DSA

Module 11

Queues

Author
Srinivas Dande

Java
Learning
Center

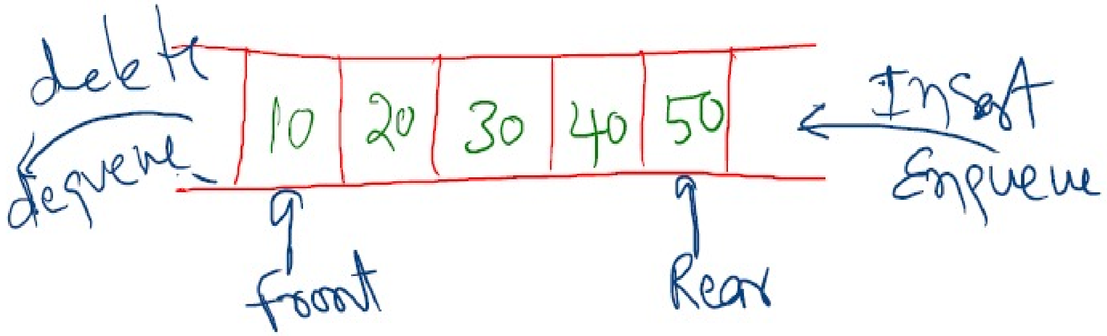
11. Introduction to Queues

- Queue is an ADT (Abstract Data Type).
- Queue is a **Linear Data Structure**.
- Queue follow the Strategy called **FIFO (First In First Out)**.
- Queue is Open at both the Ends.

  i.e You can do operations on both the ends.

- You can do **Insert Operation** in the **Back(Rear)** of the Queue
- You can do **Delete Operation** in the **Front** of the Queue

- **Insert Operation** on the Queue is called as **Enqueue**
- **Delete Operation** on the Queue is called as **Dequeue**



- Queues can be implemented using **Arrays and LinkedLists**

✓ A queue is an ordered list in which insertions are done at one end (**rear**) and deletions are done at other end (**front**).

✓ The first element to be inserted is the first one to be deleted. Hence, it is called First in First out (**FIFO**) or Last in Last out (**LILO**) list.

11.1. Queue Operations

a) isEmpty()
b) size()
c) offer(int)
d) poll()
e) peek()

a) isEmpty()

- Returns True if the Queue is Empty otherwise false..

b) size()

- Returns the size of the Queue

c) offer(object)

- Inserts the Element at the Back of the Queue

d) poll()

- Removes the Front Element and returs the same

e) peek()

- Returns the Front Element from the Queue without Remving

11.2. Corner Conditions

a) Queue Underflow
b) Queue Overflow

a) Queue Underflow

- When poll() or peek() is called on the empty Queue.

b) Queue Overflow

- When offer() is called on the full Queue.

11.3. Time Complexity of Stack Operations

Operation	Fixed Arrays	Circular Fixed Arrays	Circular Dynamic Arrays	LinkedList
offer()	O(1)	O(1)	Amortized O(1)	O(1)
poll()	O(n)	O(1)	O(1)	O(1)
peek()	O(1)	O(1)	O(1)	O(1)

11.4. Applications of Queues

- Queues can be used in many of the real-world applications
  - **a)** Single Resource and Multiple Consumers
  - **b)** Sync between slow and fast devices

a) Single Resource and Multiple Consumers:

1) Web servers:

- Web servers uses queue to manage incoming requests from web clients.

2) CPU scheduling:

- OS uses queue to schedule the CPU based on FCFS Scheduling

3) Printers:

- Printers uses queue to manage the order in which the documents are printed

b) Sync between slow and fast devices

1) Keyboard buffer:

- Buffers the Keyboard input in queue and process it later

2) Message Buffering:

- In Distributed Systems, Messages will be placed in Queue and later consumers will consume it slowly.

3) Audio and Video Players:

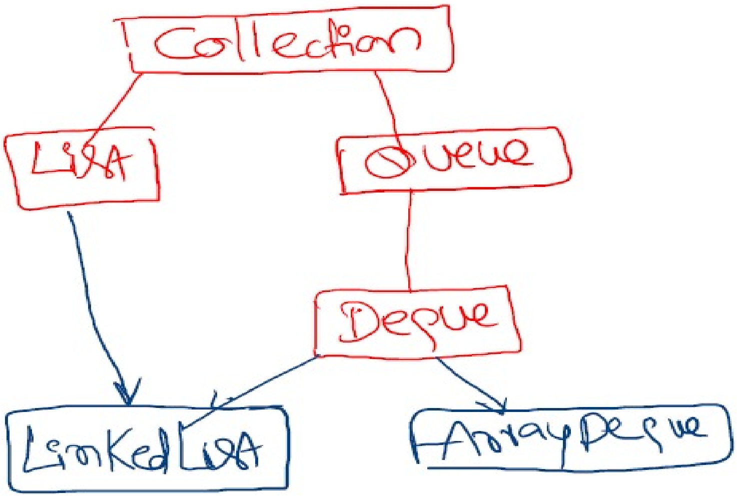
- Beffers the content and Plays it slowly

