CSC 202, Fall 2023, Midterm

Name: _____

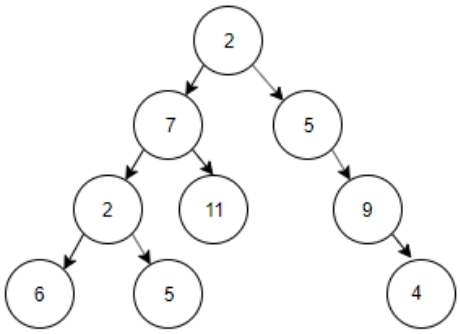
Class Time:_____

- Write the answers in the space provided in boxes for multiple choice questions.
- Please don't discuss the content of this test with others until you've received a grade on it.
- It's a close book, close notes test.
- You can't use computers, tablets or any kind of tech for this test.
- You may use all forms that you know from your homework.
- If you need a method and you don't know whether it is provided, define it.
- If you need extra space, use extra space at the back side of paper.

Good luck!

1. Multiple Choice Questions (10x1 points)

i. For the tree below, write the post-order traversal.



a)6, 2, 7, 2, 5, 11, 9, 5, 4
b) 6, 5, 11, 2, 7, 5, 9, 4, 2
c) 6, 5, 2, 11, 7, 4, 9, 5, 2
d) 6, 2, 7, 2, 11, 5, 5, 9, 4

ii. What is the runtime complexity for locating and returning the largest element (by value) in a list of N elements, assuming the values in the list are in no particular order?

a) O(1)
b) O(log N)
c) O(N)
d) O(N^2)

iii. What is the specialty about the inorder traversal of a binary search tree?
a) It traverses in a non increasing order
b) It traverses in an increasing order
c) It traverses in a random fashion
d) It traverses based on priority of the node

iv. Which of the following best describes the worst case O( ) time complexity of the following function? The parameter 'a' is a list of integers.

```
def get_answer(a):
        answer = 0
for i in range(len(a)):
        for j in range(i,len(a)):
                this_answer = (a[ j ]- a[ i ]) * (a[ j ]- a[ i ])
                if (this_answer > answer):
                answer = this_answer
        return answer
```

a) O(1)
b) O(log n)
c) O(n)
d) O(n^2)

v.
```
def fA(lst):
  for _ in range(len(lst)):
```

```
x = lst[0]
del lst[0]
lst.append(x)
```

What is the Big O of the above code

a) O(1)
b) O(log n)
c) O(n)
d) O(n^2)

vi. What is the worst-case runtime complexity of removing the first element (by index) in a doubly-linked list of N elements?

a) O(1)
b) O(log N)
c) O(N)
d) O(N log N)

vii. Given these variable definitions:

```
t = (1, 2, 3)
s = 'hello'
l = list(range(10))
```

Which of the following will result in an exception being raised?

a) s.append('!')
b) del l[2:5]
c) iter(t)
d) t[1:]

viii. What is the specialty about the inorder traversal of a binary search tree?

a) It traverses in a non-increasing order

b) It traverses in an increasing order

c) It traverses in a random fashion

d) It traverses based on priority of the node

ix. Which traversal of a Binary Search Tree would result in a sorted list of its elements?

a) Inorder

b) Preorder

c) Postorder

d) Level-order

x. Which traversal of a Binary Search Tree visits the root node first, followed by its left and right children?

a) Inorder

b) Preorder

c) Postorder

d) Level-order

2. The following programs find the sum of the numbers 1….. n using two different algorithms. For each version, describe the running time as a function of n and explain why it is what it is. (4 points)

```
def sum_of_n(n):
    the_sum = 0
    for i in range(1, n+1):
        the_sum = the_sum + i
    return ("Sum of {} numbers is {}".format(n, the_sum))


def sum_of_n2(n):
    sum = (n * (n + 1)) / 2
    return ("Sum of {} numbers is {}". format(n, sum)
```

3. Here's a Parsons Problem for a Python function that checks whether a number appears in a doubly linked list. You need to arrange the lines in the correct order to make a working function. Next to each line that you want to use, write the index of the line on which it should appear. (Remember that you don't have to use all of the lines!) (6 points)

```
_____   if head is None:
_____ new_node.prev_node = current_node
_____  current = current.next
_____ new_node.next_node = current_node.next_node
_____  current = head
_____ if current.data == number:
_____ current_node.next_node.prev_node = new_node
_____return True
_____ def has_number_in_doubly_linked_list(head, number)
_____ return False
_____ new_node.next_node = current_node.next_node
_____while current is not None:
_____ return False
```

4. Starting with an empty BST, insert the following numbers in the given order: 50, 30, 70, 20, 40, 60, 80, 10, 25, 35, 45, 55, 65, 75, 85. Draw a picture of the BST after all the insertions are complete. (2 points)

5. By using the tree drawn in question 4. Write a Python function to find the second-largest element in the BST. Just write function assume that the inputs are given to you and you are starting with the root node. (5 points)

```
def _find_second_largest(self, node):
    current = node
```

6. By using the BST in question 4. Write different steps to find the lowest common ancestor (LCA) of two given nodes in the BST. Just give steps in simple English by indexing them with numbers like Step 1: this and that Step 2: this and that and so on (5 points)

Delete the number 30 from the BST and draw the resulting tree. (2 points)

5. Draw a binary tree that contain the letters “b”, “n”, “o”, “s”, “u” such that the inorder traversal spells “bonus” and the postorder traversal spells “obuns”. (4 points)

6. Please provide constructive suggestions for this course in terms of lectures, homework, or any other aspects. (2 points)