COLLECTIONS

[ArrayList](#_e72zi3v46er3)

[LinkedList](#_d21h9jpr02y0)

[HashSet](#_nouahjoeu72)

[LinkedHashSet](#_wpf0zeugq11n)

[HashMap](#_qwduyy9idn5s)

[TreeMap](#_alv59lhnbcf)

[GENERICS IN JAVA](#_7aeif1lgl7p)

[Advantage of Java Generics](#_cu8y5vwqtot0)

[PROGRAM 1:](#_m0885lae60se)

[SOLVED PROGRAM 1:](#_q4hkn7qhrm4)

[PROGRAM 2:](#_lj33o6ht2lsj)

[SOLVED PROGRAM 2:](#_f38rrcvg2w35)

The **Collection in Java** is a framework that provides an architecture to store and manipulate the group of objects.

Java Collections can achieve all the operations that you perform on a data such as searching, sorting, insertion, manipulation, and deletion.

## ArrayList

The ArrayList class implements the List interface. It uses a dynamic array to store the duplicate element of different data types. The ArrayList class maintains the insertion order and is non-synchronized. The elements stored in the ArrayList class can be randomly accessed.

## LinkedList

LinkedList implements the Collection interface. It uses a doubly linked list internally to store the elements. It can store the duplicate elements. It maintains the insertion order and is not synchronized.

## HashSet

HashSet class implements Set Interface. It represents the collection that uses a hash table for storage. Hashing is used to store the elements in the HashSet.It contains unique items.

## LinkedHashSet

LinkedHashSet class represents the LinkedList implementation of Set Interface. It extends the HashSet class and implements Set interface. Like HashSet, It also contains unique elements. It maintains the insertion order and permits null elements.

## HashMap

In HashMap, we have a key and a value pair<Key, Value>.

## TreeMap

TreeMap can be a bit handy when we only need to store unique elements in a sorted order.

# GENERICS IN JAVA

The **Java Generics** programming is introduced in J2SE 5 to deal with type-safe objects.

Before generics, we can store any type of objects in collection i.e. non-generic. Now generics, forces the java programmer to store specific type of objects.

## Advantage of Java Generics

There are mainly 3 advantages of generics. They are as follows:

**1) Type-safety :** We can hold only a single type of objects in generics. It doesn’t allow to store other objects.

**2) Type casting is not required:** There is no need to typecast the object.

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

This says that the arraylist can accept only string.

# PROGRAM 1:

Given an array of dates. Sort them in ascending and descending order based on the

user inputs.(Take the format as DD-MM-YYYY)

## SOLVED PROGRAM 1:

[https://github.com/sruthiviswanathan/Zterns-19-Sruthi/blob/master/Collections/src/com/zilker/collections/SortingDates.java](https://github.com/sruthiviswanathan/Collections/blob/master/src/com/zilker/collections/SortingDates.java)

* Get the no.of inputs from the user.
* Then get the dates from the user.
* Override the comparator for sorting the elements in ascending and descending order.
* For sorting the dates use DateFormat object that takes the input as a date.
* For ascending order :
* dateFormat.parse(arg0).compareTo(dateFormat.parse(arg1));
  + This will arrange the dates in ascending order…(i.e)firstarg.compareTo(secondarg)
* For descending order:
* dateFormat.parse(arg1).compareTo(dateFormat.parse(arg0));
  + This will arrange the dates in ascending order…(i.e)secondarg.compareTo(firstarg)
* Return the results.

# PROGRAM 2:

Given an array of names standing for a student council election. A name in array

represents a vote casted to the candidate. Print the name of student who won. If there is

tie, print lexicographically smaller name.

Input : Input : votes[] = {john, johnny, jackie,johnny,john,jackie,jamie, jamie, john,johnny,jamie&quot,johnny,john};

Output : John

## SOLVED PROGRAM 2:

https://github.com/sruthiviswanathan/Zterns-19-Sruthi/blob/master/Collections/src/com/zilker/collections/CountingVotes.java

* For counting the votes casted to each person use Treemap which stores the entries in key value pairs.
* Have the names as key and the number of votes as value.
* Have a separate logic for counting the number of votes for each person and store it in treemap.
* The first element of treemap will have the key and value of the entry that has maximum values.
* Now iterate through the treemap to see if any other key in the treemap has same key value,if yes store that key in treeset(treeset will have unique elements and sorts them by lexicographical order)
* Thus the final output would be the first element of treeset.