```python
from exceptions import Empty

class DoublyLinkedList:
    class _Node:
        __slots__ ='_element','_prev','_next'
        def __init__(self,element,prev,next):
            self._element = element
            self._prev = prev
            self._next = next
    def __init__(self):
        self._head = self._Node(None,None,None)
        self._tail = self._Node(None,None,None)
        self._head._next = self._tail
        self._tail._prev = self._head
        self._size = 0
    def __len__(self):
        return self._size
    def is_empty(self):
        return self._size == 0

    def add_first(self, e):
        newest = self._Node(e, None, None)
        if self.is_empty():
            self._head = newest
            self._tail = newest
        else:
            newest._next = self._head
            self._head._prev = newest
        self._head = newest
        self._size +=1

    def add_last(self,e):
        newest = self._Node(e,None,None)
        if self.is_empty():
            self._head = newest
            self._tail = newest
        else:
            self._tail._next = newest
            newest._prev = self._tail
        self._tail = newest
        self._size += 1

    def add_any(self,e,pos):
        newest = self._Node(e,None, None)
        thead = self._head
        i = 1
        while i < pos:
```

```
48              thead = thead._next
49              i += 1
50          newest._next = thead._next
51          thead._next = newest
52          thead._next._prev = newest
53          newest._prev = thead
54          self._size += 1
55
56      def remove_first(self):
57          if self.is_empty():
58              raise Empty('List is Empty')
59          value = self._head._element
60          self._head = self._head._next
61          self._head._prev = None
62          self._size -= 1
63          if self.is_empty():
64              self._tail = None
65          return value
66
67      def remove_last(self):
68          if self.is_empty():
69              raise Empty('List is Empty')
70          thead = self._head
71          i = 0
72          while i < len(self)-2:
73              thead = thead._next
74              i += 1
75          self._tail = thead
76          thead = thead._next
77          value = thead._element
78          self._tail._next = None
79          self._size -= 1
80          return value
81
82      def remove_any(self, pos):
83          if self.is_empty():
84              raise Empty('List is Empty')
85          thead = self._head
86          i = 1
87          while i < pos-1:
88              thead = thead._next
89              i += 1
90          thead._next = thead._next._next
91          thead._next._next._prev = thead
92          self._size -= 1
93
94      def display(self):
```

```
 95          thead = self._head
 96          while thead:
 97              print(thead._element, end='-->')
 98              thead = thead._next
 99          print()
100
101
102 L = DoublyLinkedList()
103 L.add_last(10)
104 L.add_last(20)
105 L.add_last(30)
106 L.add_last(40)
107 L.display()
108 print('Delete: ',L.remove_first())
109 L.display()
110 L.add_first(70)
111 L.display()
112 print('Delete: ',L.remove_last())
113 L.display()
114 L.add_any(100,2)
115 L.display()
116 L.remove_any(2)
117 L.display()
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
```