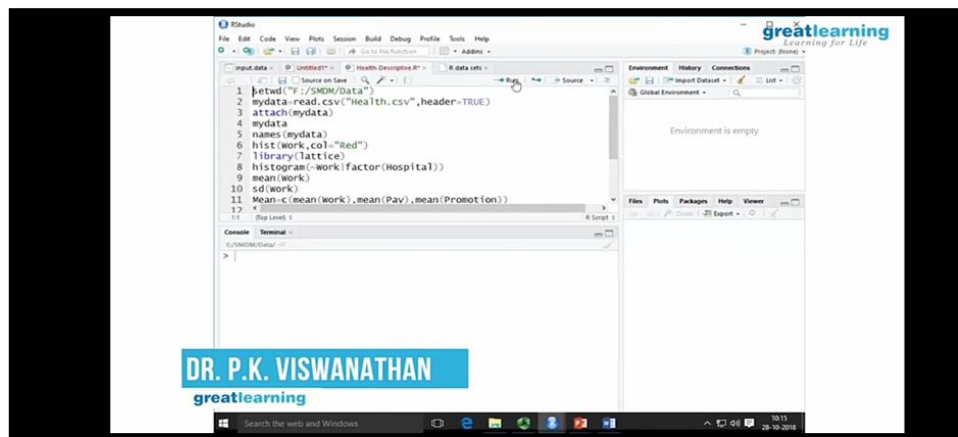
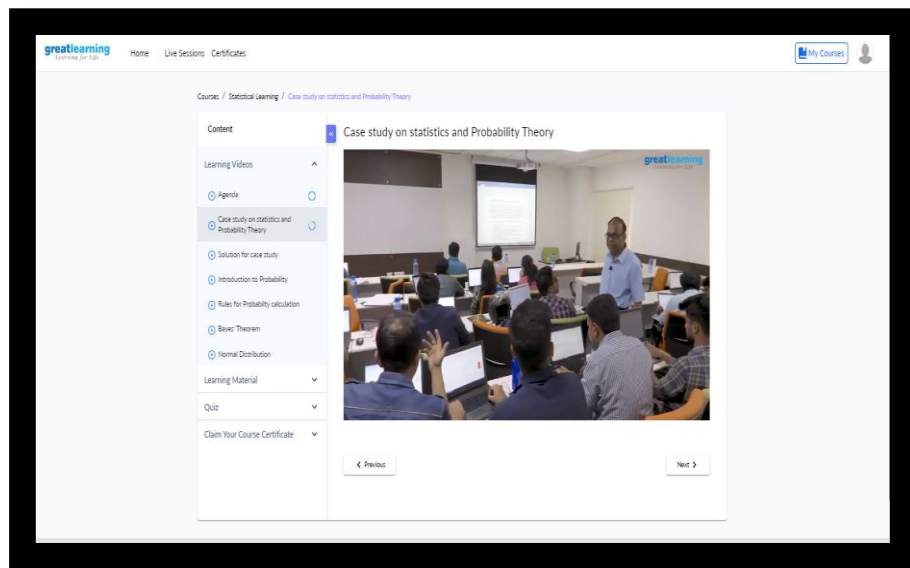


# DAILY ASSESSMENT

Date:	16-June-2020	Name:	Swastik R Gowda
Course:	Statistical Learning	USN:	4AL17EC091
Topic:	❖ Case Study on statistics & probability theory. ❖ Solution for case study	Semester & Section:	6 <sup>th</sup> Sem 'B' Sec
Github Repository:	swastik-gowda		

## FORENOON SESSION DETAILS

### Image of session



**Report – Report can be typed or hand written for up to two pages.**

### AGENDA:

- Case study for statistics
- Probability and its types
- Bayes theorem
- Normal distribution and bell curve

### Steps:

- Identify and define the research questions
- Select the cases
- Collect data
- Evaluate and analyze the data
- Result presentation

### Questions:

#### National Health Care Association

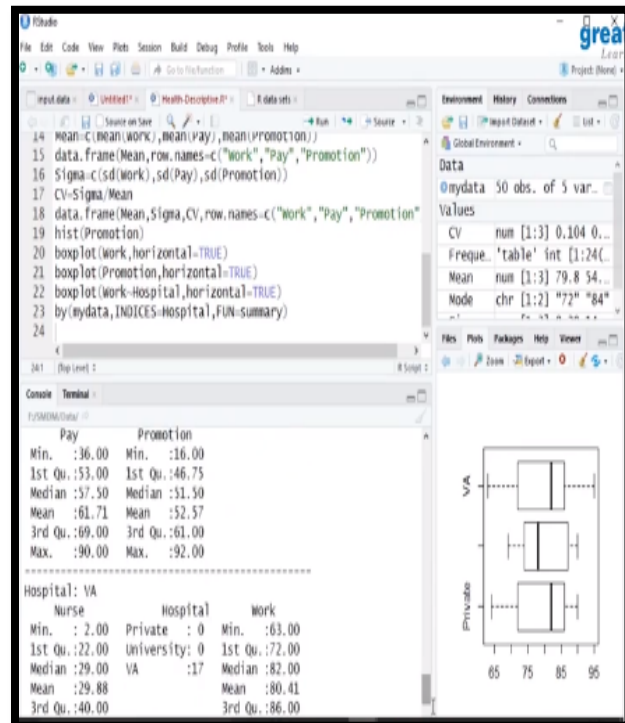
(Adapted from Anderson, Sweeney, and Williams for Classroom Discussion)

