Name: ALFREDO PEREZ
COSC 40403 - Analysis of Algorithms: Fall 2020: Homework 1
Due: 23:30 on 8/24

Question	Points	Score
1	10	
2	10	
3	10	
4	10	
5	10	
Total:	50	

For questions 1-3, use the following Node definition. This Node definition is written as a Python class.

```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None
```

1. (10 points) Write a function (Python) to concatenate two linked lists. Given lists $l1 = \langle 2, 3, 1 \rangle$ and $l2 = \langle 4, 5 \rangle$, after return from LIST_CONCATENATE($l1$, $l2$) the list $l1$ should be changed to $l1 = \langle 2, 3, 1, 4, 5 \rangle$. Your function should not change $l2$ and should not directly link nodes from $l1$ to $l2$ (i.e. the nodes inserted into $l1$ should be copies of the nodes in $l2$).

Solution:

```
// Concatenate Two Linked Lists

def copy(self):
   if self.next is None:
      return Cell(self.data)
   else:
      return Cell(self.data, self.next.copy()

def list_concatenate(l1, l2):
   new = l1.copy()

   # find the end of the copy
   last = new
   while last.next is not None:
      last = last.next
   # append a copy of the other list
   last.next = l2.copy()
```

2. (10 points) Write a function to insert a number as the new i^{th} node of a linked list. Nodes initially in positions i, $i+1$, ..., n shoudl be shifted to positions $i+1$, $i+2$, ..., $n+1$. Thus, the length of the list will increase by 1. If the original list contains fewer than $i-1$ nodes, then the number should be inserted at the end of the list.

Solution:

```
def InsertNth(head, data, position):
   start = head
   if position == 0:
      return Node(data, head)
   while position > 1:
      head = head.next
      position -= 1
   head.next = Node(data, head.next)
   return start
```

3. (10 points) Write a function to remove duplicate entries in a linked list. For example, given the list $\langle 5, 2, 2, 5, 3, 9, 2 \rangle$ as input, your function should change the list so that on return from the function it contains $\langle 5, 3, 9, 2 \rangle$.

Solution:

```
def removeDuplicates(self):
   temp = self.head
   if temp is None:
      return
   while temp.next is not None:
      if temp.data == temp.next.data:
         new = temp.next.next
         temp.next = None
         temp.next = new
      else:
         temp = temp.next
   return self.head
```

4. (10 points) (Problem 1.1 on page 22) Decide whether you think the following statement is true or false. If it is true, give a short explanation. If it is false, give a counterexample.

True or false? In every instance of the Stable Matching Problem, there is a stable matching containing a part (m, w) such that m is ranked first on the preference list of w and w is ranked first on the preference list of m.

Solution: True, in every instance of the Stable Matching Problem, there is a stable matching containing a part (m, w) such that m is ranked first on the preference list of w and w is ranked first on the preference list of m.

5. (10 points) (Problem 1.3 on page 22-23) See the textbook for a description of the problem.

Solution: The statement is True. In the stable matching problem the couples are stable when all couples find the other that is equal in its preference. This would mean that when m and w rank first on their preference list they are stable, which is possible in every instance.