Name: Varad Rane

PRN: 123B1F117

Code:

```
class Patient:

    def __init__(self, name, condition):

        self.name = name

        self.condition = condition

        self.next = None



class EmergencyRoomQueue:

    def __init__(self):

        self.head = None

        self.tail = None



    def is_empty(self):

        return self.head is None
```

```
def add_patient(self, name, condition):
    new_patient = Patient(name, condition)
    if self.is_empty():
        self.head = self.tail = new_patient
    else:
        self.tail.next = new_patient
        self.tail = new_patient
    print(f"Patient {name} added to the queue with condition: {condition}.")


def remove_patient(self):
    if self.is_empty():
        print("No patients in the queue.")
        return None
    removed_patient = self.head
    self.head = self.head.next
    if self.head is None:
```

```
        self.tail = None
    print(f"Patient {removed_patient.name} removed from the queue.")
    return removed_patient


def move_patient_up(self, name):
    if self.is_empty():
        print("No patients in the queue.")
        return

    # If the patient to move is the first one
    if self.head.name == name:
        print(f"Patient {name} is already at the front of the queue.")
        return

    prev = None
    current = self.head
```

```
while current and current.name != name:
    prev = current
    current = current.next

if current is None:
    print(f"Patient {name} not found in the queue.")
    return

if prev:
    prev.next = current.next
    if current == self.tail:
        self.tail = prev
    current.next = self.head
    self.head = current
    print(f"Patient {name} moved to the front of the queue.")
```

```
def search_patient(self, name):
    current = self.head
    while current:
        if current.name == name:
            print(f"Patient {name} found with condition: {current.condition}.")
            return current
        current = current.next
    print(f"Patient {name} not found in the queue.")
    return None


def update_patient_info(self, name, new_condition):
    patient = self.search_patient(name)
    if patient:
        patient.condition = new_condition
        print(f"Patient {name}'s condition updated to: {new_condition}.")
```

```
    def display_queue(self):
        if self.is_empty():
            print("The queue is empty.")
            return

        current = self.head
        print("Current queue:")
        while current:
            print(f"Patient Name: {current.name}, Condition: {current.condition}")
            current = current.next
```

Output:

```
        self.priority = prio
        self.next = None
ss stack:
 def __init__(self) -> None:
        self.head = None
 def add(self,disc,prio):
        new_node = node(disc,prio)
        if self.head == None:
            self.head = new_node
        else:
            new_node.next = self.head
            self.head = new_node

 def remove(self):
        ret = [self.head.desc,self.head.priority]
        self.head = self.head.next
        return ret
```

```
PS D:\downloads\DS-20240919T163116Z-001\DS> python -u "d:\downloa
ds\DS-20240919T163116Z-001\DS\Assignments\Assignment-6.py"
Prepare weekly report 1
Complete project proposal 1

Respond to client emails 2
Schedule team meeting 2

Review draft presentation 3

['Prepare weekly report', 1]
['Complete project proposal', 1]
['Respond to client emails', 2]
['Schedule team meeting', 2]
['Review draft presentation', 3]
```