

QUEUE (I) :-

- ★ It is a predefined **Interface** available in the “ **UTIL** ” package.
- ★ It will arrange the values in the form of **first in first out of order (FIFO)** .
- ★ Here we can **add the elements at the end of the Queue** remove the value from the front of the queue.
- ★ It will **use dynamic array like array list for storing the elements**.
- ★ For these interfaces we have implementation class like linked list or array dequeue.

METHODS :-

1. **public boolean add (Object)**
2. **public boolean offer (Object)**
3. **public Object peek ()**
4. **public Object poll ()**
5. **public void remove ()**
6. **public int size ()**
7. **public boolean isEmpty ()**

USE AND DESCRIPTION ABOUT METHODS :-

1. **boolean add (Object)** → Appends the specified element to the first of the Queue.
2. **boolean offer (Object)** → Appends the specified element to the first of the Queue.
3. **Object peek ()** → Returns first object from the queue.

```
8 public static void main(String[] args) {
9     Queue q=new ArrayDeque();
10    q.add(1);
11    q.add(2);
12    q.add(3);
13    q.add(4);
14    q.add(5);
15    System.out.println(q);
16    System.out.println(q.peek());
17    System.out.println(q);
--
```

```
<terminated> Queue1 [Java Application] C:\Users\vishn\Downloads\
[1, 2, 3, 4, 5]
1
[1, 2, 3, 4, 5]
```

4. **Object poll ()** → Returns and removes first object from the queue.

```
8 public static void main(String[] args) {
9     Queue q=new ArrayDeque();
10    q.add(1);
11    q.add(2);
12    q.add(3);
13    q.add(4);
14    q.add(5);
15    System.out.println(q);
16    System.out.println(q.poll());
17    System.out.println(q);
--
```

```
<terminated> Queue1 [Java Application] C:\Users\vishn\Downloads\
[1, 2, 3, 4, 5]
1
[2, 3, 4, 5]
```

5. **void remove ()** → Removes first object from the queue.
6. **int size ()** → To get the size of Queue.
7. **boolean isEmpty ()** → Returns true if the queue is empty else returns false.

PRIORITY QUEUE :-

- It is a predefined **class** present in “**UTIL**” package.
- It will **store the Objects on the basis of priority**.

CONSTRUCTORS :-

```
public PriorityQueue ()  
public PriorityQueue ( int )  
public PriorityQueue ( comparator )
```

METHODS :-

1. **public boolean add (Object)**
2. **public boolean offer (Object)**
3. **public Object peek ()**
4. **public boolean remove (Object)**
5. **public int size ()**
6. **public boolean contains (Object)**
7. **public Object [] toArray ()**
8. **public void clear ()**
9. **public Object poll ()**

USE AND DESCRIPTION ABOUT METHODS :-

1. **boolean add (Object)** → Appends the elements to the first of the queue.
2. **boolean offer (Object)** → Appends the elements to the first of the queue.
3. **Object peek ()** → Retrieves the first element from the queue.
4. **boolean remove (Object)** → Removes the element from the queue.
5. **int size ()** → Returns the size of the queue.
6. **boolean contains (Object)** → Returns true if two queue contains same elements else returns false.
7. **Object [] toArray ()** → Converts queue to array.
8. **void clear ()** → Removes all the elements from the queue.
9. **Object poll ()** → Retrieves and removes the element from the queue.

1. Create a java application where we have pair class which has 2 variables x and y taking pair as generic to the priority queue such that they get ordered by y when we print the queue.

```
class Pair  
{  
    int X;  
    String Y;  
    Pair(int X,String Y)  
    {  
        this.X=X;  
        this.Y=Y;  
    }  
    public static void main(String[]args)  
    {  
        Queue<Pair> p=new PriorityQueue<>(new Pair1());  
        Queue<Pair> P=new PriorityQueue<>(new Pair2());  
        Pair p1=new Pair(1,"JAMES");  
        Pair p2=new Pair(2,"PYTHON");
```

```

Pair p3=new Pair(5,"SCRIPT");
Pair p4=new Pair(4,"HTML");
Pair p5=new Pair(3,"CSS");
p.add(p1);
p.add(p2);
p.add(p3);
p.add(p4);
p.add(p5);
while(!p.isEmpty())
{
    Pair e=p.remove();
    System.out.println(e.X+" "+e.Y);
}
System.out.println();
P.add(p1);
P.add(p2);
P.add(p3);
P.add(p4);
P.add(p5);
while(!P.isEmpty())
{
    Pair e=P.remove();
    System.out.println(e.X+" "+e.Y);
}
System.out.println();
P.add(p1);
P.add(p2);
P.add(p3);
P.add(p4);
P.add(p5);
while(!P.isEmpty())
{
    Pair e=P.remove();
    System.out.println(e.X+" "+e.Y);
}
}

class Pair1 implements Comparator<Pair>
{
    public int compare(Pair a1,Pair a2)
    {
        if(a1.X>a2.X)
            return -1;
        else if(a1.X<a2.X)
            return 1;
        else
            return 0;
    }
}

class Pair2 implements Comparator<Pair>
{
    public int compare(Pair a1,Pair a2)
    {
        return a1.Y.compareTo(a2.Y);
    }
}

