Dr.Erdem KAYA
ekaya@metu.edu.tr
Mehmet Taha ŞAHİN
staha@metu.edu.tr

Midterm Examination
METU CENG310 Spring 2021
Data Structures and Algorithms with Python

Exam Date: May 3rd, 2021
10:40am - 11:30am
(50 Minutes)

A True/False Questions (19.5 pts.)

1. Accessor functions of a class can change state of the objects that are created from that class. (True/False)
2. Primitive data types in Python such as int, float, bool are classes and thus they can be instantiated. (True/False)
3. str, set, and frozen sets are examples of sequence types in Python. (True/False)
4. Given that `data = [1,2,3]`, the first time `next(data)` is called, it should return value 1. (True/False)
5. Modules are libraries (or containers) for values, functions, and classes that are logically related. (True/False)
6. In an abstract base class, there can be concrete methods relying on abstract (not implemented) methods. (True/False)
7. $f(n) = n - 7$ is $O(n)$ but not $\Omega(n)$. (True/False)
8. Cells of arrays need not be located contiguously in memory. (True/False)
9. "Every" `append` operation of a `list` object takes the same amount of time. (True/False)
10. Computational complexity of `push` operation of a stack implemented with `list` class is $O(1)$ (not amortized $O(1)$). (True/False)
11. Deleting the first and the last nodes of a singly linked list costs the same amount of operations. (True/False)
12. `push` and `pop` operations of a stack implemented with a singly linked list is $O(1)$ (not amortized $O(1)$). (True/False)
13. Link-based sequences tend to occupy less memory space compared to that of array-based sequences. (True/False)

B Fill in the Blanks Questions (28.5 pts.)

Please find a suitable word, expression, word, number, etc. to the blanks (denoted as five consecutive underscores _____) in the following questions. Please note that there might be more than one correct answer, any one of them will be accepted, so do not try to enumerate all possible correct answers.

14. Fill in the for loop statement so that it can work.

```
for item _____ my_list:
    # Do something.
```

15. If `data` is a sequence type then the object `iter(data)` is called an _____ object.
16. The class method that initializes a newly created object is called _____ .
17. $3^n =$_____(n^3). (Asymptotic Notation)
18. Recursive algorithms have two parts: _____ and _____.
19. `list` class can be shown as an example to _____ arrays. Python also provides a more *compact* array representation in which we can only store primitive data types.
20. The _____ and _____ operations can be used to add to and drop from a queue.
21. ADTs such as stacks, queues, and linked lists are generally considered as _____ data types whereas trees are said to be _____ data structures.
22. First and last nodes of a linked list are called _____ and _____, and the link between any two consecutive nodes is called _____.
23. By making use of a _____ linked list, we can do consecutive `dequeue` and `enqueue` on the same node very efficiently. Technically, instead of making a set of consecutive `dequeue` and `enqueue` operations, we can make a single `rotate` operation.
24. Considering general trees, if a node has one or more children, it is said to be an _____ node, otherwise it is a/an _____ node.
25. A binary tree is said to be _____ if its each node has either zero or two children.
26. A binary tree can have at most _____ nodes if its height is d.
27. While traversing a tree, if we run into a node whose parent reference is `None`, we can be sure that that node is the _____ of that tree.

C Definitions (5 pts.) (Pick any two. Answer two only.)

Please provide *formal* definitions for the following terms.

28. Provide a recursive definition for tree ADT.
29. What is the *depth* of a tree?
30. What is the *height* of a tree?
31. What is a binary tree?

D Simple Answers (10 pts.)

Answers to these questions are expected to be very brief such as one or two words, or at most one sentence.

32. Let us assume that `a` and `b` are two instances of a custom class that you implemented. In order to be able to make an operation such as `a == b`, which function should you implement in the body of your class?
33. For a successful search, what is the worst-case computational complexity of finding an item with the binary search algorithm?
34. What is the best-case complexity of insertion sort? (Hint: In the best-case scenario, how would the items in the sequence (e.g., list, array, etc.) look like?)
35. Except `size` and `is_empty`, give the names of two stack and four general tree (valid for all kinds of trees) ADT operations.
36. What is the postfix expression of the infix expression `a*b+c`.

