DSA

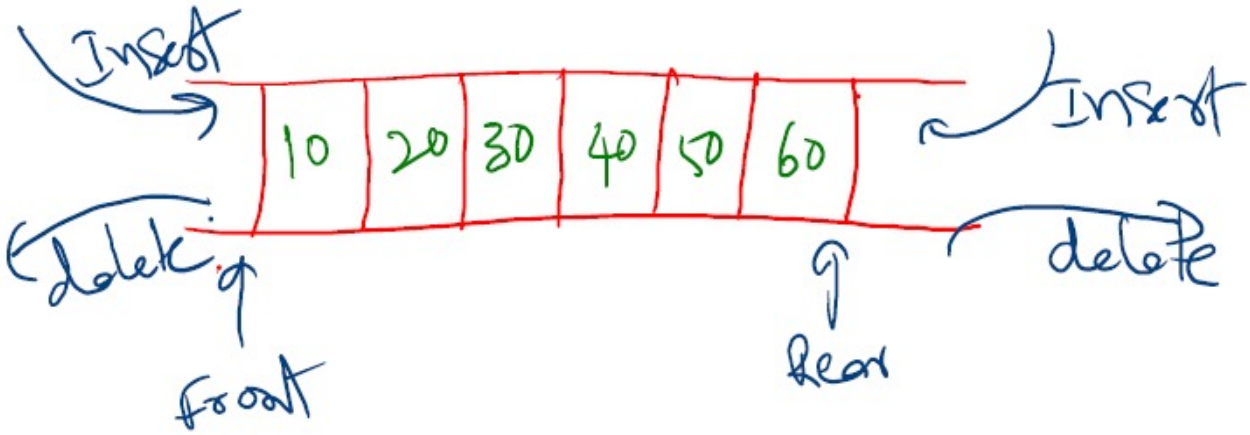
Module 12

Deques

Author
Srinivas Dande

12. Introduction to Deques

- Deque - Doubly Ended Queues.
- Deque is a special type of Queue.
- Deque does not follow FIFO Rule.
- You can do **Insert** and **Delete** Operations in the **Front of the Deque**
- You can do **Insert** and **Delete** Operations in the **Back(Rear) of the Deque**
- **Insert Operation** on the Deque is called as **Enqueue**
- **Delete Operation** on the Deque is called as **Dequeue**



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

> ✓ A **Deque** is a Special Queue which is an ordered list in which insertions are done at both **front** and **rear** and deletions are also done at both **front and rear**.

12.1. Queue Operations

a) isEmpty()
b) size()
c) offerFirst()
d) offerLast()
e) pollFirst()
f) pollLast()
g) peekFirst()
h) peekLast()

a) isEmpty():

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

b) size():

- Returns the size of the Queue

c) offerFirst():

- Inserts the Element at the Front of the Queue

d) offerLast():

- Inserts the Element at the Rear of the Queue

e) pollFirst():

- Removes the Front Element from the Queue and returns the same

f) pollLast():

- Removes the Rear Element from the Queue and returns the same

g) peekFirst():

- Removes the Front Element from the Queue and returns the same

h) peekLast():

- Removes the Front Element from the Queue and returns the same

12.2. Corner Conditions

a) Deque Underflow
b) Deque Overflow

a) Deque Underflow

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

b) Deque Overflow

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

12.3. Time Complexity of Stack Operations

Operation	Fixed Arrays	Circular Fixed Arrays	Circular Dynamic Arrays	LinkedList
offerFirst()	O(n)	O(1)	Amortized O(1)	O(1)
offerLast()	O(1)	O(1)	Amortized O(1)	O(1)
pollFirst()	O(n)	O(1)	O(1)	O(1)
pollLast()	O(1)	O(1)	O(1)	O(1)
peekFirst()	O(1)	O(1)	O(1)	O(1)
peekLast()	O(1)	O(1)	O(1)	O(1)

