Faculty of
Information Technology
King Mongkut's Institute of Technology Ladkrabang

Name:.....

Student ID:.....

Assignment # 4: Linked Lists

1. In your own words, describe the definition of a linked list and its properties.

2. What are the main differences between Arrays and Linked Lists?

3. Given a singly linked list with elements as follows:

The header node contains a counter to keep track the number of elements in the list. Suppose that it takes 1 byte to store a reference to string value, a draft memory table with references to value is provided below. The header node is located at address 6009 which is also where the counter keeps record.

3.1 Update the memory table to include the next reference/pointer for all nodes.

Address/Byte#	Value
6001	DS-1
6002	
6003	Tatooine
6004	
6005	
6006	
6007	Yavin 4
6008	
6009	4
6010	6003
6011	
6012	
6013	Alderaan
6014	

3.2 Copy the result from Question 3.1 and then add a new node which has an element “Hoth” between “Yavin 4” and “DS-1” nodes. Update the memory table in which Hoth is located at address 6011.

 # Draw an updated linked list here (Use of abbreviation for each node is allowed such as T for Tatooine)

Address/Byte#	Value
6001	DS-1
6002	
6003	Tatooine
6004	
6005	
6006	
6007	Yavin 4
6008	
6009	
6010	6003
6011	
6012	
6013	Alderaan
6014	

3.3 Copy the result from Question 3.2 and then remove two nodes: one of which has an element “Alderaan”, and another is “DS-1”. Update the memory table accordingly.

 # Draw an updated linked list here

Address/Byte#	Value
6001	
6002	
6003	Tatooine
6004	
6005	
6006	
6007	Yavin 4
6008	
6009	
6010	6003
6011	
6012	
6013	
6014	

3.4 Copy the result from Question 3.3 and then insert a node which has an element “Dagobah” as a new head node (Not to be confused with header node!). Update the memory table in which Dagobah is located at address 6005.

 # Draw an updated linked list here

Address/Byte#	Value
6001	
6002	
6003	Tatooine
6004	
6005	
6006	
6007	Yavin 4
6008	
6009	
6010	6003
6011	
6012	
6013	
6014	

3.5 Copy the result from Question 3.4 and then insert a node which has an element “Endor” as a new tail node. Update the memory table in which Endor is located at address 6013.

 # Draw an updated linked list here

Address/Byte#	Value
6001	
6002	
6003	Tatooine
6004	
6005	
6006	
6007	Yavin 4
6008	
6009	
6010	6003
6011	
6012	
6013	
6014	

4. Given a class of **singly** linked list node which is defined as follows:

```
class _Node:
"""Lightweight, nonpublic class for storing a singly linked node."""
     __slots__ = '_element' , '_next'
     def __init__ (self, element, next):
        self._element = element       # reference to user's element
        self._next = next             # reference to next node
```

A method to create an empty Node is:

N = _Node(None, None)

Here is an example of how to create a linked list with two nodes:

N1 = _Node("Node 1", None)
N2 = _Node("Node 2", N1) # N2's next pointer refer to N1

4.1 Write Python code or pseudocode to create all the following nodes and their pointers accordingly.

4.2 Based on the result from 4.1, write Python code or pseudocode to insert/delete nodes and update their pointers accordingly as depicted in the figure below.

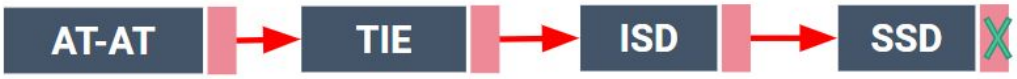
Here is an example of how to update a pointer where N1 point to N2:

```
N1._next = N2
```

4.3 Based on the result from 4.2, write Python code or pseudocode to insert/delete nodes and update their pointers accordingly as depicted in the figure below.

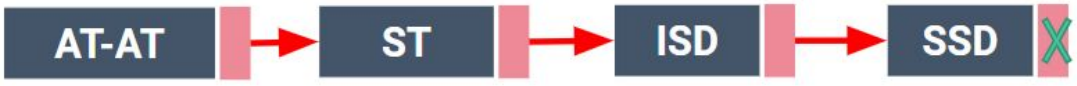
Here is an example of how to remove a node:

```
N1._element = None
N1._next = None
```



4.4 Based on the result from 4.3, write Python code or pseudocode to insert/delete nodes and update their pointers accordingly as depicted in the figure below.

