**LAB 11**

**Name: Omerullah Ansari**

**ID: 65584**

Q1

class Node:

def \_\_init\_\_(self, data):

self.data = data

self.next = None

class Queue:

def \_\_init\_\_(self):

self.front = None

self.rear = None

def is\_empty(self):

return self.front is None

def enqueue(self, data):

new\_node = Node(data)

if self.rear is None:

self.front = self.rear = new\_node

else:

self.rear.next = new\_node

self.rear = new\_node

def dequeue(self):

if self.is\_empty():

print("Queue is empty")

return None

temp = self.front

self.front = temp.next

if self.front is None:

self.rear = None

return temp.data

def search(self, data):

current = self.front

position = 0

while current is not None:

if current.data == data:

return position

current = current.next

position += 1

return -1

def update(self, old\_data, new\_data):

current = self.front

while current is not None:

if current.data == old\_data:

current.data = new\_data

return True

current = current.next

return False

def delete(self, data):

current = self.front

prev = None

while current is not None:

if current.data == data:

if prev is None:

self.front = current.next

if self.front is None:

self.rear = None

else:

prev.next = current.next

if current.next is None:

self.rear = prev

return True

prev = current

current = current.next

return False

def display(self):

current = self.front

while current is not None:

print(current.data, end=" ")

current = current.next

print("None")

queue = Queue()

queue.enqueue(10)

queue.enqueue(20)

queue.enqueue(30)

queue.display()

print("Dequeued:", queue.dequeue())

queue.display()

print("Search for 20:", queue.search(20))

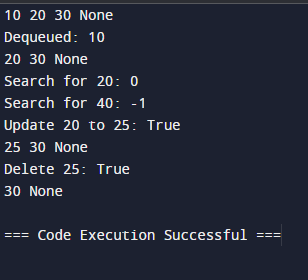
print("Search for 40:", queue.search(40))

print("Update 20 to 25:", queue.update(20, 25))

queue.display()

print("Delete 25:", queue.delete(25))

queue.display()



Q2

class QueueUsingStacks:

def \_\_init\_\_(self):

self.main\_stack = []

self.temp\_stack = []

def isEmpty(self):

return len(self.main\_stack) == 0

def enqueue(self, item):

self.main\_stack.append(item)

def dequeue(self):

if self.isEmpty():

return "Queue is empty"

while self.main\_stack:

self.temp\_stack.append(self.main\_stack.pop())

dequeued\_element = self.temp\_stack.pop()

while self.temp\_stack:

self.main\_stack.append(self.temp\_stack.pop())

return dequeued\_element

queue = QueueUsingStacks()

queue.enqueue(1)

queue.enqueue(2)

queue.enqueue(3)

print(queue.dequeue())

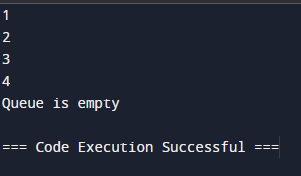
print(queue.dequeue())

queue.enqueue(4)

print(queue.dequeue())

print(queue.dequeue())

print(queue.dequeue())



Q3

from queue import Queue

ages = [18, 35, 65, 68, 45, 55, 22, 58, 67, 43]

queue = Queue()

for age in ages:

queue.put(age)

def sort\_queue(q):

temp\_list = []

while not q.empty():

temp\_list.append(q.get())

temp\_list.sort(reverse=True)

for item in temp\_list:

q.put(item)

sort\_queue(queue)

sorted\_ages = []

while not queue.empty():

sorted\_ages.append(queue.get())

print(sorted\_ages)