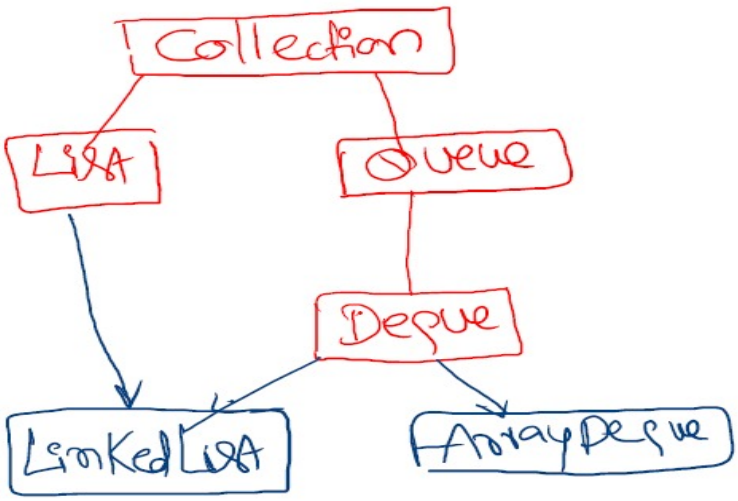
12.4. Applications of Queues

- Deques can be used in many of the real-world applications
  - **a)** Deque can be used as both Stack and Queue
  - **b)** Deque can be used to maintain recently visited URLs in the Browser History
  - **c)** Deque can be used to store list of Undo Operations
  - **d)** Deque is used in Steal Process Scheduling Algorithm
  - **e)** Clockwise and anti-clockwise operations in deque are very faster and will be useful in many problems

    etc

12.5. Deques in Java Collection

- Java Collections has two classes for Deque 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

12.5.1. Using Deque Built-In Methods

Lab1.java

```
package com.jlcindia.deques;

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

		//Deque<Integer> mydeque = new ArrayDeque<>();
		Deque<Integer> mydeque = new LinkedList<>();

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

		mydeque.offerFirst(10);
		mydeque.offerFirst(20);
		mydeque.offerFirst(30);
		System.out.println(mydeque);

		mydeque.offerLast(55);
		mydeque.offerLast(66);
		mydeque.offerLast(77);
		System.out.println(mydeque);
		System.out.println(mydeque.size());
		System.out.println(mydeque.isEmpty());

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

```
                System.out.println(mydeque.peekFirst()); //30
                System.out.println(mydeque.peekLast()); //77
                System.out.println("-------------------------");
                mydeque.pollFirst(); //Deletes 30
                mydeque.pollLast(); // Deletes 77
                System.out.println("-------------------------");
                System.out.println(mydeque.peekFirst()); //20
                System.out.println(mydeque.peekLast()); //66
                System.out.println("-------------------------");

        }
}
```

12.5.2. Using Deque Built-In Methods

Lab2.java

```
package com.jlcindia.deques;

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

                //Deque<Integer> mydeque = new ArrayDeque<>();
                Deque<Integer> mydeque = new LinkedList<>();

                System.out.println(mydeque);
                System.out.println(mydeque.size());
                System.out.println(mydeque.isEmpty());
```

```
        mydeque.addFirst(10);
        mydeque.addFirst(20);
        mydeque.addFirst(30);
        System.out.println(mydeque);

        mydeque.addLast(55);
        mydeque.addLast(66);
        mydeque.addLast(77);
        System.out.println(mydeque);
        System.out.println(mydeque.size());
        System.out.println(mydeque.isEmpty());
        System.out.println("-------------------------");
        System.out.println(mydeque.getFirst()); //30
        System.out.println(mydeque.getLast()); //77
        System.out.println("-------------------------");
        mydeque.removeFirst(); //Deletes 30
        mydeque.removeLast(); // Deletes 77
        System.out.println("-------------------------");
        System.out.println(mydeque.getFirst()); //20
        System.out.println(mydeque.getLast()); //66
        System.out.println("-------------------------");


