SLIIT
Discover Your Future

BSc (Hons) in Information Technology

Year 1

Data Structures and Algorithms – IT1170

Practical 4 - Queues

Objectives:

- Understand the queue data structure and its operations.
- Implement queue operations using arrays in Java.
- Apply queue concepts to solve real-world problems.
- Develop problem-solving skills through hands-on coding exercises.

Queues

A **queue** is a linear data structure that follows the **FIFO (First In, First Out)** principle. Elements are added at the rear and removed from the front.

Operations of a Queue

1. **Insert** - Add an element to the rear
2. **Remove** - Remove an element from the front
3. **PeekFront** - Get the front element without removing it
4. **isEmpty** - Check if the queue is empty
5. **isFull** - Check if the queue is full

Queues are commonly used in applications such as printer queues, resource allocation, and priority management.

Exercise 1: Basic Implementation of Queue

1. Create a class QueueArray that implements a queue using an array and stores integers.
2. Implement the following methods:
    - insert(int item): Adds an item to the rear.
    - remove(): Removes and returns the front item.
    - peekFront(): Returns the front item without removing it.
    - isEmpty(): Checks if the queue is empty.
    - isFull(): Checks if the queue is full.
3. Introduce a new method named getCount() to get the number of elements in the Queue.
4. Write a main method to test the queue operations.

Sample output:

```
Queue elements: 10 20 30
Removed: 10
Queue elements: 20 30
Front Element: 20
Queue Count: 2


...Program finished with exit code 0
Press ENTER to exit console.
```

Exercise 2: Modify the Linear Queue to obtain Circular Queue

5. Modify previously implemented methods to ensure the properties of circular queue.
    - insert(int item): Adds an item to the rear and reassign rear to initial state after max index.
    - remove(): Removes and returns the front item, when front reaches to max index reassign to initial index.
    - peekFront(): Returns the front item without removing it, when front reaches to max index reassign to initial index
    - isEmpty(): Checks if the queue is empty.
    - isFull(): Checks if the queue is full.

6. Write a main method to test the queue operations. Allow the user to define the size of the circular queue. Initially, allow the user to insert values to all available indexes. Afterwards, remove values and insert values appropriately to showcase the behavior of circular queue.

Sample Output: