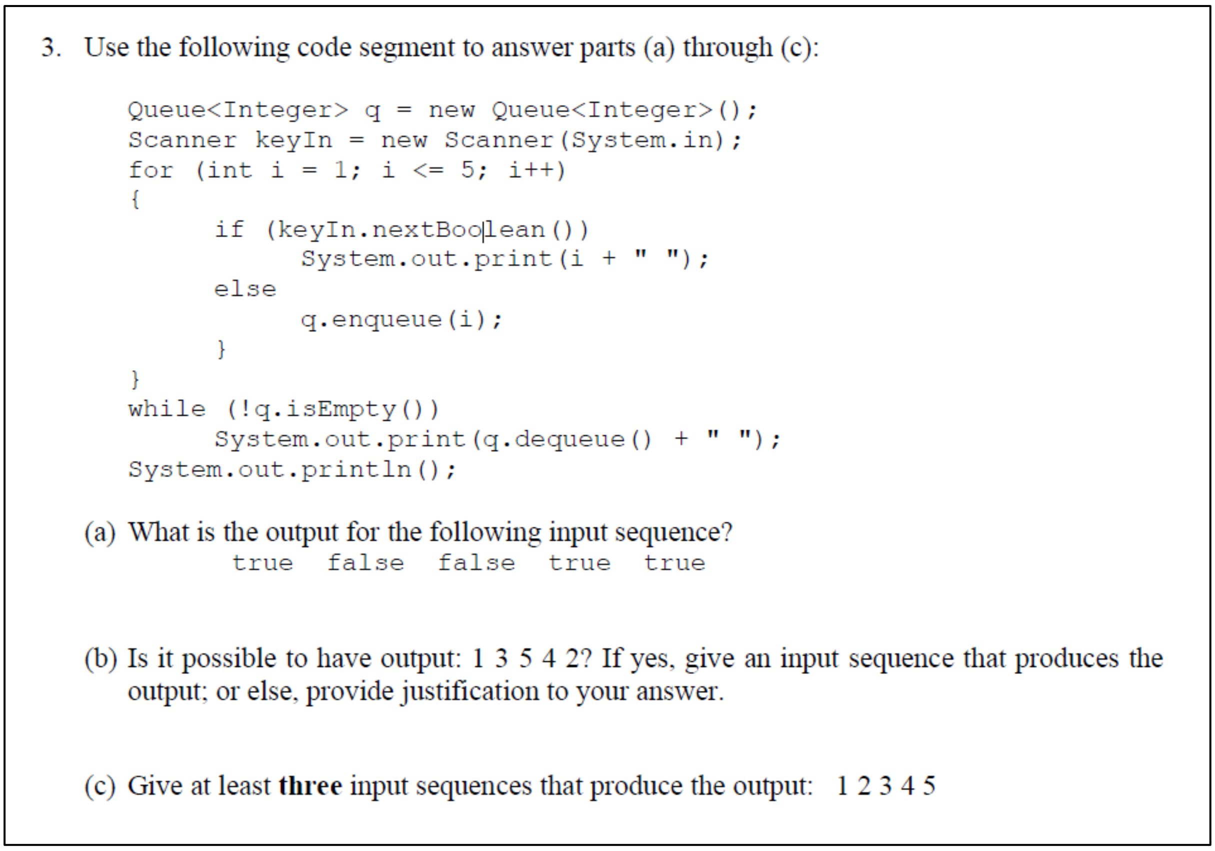
# Tutorial 7: Queue

1. Four examples:
    a. Cars lining up at a car wash.
    b. Waiting for your turn at the doctor’s
    c. Cup stack at the water dispenser (assume that it is refilled at the top and taken from the bottom)
    d. Queue at the ticketing booth of a train station.
2.

| **Queue** | **Stack** |
|---|---|
| First In First Out (FIFO) | Last In First Out (LIFO) |
| Elements are removed from the beginning of the queue | Elements are removed from the top of the stack |
| Common operations include enqueue and dequeue | Common operations include pop, push and peek |



3. Use the following code segment to answer parts (a) through (c):

```
Queue<Integer> q = new Queue<Integer>();
Scanner keyIn = new Scanner(System.in);
for (int i = 1; i <= 5; i++)
{
      if (keyIn.nextBoolean())
            System.out.print(i + " ");
      else
            q.enqueue(i);
      }
}
while (!q.isEmpty())
      System.out.print(q.dequeue() + " ");
System.out.println();
```

(a) What is the output for the following input sequence?

```
      true   false   false   true   true
```

(b) Is it possible to have output: 1 3 5 4 2? If yes, give an input sequence that produces the output; or else, provide justification to your answer.

(c) Give at least **three** input sequences that produce the output: 1 2 3 4 5

a) 1 4 5 2 3
b) No, this is due to the fact that 4 and 2 are in descending order. When the elements are queued into the queue, they will be queued in ascending order (1-5) and this will only allow 2 to be enqueued first followed by 4. Hence, the closest answer we can get is 1 3 5 2 4 (true false true false true).
c)
   a. false false false false false
   b. true true true true true
   c. true true true true false

4. Hand trace a queue X through the following operations:

```
X.enqueue(new Integer(14));
X.enqueue(new Integer(3));
X.enqueue(new Integer(5));

Object Y = X.dequeue();
X.enqueue(new Integer(7));
X.enqueue(new Integer(9));
Y = X.dequeue();
X.enqueue(new Integer(2));
X.enqueue(new Integer(4));
```