    }
}
```

12.5.3. Traversing Deque elements in Forward Order

Lab3.java

```
package com.jlcindia.deques;

import java.util.*;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab3 {
	public static void main(String[] args) {
		Deque<Integer> mydeque = new LinkedList<>();
		System.out.println(mydeque);

		mydeque.addFirst(10);
		mydeque.addFirst(20);
		mydeque.addFirst(30);
		mydeque.addLast(55);
		mydeque.addLast(66);
		mydeque.addLast(77);
		System.out.println(mydeque);

		//1. Using Iterator
		Iterator<Integer> it = mydeque.iterator();
		while(it.hasNext()) {
			System.out.print(it.next()+"\t");
		}
		System.out.println("\n-------------------------");
		//2.Using enhanced for loop
		for(Integer x: mydeque) {
			System.out.print(x+"\t");
		}
	}
}
```

12.5.4. Reverse the Deque Elements

Lab4.java

```
package com.jlcindia.deques;

import java.util.Deque;
import java.util.Iterator;
import java.util.LinkedList;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab4 {
	public static void main(String[] args) {

		Deque<Integer> mydeque = new LinkedList<>();

		System.out.println(mydeque);

		mydeque.addFirst(10);
		mydeque.addFirst(20);
		mydeque.addFirst(30);
		mydeque.addLast(55);
		mydeque.addLast(66);
		mydeque.addLast(77);
		System.out.println(mydeque);

		//1. Using Iterator
		Iterator<Integer> it = mydeque.descendingIterator();
		while(it.hasNext()) {
			System.out.print(it.next()+"\t");
		}

	}
}
```

12.5.5. ArrayDeque can be used as Stack

Lab5.java

```
package com.jlcindia.deques;

import java.util.*;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab5 {
	public static void main(String[] args) {

		Deque<Integer> mystack = new ArrayDeque<>();
		System.out.println(mystack);

		mystack.push(10);
		mystack.push(20);
		mystack.push(30);
		mystack.push(40);
		mystack.push(50);

		System.out.println(mystack);
		System.out.println(mystack.peek());

		mystack.pop();
		System.out.println(mystack);
		System.out.println(mystack.peek());

	}
}
```

12.5.6. ArrayDeque can be used as Queue

Lab6.java

```
package com.jlcindia.deques;

import java.util.*;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab6 {
	public static void main(String[] args) {

		Deque<Integer> myque = new ArrayDeque<>();
		System.out.println(myque);

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

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

		myque.poll();
		System.out.println(myque);
		System.out.println(myque.peek());

	}
}
```

12.6. Deque Implementation using Arrays

MyDeque.java

```
package com.jlcindia.deques.arrays1;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */

//Design Deque using Arrays
// Non-Circular Deque or Non-Circular Array

class MyDeque {

	int capacity;
	int size;
	Integer myarray[];

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

        return 0;
}

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

        return size-1;
}

public boolean offerFirst(int element) {

        if(isFull()) {
                return false;
        }

        for(int i= size-1;i>=0;i--) {
                myarray[i+1] = myarray[i];
        }

        myarray[0] = element;
        size++;
        return true;

}
```

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

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

}

public Integer pollFirst() {
        if(isEmpty())
                return null;

        int element = myarray[0];

        for(int i=0;i<size-1;i++) {
                myarray[i] = myarray[i+1];
        }

        myarray[size-1] = null;
        size--;
        return element;
}

public Integer pollLast() {
        if(isEmpty())
                return null;

        int element = myarray[size-1];

        myarray[size-1] = null;
        size--;
        return element;
}
```

```
	public Integer peekFirst() {
		if(isEmpty()) {
			return null;
		}

		return myarray[0];
	}

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

		return myarray[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;
	}

}
```

