First Name (Print): _____ **Last Name (Print):** _____

Student Number: _____

COMP 2150 Fall 2022

Final Exam (Dec 7, Wednesday)

Maximum 94

INSTRUCTIONS:

- DO NOT zip your file (this file and two python programs from part 3 (Q23, Q24)
- Submit this file with your answers.
- For the python program, submit the .py file ONLY (No other files)

You are submitting three files: (DO NOT ZIP them into one)

1. This exam paper with your answers
2. Python program for part 3 Q22
3. Python program for part 3 Q23

Part 1 (42 points) Multiple Choice Questions (type the correct answer at Ans:)

(2 mark each unless stated otherwise)

1) How many comparisons will the linear search algorithm make if the search key is *not* in an array of 100 elements?

   A. Log_2100
   B. 100
   C. 99
   D. 1
   E. Not enough information to compare

   Ans:

2) Using a binary search, what is the maximum number of comparisons required to find a search key in a 124 -element sorted array?

   A. 5
   B. 7
   C. 6
   D. 62
   E. 123

   ANS:

3) What is the term used for binary search's run time?

   A. Linear run time.
   B. Quadratic run time.
   C. Constant run time.
   D. Logarithmic run time.
   E. Division run time.

   ANS:

4) Which of the following is true about a queue?

   A. Enqueue, dequeue, and peek are all done from the front of the queue.
   B. Enqueue and dequeue are done from the front of the queue; peek is done from the back.
   C. Peek is done from the front of the queue; enqueue and dequeue are done from the back.
   D. Enqueue is done from the front of the queue; dequeue and peek are done from the back.
   E. Dequeue and peek are done from the front of the queue; enqueue is done from the back.

   Ans:

5) **Big O** notation describes _____.

   A. the amount of memory required by an algorithm.
   B. the difficulty of writing an algorithm to solve a specific problem.
   C. an algorithm's comlexity in terms of the system model.
   D. an algorithm's efficiency in terms of the work required to solve a problem.
   E. the length of time for an algorithm for solving a specific problem.

   Ans

6) (3 points) Which of the following sorting algorithms is the fastest and why?

A. Selection sort.
B. Insertion sort.
C. Merge sort.
D. Insertion sort.
E. They all run at roughly the same speed with a large N (data sample)

Ans: **why:**

7) What does the first pass of selection sort do?

A. Splits the array into two approximately equal pieces.
B. Orders the first two elements of the array.
C. Partitions the array into two unequal pieces depending on whether each element in the array is greater or less that some pivot element.
D. Locates the smallest element in the array and swaps it into the zeroth position.
E. Locates the largest element in the array and swaps it into the zeroth position

Ans:

8) Which of the following statement of a LinkedList DS is ***false***?

A. A linked list is a linear collection of self-referential class objects called nodes connected by reference links.
B. A linked list is appropriate when the number of data elements to be represented in the data structure is unpredictable.
C. A linked list is a fixed-size data structure.
D. By convention, the link reference in the last node of a list is set to null to mark the end of the list.

Ans:

9) (4 points) What is the output of the following code:

```
class Orange:
  def __init__(self, orange):
    self.orange = orange
a1 = Orange(375)
a2 = Orange(35)
a3 = Orange(a1.orange % (Orange(10).orange + a2.orange))
print(a3.orange)
```

A. 45.71
B. 25
C. 35
D. 15
E. 45

Ans:

10) Which of the following data structures **are** not ordered: (more than one answer)

A. Tuple
B. Sets
C. Linkedlist
D. Dictionaries
E. List/Arrays

Ans:

11) What do you call the attributes that create unique object in python:

A. class attributes
B. super attributes
C. static attributes
D. instance attributes
E. global attributes
F. None of the above

Ans:

12) Recursion Question: What is the output of the below code?

```
def func(x):
  if x==0:
    return 0
  return x+func(x-1)
func(-5)
```

A) 15 B) -15 C) -10 D) syntaxError E) RecursionError

Ans:

Quick Sort question:

13) (3 point) Suppose you choose the first element as a pivot in the list {`45, 11, 50, 59, 60, 2, 4, 7, 10`}. Using the quick sort **partition algorithm** that we discussed in the lecture, what is the new list after the first partition? What is the pivot selected for the partition?

