**NAME :SAAD ULLAH**

**SECTION: C**

**LAB TASK: 4**

**SUBMITTED TO: JAMAL ABDUL AHAD**

**DATE : 12-11-2023**

**QUESTION:1**

class Queue:

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

self.items = []

def enqueue(self, item):

self.items.append(item)

def dequeue(self):

if not self.is\_empty():

return self.items.pop(0)

else:

return None # Queue is empty

def is\_empty(self):

return len(self.items) == 0

def size(self):

return len(self.items)

# Example usage:

my\_queue = Queue()

my\_queue.enqueue(1)

my\_queue.enqueue(2)

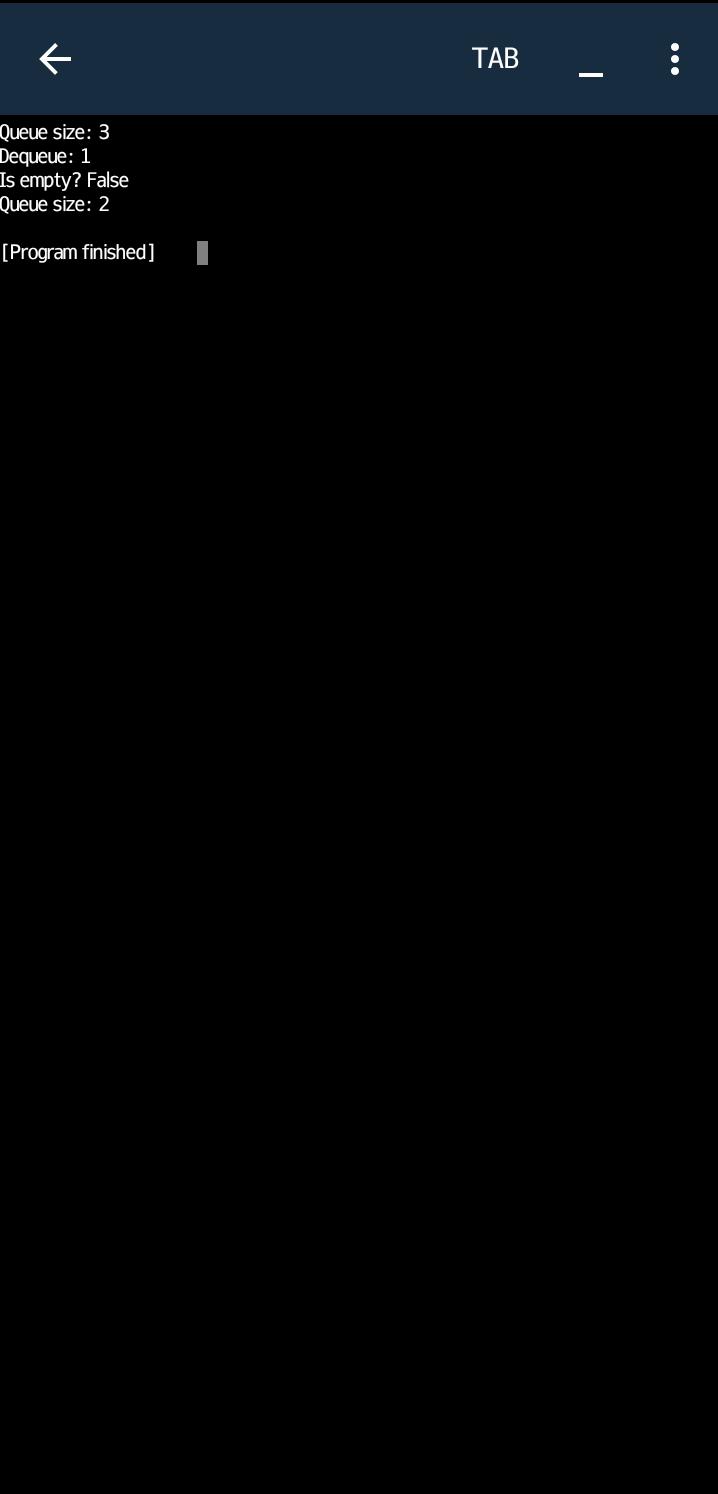
my\_queue.enqueue(3)

print("Queue size:", my\_queue.size()) # Output: 3

print("Dequeue:", my\_queue.dequeue()) # Output: 1

print("Is empty?", my\_queue.is\_empty()) # Output: False

print("Queue size:", my\_queue.size()) # Output: 2



class CircularQueue:

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

self.capacity = capacity

self.queue = [None] \* capacity

self.front = self.rear = -1

def is\_empty(self):

return self.front == -1

def is\_full(self):

return (self.rear + 1) % self.capacity == self.front

def size(self):

if self.is\_empty():

return 0

return (self.rear - self.front + self.capacity) % self.capacity + 1

def enqueue(self, item):

if self.is\_full():

print("Queue is full. Cannot enqueue.")

else:

if self.is\_empty():

self.front = self.rear = 0

else:

self.rear = (self.rear + 1) % self.capacity

self.queue[self.rear] = item

print(f"Enqueued: {item}")

def dequeue(self):

if self.is\_empty():

print("Queue is empty. Cannot dequeue.")

return None

else:

item = self.queue[self.front]

if self.front == self.rear:

self.front = self.rear = -1

else:

self.front = (self.front + 1) % self.capacity

print(f"Dequeued: {item}")

return item

# Example usage:

cq = CircularQueue(5)

cq.enqueue(1)

cq.enqueue(2)

cq.enqueue(3)

cq.enqueue(4)

cq.enqueue(5)

cq.enqueue(6) # This will print "Queue is full. Cannot enqueue."

cq.dequeue()

cq.dequeue()

cq.enqueue(7)

print(f"Size of the queue: {cq.size()}")