4) Event Handling:

- Queues can be used to handle events in event-driven systems

11.5. Queues in Java Collection

- Java Collections has two classes for Queues use-cases.
  a) ArrayDeque
  b) LinkedList



a) ArrayDeque

- ArrayDeque is a latest class from java.util package.
- ArrayDeque is implemented with Arrays.
- Use the ArrayDeque class in single-threaded environment
- If you want to use ArrayDeque in multi-threaded environment then you need to provide external synchronization.
- Most ArrayDeque operations run in amortized O(1).
- ArrayDeque class is likely to be faster than LinkedList
- ArrayDeque can be used as
    - Stack
    - Queue

b) LinkedList

- LinkedList is the class from java.util package.
- LinkedList is implemented with Nodes.
- Use the LinkedList class in single-threaded environment
- If you want to use LinkedList in multi-threaded environment then you need to provide external synchronization.
- Most LinkedList operations run in O(1).
- LinkedList class is likely to be slower than ArrayDeque
- LinkedList can be used as
    - List
    - Queue

11.5.1. Using Queue Built-In Methods

Lab1.java

```
package com.jlcindia.queues;

import java.util.ArrayDeque;
import java.util.LinkedList;
import java.util.Queue;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab1 {
        public static void main(String[] args) {

                //Queue<Integer> myque= new LinkedList<>();
                Queue<Integer> myque= new ArrayDeque<>();

                System.out.println(myque);
                System.out.println(myque.size());
                System.out.println(myque.isEmpty());

                myque.offer(10);
                myque.offer(20);
                myque.offer(30);
                myque.offer(40);
                myque.offer(50);

                System.out.println("-----------------------------");
                System.out.println(myque);
                System.out.println(myque.size());
                System.out.println(myque.isEmpty());

                System.out.println("-----------------------------");
                System.out.println(myque.peek());
                System.out.println("-----------------------------");
        myque.poll();
        myque.poll();
        System.out.println("-----------------------------");
        System.out.println(myque.peek());
        System.out.println("-----------------------------");
    }
}
```

11.5.2. Using Queue Built-In Methods

Lab2.java

```
package com.jlcindia.queues;

import java.util.ArrayDeque;
import java.util.LinkedList;
import java.util.Queue;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab2 {
    public static void main(String[] args) {

        Queue<Integer> myque= new LinkedList<>();
        //Queue<Integer> myque= new ArrayDeque<>();

        System.out.println(myque);
        System.out.println(myque.size());
        System.out.println(myque.isEmpty());

        myque.add(10);
        myque.add(20);
        myque.add(30);
        myque.add(40);
        myque.add(50);
```

```
                System.out.println("----------------------------");
                System.out.println(myque);
                System.out.println(myque.size());
                System.out.println(myque.isEmpty());

                System.out.println("----------------------------");
                System.out.println(myque.element());
                System.out.println("----------------------------");

                myque.remove();
                myque.remove();

                System.out.println("----------------------------");
                System.out.println(myque.element());
                System.out.println("----------------------------");
        }
}
```

11.5.3. Traversing Queue elements in Forward Order

Lab3.java

```
package com.jlcindia.queues;

import java.util.Iterator;
import java.util.LinkedList;
import java.util.Queue;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab3 {
        public static void main(String[] args) {

                Queue<Integer> myque= new LinkedList<>();
                //Queue<Integer> myque= new ArrayDeque<>();
```

```
		myque.add(10);
		myque.add(20);
		myque.add(30);
		myque.add(40);
		myque.add(50);

		System.out.println(myque);
		System.out.println("------------------------------");

		Iterator<Integer> it = myque.iterator();
		while(it.hasNext()) {
			System.out.print(it.next()+"\t");
		}

		System.out.println("\n---------------------------");
		for(Integer x:myque) {
			System.out.print(x+"\t");

		}
	}
}
```

