

Roman Hrabovskyi

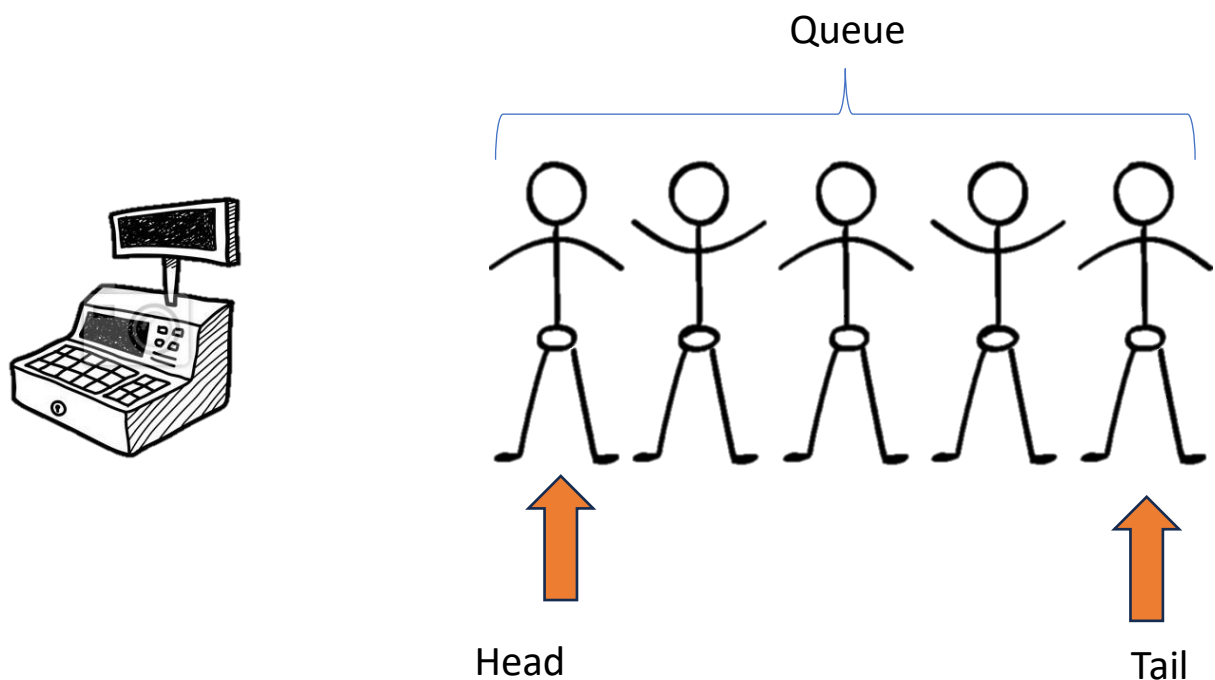
Student id 19385

## Queues

Queue = FIFO data structure. First-In First-Out

A collection designed for holding elements prior to processing

For example a line of people in the shop :



When the first person leaves the queue (dequeues), the size of the queue decreases by one, and the second person becomes the new head of the queue. If

someone joins the queue (enqueues), they become the new tail of the queue. It is the principle of how queue work.

The next step is to understand how it works in the code. For that we should know the basic commands like :

Add = enqueue, offer()

Remove = dequeue, poll()


Element = peek()

	<i>Throws exception</i>	<i>Returns special value</i>
Insert	<u><code>add(e)</code></u>	<u><code>offer(e)</code></u>
Remove	<u><code>remove()</code></u>	<u><code>poll()</code></u>
Examine	<u><code>element()</code></u>	<u><code>peek()</code></u>

And of course how to create the queue:

```
1 import java.util.Queue;
2 import java.util.LinkedList;
3
4 public class Main {
5     public static void main(String[] args) {
6         Queue<String> queue = new LinkedList<String>();
7     }
8 }
```

Then use the basic commands for adding a user in the queue:




```
1  import java.util.Queue;
2  import java.util.LinkedList;
3
4  public class Main {
5      public static void main(String[] args) {
6          Queue<String> queue = new LinkedList<String>();
7          queue.offer("Steave");
8          queue.offer("Karen");
9          queue.offer("Bob");
10         queue.offer("Liza");
11         System.out.println(queue);
12         queue.poll();
13         System.out.println(queue);
14         queue.poll();
15         System.out.println(queue);
16     }
17 }
```

Out:

```
[Steave, Karen, Bob, Liza]
[Karen, Bob, Liza]
[Bob, Liza]
```

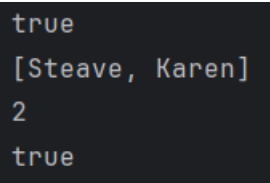
Important and very useful moment that we can use collection methods for our queue.

Ex:



```
1 public class Main {  
2     public static void main(String[] args) {  
3         Queue<String> queue = new LinkedList<String>();  
4         System.out.println(queue.isEmpty());  
5         queue.offer("Steave");  
6         queue.offer("Karen");  
7         System.out.println(queue);  
8         System.out.println(queue.size());  
9         System.out.println(queue.contains("Karen"));  
10    }  
11 }
```

Out:



```
true  
[Steave, Karen]  
2  
true
```

Where the queue is useful ?

1. Keyboard Buffer (letters should appear on the screen in the order they're pressed)
2. Printer Queue ( Print jobs should be completed in order)
3. Used in LinkedLists, PriorityQueues, Breadth-first search