Collections class frequency() method

In this post we will see the frequency () method defined in Collections class. Previously we had studied methods like [swap(..)](http://data-structure-learning.blogspot.com/2015/06/collections-class-swap-method.html), [synchronized](http://data-structure-learning.blogspot.com/2015/06/synchronized-collections-by-collections.html), [binarySearch(..)](http://data-structure-learning.blogspot.com/2015/06/collections-class-binarysearch-method.html), [copy(..)](http://data-structure-learning.blogspot.com/2015/06/collections-class-copy-method.html), [fill(..)](http://data-structure-learning.blogspot.com/2015/06/collections-class-fill-method.html) , [addAll(..)](http://data-structure-learning.blogspot.com/2015/05/collections-class-addall-method.html), [min() with Comparable](http://data-structure-learning.blogspot.com/2015/06/collections-min-method-with-comparable.html), [min() with Comparator](http://data-structure-learning.blogspot.com/2015/06/collections-min-method-with-comparator.html), [max() with Comparable](http://data-structure-learning.blogspot.com/2015/06/collections-max-method-with-comparable.html), [max() with Comparator](http://data-structure-learning.blogspot.com/2015/06/collection-max-method-with-comparator.html) methods.

I have also written about the [Comparable](http://data-structure-learning.blogspot.com/2015/06/comparable-interface-for-sorting.html) and [Comparator](http://data-structure-learning.blogspot.com/2015/06/comparator-interface.html) Interfaces and [differences between them](http://data-structure-learning.blogspot.com/2015/06/difference-between-comparable-and.html). I would highly recommend that you read those posts.

frequency() method is used to count the specified Object in Collection. As the frequency() method accepts parameter as Collection, so we can pass any type of Collection as parameter to frequency() method.

Below is the code of frequency() method in Collections class. Comments are inserted by me for understanding the method.

/\*\*

\* Returns number of times the Object o occurs in specified Collection.

\* The equality depends on Object passed as parameter.

\* More specifically if o==null ? e==null : o.equals(e)

\* \*/

**public** **static** **int** frequency(Collection<?> c, Object o) {

//Initialize result to 0.

**int** result = 0;

/\*\*

\* If specified parameter is null check for null elements

\* \*/

**if** (o == **null**) {

**for** (Object e : c)

**if** (e == **null**)

result++;

}

/\*\*

\* If element is not null then check comparison based on equals

\* more specifically o.equals(e)

\* \*/

**else** {

**for** (Object e : c)

**if** (o.equals(e))

result++;

}

**return** result;

}

Let us built a List of days.

List<String> names=**new** ArrayList<String>();

names.add("Monday");

names.add("Tuesday");

names.add("Wednesday");

names.add("Monday");

names.add("Tuesday");

names.add("Monday");

**int** result=*count*(names, "Monday");

System.***out***.println(result); //Outputs 3

Now, let us write count method with two arguments as List and day.

**public** **static** **int** count(**final** List<String> days, **final** String day){

**return** Collections.*frequency*(days, day);

}

That’s all on frequency() method of Collections class.