A. `45, 11, 50, 59, 60, 2, 4, 7, 10`
B. `60, 11, 50, 59, 45, 2, 4, 7, 10`
C. `60, 50, 11, 59, 45, 2, 4, 7, 10`
D. `2, 4, 10, 7, 11, 45, 60, 59, 50`
E. `2, 7, 4, 11, 10, 45, 60, 59, 50`
F. `2, 4, 7, 10, 11, 45, 50, 59, 60`

Ans: pivot point =

14) BST question: There are three BT traversal orders (pre-order, in-order, post-order), Which of the following will you select to perform (1) order sequence of the node (2) process the children of a node first, and (3) process a node before its children:

A. post-order, pre-order, in-order
B. pre-order, post-order, in-order
C. in-order, pre-order, post-order
D. in-order, post-order, pre-order
E. pre-order, in-order, post-order
F. None of the above

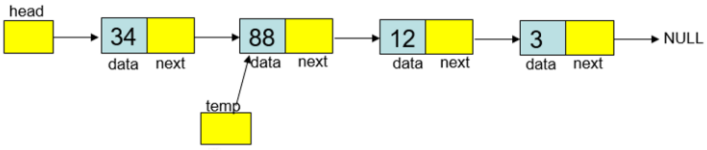
Ans:

15) BST question: for a BST tree, the predecessor is:

A. The minimum value of the right sub-tree from the root
B. The minimum value of the left sub-tree from the root
C. The maximum value of the right sub-tree from the root
D. The maximum value of the left sub-tree from the root
E. The minimum value of the left sub-tree from the right subtree

Ans:

16) What is in the head -> next -> next -> data below?



a. 34
b. 3
c. 88
d. 12
e. Null

Ans:

17) In LinkedList implementation, a node carries information regarding:

a. Data
b. Link
c. Data and Link
d. Another linkedList
e. None of the above

Ans:

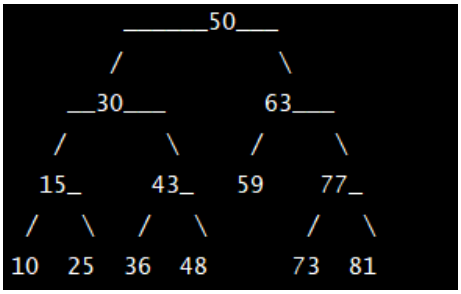
18) What is the operation of the below statements:

```
def misty (self, node)
    Int x = 0
    while node:
      x += 1
      node = node.next
    return x
```

a. Insert a new node to the front of a linked list
b. Create a new node to the linked list
c. Insert a new node to the end of the linked list
d. Insert a new node when x is > 1
e. Calculate the linked list size

Ans:

19) Consider the following BST, if the root node is deleted, the new root can be:



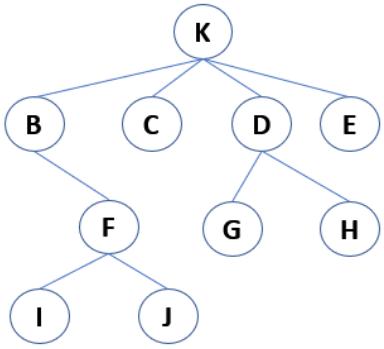
a. 30 or 63
b. 43 or 48
c. 63 or 73
d. 48 or 73
e. 48 or 59
f. None of the above.

Ans:

Part 2: (12 points) Short Answers and Code Analysis

BST question

20) (6 points) The following tree diagram does not represent a Binary Search or a Binary Tree, however the tree traverse mechanism is the same:



a. (1 point) Which node(s) is(are) the roots of this tree? Ans:

b. (2 points) What node(s) is(are) the leaves of this tree? Ans:

c. (3 points) Starting with Q, write the nodes in the order of ***postorder*** traversal:
Ans:
(no partial correct – 0 point for any mismatch)

21) (6 points) You can implement a queue data structure (DS) using a list/array or a LinkedList DS. Nonetheless, the performance (in O(..) factor) can be drastically different depending on your data manipulation pattern. Which of the following Queue implementation is more efficient in terms of O(x) factor?