The National Health Care Association is concerned about the shortage of nurses the health care profession is projecting for the future. To learn the current degree of job satisfaction among nurses, the association has sponsored a study of hospital nurses throughout the country. As part of this study, a sample of 50 nurses was asked to indicate their degree of satisfaction in their work, their pay and their opportunities for promotion. Each of the three aspects of satisfaction was measured on a scale from 0 to 100, with larger values indicating higher degrees of satisfaction. The data collected also showed the type of hospital employing the nurses. The types of hospitals were private (P), Veterans Administration (VA) and University (U). The complete data set is on the file named "Health.csv".

How do you make insights or wisdom out of this data set? What are the insights?

- 1) What is the mode for work?
- 2) Which of the three attributes has the highest mean satisfactory score?/ lowest mean satisfaction score?
- 3) Find out the coefficient of variation for work, pay, and promotion
- 4) In the histogram for Promotion, which class has the highest concentration?
- 5) Is the shape of box plot for Work is skewed? If so, which direction?
- 6) How many points are outliers in Promotion Box Plot?
- 7) If All the Box plots for Work are drawn for all the hospitals, which hospital type has the best median value?
- 8) If box plots for Work, Pay, Promotion are drawn in the same space, how many outliers are there for promotion, and Pay?

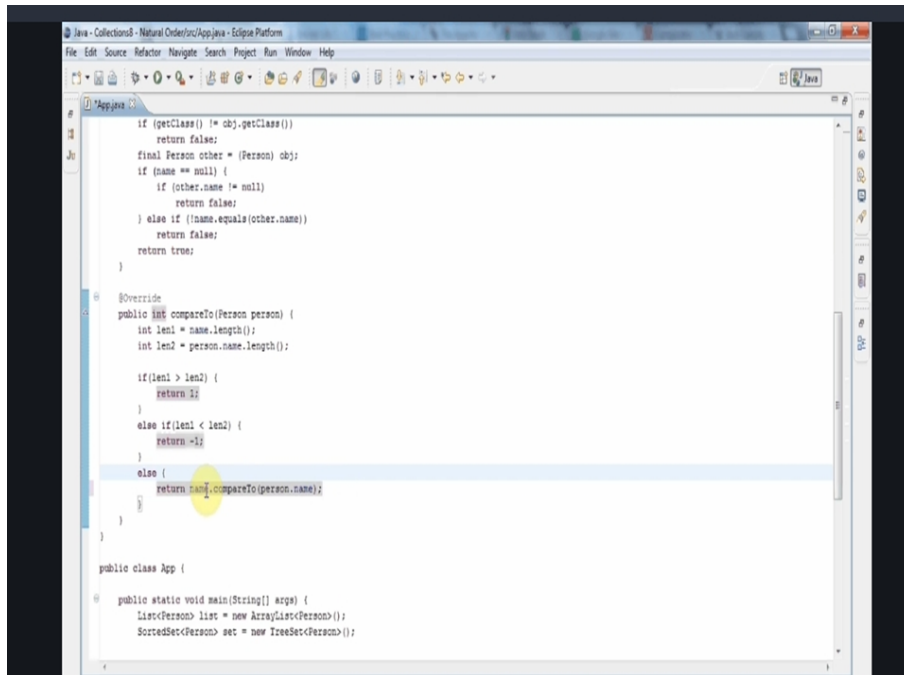
### Solutions of case study:



<b>Date:</b>	<b>16-June-2020</b>	<b>Name:</b>	<b>Swastik R Gowda</b>
<b>Course:</b>	<b>Java</b>	<b>USN:</b>	<b>4AL17EC091</b>
<b>Topic:</b>	<b>Programming</b>	<b>Semester &amp; Section:</b>	<b>6<sup>th</sup> Sem 'B' Sec</b>

### AFTERNOON SESSION DETAILS

#### Image of session



Java Tutorial for Complete Beginners
★ Leave a rating

doccradice.com/javaee/7/docs/api/java/util/concurrent/BlockingQueue.html

All Known Implementing Classes:

ArrayBlockingQueue, DelayQueue, LinkedBlockingQueue, LinkedTransferQueue, PriorityBlockingQueue, SynchronousQueue

```

public interface BlockingQueue<E>
    extends Queue<E>

```

A Queue that additionally supports operations that wait for the queue to become non-empty when retrieving an element, and wait for space to become available in the queue when storing an element.