```



OUTPUT :-

5 SCRIPT
4 HTML
3 CSS
2 PYTHON
1 JAMES

3 CSS
4 HTML
1 JAMES
2 PYTHON
5 SCRIPT

3 CSS
4 HTML
1 JAMES
2 PYTHON
5 SCRIPT



DEQUEUE (I):-

- ♣ It is a predefined **Interface** present in “ **UTIL** ” package.
- ♣ It is the **child Interface** of queue interface.
- ♣ It supports the **addition as well as the removal of elements from both ends of the data structure.**
- ♣ For this interface we have **implementation classes like array dequeue and linked list.**



ARRAY DEQUEUE :-

- It is a predefined **class** present in “**UTIL**” package.
- By using this class we can **add or remove elements from both sides**.

CONSTRUCTORS :-

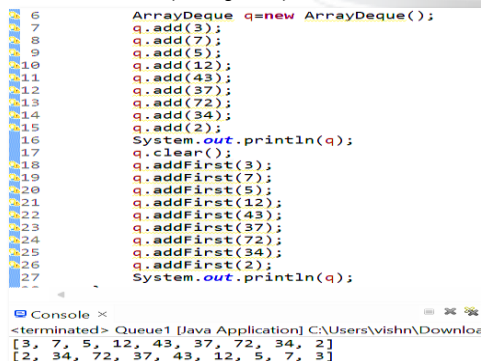
ArrayDeque ()
ArrayDeque (int)
ArrayDeque (collection)

METHODS :-

1. **public void addFirst (Object)**
2. **public void addLast (Object)**
3. **public boolean offerFirst (Object)**
4. **public boolean offerLast (Object)**
5. **public Object removeFirst ()**
6. **public Object removeLast ()**
7. **public Object pollFirst ()**
8. **public Object pollLast ()**
9. **public Object peekFirst ()**
10. **public Object peekLast ()**
11. **public Object getFirst ()**
12. **public Object getLast ()**

USE AND DESCRIPTION ABOUT METHODS :-

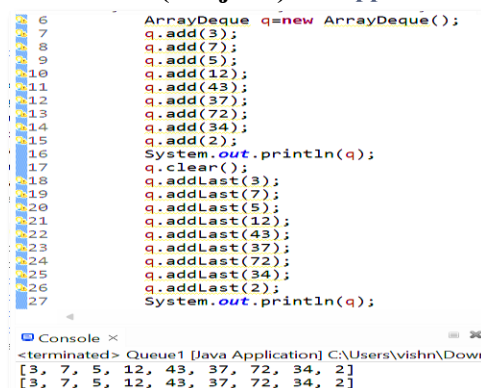
1. **void addFirst (Object)** → Appends the elements to the first of the queue.



```
6      ArrayDeque q=new ArrayDeque();
7      q.add(3);
8      q.add(7);
9      q.add(5);
10     q.add(12);
11     q.add(43);
12     q.add(37);
13     q.add(72);
14     q.add(34);
15     q.add(2);
16     System.out.println(q);
17     q.clear();
18     q.addFirst(3);
19     q.addFirst(7);
20     q.addFirst(5);
21     q.addFirst(12);
22     q.addFirst(43);
23     q.addFirst(37);
24     q.addFirst(72);
25     q.addFirst(34);
26     q.addFirst(2);
27     System.out.println(q);
```

Console ×
<terminated> Queue1 [Java Application] C:\Users\vishn\Downloa
[3, 7, 5, 12, 43, 37, 72, 34, 2]
[2, 34, 72, 37, 43, 12, 5, 7, 3]

