NAMAL UNIVERSITY MIANWALI

DEPARTMENT OF ELECTRICAL ENGINEERING

DATA STRUCTURE AND ALGORITHM

LAB # 07

REPORT

Title : Queues in Python

Name	*Fahim-Ur-Rehman Shah*
Roll No	*NIM-BSEE-2021-24*
Intructor	*Ms. Naureen Shaukat*
Lab Engineer	*Mr .Ali Hasnain*
Date	*13-May-2023*
Marks	

Instructions:

1. This is an individual lab. You will perform the tasks individually and submit a report.
2. Some of these tasks (marked as 'Example') are for practice purposes only while others (marked as 'Task') have to be answered in the report.
3. When asked to display an output in the task, either save it as jpeg or take a screenshot, in order to insert it in the report.
4. The report should be submitted on the given template, including:
    a) Code (copy and pasted, NOT a screenshot)
    b) Output figure (as instructed in 3)
    c) Explanation where required
5. The report should be properly formatted, with easy to read code and easy to see figures.
6. Plagiarism or any hint thereof will be dealt with strictly. Any incident where plagiarism is caught, both (or all) students involved will be given zero marks, regardless of who copied whom. Multiple such incidents will result in disciplinary action being taken.
7. Late submission of report is allowed within 03 days after lab with 20% deduction of marks every day.
8. You have to submit report in pdf format (Reg.X_DSA_LabReportX.pdf).

Complete Code

Python Code :

```
class ArrayQueue:
    DEFAULT_CAPACITY = 5    # The default capacity of the queue.

    def __init__(self):
        # Initialize the queue with the default capacity.
        self.data = [None] * ArrayQueue.DEFAULT_CAPACITY
        self.size = 0
        self.front = 0

    def __len__(self):
        # Return the number of elements in the queue.
        return self.size

    def is_empty(self):
        # Return True if the queue is empty, False otherwise.
        return self.size == 0

    def dequeue(self):
        if self.size > 0 :
          del(self.data[0])
          self.size = self.size - 1

    def enqueue(self, e):
```

```
        # Add an element to the back of the queue.
        if self.size == len(self.data):
            # Resize the array if it is full.
            self.resize(2 * len(self.data))
        avail = (self.front + self.size) % len(self.data)
        self.data[avail] = e
        self.size += 1

    def resize(self, cap): # Resize the array to the specified capacity.
        old = self.data
        self.data = [None] * cap
        walk = self.front
        for k in range(self.size):
            self.data[k] = old[walk]
            walk = (1 + walk) % len(old)
        self.front = 0

    def print(self):        # Print the contents of the queue.
        if self.is_empty():
            print("Queue is empty")
        else:
            print("[",end="")
            for i in range(0,len(self.data)):
               if self.data[i] != None :
                 print(f" {self.data[i]} ",end="")
               if i <= (self.size-2):
                 print(",",end="")
            print("]")

# Make code for the main
myQ = ArrayQueue()

print("\n")
for i in range(1,8):
    myQ.enqueue(i*10)
print("Elements After Enqueue : ",end="")
myQ.print()
for i in range(1,8):
    myQ.dequeue()
    print(i," Time Dequeue Called : ",end="")
    myQ.print()

print("\n")
```

Output Screenshot:

```
thm\Lab\Lab 07\Example.py"


Elements After Enqueue : [ 10 , 20 , 30 , 40 , 50 , 60 , 70 ]
1  Time Dequeue Called : [ 20 , 30 , 40 , 50 , 60 , 70 ]
2  Time Dequeue Called : [ 30 , 40 , 50 , 60 , 70 ]
3  Time Dequeue Called : [ 40 , 50 , 60 , 70 ]
4  Time Dequeue Called : [ 50 , 60 , 70 ]
5  Time Dequeue Called : [ 60 , 70 ]
6  Time Dequeue Called : [ 70 ]
7  Time Dequeue Called : Queue is empty


PS E:\Semester 4\Data Structure and Algorithm\Lab\Lab 07>
```

Explanation :

The given code implements a queue data structure using an array in the ArrayQueue class. It provides methods to enqueue and dequeue elements, check if the queue is empty, resize the array when needed, and print the contents of the queue. In the main code, an instance of ArrayQueue is created, elements are enqueued, and the queue is printed after each enqueue operation. Elements are then dequeued, and the queue is printed again after each dequeue operation.

