Lesson 7: Array Methods

1. The *Arrays* class, belonging to java, gives methods that are static so as to create as well as access Java arrays dynamically. Arrays have got only static methods as well as methods of Object class.

2. The *util package* belongs to the Java Collection Framework.

3. asList method- used to return the fixed-size list that mentioned Arrays back

```
int Arr [] = { 10, 30, 35, 52, 75 };

System.out.println("The Integer Array as a List = "
+ Arrays.asList(Arr));

Output: The Integer Array as a List = [[I@568db2f2]
```

4. static int binarySearch(itemToSearch)- search for a mentioned element in the array through the Binary Search algorithm

```
int Arr[] = { 10, 30, 35, 52, 75 };
Arrays.sort(Arr);
int ele = 35;

System.out.println (ele + " is found at index = "
+ Arrays.binarySearch(Arr, ele));

Output: 35 is found at index = 2
```

5. compareUnsigned(arr 1, arr 2) -This method of Integer class would compare two integer values treating them as unsigned and then returning zero in case x is equal to y.

int m = 10;
int n = 20;
// 10 < 20
System.out.println(Integer.compareUnsigned(m, n));
Output: -1
-1
	int x = 8;
	int y = 8;
	// 8 = 8
	System.out.println(Integer.compareUnsigned(x, y));
	Output: 0

int e = 25;
int f = 8;
//25 > 8
System.out.println(Integer.compareUnsigned(e, f));
 	
Output: 1
	int o = 15;
	int p = -7;
	// 15 > 7, -7 would be treated as an unsigned number
	// which will be greater than 15
	System.out.println(Integer.compareUnsigned(o, p));
	Output: -1

6. copyOf method (original array, new length)- copies the mentioned array, truncates it or pads it with a default value but only if necessary, so that copy has got the mentioned length.

import java.util.Arrays;
public class Example {
public static void main(String[] args)
{
 	
// Fetching Array
int Arr[] = { 10, 25, 55, 22, 35};
 	
// Printing the elements in a single line
System.out.println("The Integer Array is: "
+ Arrays.toString(Arr));
 	
System.out.println("\nThe new Arrays fetched by copyOf is :\n");
 	
System.out.println("Integer Array is: "
+ Arrays.toString(Arrays.copyOf(Arr, 10)));
}
}
	Output:
	The Integer Array is: [10, 25, 55, 22, 35]
	The new Arrays fetched by copyOf is:
	Integer Array is: [10, 25, 55, 22, 35, 0, 0, 0, 0, 0]

7. static boolean deepEquals (Object [] m1, Object [] m2): deepEquals method would return true in case the two mentioned arrays are deeply equal to the other array or not.

public class Array{
public static void main(String[] args)
{
// Fetching first Array
int Arr[][] = { {10, 20, 35, 82, 95} };
// Fetching second Array
int Arr2[][] = { { 10, 15, 22 } };
// Comparing both arrays
System.out.println("Arrays when compared: "+
Arrays.deepEquals(Arr, Arr2));
}
}
	Output:
	Arrays when compared: false