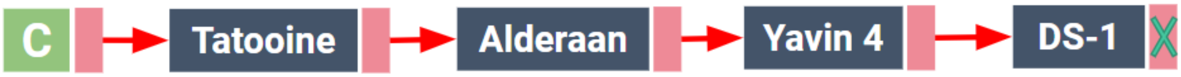
Assignment # 6: Linked Lists

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

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



Suppose that it takes 1 byte to store a reference to a string value, a draft memory table with references to value is provided below. Therefore, in this case, a node consists of 2 addresses that are next to each other: one for reference to a value, another for reference to a next node. The header node is located at address 6009 which is also where the counter keeps record of the number of elements in the list.

2.1 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	

2.2 Copy the result from Question 2.1 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	

2.3 Copy the result from Question 2.2 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	

2.4 Copy the result from Question 2.3 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	?
6011	
6012	
6013	
6014	

2.5 Copy the result from Question 2.4 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	?
6011	
6012	
6013	
6014	

3. 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:

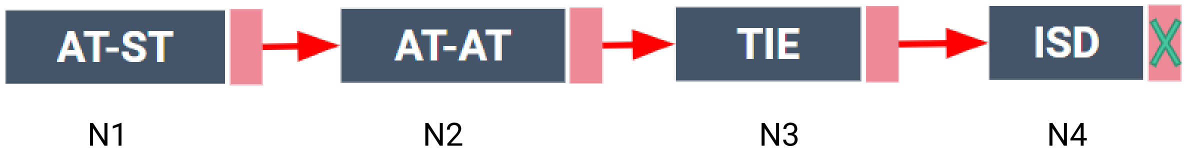
N = _Node(None, None)

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

N2 = _Node(“Node 2”, None)   # Tail node
N1 = _Node(“Node 1”, N2)   # Current head node, N1’s next pointer refer to N2

Hint: it is suggested to create a tail node first because when a new node has to point to a next node, that next node should exist first. In other words, a node has to be created first before being referred to.

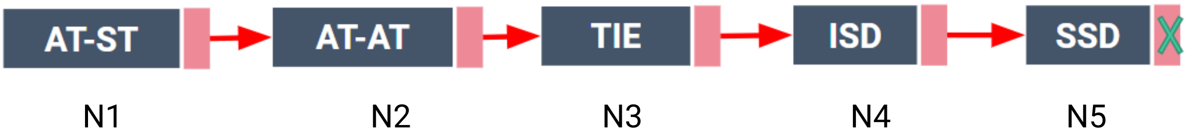
3.1 Write Python code or pseudocode to create all the following nodes and their pointers accordingly.



3.2 Based on the result from 3.1, write an additional 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:

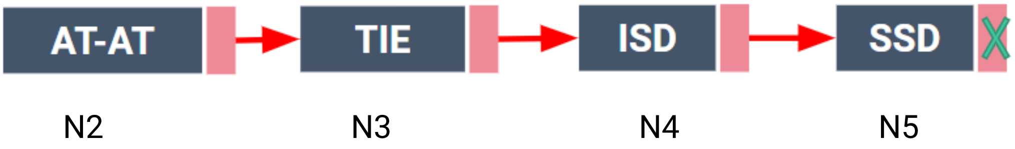
```
N1._next = N2
```



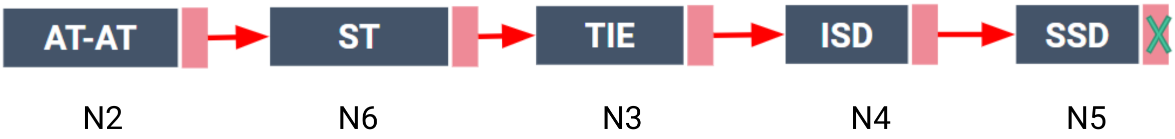
3.3 Based on the result from 3.2, write an additional Python code or pseudocode to insert/delete nodes and update their pointers accordingly as depicted in the figure below.

Here is an example of how to remove a node:

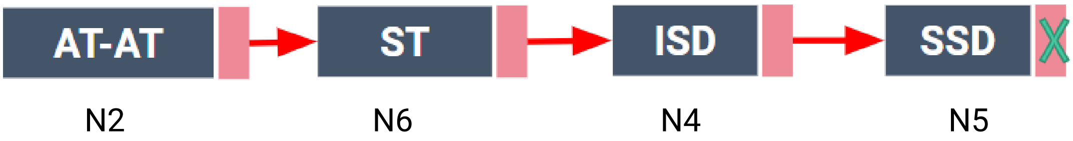
```
N1._element = None
N1._next = None
```



3.4 Based on the result from 3.3, write an additional Python code or pseudocode to insert/delete nodes and update their pointers accordingly as depicted in the figure below.



