Simon Fraser University

CMPT 225 D200

Fall 2022

Practice Quiz 3

Time: 45 minutes

Total: out of 45 marks

SOLUTION

This examination has 6 pages.

Read each question carefully before answering it.

- No textbooks, cheat sheets, calculators, computers, cell phones or other materials may be used.
- ADT means Abstract Data Type.
- List any assumptions you make when answering a question.
- Always comment your code.
- The marks for each question are given in [ ]. Use this to manage your time:
  - One (1) mark corresponds to one (1) minute of work.
  - Do not spend more time on a question than the number of marks assigned to it.
  - Time yourself!

Good luck!

Part 1– Questions that require short answers

1. a. [1 mark each] Express, using the Big O notation, the time efficiency of the following operations:

	Time Efficiency (using Big O notation)
1) Retrieving a member of a Fitness Studio (i.e., an object of the Member class) using the member's phone number from a Fitness Studio Registration system that uses a List.	O(n)
2) Popping an element from a Stack.	O(1)
3) Displaying the elements stored in a binary search tree.	O(n)
4) Determining whether a List is empty.	O(1)
5) Cloning a Queue.	O(n)
6) Deleting the first element in a sorted array-based List.	O(n)
7) Expanding a *min* binary heap (i.e., expanding by doubling the size of its full underlying data structure).	O(n)

b. [1 mark] In order to answer the question 4) in the above question 1. a), which assumption are you making?

Answer: The List has a data member that keeps track of the number of elements currently stored in the List (for example elementCount)

c. [1 mark] See question 5) in the above question 1. a). In order to allow a client code to clone a Queue, which Queue method must we be offering (implementing)?

Answer: At least a copy constructor.

d. [1 mark] Fill in the blank with the most appropriate set of words:

The goal of Step 1 of the software development process is to

____ .

Answer: *The goal of Step 1 of the software development process is to* understand (clarify, detail, disambiguate) the problem statement and requirements.

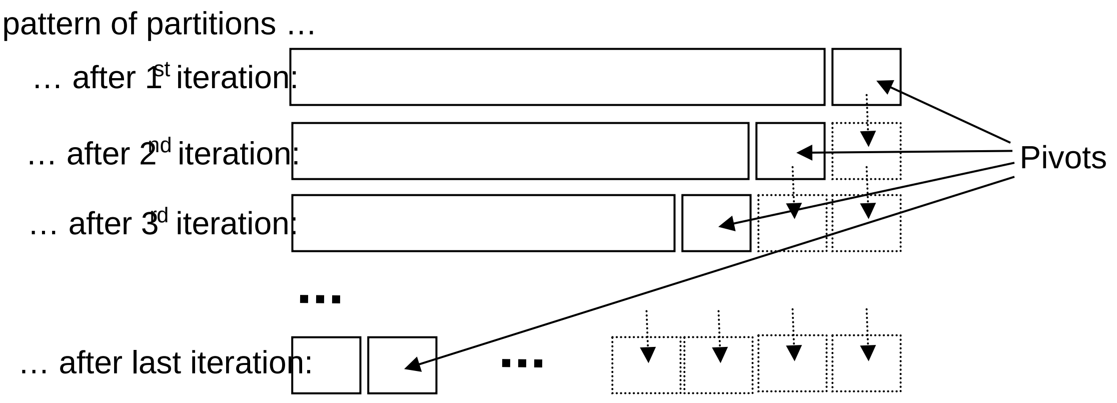
e. [1 mark] State a possible class invariant for the Stack ADT class

Possible answer: LIFO or FILO.

Part 2 – Questions that require longer answers.

1. [Total: 9 marks]

   a. [3 marks] Give an example of an array that would create the following pattern of partitions while it is being sorted using the quick sort algorithm we saw during our lectures:



   Possible Answer: 1, 2, 3, 4, 5, 6, 7, 8

   b. [3 marks] Consider the following binary heap (here, we show only the search key value of each element): A, B, C, D, E, F, G, H

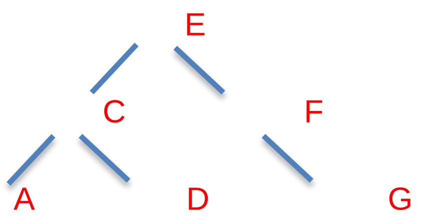
   Fill in the blank: The above is a _______minimum_______ binary heap.

   What would this binary heap look like once we have inserted another element with the search key value "A". Draw the resulting binary heap as an array.

   Answer: A, A, C, B, E, F, G, H, D

   c. [3 marks] The preorder traversal sequence of some binary tree is given as E, C, A, D, F, G, and its postorder sequence is A, D, C, G, F, E. Draw this binary tree.

   Answer:



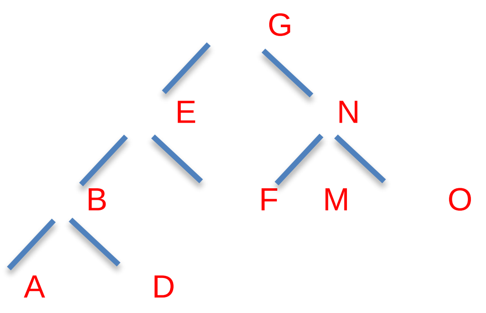
2. [9 marks] Sort the following sequence of elements (here, we show only the search key value of each element): G, D, M, E, F, O, A, N, B
using the tree sort algorithm as we have seen it in class. As part of the tree sort algorithm, make use of an AVL. Clearly label each part of the tree sort algorithm. As you are performing the tree sort algorithm, show every step i.e., show the resulting AVL tree after you have inserted each element and if rebalancing is required, show the AVL tree before and after each rotation. No marks will be given if only the final result is shown.