BlockingQueue methods come in four forms, with different ways of handling operations that cannot be satisfied immediately, but may be satisfied at some point in the future: one throws an exception, the second returns a special value (either null or false, depending on the operation), the third blocks the current thread indefinitely until the operation can succeed, and the fourth blocks for only a given maximum time limit before giving up. These methods are summarized in the following table:

	Throws exception	Special value	Blocks	Times out
Insert	add(E)	add(E)	put(E)	offer(E, time, unit)
Remove	remove()	poll()	take()	poll(time, unit)
Examine	peek()	peek()	not applicable	not applicable

A BlockingQueue does not accept null elements. Implementations throw NullPointerException on attempts to add, put or offer a null. A null is used as a sentinel value to indicate failure of poll operations.

A BlockingQueue may be capacity bounded. At any given time it may have a remainingCapacity beyond which no additional elements can be put without blocking. A BlockingQueue without any intrinsic capacity constraints always reports a remaining capacity of Integer.MAX\_VALUE.

BlockingQueue implementations are designed to be used primarily for producer-consumer queues, but additionally support the Collection interface. So, for example, it is possible to remove an arbitrary element from a queue using remove(). However, such operations are in general not performed very efficiently, and are intended for only occasional use, such as when a queued message is cancelled.

BlockingQueue implementations are thread-safe. All queueing methods achieve their effects atomically using internal locks or other forms of concurrency control. However, the bulk Collection operations removeAll, retainAll and removeAll are not necessarily performed atomically unless specified otherwise in an implementation. So it is possible, for example, for removeAll() to fail (throwing an exception) after adding only some of the elements to a.

A BlockingQueue does not necessarily support any kind of "dead" or "shutdown" operation to indicate that no more items will be added. The needs and usage of such features tend to be implementation-dependent. For example, a common tactic is for producers to insert special end-of-stream or poison objects, that are interpreted accordingly when taken by consumers.

Usage example, based on a typical producer-consumer scenario. Note that a BlockingQueue can safely be used with multiple producers and multiple consumers.

```

class Producer implements Runnable {
    private final BlockingQueue queue;
    Producer(BlockingQueue q) { queue = q; }
    public void run() {
        try {
            while (true) { queue.put(produce()); }
        }
    }
}

```

Report – Report can be typed or hand written for up to two pages.

## The Java collections Framework:

- Natural Ordering
- Queues
- Using Iterators
- Implementing iteration
- Deciding Which Collection to Use
- Complex Data Structures

### Using Iterators:

```
public class App {  
    public static void main(String[] args) {  
        LinkedList<String> animals = new LinkedList<String>();  
        animals.add("fox");  
        animals.add("cat");  
        animals.add("dog");  
        animals.add("rabbit");  
  
        Iterator<String> it = animals.iterator();  
        while (it.hasNext()) {  
            String value = it.next();  
            System.out.println(value);  
            if(value.equals("cat")) {  
                it.remove();  
            }  
        }  
        System.out.println();  
        // Modern Iteration, Java 5 and later  
        for (String animal : animals) {  
            System.out.println(animal);  
        }  
        // animals.remove(2);  
    }  
}
```

### Complex data structures:

```
public static String[] vehicles = { "ambulance", "helicopter", "lifeboat" };  
  
public static String[][] drivers = {  
    { "Fred", "Sue", "Pete" },  
    { "Sue", "Richard", "Bob", "Fred" },  
    { "Pete", "Mary", "Bob" },  
};  
  
public static void main(String[] args) {  
    Map<String, Set<String>> personnel = new HashMap<String, Set<String>>();  
  
    for (int i = 0; i < vehicles.length; i++) {  
        String vehicle = vehicles[i];  
        String[] driversList = drivers[i];  
  
        Set<String> driverSet = new LinkedHashSet<String>();  
  
        for (String driver : driversList) {  
            driverSet.add(driver);  
        }  
  
        personnel.put(vehicle, driverSet);  
    }  
  
    { // Brackets just to scope driversList variable so can use again later  
        // Example usage  
        Set<String> driversList = personnel.get("helicopter");  
  
        for (String driver : driversList) {  
            System.out.println(driver);  
        }  
    }  
}
```

Date:	16-June-2020	Name:	Swastik R Gowda
Course:	Webinar on "Future Ahead For Electronic Engineers"	USN:	4AL17EC091
Topic:	An Overview of Avionics in Electronics Industry	Semester & Section:	6 <sup>th</sup> Sem 'B' Sec

#### AFTERNOON SESSION DETAILS

Image of session

**Who Are We**

**PCC PARK CONTROLS & COMMUNICATIONS (P) LTD.**  
Empowered vision. Embedded values.  
AS 9100, ISO 9001, CEMILAC CERTIFIED DEFENCE AVIONICS COMPANY

*A Defense Avionics Company for Design Services, Product Engineering, Production & Supply, System Integration & Maintenance*

radhakrishnan's screen

Report – Report can be typed or hand written for up to two pages.