11.5.4. Reverse the Queue using Iteration

Lab4.java

```
package com.jlcindia.queues;

import java.util.LinkedList;
import java.util.Queue;
import java.util.Stack;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
```

```
public class Lab4 {
        static void reverseQueue(Queue<Integer> myque) {
                Stack<Integer> mystack = new Stack<>();

                while(!myque.isEmpty()) {
                        mystack.push(myque.poll());
                }

                while(!mystack.isEmpty()) {
                        myque.offer(mystack.pop());
                }
        }

        public static void main(String[] args) {

                Queue<Integer> myque= new LinkedList<>();

                myque.add(10);
                myque.add(20);
                myque.add(30);
                myque.add(40);
                myque.add(50);

                System.out.println(myque);
                System.out.println("-----------------------------");
                reverseQueue(myque);
                System.out.println(myque);

                }

}
```

11.5.5. Reverse the Queue using Recursion

Lab5.java

```
package com.jlcindia.queues;

import java.util.LinkedList;
import java.util.Queue;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab5 {
        static void reverseQueue(Queue<Integer> myque) {
                if(myque.isEmpty()) {
                        return;
                }

                int x= myque.poll();
                reverseQueue(myque);
                myque.offer(x);
        }
        public static void main(String[] args) {
                Queue<Integer> myque= new LinkedList<>();

                myque.add(10);                myque.add(20);
                myque.add(30);                myque.add(40);
                myque.add(50);

                System.out.println(myque);
                System.out.println("----------------------------");
                reverseQueue(myque);
                System.out.println(myque);
                }
}
```

11.6. Queue Implementation using Arrays

MyQueue.java

```
package com.jlcindia.queues.arrays1;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
class MyQueue {

        int capacity; // size of array
        int size; //size of the Queue
        Integer myarray[];

        public MyQueue(int capacity) {
                this.capacity = capacity;
                this.size = 0;
                this.myarray = new Integer[capacity];
        }

        public int size() {
                return size;
        }

        public boolean isEmpty() {
                return (size == 0);
        }

        public boolean isFull() {
                return (size == capacity);
        }
```

```
    public boolean offer(int element) {
        if (isFull()) {
            return false;
        }

        myarray[size] = element;
        size++;
        return true;
    }

    public Integer poll() {
        if(isEmpty()) {
            return null;
        }
        int element = myarray[0];
        for(int i=0;i<size-1;i++) {
            myarray[i] = myarray[i+1];
        }
        size--;
        myarray[size] = null;
        return element;
    }

    public Integer peek() {
        if(isEmpty()) {
            return null;
        }

        return  myarray[0];
    }

    public int getFront() {
        if(isEmpty())
            return -1;

        return 0;
    }
    public int getRear() {
        if(isEmpty())
            return -1;

        return size-1;
    }

    public String toString() {
        String str = "[";
        if (size != -1) {
            for (Integer x : myarray) {
                if (x != null)
                    str = str + x + ",";
                else
                    str= str+" null , ";
            }
        }
        str = str + "]";
        return str;
    }
}
```

Lab6.java

```
package com.jlcindia.queues.arrays1;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab6 {
    public static void main(String[] args) {
        MyQueue myque = new MyQueue(5);

        System.out.println(myque);
        System.out.println(myque.size());
        System.out.println(myque.isEmpty());
        System.out.println(myque.isFull());
        myque.offer(10);
        myque.offer(20);
        myque.offer(30);
        myque.offer(40);
        myque.offer(50);

        System.out.println(myque);
        System.out.println(myque.size());
        System.out.println(myque.isEmpty());
        System.out.println(myque.isFull());
        System.out.println("--------------------");
        System.out.println(myque.peek()); //10
        System.out.println("--------------------");
        myque.poll();
        myque.poll();
        System.out.println(myque);
        System.out.println("--------------------");
        System.out.println(myque.peek()); //30
        System.out.println("--------------------");

    }

}
```

11.7. Queue Implementation using Arrays

MyQueue.java

```
package com.jlcindia.queues.arrays2;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
class MyQueue {
        private int capacity; // size of array
        private int size; // size of the Queue
        private int front;
        private Integer myarray[];

        public MyQueue(int capacity) {
                this.capacity = capacity;
                this.size = 0;
                this.front = 0;
                this.myarray = new Integer[capacity];
        }
        public int size() {
                return size;
        }
        public boolean isEmpty() {
                return (size == 0);
        }
        public boolean isFull() {
                return (size == capacity);
        }
        public int getFront() {
                if (isEmpty())
                        return -1;

                return front;
        }
```

```
    public int getRear() {
        if (isEmpty())
            return -1;

        return (front+size-1)%capacity;
    }

    public boolean offer(int element) {
        if (isFull()) {
            //Resize Code
            //Dont return false
            return false;
        }

        int currRear= getRear();
        int newRear= (currRear+1)%capacity;

        myarray[newRear] = element;
        size++;
        return true;
    }

    public Integer poll() {
        if (isEmpty()) {
            return null;
        }

        int element = myarray[front];

        size--;
        myarray[front] = null;
        front = (front+1)%capacity;
        return element;
    }
    public Integer peek() {
        if (isEmpty()) {
            return null;
        }

        return myarray[front];
    }

    public String toString() {
        String str = "[";
        if (size != -1) {
            for (Integer x : myarray) {
                if (x != null)
                    str = str + x + ",";
                else
                    str = str + " null , ";
            }
        }
        str = str + "]";
        return str;
    }
}
```