2. **void addLast (Object)** → Appends the elements to the last of the queue.



```
6      ArrayDeque q=new ArrayDeque();
7      q.add(3);
8      q.add(7);
9      q.add(5);
10     q.add(12);
11     q.add(43);
12     q.add(37);
13     q.add(72);
14     q.add(34);
15     q.add(2);
16     System.out.println(q);
17     q.clear();
18     q.addLast(3);
19     q.addLast(7);
20     q.addLast(5);
21     q.addLast(12);
22     q.addLast(43);
23     q.addLast(37);
24     q.addLast(72);
25     q.addLast(34);
26     q.addLast(2);
27     System.out.println(q);
```

Console ×
<terminated> Queue1 [Java Application] C:\Users\vishn\Downloa
[3, 7, 5, 12, 43, 37, 72, 34, 2]
[3, 7, 5, 12, 43, 37, 72, 34, 2]

3. **boolean offerFirst (Object)** → Appends the elements to the first of the queue.

```
8- public static void main(String[] args) {
9-     ArrayDeque q=new ArrayDeque();
10-    q.add(3);
11-    q.add(7);
12-    q.add(5);
13-    q.add(12);
14-    q.add(43);
15-    q.add(37);
16-    q.add(72);
17-    q.add(34);
18-    q.add(2);
19-    System.out.println(q);
20-    System.out.println(q.offerFirst(7587));
21-    System.out.println(q);
}
```

Console ×

<terminated> Queue1 [Java Application] C:\Users\vishn\Downloads\ect
[3, 7, 5, 12, 43, 37, 72, 34, 2]
true
[7587, 3, 7, 5, 12, 43, 37, 72, 34, 2]

4. **boolean offerLast (Object)** → Appends the elements to the last of the queue.

5. **Object removeFirst ()** → Retrieves and removes first elements from the queue.

```
5- public static void main(String[] args) {
6-     ArrayDeque q=new ArrayDeque();
7-     q.add(3);
8-     q.add(7);
9-     q.add(5);
10-    q.add(12);
11-    q.add(43);
12-    q.add(37);
13-    q.add(72);
14-    q.add(34);
15-    q.add(2);
16-    System.out.println(q);
17-    System.out.println(q.removeFirst());
18-    System.out.println(q);
}
```

Console ×

<terminated> Queue1 [Java Application] C:\Users\vishn\Downloads\ect
[3, 7, 5, 12, 43, 37, 72, 34, 2]
3
[7, 5, 12, 43, 37, 72, 34, 2]

6. **Object removeLast ()** → Retrieves and removes last elements from the queue.

```
5- public static void main(String[] args) {
6-     ArrayDeque q=new ArrayDeque();
7-     q.add(3);
8-     q.add(7);
9-     q.add(5);
10-    q.add(12);
11-    q.add(43);
12-    q.add(37);
13-    q.add(72);
14-    q.add(34);
15-    q.add(2);
16-    System.out.println(q);
17-    System.out.println(q.removeLast());
18-    System.out.println(q);
}
```

Console ×

<terminated> Queue1 [Java Application] C:\Users\vishn\Downloads\ect
[3, 7, 5, 12, 43, 37, 72, 34, 2]
2
[3, 7, 5, 12, 43, 37, 72, 34]

7. **Object pollFirst ()** → Retrieves and removes first elements from the queue.

```
5- public static void main(String[] args) {
6-     ArrayDeque q=new ArrayDeque();
7-     q.add(3);
8-     q.add(7);
9-     q.add(5);
10-    q.add(12);
11-    q.add(43);
12-    q.add(37);
13-    q.add(72);
14-    q.add(34);
15-    q.add(2);
16-    System.out.println(q);
17-    System.out.println(q.pollFirst());
18-    System.out.println(q);
}
```

Console ×

<terminated> Queue1 [Java Application] C:\Users\vishn\Downloads\ect
[3, 7, 5, 12, 43, 37, 72, 34, 2]
3
[7, 5, 12, 43, 37, 72, 34, 2]

8. Object pollLast () → Retrieves and removes last elements from the queue.

```
5- public static void main(String[] args) {
6     ArrayDeque q=new ArrayDeque();
7     q.add(3);
8     q.add(7);
9     q.add(5);
10    q.add(12);
11    q.add(43);
12    q.add(37);
13    q.add(72);
14    q.add(34);
15    q.add(2);
16    System.out.println(q);
17    System.out.println(q.pollLast());
18    System.out.println(q);
}
```

