Experiment no 7

Queue Implementation

OBJECTIVES:

- To learn about the concept of Queue.
- To learn different ways for implementing a Queue.
- To implement Queue using deqeue, lists and singly linked list.

1. QUEUE:

Like a stack, the queue is a linear data structure that stores items in a First In First Out (FIFO) manner. With a queue, the least recently added item is removed first. A good example of a queue is any queue of consumers for a resource where the consumer that came first is served first.

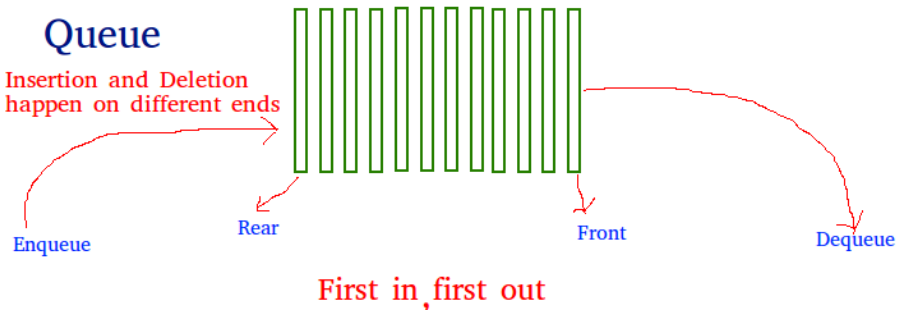


Figure 1: Stack Insertion and Deletion

Operations associated with queue are:

- Enqueue: Adds an item to the queue. If the queue is full, then it is said to be an Overflow condition – Time Complexity : O(1)
- Dequeue: Removes an item from the queue. The items are popped in the same order in which they are pushed. If the queue is empty, then it is said to be an Underflow condition – Time Complexity : O(1)
- Front: Get the front item from queue – Time Complexity : O(1)
- Rear: Get the last item from queue – Time Complexity : O(1)

There are various ways to implement a queue in Python. This article covers the implementation of queue using data structures and modules from Python library. Python Queue can be implemented by the following ways:

- **list**
- **collections.deque**
- **queue.Queue**

2. IMPLEMENTATION USING LIST:

List is a Python's built-in data structure that can be used as a queue. Instead of enqueue() and dequeue(), append() and pop() function is used. However, lists are quite slow for this purpose because inserting or deleting an element at the beginning requires shifting all of the other elements by one, requiring O(n) time.
The code simulates a queue using a Python list. It adds elements 'a', 'b', and 'c' to the queue and then dequeues them, resulting in an empty queue at the end. The output shows the initial queue, elements dequeued ('a', 'b', 'c'), and the queue's empty state.

```
queue = []
queue.append('a')
queue.append('b')
queue.append('c')
print("Initial queue")
print(queue)
print("\nElements dequeued from queue")
print(queue.pop(0))
print(queue.pop(0))
print(queue.pop(0))
print("\nQueue after removing elements")
print(queue)
```

Figure 2: Implementation using Lists

3. IMPLEMENTATION USING collections.deque:

Queue in Python can be implemented using deque class from the collections module. Deque is preferred over list in the cases where we need quicker append and pop operations from both the ends of container, as deque provides an O(1) time complexity for append and pop operations as compared to list which provides O(n) time complexity. Instead of enqueue and deque, append() and popleft() functions are used.
The code uses a deque from the collections module to represent a queue. It appends **'a'**, **'b'**, and **'c'** to the queue and dequeues them with **q.popleft()**, resulting in an empty queue. Uncommenting **q.popleft()** after the queue is empty would raise an IndexError. The code demonstrates queue operations and handles an empty queue scenario.

```
from collections import deque
q = deque()
q.append('a')
q.append('b')
q.append('c')
print("Initial queue")
print(q)
print("\nElements dequeued from the queue")
print(q.popleft())
print(q.popleft())
print(q.popleft())

print("\nQueue after removing elements")
print(q)
```

Figure 3: Implementation using deque

4. IMPLEMENTATION USING QUEUE MODULE:

Queue is built-in module of Python which is used to implement a queue. queue.Queue(maxsize) initializes a variable to a maximum size of maxsize. A maxsize of zero '0' means a infinite queue. This Queue follows FIFO rule.

