First Name:
Last Name:
Student #

University of Toronto, Mississauga
CSC 148, Midterm Exam 2
20th March 5:10pm to 6:00pm
Good Luck!

In addition to the correct answer, you MUST show all your work in order to receive full credit.

Questions	Mark:
Question1) Multiple Choice Questions	/15
Question 2) Linked List	/15
Question 3) Binary Search Trees	/15

CIRCLE YOUR INSTRUCTOR:

T.Tiffany (10am to 11am, MWF)

A.Attarwala (11am to 12pm, MWF)

THIS EXAM CONTAINS A TOTAL OF 10 PAGES. PAGE 9-10/11 ARE THE APPENDIX OF YOUR EXAM

First Name:
Last Name:
Student #

Question 1) Multiple Choice

Note: For all multiple choice questions, in addition to the correct answer, you MUST also explain your answer correctly to receive full credit.

1.a) Given a binary search tree, which traversal type would print the values in the nodes in sorted order? [2]

a) PreOrder
b) PostOrder
c) InOrder
d) None of the above

Explain your answer (in 1 or 2 sentences):

2.a) What is the running time of the following code fragment? [3]

```
def printValues(N):
        for i in range (0,10,1):
            for j in range (0,N,1):
              for k in range (N-2,N+2,1):
                print ("HelloWorld!")
```

a) O(log N)
b) O(N log N)
c) O(N)
d) O(N^2)
e) O(N^3)

Explain your answer (in 1 or 2 sentences):

3.a) Which of the following statements about binary trees is NOT true? [1]

a) Every binary tree has atleast one node.
b) Every non empty tree has exactly one root node.
c) Every node has at most two children.
d) Every non root node has exactly one parent.

Explain your answer (in 1 or 2 sentences):

4.a) How many times is the string “foo” printed by the function call `foo(4)` ? [3]

```
def foo(i):
    if i>1:
        foo(i/2)
        foo(i/2)
    print "foo"
```

a) 3
b) 4
c) 7
d) 8
e) Something else

Explain your answer (in 1 or 2 sentences):

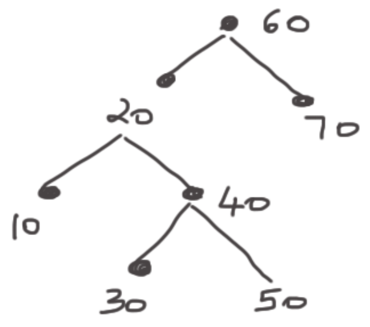
First Name:
Last Name:
Student #

5.a) Which of the following rankings orders the algorithms from fastest to slowest? [2]

a) $O(N)$, $O(\log_2 N)$, $O(N^2)$, $O(2^N)$, $O(N!)$
b) $O(\log_2 N)$, $O(N)$, $O(N^2)$, $O(2^N)$, $O(N!)$
c) $O(\log_2 N)$, $O(N)$, $O(2^N)$, $O(N^2)$, $O(N!)$
d) $O(\log_2 N)$, $O(N)$, $O(N!)$, $O(N^2)$, $O(2^N)$
e) $O(\log_2 N)$, $O(N)$, $O(N^2)$, $O(N!)$, $O(2^N)$

Explain your answer (in 1 or 2 sentences):

6) Refer to the following tree when answering 6.a) and 6.b) of the multiple choice question.



6.a) What would the result of a *pre-order* traversal of the above tree? [2]

a) 60 20 10 40 30 50 70
b)10 20 30 40 50 60 70
c)10 30 50 40 20 70 60
d) 70 60 50 40 30 20 10
e) None of the above

Explain your answer (in 1 or 2 sentences):

6.a) What would the result of a *post-order* traversal of the above tree? [2]

a) 60 20 10 40 30 50 70
b)10 20 30 40 50 60 70
c)10 30 50 40 20 70 60
d) 70 60 50 40 30 20 10
e) None of the above

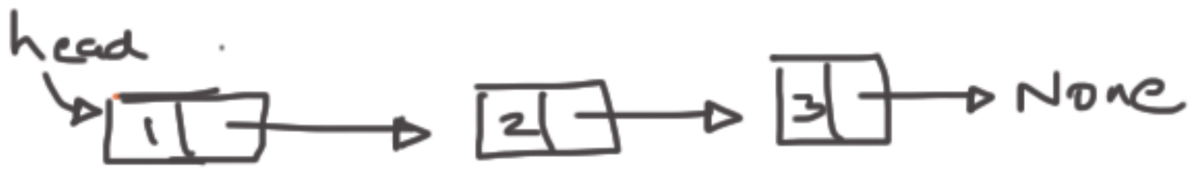
Explain your answer (in 1 or 2 sentences):