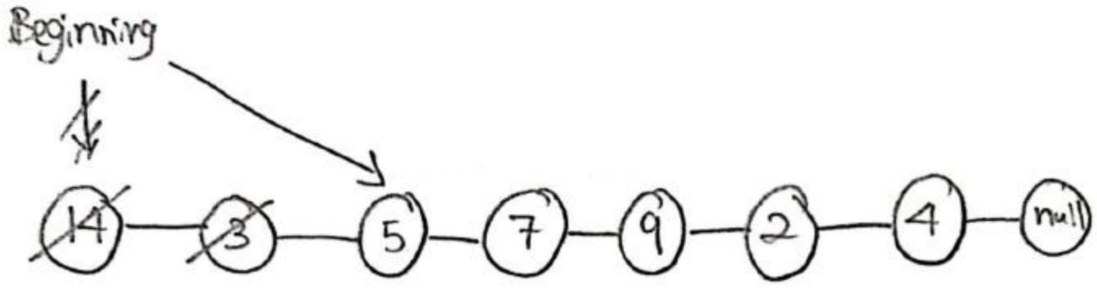
Given the resulting queue X above, what would be the result of each of the following?

```
a) X.front();

b) Y = X.dequeue();
   X.enqueue(new Integer(10));
   X.front();

c) Y = X.dequeue();

d) X.front();
```

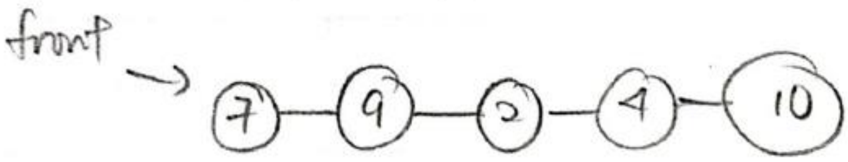


a) X.front();= 5

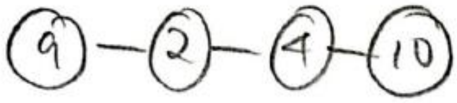
b) Y= X.dequeue(); → 5

X. enqueue (new Integer(10));

X.front(); → 7



c) Y = X.dequeue. → 7



d) X.front(); → 9.

5. Provide and explain three operations / functions of Linked List based Queue that you can add to the GenericQueue class, other than those discussed in the lecture (i.e., enqueue, dequeue and getSize – from Slide 10 in the lecture slide).

1. **Public boolean contains(E e)**
   Checks to see if the queue contains a certain entry. Returns true if the queue contains the entry, false otherwise.
2. **Public E peek();**
   Returns the first element of the queue.
3. **Public boolean remove(E e);**
   Removes a specific element from the queue. If there are more than two identical elements, remove the first occurring one. If remove successful, returns true, false otherwise.

## Tutorial 7b: Priority queue

1. Describe the main difference between Queue and PriorityQueue.

**In a queue**, the first-in-first-out rule is implemented whereas, **in a priority queue**, the values are removed on the basis of **priority**. The element with the highest **priority** is removed first.

2. Briefly provide THREE (3) real-life example in using PriorityQueue.

a) ExpressPass used in Theme parks.
b) Filtering section when you are selecting a restaurant from google maps or some other applications (ranks based on what you want – reviews, cost, etc)
c) Teachers selecting students to participate in a prestigious event based on merits.

3. Show the output for every System.out.println ((a) – (f)) in the following code:

```
 import java.util.*;
public static void main(String args[])
{
   PriorityQueue<String> pQueue = new PriorityQueue<String>();

   pQueue.offer("C++");
   pQueue.offer("Python");
   pQueue.offer("Java");
   pQueue.offer("Fortran");

   System.out.println("peek() gives us: "+ pQueue.peek());    //(a)

   System.out.println("The queue elements:");                 //(b)
   Iterator itr = pQueue.iterator();
   while (itr.hasNext())
      System.out.println(itr.next());                         //(b)

   pQueue.poll();
   System.out.println("After poll():");                      //(c)
   Iterator<String> itr2 = pQueue.iterator();
   while (itr2.hasNext())
      System.out.println(itr2.next());                        //(c)

   pQueue.remove("Java");
   System.out.println("After remove():");                    //(d)
   Iterator<String> itr3 = pQueue.iterator();
   while (itr3.hasNext())
      System.out.println(itr3.next());                        //(d)

   boolean b = pQueue.contains("Ruby");
   System.out.println ( "Priority queue contains Ruby or not?: " + b);        //(e)

   Object[] arr = pQueue.toArray();
   System.out.println ( "Value in array: ");                  //(f)
   for (int i = 0; i<arr.length; i++)
     System.out.println ( "Value: " + arr[i].toString()) ;    //(f)
```

a) peek() gives us: C++

b)

```
The queue elements:
C++
Fortran
Java
Python
```

c)

```
After poll():
Fortran
Python
Java
```

d)

```
After remove():
Fortran
Python
```

e)

```
Priority queue contains Ruby or not?: false
```

f)

```
Value in array:
Value: Fortran
Value: Python
```

4. Answer the following sub-questions with referring to the following code:

```
public class PriorityQueue2 {
   public static void main(String... args ){
      PriorityQueueComparator pqc=new PriorityQueueComparator();
      PriorityQueue<String> pq=new PriorityQueue<String>(5,pqc);
      pq.add("Jason");
      pq.add("Ali");
      pq.add("Muhamad");
      for(String s:pq){
            System.out.println(s);
      }
   }
}

public class PriorityQueueComparator implements Comparator<String>{
         public int compare(String s1, String s2) {
      if (s1.length() < s2.length()) {
         return -1;
      }
      if (s1.length() > s2.length()) {
         return 1;
      }
      return 0;
   }
}
```

a) What is the purpose of the PriorityQueueComparator in the code?
b) What is the output for the code?

a) Purpose of the PQC is to have control over how we rank our elements in the priority queue, in this case the shortest string has the highest priority over longer ones.

b)

```
Ali
Jason
Muhamad
```