Console ×

<terminated> Queue1 [Java Application] C:\Users\vishn\Downloads
[3, 7, 5, 12, 43, 37, 72, 34, 2]
2
[3, 7, 5, 12, 43, 37, 72, 34]

9. Object peekFirst () → Retrieves first elements from the queue.

```
5- public static void main(String[] args) {
6     ArrayDeque q=new ArrayDeque();
7     q.add(3);
8     q.add(7);
9     q.add(5);
10    q.add(12);
11    q.add(43);
12    q.add(37);
13    q.add(72);
14    q.add(34);
15    q.add(2);
16    System.out.println(q);
17    System.out.println(q.peekFirst());
18    System.out.println(q);
}
```

Console ×

<terminated> Queue1 [Java Application] C:\Users\vishn\Downloads
[3, 7, 5, 12, 43, 37, 72, 34, 2]
3
[3, 7, 5, 12, 43, 37, 72, 34, 2]

10. Object peekLast () → Retrieves last elements from the queue.

```
5- public static void main(String[] args) {
6     ArrayDeque q=new ArrayDeque();
7     q.add(3);
8     q.add(7);
9     q.add(5);
10    q.add(12);
11    q.add(43);
12    q.add(37);
13    q.add(72);
14    q.add(34);
15    q.add(2);
16    System.out.println(q);
17    System.out.println(q.peekLast());
18    System.out.println(q);
}
```

Console ×

<terminated> Queue1 [Java Application] C:\Users\vishn\Downloads
[3, 7, 5, 12, 43, 37, 72, 34, 2]
2
[3, 7, 5, 12, 43, 37, 72, 34, 2]

11. Object getFirst () → Retrieves first elements from the queue.

```
6     ArrayDeque q=new ArrayDeque();
7     q.add(3);
8     q.add(7);
9     q.add(5);
10    q.add(12);
11    q.add(43);
12    q.add(37);
13    q.add(72);
14    q.add(34);
15    q.add(2);
16    System.out.println(q);
17    System.out.println(q.getFirst());
18    System.out.println(q);
}
```

Console ×

<terminated> Queue1 [Java Application] C:\Users\vishn\Downlo
[3, 7, 5, 12, 43, 37, 72, 34, 2]
3
[3, 7, 5, 12, 43, 37, 72, 34, 2]

12. Object getLast () → Retrieves last elements from the queue.

```
6      ArrayDeque q=new ArrayDeque();
7      q.add(3);
8      q.add(7);
9      q.add(5);
10     q.add(12);
11     q.add(43);
12     q.add(37);
13     q.add(72);
14     q.add(34);
15     q.add(2);
16     System.out.println(q);
17     System.out.println(q.getLast());
18     System.out.println(q);
```

Console ×

```
<terminated> Queue1 [Java Application] C:\Users\vishn\Downl
[3, 7, 5, 12, 43, 37, 72, 34, 2]
2
[3, 7, 5, 12, 43, 37, 72, 34, 2]
```

1. Create a java application where we need to create object of Array dequeue class add elements in this object, Create second object for Array dequeue class add elements from previous object, add and remove elements then display all the values.

```
import java.util.*;
public class Queue
{
    public static void main(String[] args)
    {
        ArrayDeque<Integer> ad1=new ArrayDeque<>();
        ad1.add(2);
        ad1.add(5);
        ad1.add(3);
        ad1.add(1);
        ad1.add(4);
        ArrayDeque<Integer> ad2=new ArrayDeque<>(ad1);
        System.out.println(ad1.peekFirst());
        System.out.println(ad1.peekLast());
        ad1.removeFirst();
        System.out.println(ad1.peekFirst());
        ad1.removeLast();
        System.out.println(ad1.peekLast());
        while(!ad1.isEmpty())
        {
            System.out.print(ad1.pollFirst()+" ");
        }
        System.out.println();
        ad2.add(7);
        ad2.add(6);
        ad2.add(9);
        ad2.add(8);
        ad2.remove(6);
        ad2.add(10);
        while(!ad2.isEmpty())
        {
            System.out.print(ad2.pollFirst()+" ");
        }
    }
}
```

OUT PUT :-

```
2
4
5
1
5 3 1
2 5 3 1 4 7 9 8 10
```