Maya G C

13/09/2023

Kodnest

ASSIGNMENT

1) Arrays:

The 'Arrays' class is a member of the 'java.util' package. This is a part of the Java Collections framework and provides methods to create, access and manipulate Java arrays dynamically.

All the methods provided by the Arrays class are static in nature and are methods of the 'Object' class. As the methods are static, they can be accessed using the class name itself.

The class hierarchy is as follows:

java. lang. Object

↳ java.util.Arrays

Syntax: Class declaration public class Arrays extends Object

Syntax: In order to use Arrays Arrays.;

1) asList

Prototype: static <T> List<T> asList (Object[] a)
Parameters: a – array of objects from which the list will be backed.
Return Value: List<T> => fixed-size list of specified array

Description: Returns a fixed-size serializable list backed by an array provided as an argument.

Example:

```
import java.util.Arrays;

import java.util.List;
```

```
public class Main {

   public static void main(String[] args) {

      String[] months = {"January", "February", "March", "April", "May"};

      System.out.println("The string array converted to list:");

      List&lt;String&gt; month_list = Arrays.asList(months);

      System.out.println(month_list);

   }

}
```

2) Binary Search

Prototype: static int binarySearch (int[] a, int key)

Parameters:

a => array in which the key is to be searched
Key=> element value to be searched

Return Value: int=>position (index) at which key is found, else returns (-(the “insertion point”) – 1).

Description: Searches for the specified key in the given array using a binary search algorithm. The array needs to be sorted for the binary search to work. If the array is not sorted then the results are undefined. Also, if there are multiple locations in the array for the same key value, the position returned is not guaranteed.

Example:

```
import java.util.Arrays;

import java.util.List;

public class Main {

   public static void main(String[] args)
```

```
    {
       int numArr[] = { 23,43,26,65,35,16,74,27,98 };
       Arrays.sort(numArr);
       System.out.println("Input array:" + Arrays.toString(numArr
      int key =i
     System.out.println("Key " + key + " found at index = " + Arrays
                              .binarySearch(numArr, key)
}
}
```

3) CopyOf

Prototype: static int[] copyOf(int[] original, int newLength)

Parameters:

original=> array to be copied
newLength=> length of the copied array

Return Value: A new array copied from the original and padded or truncated with zeros depending on a specified length.

Description: Copies the array original into a new array and pads or truncates it with zeros depending on the length specified.

Example:

```
import java.util.Arrays;
public class Main {
   public static void main(String[] args)
   {
String strArr[] = {"Java", "Python", "Perl", "C", "Ruby"};
System.out.println("Original String Array: “ + Arrays.toString(strArr));
```

```
System.out.println("Copied Array:" + Arrays.toString(Arrays.copyOf(strArr, 5)));
  }
}
```

4) copyOfRange

Prototype: static int[] copyOfRange(int[] original, int from, int to)

Parameters:

original => array from which values in the range are to be copied
From=> first index of the range
To=> last index of the range

Return Value: New array with values from the specified range with zeros truncated or padded to obtain the desired length.

Description: Copies the range specified from a given array into a new array. The started index of the array should be inclusive between 0 to original.length. The end index can be exclusive.

Example:

```
import java.util.Arrays;
public class Main {
public static void main(String[] args)
  {
 String strArr[] = {"Java", "Python", "Perl", "C", "Ruby"};
 System.out.println("Original String Array: "  + Arrays.toString(strArr));
 System.out.println("Copied Range of Array: " + Arrays.toString Arrays.copyOfRange(strArr,1,3)));
  }
}
```

5). DeepEquals(Object[] a1, Object[] a2)

Method o Returns true if the two specified arrays are deeply equal to one another.

Example:

```
import java.util.Arrays;
public class Main {
 public static void main(String[] args) {
int intArr[][] = { { 10, 20, 15, 22, 35 } };
int intArr1[][] = { { 10, 15, 22 } };
System.out.println("Integer Arrays on comparison: " + Arrays.deepEquals(intArr, intArr1));
 }
 }
```

6). compare(array 1, array 2)

Method o Compares two arrays passed as parameters lexicographically.

Example:

```
import java.util.Arrays;
public class Main {
 public static void main(String[] args) {
 int intArr[] = { 10, 20, 15, 22, 35 };
 int intArr1[] = { 10, 15, 22 };
 System.out.println("Integer Arrays on comparison: " + Arrays.compare(intArr, intArr1));
}
}
```

7). binarySearch(array, fromIndex, toIndex, key, Comparator)

Method o This method searches a range of the specified array for the specified object using the binary search algorithm.

Example:

```
 import java.util.Arrays;

public class Main {

public static void main(String[] args) {

 int intArr[] = { 10, 20, 15, 22, 35 };

Arrays.sort(intArr);

int intKey = 22;

 System.out.println(intKey " found at index = + Arrays .binarySearch(intArr, 1, 3, intKey));

}

}
```

3)Use of Array

→The class Arrays belongs to java.util package has numerous static methods that are useful in filling, sorting, searching, and many other things in arrays.