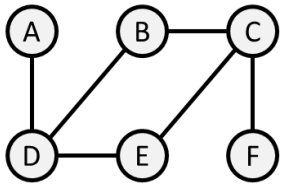
Abstract Data Type: Java class:

h.) A tax agency needs to process the returns from individual taxpayers. The forms are read in one at a time and refunds are issued if necessary. At the end, 2% of taxpayers are chosen at random to be audited. The data for their returns is sent to the audit department.

Abstract Data Type: Java class:

Graph Traversal (16 points)

For the graph shown, please provide the following:



a.) TWO possible breadth-first traversals starting from node A

b.) FOUR possible depth-first traversals starting from node A

Sorting Algorithms (12 points)

You are trying three approaches to sorting the following array:

3	8	2	4	1

Below, the first few steps of three sorting algorithms are shown. First, identify which sorting algorithm is shown (you will not use all of the algorithms). Then, fill in the boxes to show the steps needed to finish sorting the array (you may not need all of the rows.)

Algorithms:

.	Selection Sort	C. Insertion Sort	E. Quicksort
A.	Merge Sort	D. Heap Sort	F. PanicSort*

*The "ineffective sorting algorithm" from the XKCD comic that erases your whole hard drive

ANSWER TWO OUT OF THREE

3	8	2	4	1
3	8	2	4	1
3	8	2	4	1
3	2	8	4	1
2	3	8	4	1

Algorithm: _____

3	8	2	4	1
3	1	2	4	8
3	1	2	4	8

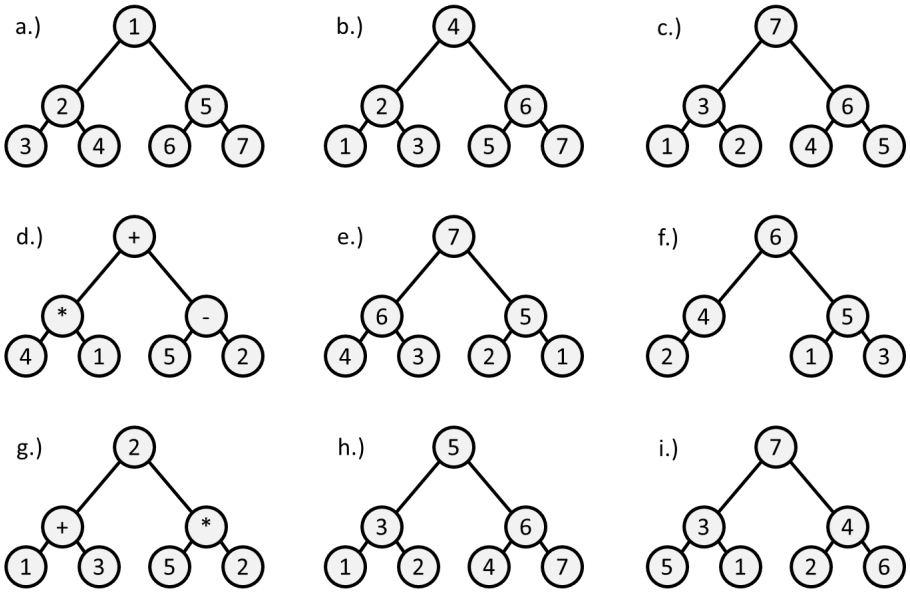
Algorithm: _____

3	8	2	4	1
1	8	2	4	3
1	3	2	4	8
1	2	3	4	8
1	2	3	4	8
1	2	3	4	8

Algorithm: _____

Trees (18 points)

On this question you will answer some questions about the trees shown below.



In the questions below, please circle the letters corresponding to the tree structures that are consistent with each of the following categories. (Each tree may or may not qualify for more than one category.)

I. Arithmetic Expression Tree

A B C D E F G H I

II. Binary Search Tree

A B C D E F G H I

III. Heap (either min-heap or max-heap)

A B C D E F G H I

IV. Sorted (either ascending or descending) when traversed in preorder
For this question and the next two, ignore the trees with non-numeric values.

A B C D E F G H I

V. Sorted (either ascending or descending) when traversed in postorder

A B C D E F G H I

VI. Sorted (either ascending or descending) when traversed breadth first

A B C D E F G H I

Hash Tables (12 points)

Consider the hash table shown below, which uses the simple hash function h(k) = k **mod** 8.

Key	Value
48	Brown Bear
57	Raccoon
42	Opossum
50	Skunk
39	Red Fox
47	Wombat

a.) List all the (key,value) pairs that are not stored at their home position in the table.

b.) Which (key,value) pairs would, if removed, cause other (key,value) pairs to change their position?

c.) Below, a snippet of the code used to create this hash table is provided. You can infer from the hash table the order that some (key, value) pairs were added. Provide an example of what the file `**localAnimals.csv**` could look like to produce this hash table.

```
import java.util.HashMap;
import java.util.Scanner;

public class readAnimals {
   public static void main(String[] args){
        Scanner file = readFile.read("localAnimals.csv");
        HashMap<Integer, String> localAnimals = new HashMap<Integer, String>();

        // Read file line by line
        while (file.hasNextLine()) {
            String line = file.nextLine();
            String[] fields = line.split(",");
            System.out.println(fields[0]);
            localAnimals.put(Integer.valueOf(fields[0]), fields[1]);
        }
        file.close();
   }
}
```

(question continues on next page…)

Show a possible order of animals in the localAnimals.csv file by placing one of the six lines into each of the rows.

Lines:

```
39, Red Fox

42, Opossum

47, Wombat

48, Brown Bear

50, Skunk

57, Raccoon
```

Rows:

Row 1:	
Row 2:	
Row 3:	
Row 4:	
Row 5:	
Row 6:	

END OF QUESTIONS

EXTRA PAGE FOR SCRATCH WORK:

EXTRA PAGE FOR SCRATCH WORK: