**Collection Framework – Part\_01**

* **Introduction:**

An array is indexed collection of fixed number of homogeneous data elements.

The main advantage of arrays is, we can represent multiple values by using single variable. So that readability of the code will be improved.

Limitations of Arrays:

1. Arrays are fixed in size. That is, once we create an array there is no chance of increasing or decreasing the size based on our requirement. Due to this, to use arrays concept compulsory we should know the size in advance. Which may not possible always.
2. Array can hold only homogeneous data type elements. Example:

Student[] s =new Student[10000];

S[0] = new Student(); // valid

S[1] = new Customer(); // Invalid

CE: incompatible types

found: Customer

required: Student

We can solve this problem by using Object type arrays.

Object[] a = new Object[1000];

a[0] = new Student(); // Valid

a[1] = new Customer(); // Valid

1. Arrays concept is not implemented based on some standard data structure and hence readymade method support is not available. For every requirement we have to write the code explicitly which increases complexity of programming.

To overcome above problems of arrays, we should go for Collections concept.

Collections are growable in nature that is, based on our requirement, we can increase or decrease the size.

Collections can hold both homogeneous and heterogeneous elements.

Every collection class is implemented based on some standard data structure, hence for every requirement readymade method support is available. Being a programmer, we are responsible to use those methods and we are not responsible to implement those methods.

* **Differences between Arrays & Collections:**

|  |  |  |
| --- | --- | --- |
| S.No | Arrays | Collections |
| 1 | Arrays are fixed in size, that is once we an array we can’t increase or decrease the size based on our requirement. | Collections are growable in nature. That is, based on our requirement we can increase or decrease the size. |
| 2 | With respect to memory arrays are not recommended to use. | With respect to memory collections are recommended to use. |
| 3 | With respect to performance arrays are recommended to use. | With respect to performance Collections are not recommended to use. |
| 4 | Arrays can hold only homogeneous data type elements. | Collections can hold both homogeneous and heterogeneous elements. |
| 5 | There is no underlying data structure for arrays and hence readymade method support is not available. For every requirement we have to write the code explicitly which increases complexity of programming. | Every collection class is implemented based on some standard data structure and hence for requirement readymade method support is available. Being a programmer, we can use these methods directly and we are not responsible to implement those methods. |
| 6 | Array can hold both primitives and objects. | Collections can hold only object type not primitives. |

* **Collection:**

If we want to represent a group of individual objects as a single entity then we should go for collection.

* **Collection Framework:**

It contains several classes and interfaces which can be used to represent a group of individual objects as a single entity.

|  |  |
| --- | --- |
| Java | C++ |
| Collection | Container |
| Collection Framework | STL (Standard Template Library) |