First Name:
Last Name:
Student #

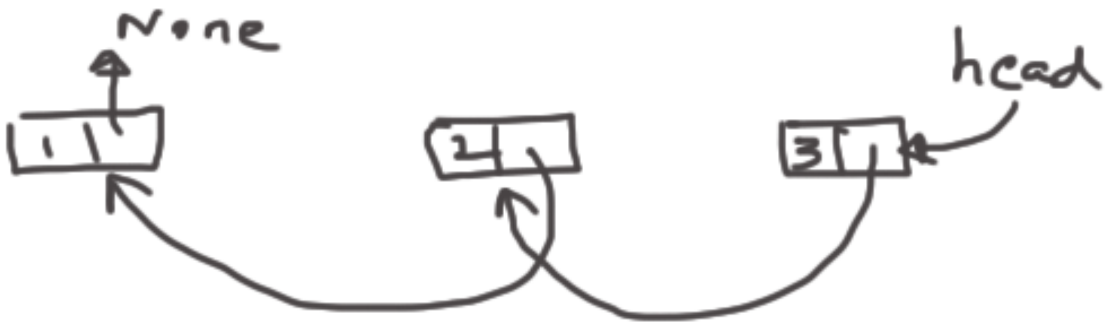
Question 2) **[15]**

Given any arbitrary singly linked list, write a function that reverses the linked list. Your solution MUST be recursive and NOT iterative. You cannot use a stack or a queue to answer this question.

i.e. for the following linked list:



The reversal of the above linked list is as follows:



```
def reverseList(head, prev=None):
    '''head is the reference to the head
    node of the linked list. prev is the reference
    to the node just before head. prev is set to None
    when this function is called for the very first time'''
```

First Name:
Last Name:
Student #

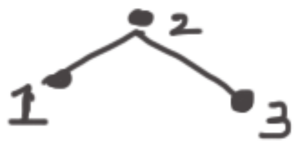
You can use this page for Question2)

First Name:
Last Name:
Student #

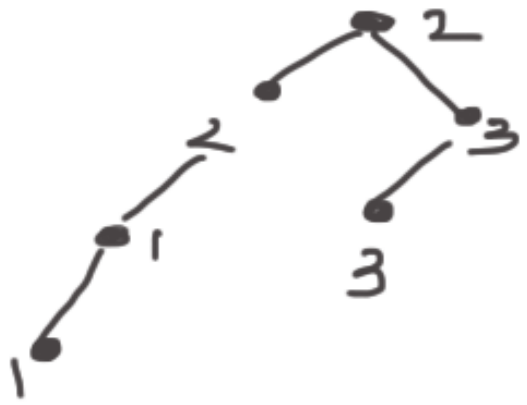
Question 3) Binary Search Trees: [15]

You may first want to revisit the definition of binary search tree (in the appendix) before answering this question.

For each node in a binary search tree, create a new duplicate node, and insert the duplicate as the left child of the original node. The resulting tree MUST still be a binary search tree.



will be changed to:



You MUST use the nodes and references version of binary search trees to answer this question.

```
def doubleTree(t):
    '''t is the reference to the BST. After this function returns t MUST refer to the
resulting BST tree as defined in the specs of the question'''
```

First Name:
Last Name:
Student #

You can use this page for Question3)

Appendix

For Question1, 2a)

Python's range() Parameters

The range() function has two sets of parameters, as follows:

range(stop)
stop: Number of integers (whole numbers) to generate, starting from zero. eg. range(3) == [0, 1, 2].

range([start], stop[, step])
start: Starting number of the sequence.
stop: Generate numbers up to, but not including this number.
step: Difference between each number in the sequence.

For Question 2):

```
class Node:
    def __init__(self, e,n=None):
        '''e is the integer that will reside in the node.
        n is the reference to the next node'''
        self.element = e
        self.next = n
    def __str__(self):
        return str(self.element)
```

For Question 3):

A binary search tree is a binary tree, with the following constraints on each node P of the binary tree.

Value stored on the left subtree of P are **less than or equal** to P.
Value stored on the right subtree of P are greater than P.

The equality constraint has been added which allows the binary search tree to contain duplicate elements in its left tree.

```
class BinarySearchTree:
 '''You can safely assume that this class has been
          correctly and fully implemented as per the above definition of BST.'''

    def __init__(self,rootValue):
        '''Creates a new BST where, rootValue refers to the integer value at root
        node. The left child is set to None, and the right child is set to None'''
        self.root=rootValue
        self.left=None
        self.right=None
    .
    .
    .
```

From the above code of Binary Search Tree:

`t.root` refers to the integer value at the root node of the tree `t`.
`t.left` is a reference to the left subtree of tree `t`.
`t.right` is a reference to the right subtree of tree `t`.

You can directly refer to the instance variables of the binary search tree when writing your code.