3.5 Based on the result from 3.4, write an additional Python code or pseudocode to insert/delete nodes and update their pointers accordingly as depicted in the figure below.



4. Based on a singly linked list class as defined in Question 3, draw an output linked list after the following Python codes.

4.1
```
N1 = _Node("75192", None)
N2 = _Node("75252", N1)
N3 = _Node("75159", N2)
N4 = _Node("10221", N3)
```

4.2
```
N4._next._next._next._element = 10179
N3._next._next = N4
N4._next = N1
```

4.3

```
N3._next = None

N2._next = N4._next

N1._next = N4

N1._next._next = N3
```

5. According to the singly linked list lecture, describe the algorithm steps to delete a node between nodes.

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

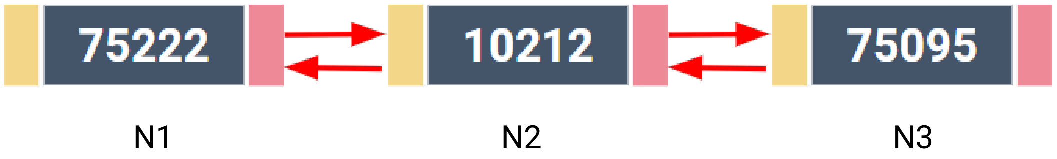
```
class  _Node:
"""Lightweight, nonpublic class for storing a singly linked node."""
  __slots__ = '_element' , '_next', '_prev'
  def __init__ (self, element, prev, next):
    self._element = element              # reference to user’s element
    self._prev = prev                    # reference to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

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

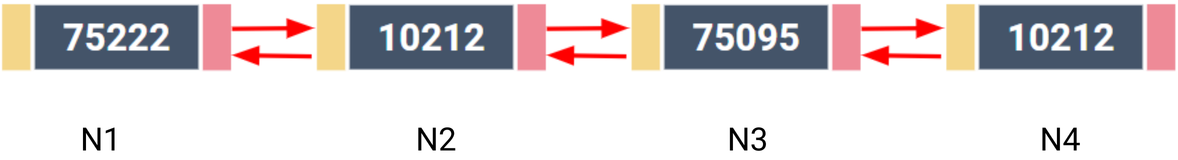
```
N1 = _Node(“Node 1”, None, None)        # N1
N2 = _Node(“Node 2”, None, N1)          # N2’s next pointer refer to N1
N1._prev = N2                           # Set N1’s prev pointer refer to N2
```

For N1, the pointers are initially set to None because a node has to be created first before being referred to.

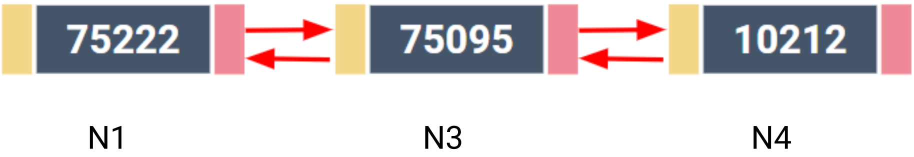
6.1 Write Python code or pseudocode to create all the following nodes and their pointers accordingly.



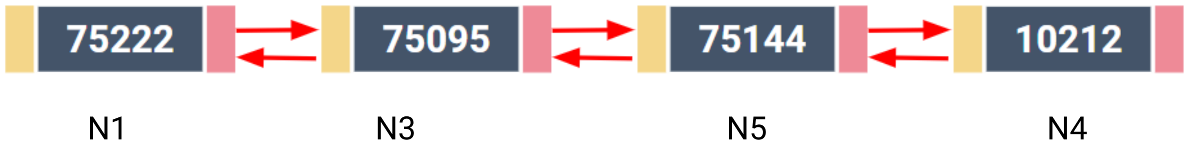
6.2 Based on the result from 6.1, write an additional Python code or pseudocode to insert/delete nodes and update their pointers accordingly as depicted in the figure below.



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


75222
75095
75144
10212
N1
N3
N5
N4