**Java Collections Framework:**

**Student1, student2, stu3, stu4 – Student is collection**

**Collection is group of objects. Here student is a collection which has stu1, stu2 etc objects**

**We can call collection as a single entity which can be used to refer multiple objects.**

**Collections framework is nothing but a group of collection.**

**Collections is nothing but a group of classes and interfaces by which we can represent group of objects into a single entity**

**Why we need this collection?**

**int a =10;**

**int b=20;**

**int c=30;**

**Suppose we want to store 100 values, in that case creating 100 variables and storing its value is not the efficient way. This can be efficiently done using array concept**

**int stu[]= new int[100]; 🡪 we can store only integer values**

**Object stu[] = new Object[100]; 🡪 we can store values of multiple data types but the memory is fixed**

**Disadvantages of using an array:**

1. **In an array memory is fixed, we cant extend the memory at runtime**
2. **Array is homogenous which means that I can store values of a single data type. In a integer array, we can store only integer value and cannot store float values**
3. **In an array, there are no predefined or readymade methods available in array.**

**To overcome the above challenges, we are going to use collection framework**

1. **We can extend the memory size at runtime using collection**
2. **we can store values of multiple data types in a single collection**
3. **In collection, we can many predefined methods using which we can manipalute values at a faster rate**

**Collection (Interface)**

Queue

(Interface)

Set (Interface)

List (Interface)

Classes implementing the List Interface are ArrayList, LinkedList, Vector which extends Stack interface. Vector is legacy class which is available in the older versions of Java

Classes implementing Set are HashSet, LinkedHashset, Tree set

Classes implementing Queue are PriorityQueue

|  |  |
| --- | --- |
| **List(Interface)** | **Set (Interface)** |
| List allows insertion of dupplicate values | Set accepts only unique values |
| List maintains insertion order | Set doesn’t main order |

**Queue** follows the FIFO (first in first out) order.

**Map** **Interface**:

Map interface is not a child interface of Collection interface and it a independent interface in Java.

Map follows Key and value pair structure. For each value, there would be key associated with it.

Keys should be unique but values can be duplicate

|  |  |
| --- | --- |
| Key | Value |
| 101 | Java |
| 102 | C |
| 102 | Python |