**Assignment**

**Air Traffic Controller using a Priority Queue**

* **Code:**

class Node(object):

def \_\_init\_\_(self,data=None,next=None):

self.data=data

self.next=next

#class implementing logic for enqueue,dequeue

class Queue(object):

def \_\_init\_\_(self,front=None,rear=None):

self.front=None

self.rear=None

# enqueue method to add from rear

def enqueue(self,data):

NewNode=Node(data)

if self.front==None and self.rear==None:

self.front=self.rear=NewNode

return

tmp=self.front

c=0

max=NewNode.data[1]

while tmp!=self.rear:

if tmp.data[1]>max:

curr=tmp

c=c+1

tmp=tmp.next

continue

elif c==0:

NewNode.next=self.front

self.front=NewNode

print("\nEntry added successfully")

return

else:

curr.next=NewNode

NewNode.next=tmp

print("\nEntry Successfully Added")

return

if self.front==self.rear:

if self.front.data[1]>max:

self.front.next=NewNode

self.rear=NewNode

print("\nEntry Successfully Added")

return

else:

NewNode.next=self.front

self.front=NewNode

print("\nEntry Successfully Added")

return

if tmp==self.rear:

if tmp.data[1]<max:

curr.next=NewNode

NewNode.next=tmp

print("\nEntry Successfully Added")

return

elif self.rear.data[1]>max:

self.rear.next=NewNode

self.rear=NewNode

print("\nEntry Successfully Added")

return

# dequeue method to delete from front

def dequeue(self):

current=self.front

if self.front==None:

print("\nQueue is Empty")

return

if self.front==self.rear:

self.front=self.rear=None

print("\nEntry Successfully Deleted")

else:

self.front=self.front.next

print("\nEntry Successfully Deleted")

# showelements method to display details on plane

def showelements(self):

front1 = self.front

if ((front1 == None) and (self.rear == None)):

print("\nQueue is empty")

return

print("\n Plane Number\t Priority Number")

while (front1 != self.rear):

print(" ",front1.data[0],"\t\t",front1.data[1] )

front1 = front1.next

if (front1 == self.rear):

print(" ",front1.data[0],"\t\t",front1.data[1] )

while True:

try:

#Select valid option

print("\n1.Create Queue")

print("\n2.Adding plane information with priority")

print("\n3.Delete Plane from Queue")

print("\n4.Display plane landing with highest priority ")

print("\n5.Exit\n")

ch=int(input())

if ch==1:

q=Queue()

#Add plane information(Enqueue)

elif ch==2:

plane\_number=input("\nEnter Plane Number : \n")

while True:

try:

if len(plane\_number)==4 and plane\_number!=None and type(int(plane\_number))==int:

plane\_number=plane\_number

break

else:

raise Exception("\nPlease Eneter Valid \

Plane Number : \n")

except Exception as e:

plane\_number=input(e)

continue

priority=input("\nEnter Priority Number : \n")

while True:

try:

if len(priority)>=1 and priority!=None and type(int(priority))\

==int:

priority=priority

break

else:

raise Exception("\nPlease Enter Valid Priority\

Number : \n")

except Exception as e:

priority=input(e)

continue

q.enqueue([plane\_number,priority])

#Delete plane entry(dequeue)

elif ch==3:

q.dequeue()

#call to showelements method

elif ch==4:

print("\nDisplay plane entries in queue: ")

q.showelements()

#Exit

elif ch==5:

break

else:

print("\nEnter Right Choice")

except:

continue

* **Output:**





