

LAB#10

Example#1: Write a program to create a deque (double ended queue) by using list.

Solution:

```
1  class Deque:
2      def __init__(self):
3          self.items = []
4
5      def is_empty(self):
6          return len(self.items) == 0
7
8      def push_front(self, item):
9          self.items.insert(0, item)
10
11     def push_back(self, item):
12         self.items.append(item)
13
14     def pop_front(self):
15         if self.is_empty():
16             print("Deque is empty")
17             return None
18         return self.items.pop(0)
19
20     def pop_back(self):
21         if self.is_empty():
22             print("Deque is empty")
23             return None
24         return self.items.pop()
25
26     def peek_front(self):
```

```

27         if self.is_empty():
28             print("Deque is empty")
29             return None
30         return self.items[0]
31
32     def peek_back(self):
33         if self.is_empty():
34             print("Deque is empty")
35             return None
36         return self.items[-1]
37
38     def display(self):
39         if self.is_empty():
40             print("Deque is empty")
41             return
42         print("Items:", self.items)
43         print("Front:", self.items[0], "& Back:", self.items[-1])
44
45     d1 = Deque()
46     d1.push_front(1)
47     d1.push_back(2)
48     d1.push_front(3)
49     d1.push_back(4)

```

```

50     d1.display()
51     element=d1.pop_front()
52     print('The popped element is:',element)
53     d1.display()
54     element=d1.pop_back()
55     print('The popped element is:',element)
56     d1.display()

```

Result:

```

Items: [3, 1, 2, 4]
Front: 3 & Back: 4
The popped element is: 3
Items: [1, 2, 4]
Front: 1 & Back: 4
The popped element is: 4
Items: [1, 2]
Front: 1 & Back: 2

```

Example#2: Write a program to create a deque (double ended queue) by using doubly linked list.

Solution:

```
1  class Node:
2      def __init__(self,pre=None,item=None,next=None):
3          self.pre=pre
4          self.item=item
5          self.next=next
6  class Queue:
7      def __init__(self):
8          self.rear=None
9          self.front=None
10
11         self.itemcount=0
12     def is_empty(self):
13         return self.itemcount==0
14
15         def insert_front(self,data):
16             n=Node(None,data,self.front)
17             if self.is_empty():
18                 self.rear=n
19             else:
20                 self.front.pre=n
21                 self.front=n
22                 self.itemcount+=1
23         def insert_rear(self,data):
24             n=Node(self.rear,data,None)
25             if self.is_empty():
26                 self.front=n
27             else:
28                 self.rear.next=n
29                 self.rear=n
30                 self.itemcount+=1
```

```
30     def delete_front(self):
31         if self.is_empty():
32             print('deque is empty')
33         elif self.front==self.rear:
34             self.front=None
35             self.rear=None
36         else:
37             self.front=self.front.next
38             self.front.pre=None
39         self.itemcount-=1
40     def delete_rear(self):
41         if self.is_empty():
42             print('deque is empty')
43         elif self.front==self.rear:
44             self.front=None
45             self.rear=None
```

```
46         else:
47             self.rear=self.rear.pre
48             self.rear.next=None
49         self.itemcount-=1
50     def getfront(self):
51         if self.is_empty():
52             print('deque is empty')
53         else:
54             return self.front.item
55     def getrear(self):
56         if self.is_empty():
57             print('deque is empty')
58         else:
59             return self.rear.item
```

```
60     def size(self):
61         return self.itemcount
62
63     def print_queue(self):
64         if self.is_empty():
65             print('deque is empty')
66         else:
67             temp = self.front
68             while temp is not None:
69                 print(temp.item, end=' ')
70                 temp = temp.next
71
72 q1=Queue()
73 q1.insert_front(10)
74 q1.insert_rear(20)
75 q1.insert_rear(30)
76 q1.insert_front(110)
```

```
76 print('The list is:')
77 q1.print_queue()
78 print('\nsize:',q1.size())
79 print('front:',q1.getfront())
80 print('rear:',q1.getrear())
81 q1.delete_rear()
82 print('\nAfter deleting rear')
83 print('The list is:')
84 q1.print_queue()
85 q1.delete_front()
86 print('\nAfter deleting front:')
87 print('The list is:')
88 q1.print_queue()
```

Result:

The list is:

110 10 20 30

size: 4

front: 110

rear: 30

After deleting rear

The list is:

110 10 20

After deleting front:

The list is:

10 20

Class Assignment

Q.1: Modify the above program example#1 lab#10 by using array to create the deque.

Q.2: Modify the above program example#2 lab#10 by using singly linked list to create the deque.

Q.3: Modify the print_queue() function in the example#2 lab#10 by printing the elements in the list from rear to front.