Lab7.java

```
package com.jlcindia.queues.arrays2;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab7 {
    public static void main(String[] args) {
        MyQueue myque = new MyQueue(5);

        System.out.println(myque);
        System.out.println(myque.size());
        System.out.println(myque.isEmpty());
        System.out.println(myque.isFull());
        System.out.println(myque.getFront());
        System.out.println(myque.getRear());

        myque.offer(10);
        myque.offer(20);
        myque.offer(30);
        myque.offer(40);
        myque.offer(50);

        System.out.println(myque);
        System.out.println(myque.size());
        System.out.println(myque.isEmpty());
        System.out.println(myque.isFull());
        System.out.println(myque.getFront());
        System.out.println(myque.getRear());
        System.out.println("-------------------");
        System.out.println(myque.peek()); //10
        System.out.println("-------------------");

        myque.poll();
        myque.poll();

        System.out.println(myque);
        System.out.println("-------------------");
        System.out.println(myque.peek()); //30
        System.out.println(myque.getFront()); //2
        System.out.println(myque.getRear()); //4
        System.out.println("-------------------");

        myque.offer(55);
        boolean x =myque.offer(66);
        System.out.println(x);

        System.out.println(myque);
```

```
                System.out.println("------------------");
                System.out.println(myque.peek()); //30
                System.out.println(myque.getFront()); //2
                System.out.println(myque.getRear()); //1
                System.out.println("------------------");

                boolean b= myque.offer(99);
                System.out.println(b);
        }
}
```

11.8. Queue Implementation using LinkedLists

Node.java

```
package com.jlcindia.queues.linkedlist;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */

public class Node {

        int data;
        Node next;

        Node(int data) {
                this.data = data;
                this.next = null;
        }
}
```

MyQueue.java

```
package com.jlcindia.queues.linkedlists;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
class MyQueue {

    private int size; // size of the Queue
    private Node frontNode;
    private Node rearNode;

    public MyQueue() {
        this.size = 0;
        this.frontNode=null;
        this.rearNode=null;
    }

    public int size() {
        return size;
    }

    public boolean isEmpty() {
        //return (size == 0);
         return (frontNode==null && rearNode==null);
    }

    public Integer peek() {
        if (isEmpty()) {
            return null;
        }

        return frontNode.data;
    }
```

```
public void offer(int element) {

        Node temp = new Node(element);

        //1.Empty List
        if(isEmpty()) {
                frontNode=temp;
                rearNode=temp;
                size++;
                return;
        }
        //2.Non-Empty List
        rearNode.next=temp;
        rearNode=temp;
        size++;
}

public Integer poll() {
        if (isEmpty()) {
                return null;
        }

        int element = frontNode.data;
        Node temp = frontNode;
        frontNode=frontNode.next;
        temp.next=null;
        size--;

        if(frontNode==null) {
                rearNode=null;
        }

        return element;

}
```

```
	public String toString() {
		if (this.frontNode == null) {
			return "[]";
		}

		String str = "[";
		Node currentNode = this.frontNode;
		while (currentNode != null) {
			str = str + "" + currentNode.data + " , ";
			currentNode = currentNode.next;
		}
		str = str.substring(0, str.length() - 2);
		str = str + "]";

		return str;
	}

}
```

Lab4.java

```
package com.jlcindia.queues.linkedlists;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab8 {
	public static void main(String[] args) {

		MyQueue myque = new MyQueue();

		System.out.println(myque);
		System.out.println(myque.size());
		System.out.println(myque.isEmpty());

		myque.offer(10);
		myque.offer(20);
```

```
                myque.offer(30);
                myque.offer(40);
                myque.offer(50);

                System.out.println(myque);
                System.out.println(myque.size());
                System.out.println(myque.isEmpty());

                System.out.println("--------------------");
                System.out.println(myque.peek()); //10
                System.out.println("--------------------");

                myque.poll();
                myque.poll();

                System.out.println(myque);
                System.out.println("--------------------");
                System.out.println(myque.peek()); //30
                System.out.println("--------------------");

                myque.offer(60);
                myque.offer(70);

                System.out.println(myque);
                System.out.println("--------------------");
                System.out.println(myque.peek()); //30
                System.out.println("--------------------");

                myque.offer(99);
        }
}
```