Duration: **50 minutes**
Aids Allowed: ***None***

Student Number: |_|_|_|_|_|_|_|_|_|_|

Last (Family) Name(s): ____________________

First (Given) Name(s): ____________________

---

*Do **not** turn this page until you have received the signal to start.*
In the meantime, please read the instructions below *carefully*.

Please put an x beside your tutorial section

☐	TUT001	Ainsley Lawson	MO 13:00-15:00
☐	TUT002	Ainsley Lawson	MO 15:00-17:00
☐	TUT003	Joyce Woo	TU 09:00-11:00
☐	TUT004	Kevin Lin	TU 09:00-11:00
☐	TUT005	Bonnie Han	TU 10:00-12:00
☐	TUT006	Jacey Wu	WE 12:00-14:00
☐	TUT007	Leo Zhao	WE 13:00-15:00
☐	TUT008	Jacey Wu	WE 14:00-16:00
☐	TUT009	Johnny Scialdone	WE 15:00-17:00
☐	TUT010	Philip Yang	TU 11:00-13:00
☐	TUT011	Bonnie Han	WE 15:00-17:00

This term test consists of 3 questions on 8 pages (including this one), printed on both sides of the paper. *When you receive the signal to start, please make sure that your copy of the test is complete, fill in the identification section above, and write your name on the back of the last page.*

Answer each question directly on the test paper, in the space provided, and use one of the "blank" pages for rough work. If you need more space for one of your solutions, use a "blank" page and *indicate clearly the part of your work that should be marked.*

MARKING GUIDE

# 1:	______/10
# 2:	______/20
# 3:	______/20
TOTAL:	______/50

Good Luck!

For the duration of this exam, you can assume that you have access to the following Python classes:

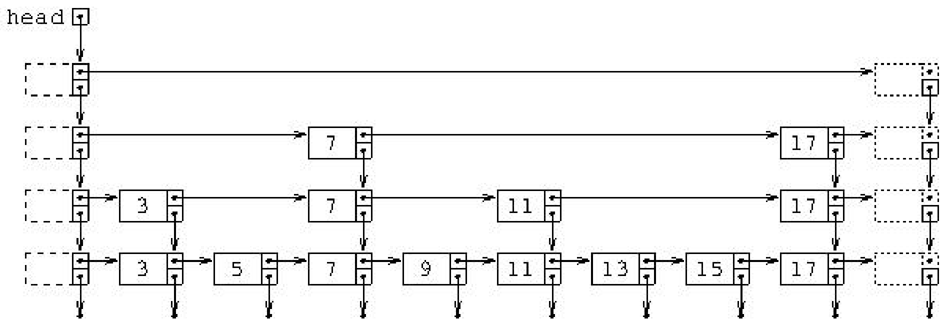
```
class LLNode(object):
    '''A Node in a singly-linked list'''

    def __init__(self, data):
        '''(LLNode, object) -> NoneType
        Create a new node to hold data
        '''
        self.data = data
        self.link = None


class DLLNode(object):
    '''A Node in a doubly-linked list'''

    def __init__(self, data):
        '''(DLLNode, object) -> NoneType
        Create a new node to hold data
        '''
        self.data = data
        self.next = None
        self.prev = None


class BTNode(object):
    '''A Node in a binary tree''

    def __init__(self, data):
        '''(BTNode, object) -> NoneType
        Create a new node to hold data
        '''
        self.data = data
        self.left = None
        self.right = None
```

Question 1. [10 marks]

Given the skip-list below, draw a representation of what the list would look like after the following series of operations:

- `delete(5)`
- `delete(11)`
- `insert(14)`

You can assume that `random.random()` will return the following series of random numbers (in order):
`0.25, 0.15, 0.01, 0.73, 0.10, 0.95, 0.02, 0.74, 0.53`



Use the space on this "blank" page for scratch work, or for any solution that did not fit elsewhere.
Clearly label each such solution with the appropriate question and part number.

Question 2. [20 marks]

Write the body of the function below so that it satisfies its docstring.

```
def make_double_copy(ll_head):
    """(LLNode) -> DLLNode
    Return a the head of a properly connected doubly-linked list with
    the same elements in the same order, as those in the singly-linked
    list beginning with ll_head
    """
    # Hint: Draw some examples first
```

Use the space on this "blank" page for scratch work, or for any solution that did not fit elsewhere.
Clearly label each such solution with the appropriate question and part number.

Question 3. [20 marks]

Write the body of the function below so that it satisfies its docstring Your code *must* be recursive, and in order to obtain full marks, it must visit each node at most 1 time.

```
def tree_average(root):
    """(BTNode) -> float
    Return the average value of all nodes in the tree rooted at root
    REQ: all nodes in the tree contain float values
    """
    #HINT: You can create a helper function
```

On this page, please write nothing except your name.

Last (Family) Name(s):	
First (Given) Name(s):	

Total Marks = 50