Answer:

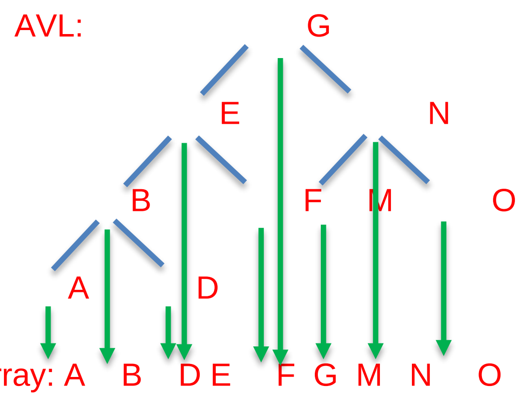
Part 1: array -> AVL

Array: G, D, M, E, F, O, A, N, B

resulting AVL:



Part 2: AVL -> arrayusing inorder traverse



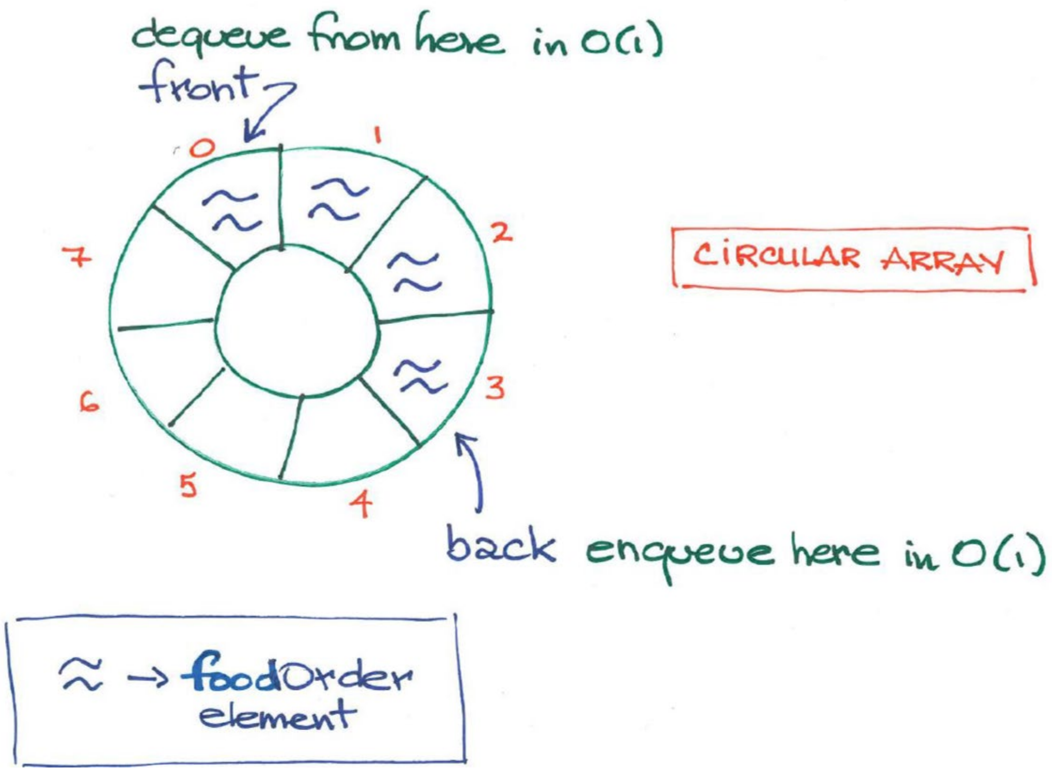
3. The owner of a fast food restaurant asks you to develop a fully automated Drive-Through application. This application is to allow drive-through customers to order their food from their car at a menu posted at the entrance of the drive-through. As the driver orders the food, your Drive-Through application is to convert the verbal food order into an electronic food order, i.e., an object of foodOrder class, and quickly put this foodOrder element into a data collection (implemented as an ADT class). When the kitchen staff is ready to prepare the food order of the next car going through the drive-through, your Drive-Through application is to quickly retrieve the next foodOrder element out of the data collection and displays its list of ordered items on the screen in the kitchen so that the short order cooks can prepare the food. When the driver reaches the payment machine at the end of the drive-through, your Drive-Through application is to automatically total the bill for this particular foodOrder element and display the list of ordered items along with this total on the screen of the payment machine. Once the driver has paid, the food is then made available for pick up.

   a. [1 mark] Which data collection would be the most appropriate to use in this Drive-Through application?

   Answer: Queue

   b. [5 marks] You decide to implement the data collection you have named in a. above with an array as its underlying data structure. Below, draw this underlying data structure. Indicate where, in this data structure, would a new foodOrder element be put and where would a foodOrder element be retrieved from. Add any details you deem necessary for a reader to understand the drawing of your underlying data structure.

Solution:



c. [5 marks] Write the algorithm of the method that will take care of putting a new foodOrder element into this underlying data structure (Note: Your algorithm must work with the drawing of the underlying data structure you have drawn in b.):

```
Private data members:
const unsigned int SIZE = 8;
ElementType * elements = nullptr;
unsigned int front = 0;
unsigned int back = 0;
```

```
enqueue:
if FULL -> expand or error
elements[back] = newElement;
back = (back + 1) % capacity;
elementCount++;
```

d. [5 marks] Write the algorithm of the method that will take care of retrieving a foodOrder element out of this underlying data structure. (Note: Your algorithm must work with the drawing of the underlying data structure you have drawn in b.):

```
dequeue:
if ( isEmpty( ) ) -> error
front = (front + 1) % capacity;
elementCount--;
```