Task 1: Make DEQUEUE method for the class provided in Example.

Python Code :

```
def dequeue(self):
     if self.size > 0 :
       del(self.data[0])
       self.size = self.size - 1
```

Explanation :

The dequeue method removes and returns the element at the front of the queue if the queue is not empty. It accomplishes this by deleting the first element of the data array and updating the size accordingly.

Task 2: Design PRINT method for printing the whole

Python Code :

```
    def print(self):        # Print the contents of the queue.
        if self.is_empty():
            print("Queue is empty")
        else:
            print("[",end="")
            for i in range(0,len(self.data)):
               if self.data[i] != None :
                 print(f" {self.data[i]} ",end="")
               if i <= (self.size-2):
                 print(",",end="")
            print("]")
```

Explanation :

This code defines a method named "print" within a class. The purpose of this method is to print the contents of a queue. It first checks if the queue is empty, and if so, it prints the message "Queue is empty". Otherwise, it starts printing the queue elements enclosed within square brackets. It iterates over each element in the queue, ignoring any elements that are None. It prints each non-None element with surrounding spaces, and if it's not the last element, it also prints a comma to separate the elements. Finally, it prints a closing square bracket to complete the representation of the queue.

Task 3: Write the main portion for making a queue. Call all the methods to see the behavior of the code.

Python Code :

```
myQ = ArrayQueue()
print("\n")
for i in range(1,8):
    myQ.enqueue(i*10)
print("Elements After Enqueue : ",end="")
myQ.print()
for i in range(1,8):
    myQ.dequeue()
    print(i," Time Dequeue Called : ",end="")
    myQ.print()
print("\n")
```

Explanation :

This code creates an instance of the ArrayQueue class named myQ. It then enters a loop that runs seven times, enqueueing values that are multiples of 10 into the queue. After each enqueue operation, it calls the print method of the queue to display its contents. Following that, it enters another loop that also runs seven times, dequeues an element from the queue, prints the number of times the dequeue operation has been called, and displays the updated contents of the queue. Finally, it prints a new line character to separate the output.

Lab Evaluation Rubrics							
Domain	**CLOs/ Rubric**	**Performance Indicator**	**Unsatisfactory 0-5**	**Marginal 5-10**	**Satisfactory 11-15**	**Exemplary 16-20**	**Allocated Marks**
Psychomotor	**CLO:1 R2**	Implementation with Results **(P)**	Does not try to solve problems. Many mistakes in code and difficult to comprehend for the instructor. There is not result of the problem.	Does not suggests or refine solutions but is willing to try out solutions suggested by others. Few mistakes in code, but done along with comments, and easy to comprehend for the instructor. Few mistake in result.	Refines solutions suggested by others. Complete and error-free code is done. No comments in the code, but easy to comprehend for the instructor. Results are correctly produced.	Actively looks for and suggests solution to problems. Complete and error free code is done, easy to comprehend for the instructor. Results are correctly produced. Student incorporated comments in the code.	
Affective	**CLO:3 R3**	Lab Report **(A)**	Code of the problem is not given. Outputs are not provided. Explanation of the solution is not stated.	Code of the problem is not given. Output is not complete. Explanation of the solution is not satisfactory.	Code of the problem is not given. Output is completely given. Explanation of the solution is not satisfactory.	Code of the problem is not given. Output is completely given. Explanation of the solution is satisfactory.	
	CLO:1 R5	Discipline and Behavior **(A)**	Got and wandered around.Chased others, ran, or played around. More than two incidents of talking non-lab related stuff in laband/or any talk with other groups, voice level exceeding the appropriate level, use of cell phones and involvement in any non lab activity.	Got out of seat and wander around for some time. No more than two incidents of talking non-lab related stuff inlab. Voice level exceeding theappropriate level, use of cell phones and involvement in any non-lab related activity.	Stayed in seat and got up for a specific lab related reason, but took more time than required to do the job. No more than one incidents of talking non-lab related stuff in lab. Voice level exceedingthe appropriate level, use of cell phones and involvementin any non-lab related activity.	Stayed in seat and got up for a specific lab related reason. Tookcare of lab related business and sat down right away. Voice level kept appropriate. Not used cell phones or involved in any non- lab related activity.	