Lab7.java

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

public class Lab7 {
	public static void main(String[] args) {

		MyDeque mydeque = new MyDeque(6);

		System.out.println(mydeque);
		System.out.println(mydeque.size());
		System.out.println(mydeque.isEmpty());
		System.out.println(mydeque.isFull());
		System.out.println(mydeque.getFront());
		System.out.println(mydeque.getRear());
		System.out.println("------------------------------");


		mydeque.offerFirst(10);
		mydeque.offerFirst(20);
		mydeque.offerFirst(30);

		System.out.println(mydeque);
		System.out.println(mydeque.isEmpty());
		System.out.println(mydeque.isFull());
		System.out.println(mydeque.getFront()); //0
		System.out.println(mydeque.getRear()); //2
		System.out.println("------------------------------");


		mydeque.offerLast(55);
		mydeque.offerLast(66);
		mydeque.offerLast(77);
```

```
            System.out.println(mydeque);
            System.out.println(mydeque.size());
            System.out.println(mydeque.isEmpty());
            System.out.println(mydeque.isFull());
            System.out.println(mydeque.getFront()); //0
            System.out.println(mydeque.getRear()); //5

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

            System.out.println(mydeque.peekFirst()); //30
            System.out.println(mydeque.peekLast()); //77
            System.out.println("-------------------------");

            mydeque.pollFirst(); //Deletes 30
            mydeque.pollLast(); // Deletes 77
            System.out.println(mydeque);

            System.out.println("-------------------------");
            System.out.println(mydeque.peekFirst()); //20
            System.out.println(mydeque.peekLast()); //66
            System.out.println("-------------------------");

        }
}
```

12.7. Deque Implementation using Arrays

MyDeque.java

```
package com.jlcindia.deques.arrays2;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */

//Design Circular Deque using Arrays

class MyDeque {

	int front;
	int rear;
	int capacity;
	int size;
	Integer myarray[];

	public MyDeque(int capacity) {
		this.capacity = capacity;
		this.myarray = new Integer[capacity];

		this.size = 0;
		this.front = -1;
		this.rear = -1;
	}

	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 rear;
}

public boolean offerFirst(int element) {

        if (isFull())
                return false;

        if (front == -1) {
                front = 0;
                rear = 0;
        } else if (front == 0) {
                front = capacity - 1;
        } else {
                front = front - 1;
        }

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

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

        if (rear == -1) {
                front = 0;
                rear = 0;
        }
        if (rear == capacity - 1) {
                rear = 0;
        } else {
                rear = rear + 1;
        }

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

public Integer pollFirst() {
        if (isEmpty())
                return null;

        int element = myarray[front];
        myarray[front] = null;
        size--;

        if(front==rear) {
                front=-1;
                rear=-1;
        }else   if (front == capacity - 1) {
                front = 0;
        } else {
                front = front + 1;
        }
        return element;
}
```

```
public Integer pollLast() {
        if (isEmpty()) {
                return null;
        }

        int element = myarray[rear];
        myarray[rear] = null;
        size--;

        if(front==rear) {
                front=-1;
                rear=-1;
        }else   if (rear == 0) {
                rear = capacity-1;
        } else {
                rear = rear - 1;
        }

        return element;
}

public Integer peekFirst() {
        if (isEmpty())
                return null;

        return myarray[front];
}

public Integer peekLast() {
        if (isEmpty())
                return null;

        return myarray[rear];
}
```

```
	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;
	}
}
```

Lab8.java

```
package com.jlcindia.deques.arrays2;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
public class Lab8 {
	public static void main(String[] args) {
		MyDeque mydeque = new MyDeque(6);

		System.out.println(mydeque);
		System.out.println(mydeque.size());
		System.out.println(mydeque.isEmpty());
		System.out.println(mydeque.isFull());
		System.out.println(mydeque.getFront()); //-1
		System.out.println(mydeque.getRear()); //-1
		System.out.println("------------------------------");

		mydeque.offerFirst(10);
		mydeque.offerFirst(20);
		mydeque.offerFirst(30);
		System.out.println(mydeque);
```