Which data structure (MyQ1, or MyQ2) is better for the following situations (Use big O factor with explanations)?

```
class MyQ1:  (using the doubly linked list implementation)
class MyQ2:  (using the list append(ele) and pop(0) implementation)
```

(a) Most of the data are adding and removing activities in the front of the queue

Ans:

(b) Most of the data are adding and removing activities at the rear of the queue

Ans:

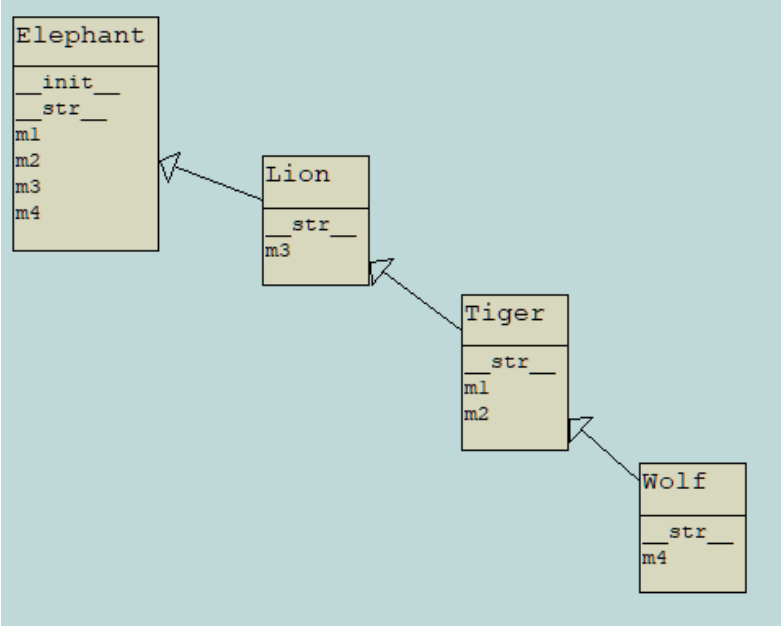
Part 3 (40 points) Programming:

22) (**16 points**) Write a python function for a LinkedList data structure that returns the sum of the nodes greater than the node next to them. I am providing you with a fully functional LinkedList program we used in our inClass exercises (Q22_Sum_above_LinkedList_student.py). The program has all the functions to convert a list to a LinkedList DS with the display. For this question, an empty function (sum (root)) with some skeletons is available for you to start. You must convert the provided list to a LinkedList and display it before coding the sum(root) function.

Results:

original list: [17, 0, 4, 1, 20, 9, -1, 23, 18, -5, 34, 35]
linked_list: 17 -> 0 -> 4 -> 1 -> 20 -> 9 -> -1 -> 23 -> 18 -> -5 -> 34 -> 35 -> None
Nodes that are greater than the next node
17 4 20 9 23 18
last node 35
Sum of nodes greater than the next = 126

23) (**24 points**) Write a python program based on the following UML diagram:
Class Elephant is the parent of Lion; Lion is the parent of Tiger; Tiger is the parent of Wolf



(i) (2 points) Create four classes (Elephant, Lion, Tiger, Wolf)
Only the Elephant class has:
def __init__(self):
pass

(ii) (2 points) Define a __str__(self) function for each class:

The __ str__ (self): in each of the **multi-level hierarchy classe**s returns its class name:
For example, for the Elephant class,

```
def __str__(self):
    return  'from Elephant'
```

(iii) (4 points) Define the mx(self) function in each of the classes to display a message like the following:
(x in a number for a mx( ); eg, m1(), m2()…). Refer to the UML diagram above for the mx( ) definition specifications.
For example:
m1(self) in the Elephant class is:

```
def m1(self):
    print('I am an elephant m1( )')
```

m4(self) in the Wolf class is:

```
def m4(self):
    print('I am a wolf m4( )')
```

NOTE: Not every class has the same number of specific mx() methods.

Write a py program to demonstrate the multiple-level class dynamic method binding behavior. For example, a child (derived) class method can override the parent (inherited) class if it exists but use the parent class method if otherwise.

Driver Program:

- (2 points) Create an object for each of the classes
- (2 points) Create a list of the four objects (Elephant, Lion, Tiger, Wolf)
- (4 points) Loop through the list to display the following:
- (8 points) Display the results:

Your result must be the same as the following. (in terms of the mx( ) display order)

from an elephant
I am an elephant m1()
I am an elephant m2()
I am an elephant m3()
from a lion
I am an elephant m1()
I am an elephant m2()
I am a lion m3()
from a tiger
I am a tiger m1()
I am an elephant m2()
I am a tiger m2()
I am a lion m3()
from a wolf
I am a tiger m1()
I am an elephant m2()
I am a tiger m2()
I am a wolf m4()
I am a lion m3()