There are various functions available in this module:

- maxsize – Number of items allowed in the queue.
- empty() – Return True if the queue is empty, False otherwise.
- full() – Return True if there are maxsize items in the queue. If the queue was initialized with maxsize=0 (the default), then full() never returns True.
- get() – Remove and return an item from the queue. If queue is empty, wait until an item is available.
- get_nowait() – Return an item if one is immediately available, else raise QueueEmpty.
- put(item) – Put an item into the queue. If the queue is full, wait until a free slot is available before adding the item.
- put_nowait(item) – Put an item into the queue without blocking. If no free slot is immediately available, raise QueueFull.
- qsize() – Return the number of items in the queue.

```
from queue import Queue
q = Queue(maxsize = 3)
print(q.qsize())
q.put('a')
q.put('b')
q.put('c')
print("\nFull: ", q.full())
print("\nElements dequeued from the queue")
print(q.get())
print(q.get())
print(q.get())
print("\nEmpty: ", q.empty())
q.put(1)
print("\nEmpty: ", q.empty())
print("Full: ", q.full())
```

Figure 4: Implementation using queue module

5. ACTIVITIES:

A1) Write a code for queue implementation using lists for your registration number.

A2) Write a code for queue implementation using deque for your department name i.e "Electrical and Computer Engineering"

A3) Write a code for queue implementation using queue module for name of all the courses of your current semester. (Make a queue for the names of your courses of current semester).

A4) Write a python program to implement a queue using linked list, classes and functions with the methods should be used for queue.

IMPORTANT: All the activities` code should be attached to the manual before summary section.

Lab Exercise and Summary

Summary should cover Introduction, Procedure, Data Analysis and Evaluation.

Student's Signature:_____ **Date:**_____

LABORATORY SKILLS ASSESSMENT (Psychomotor)

Total Marks:100

Criteria (Max Marks)	Level 1 0% ≤ S < 50%	Level 2 50% ≤ S< 70%	Level 3 70% ≤ S< 90%	Level 4 90%≤ S ≤100%	Score (S)
Procedural Awareness (30)	Selects inappropriate skills and/or strategies Required by the task.	Selects and applies appropriate skills and/or strategies required by the task with major errors.	Selects and applies the appropriate strategies and/or skills specific to the task without significant errors.	Selects and applies appropriate strategies and/or skills specific to the task without any error.	
Practical Implementation (30)	Makes major critical errors in applying procedural knowledge related to python Queue Implementation.	Makes numerous critical errors in applying procedural knowledge related to python Queue Implementation.	Makes some non-critical errors in applying procedural knowledge related to python Queue Implementation.	Applies the procedural knowledge in optimized ways related to python Lists, Queue Implementation.	
Use of Tool/Equipment (30)	Uses tools, equipment and materials with limited competence.	Uses tools, equipment and materials with some competence.	Uses tools, equipment and materials with considerable competence.	Uses tools, equipment and materials with a high degree of competence.	
Safety (10)	Requires constant reminders to follow safety procedures.	Requires some reminders to follow safety procedures.	Follows safety procedures with only minimal reminders.	Routinely follows safety procedures.	
				Marks Obtained	

Instructor Name:_____ **Sign:** _____

LABORATORY SKILLS ASSESSMENT (Affective)

Total Marks: 40

Criteria (Max. Marks)	Level 1 0% ≤ S < 50%	Level 2 50% ≤ S < 70%	Level 3 70% ≤ S < 90%	Level 4 90% ≤ S ≤ 100%	Score
Introduction (5)	Very little background information provided or information is incorrect	Introduction is brief with some minor mistakes	Introduction is nearly complete, missing some minor points	Introduction complete and well-written; provides all necessary background principles for the experiment	
Procedure (5)	Many stages of the procedure are not entered on the lab report.	Many stages of the procedure are entered on the lab report.	The procedure could be more efficiently designed but most stages of the procedure are entered on the lab report.	The procedure is well designed and all stages of the procedure are entered on the lab report.	
Data Record (10)	Data is brief and missing significant pieces of information.	Data provides some significant information and has few critical mistakes.	Data is almost complete but has some minor mistakes.	Data is complete and relevant. Tables with units are provided. Graphs are labeled. All questions are answered correctly.	
Data Analysis (10)	Data is presented in very unclear manner.	Data is presented in ways that are not clear enough.	Data is presented in ways that can be understood and interpreted.	Data is presented in ways that best facilitate understanding and interpretation.	
Report Quality (10)	Report contains many errors.	Report is somewhat organized with some spelling or grammatical errors.	Report is well organized and cohesive but contains some grammatical errors.	Report is well organized and cohesive and contains no grammatical errors. Presentation seems polished.	
				Marks Obtained	

LABORATORY SKILLS ASSESSMENT (Cognitive)

Total Marks: 10

(If any) **Marks Obtained**	

Instructor's Signature:_____ **Date:**_____