```
        System.out.println(mydeque.isEmpty());
        System.out.println(mydeque.isFull());
        System.out.println(mydeque.getFront()); //4
        System.out.println(mydeque.getRear()); //0
        System.out.println("-----------------------------");

        mydeque.offerLast(55);
        mydeque.offerLast(66);
        mydeque.offerLast(77);

        System.out.println(mydeque);
        System.out.println(mydeque.size());
        System.out.println(mydeque.isEmpty());
        System.out.println(mydeque.isFull());
        System.out.println(mydeque.getFront()); //4
        System.out.println(mydeque.getRear()); //3

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

        System.out.println(mydeque.peekFirst()); //30
        System.out.println(mydeque.peekLast()); //77
        System.out.println("-------------------------");

        mydeque.pollFirst();
        mydeque.pollFirst();
        mydeque.pollLast();
        mydeque.pollLast();
        System.out.println(mydeque);

        System.out.println(mydeque.getFront()); //4
        System.out.println(mydeque.getRear()); //3
        System.out.println("-------------------------");
        System.out.println(mydeque.peekFirst()); //10
        System.out.println(mydeque.peekLast()); //55
        System.out.println("-------------------------");
    }
}
```

12.8. Deque Implementation using LinkedLists

Node.java

```
package com.jlcindia.deques.linkedlist;
/*
* @Author : Srinivas Dande
* @Company: Java Learning Center
* */
class Node{

	int data;
	Node next;
	Node prev;

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

MyDeque.java

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

class MyDeque {

	Node front;
	Node rear;
	int size;
```

```
public MyDeque() {
        this.size = 0;
        this.front = null;
        this.rear = null;
}

public int size() {
        return size;
}

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

public boolean offerFirst(int element) {

        Node temp =new Node(element);

        if(front==null) {
                front=temp;
                rear=temp;
                size++;
                return true;
        }

        temp.next = front;
        front.prev=temp;
        front=front.prev;

        size++;
        return true;
}
```

```
public boolean offerLast(int element) {

Node temp =new Node(element);

        if(front==null) {
                front=temp;
                rear=temp;
                size++;
                return true;
        }

        rear.next = temp;
        temp.prev = rear;
        rear = rear.next;

        size++;
        return true;
}

public Integer pollFirst() {
        if(size==0) {
                return null;
        }

        Node temp = front;
        front = front.next;
        front.prev=null;

        temp.next=null;
        size--;

        return temp.data;
}
```

```
public Integer pollLast() {

        if(size==0) {
                return null;
        }

        Node temp = rear;
        rear = rear.prev;
        rear.next=null;

        temp.prev=null;
        size--;

        return temp.data;
}

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

        return front.data;
}

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

        return rear.data;
}
```

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

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

                return str;
        }
}
```

Lab9.java

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

public class Lab9 {
        public static void main(String[] args) {

                MyDeque mydeque = new MyDeque();

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

```
        mydeque.offerFirst(10);
        mydeque.offerFirst(20);
        mydeque.offerFirst(30);

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

        mydeque.offerLast(55);
        mydeque.offerLast(66);
        mydeque.offerLast(77);

        System.out.println(mydeque);

        System.out.println("-------------------------");
        System.out.println(mydeque.peekFirst()); //30
        System.out.println(mydeque.peekLast()); //77
        System.out.println("-------------------------");

        mydeque.pollFirst();
        mydeque.pollFirst();
        mydeque.pollLast();
        mydeque.pollLast();
        System.out.println(mydeque);

        System.out.println("-------------------------");
        System.out.println(mydeque.peekFirst()); //10
        System.out.println(mydeque.peekLast()); //55
        System.out.println("-------------------------");
    }
}
```