4.5 Based on the result from 4.4, write Python code or pseudocode to insert/delete nodes and update their pointers accordingly as depicted in the figure below.



5. Based on a singly linked list class as defined in Question 4, draw an output linked list after the following Python codes for various tasks.

5.1 N1 = _Node(“75192”, None)

N2 = _Node(“75252”, N1)

N3 = _Node(“75159”, N2)

N4 = _Node(“10221”, N3)

5.2 N4._next._next._next._element = 10179

N3._next._next = N4

N4._next = N1

5.3 N3._next = None

N2._next = N4._next

N1._next = N4

N1._next._next = N3

6. Given a class of **doubly** linked list node which is defined as follows:

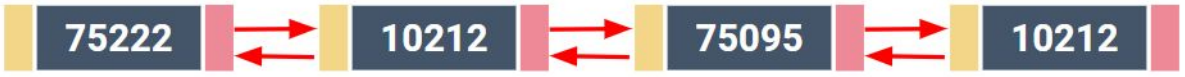
```
class _Node:
"""Lightweight, nonpublic class for storing a singly linked node."""
  __slots__ = '_element' , '_next', '_prev'
  def __init__ (self, element, prev, next):
    self._element = element             # reference to user's element
    self._prev = prev                   # reference to previous node
    self._next = next                   # reference to next node
```

Here is an example of how to create a doubly linked list with two nodes:

N1 = _Node("Node 1", N2, None) # N1's prev pointer refer to N2
N2 = _Node("Node 2", None, N1) # N2's next pointer refer to N1

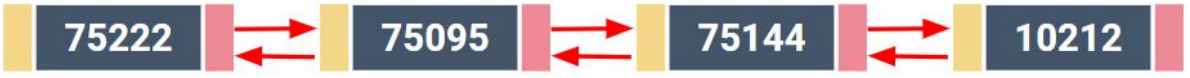
6.1 Write Python code or pseudocode to create all the following nodes and their pointers accordingly.

6.2 Based on the result from 6.1, write Python code or pseudocode to insert/delete nodes and update their pointers accordingly as depicted in the figure below.



6.3 Based on the result from 6.2, write Python code or pseudocode to insert/delete nodes and update their pointers accordingly as depicted in the figure below.

6.4 Based on the result from 6.3, write Python code or pseudocode to insert/delete nodes and update their pointers accordingly as depicted in the figure below.



7. Using Python or pseudocode, write an algorithm for **sequential search** on a singly linked list. The input is the head node and the value to find. The expected output is True if a search is found, False if the element is nowhere to be found.

Hint: Use while loop to traverse the linked list from the beginning to the end. Also, a temporary variable is required to keep track of the current node.

Hint 2: The algorithm should return False if the list is empty

Example input: findNode(head_N, 20) where head_N is a head node in a linked list

Expected output: True or False

```
class _Node:
"""Lightweight, nonpublic class for storing a singly linked node."""
  __slots__ = '_element' , '_next'
 def __init__ (self, element, next):
      self._element = element             # reference to user's element
      self._next = next                   # reference to next node
```

def findNode(head, value):

""" Find value in a linked list """

#Add your code here

8. Using Python or pseudocode, implement the missing methods of LinkedDeque class in Week 5's Code Examples with **doubly** linked list in which its methods are defined in **_DoublyLinkedBase** class.

Hint: Have a look at the given codes for first(), insert_first(), and delete_first() methods.

```
def last(self):
  """ Return (but do not remove) the element at the back of the deque."""
  # Add your code here
  # Hint: Use inherited method from _DoublyLinkedBase class
```

```
def insert_last(self, e):
  """ Add an element to the back of the deque."""
  # Add your code here
  # Hint: Use inherited method from _DoublyLinkedBase class
```

```
def delete_last(self):
  """ Remove and return the element from the front of the deque.
  Raise Empty exception if the deque is empty."""
  # Add your code here
  # Hint: Use inherited method from _DoublyLinkedBase class
```