Problem Set 1

Name: Vivek Verma

Collaborators: None

Problem 1-1.

(a) $(f_5, f_3, f_4, f_1, f2)$

(b) $(f_1, f_2, f_5, f_4, f_3)$

(c) $(\{f_2, f_5\}, f_4, f_1, f_3)$

(d) $(f_5, f_3, f_4, f_2, f_1)$ WRONG
$(f_5, f_2, f_1, f_3, f_4)$ From solution

Problem 1-2.

(a)

(b)

Problem 1-3.

Problem 1-4.

(a) $insert\_first(x)$
Create a Node containing the item x.
Check the head of the list to see if list is empty.
If list is empty, set the Node as head and tail of the list and return.
If list is non-empty, get the head of the list called $first$, Connect $first$ as next element of x and x as previous element of $first$.
Set the head of the list to x.

$insert\_last(x)$
Create a Node containing the item x.
Check the tail of the list to see if list empty.
If list is empty, set the Node as head and tail of the list and return.
If list is non-empty, get the tail of list called $last$, Connect $last$ as previous element of x and x as next element of $last$.
Set the tail of the list to x.

$delete\_first()$
Get the first two elements as $first$ and $second$.
Set $second$ as the head of the list and set $second.prev$ as None.
Return $first.item$.

$delete\_last()$
Get the last two elements as $last$ and $second\_last$.
Set $second\_last$ as tail of the list and set $second\_last.next$ as None.
Return $last.item$.

(b) Construct a new empty list $L2$.
There are four cases based on the location of x_1 and x_2 in the list:

1. Neither x_1 is head nor x_2 is tail:
   Connect the element before x_1 to the element after x_2.
2. x_1 is the head:
   Set the head of the list to the element after x_2 and set the previous element of head as None.
3. x_2 is the tail:
   Set the tail of the list to the element before and set the next element of tail as None.
4. x_1 is the head and x_2 is the tail:
   Set the head and tail of the list to None.

Independent of which case above was executed,
Set the previous element of x_1 and next element of x_2 as None.
Set the head of $L2$ to x_1 and the tail of $L2$ to x_2.

(c) If $L2$ is an empty list, nothing needs to be done, hence return None.
Otherwise,
Get the element after x in a variable x_next.
Set the next element of x to head of $L2$ and previous element of $L2.head$ to x.

If x_next is not None, Set previous element of x_next to tail of $L2$ and next next element of $L2's$ tail to x_next.
Set the tail of L as tail of $L2$.
Set head and tail of $L2$ to None.

(d)

```
    def insert_first(self, x):
        # create node with item x
        x = Doubly_Linked_List_Node(x)

        # if list is empty, add x as only element
        if self.head is None:
            self.head = x
            self.tail = x
            return

        # get the first element and connect it to x
        first = self.head
        first.prev = x

        # connect x to first
        x.next = first

        # set x as the head
        self.head = x

    def insert_last(self, x):
        # create node with item x
        x = Doubly_Linked_List_Node(x)

        # if list is empty add, x as the only element
        if self.tail is None:
            self.tail = x
            self.head = x

        # get the last element and connect it to x
        last = self.tail
        last.next = x

        # connect x to last element
        x.prev = last

        # set x as the tail
        self.tail = x

    def delete_first(self):
        # get first two elements
        first = self.head
        second = first.next

        # set second element as head ot the list
        second.prev = None
        self.head = second
        return first.item

    def delete_last(self):
        # get last two elements
```

```
52          last = self.tail
53          second_last = last.prev
54
55          # set second last element as tail of the list
56          second_last.next = None
57          self.tail = second_last
58          return last.item
59
60      def remove(self, x1, x2):
61          L2 = Doubly_Linked_List_Seq()
62
63          # Neither x1 nor x2 is the head or tail
64          if x1.prev is not None and x2.next is not None:
65              # connect the element before x1 to the element after x2
66              x1.prev.next = x2.next
67              x2.next.prev = x1.prev
68          # if x1 is the head but x2 is not tail
69          elif x1.prev is None and x2.next is not None:
70              # disconnect elements x1 to x2 from the list
71              self.head = x2.next
72              self.head.prev = None
73          # if x2 is the tail but x1 is not head
74          elif x2.next is None and x1.prev is not None:
75              # disconnect elements from x1 to x2 from the list
76              self.tail = x1.prev
77              self.tail.next = None
78          # x1 is head and x2 is tail
79          else:
80              self.head = None
81              self.tail = None
82
83          # disconnect the links to any element before x1 and any elment
84          x1.prev = None
85          x2.next = None
86
87          # set the head and tail of L2
88          L2.head = x1
89          L2.tail = x2
90          return L2
91
92      def splice(self, x, L2):
93          # if L2 is empty, return None
94          if L2.head is None:
95              return
96
97          # get the element after x
98          x_next = x.next
99
100         # connect x to head of L2
101         x.next = L2.head
102         L2.head.prev = x
```

```
103
104         # there was an element after x, connect it to tail of L2
105         if x_next is not None:
106             x_next.prev = L2.tail
107             L2.tail.next = x_next
108
109         # set the tail of current list as tail of L2
110         self.tail = L2.tail
111
112         # remove all elements from L2
113         L2.head = None
114         L2.tail = None
```