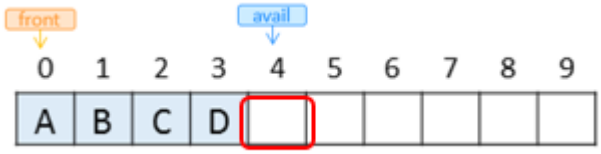
E Open-ended Questions (5 pts.)

37. Given a binary tree T with height h, what could be the minimum and maximum number of nodes of T? Please prove your answer.

F Simple Coding (16 pts)

In this part, you are expected to provide Python code snippets depending on what is asked in question. Typical answers should not be longer than 8-10 lines.

38. (2 pts.) Assume that a class named `Building` is implemented and we need to create a new class `House` inheriting `Building`. Write only the signature part of the `House` class. (Expectation: 1 line of code)
39. (2 pts.) Write a *recursive* function that calculates factorial of its input. Assume that the input will always be a natural number, hence, no need for type check.
40. (2 pts.) As you have already learned so far, queue class implementations based on *circular arrays* have `_front` (head of the queue) and `_size` (number of items in queue) members. By making use of these members, the index of the next available slot for new enqueue operation (`avail`) can be easily calculated. Assuming that the length of underlying array can be found by `len(self._data)`, write a single line of code to find the index of the next available slot.



41. (5 pts.) Consider a singly linked list object `L` whose nodes are instances of `Node` class. `L` has also a `tail` pointer which points to the last node. `Node` class' constructor accepts any type of element as parameter, so an instantiation such as `Node(e)` can be done. Write a function `add_last(L, e)` that adds a new node to the end of the linked list `L`.
42. (5 pts.) Below is an implementation of doubly linked list. Please fill in the blanks.

```
1  class DoublyLinkedList:
2    class _Node:
3      def __init__(self, element, prev, next):
4        self._element = element
5        self._prev = prev
6        self._next = next
7
8    def __init__(self):
9      self._header = self._Node(None,None,None)
10     self._trailer = self._Node(None,None,None)
11     self._header._next = self._trailer
12     self._trailer._prev = self._header
13     self._size = 0
14
15   def __len__(self):
16     return ______
```

```
17
18   def is_empty(self):
19     return ______
20
21   def _insert_between(self, e, predecessor, successor):
22     _______
23     _______
24     _______
25     _______
26     _______
```

G Complexity (6 pts)

Please answer the questions by showing the necessary calculation steps.

43. If an algorithm requires $f(x) = x^3 + x + 7$ many operations to solve a problem, what is its upper bound (i.e., big-oh notation) and show your calculation steps.
44. What is the computational complexity of the function below? Please show your calculation steps.

```
1 i=1
2 while (i <= n):
3     for j in range(i):
4         print(j)
5     i += 1
```

H Recursion (3 pts)

45. Write the recurrence function for the code snippet below. Just write the recurrence function ($T(n)$). You are not expected to solve the recurrence equation. Note that the input parameter `n` can be zero, too. So the recurrence function may have two parts.

```
1 def hanoi(n, source, dest, spare):
2     if (n > 0):
3         hanoi(n-1, source, spare, dest)
4         print(f' Move top disk from pole {source} to pole {dest}')
5         hanoi(n-1, spare, dest, source)
```

I Algorithms (7 pts.)

46. Fill in the blanks in the code snippet below so that binary search algorithm can work as expected.

```
1  def binary_search(arr, low, high, x):
2      if high >= low:
3          mid = (high + low) // 2
4          if arr[mid] == x:
5              return mid
6          elif ____ > x:
7              return binary_search(arr, ____, ____, x)
8          else:
9              return binary_search(arr, ____, ____, x)
10     else:
11         return -1
```

47. Show iteration steps of *insertion sort* for this sequence of data: $B - C - A - E - F - D$. How would this sequence look like after each iteration step? How many iteration steps would there be? (Please show each iteration step even if the array does not change.)

Good luck!