Collections Framework

1. Collections

DEFINITION/WHAT

Collection framework provides an architecture to store and manipulate the group of objects.

NEED/WHY

- ✓ Need 1:Improved efficiency of the code
- ✓ Need 2: Easy to code
- ✓ Need 3: Allows interoperability between different API's

REAL TIME EXAMPLE

A classroom is a collection of students. Therefore, the classroom is nothing but a collection and students are objects.

ADDITIONAL INFORMATION

1. Refer to Collections hiearchy.All classes and interfaces are defined in java.util package

Limitations of the collections framework in Java.

1. Care must be taken to use the appropriate cast operation.
2. Compile time type checking is not possible.

2. ArrayList

DEFINITION/WHAT

Implements the List interface. **ArrayList** class uses a *dynamic array* for storing the elements. It is like an array, but there is *no size limit*.

ArrayList can have the duplicate elements(objects).

NEED/WHY

- ✓ Need 1:Provides Ordered Collections
- ✓ Need 2:Provides many built-in methods to store,iterate,remove,search for an element in collections

REAL TIME EXAMPLE

Eg:Amazon website has many customer's records in its database. If they want to retrieve the customers from the database and display it to the admin.

ArrayList can be used to add customer records from ResultSet.

Alternatives/Differences between Array and ArrayList

Array	ArrayList
Array is static in size.	ArrayList is dynamic in size.
It performs fast in comparison to ArrayList because of fixed size.	ArrayList is internally backed by the array in Java. The resize operation in ArrayList slows down the performance.
An array can store both objects and primitives type.	We cannot store primitive type in ArrayList. It automatically converts primitive type to object.
We cannot use generics along with array because it is not a convertible type of array.	ArrayList allows us to store only generic/ type, that's why it is type-safe.
Array provides a length variable which denotes the length of an array.	ArrayList provides the size() method to determine the size of ArrayList.

ADDITIONAL INFORMATION

We can sort objects of arrayList using Comparable/Comparator interface

3. HashSet

DEFINITION/WHAT

Implements Set interface. HashSet stores the elements by using a mechanism called **hashing.**

It contains unique elements only and allows null value.

It is non-synchronized.

HashSet does not maintain the insertion order.

Elements are inserted based on their hash code.

NEED/WHY

- ✓ Need 1:Best Collection to be used for Search operations
- ✓ **HashSet** is commonly **used** if we have to access elements randomly. It is because elements in a hash table are accessed using hash codes.

REAL TIME EXAMPLE

A College university storing student details, so that no two student id's are same

ADDITIONAL INFORMATION

Always override hashCode() and equals() methods when using HashSet

4. HashMap

DEFINITION/WHAT

Implements the Map interface, which allows us *to store key and value pair*.

Keys should be unique.

It is not synchronized. It allows us to store the null values as well, but there should be only one null key.

NEED/WHY

- ✓ Need 1: easy to perform operations using the key index like updation, deletion, etc.
- ✓ Need 2: We cannot store duplicate keys in HashMap. However, if we try to store duplicate key with another value, it will replace the value

IMPLEMENTATION/HOW IT WORKS

1. Find the hash code for the key.
2. Store the key and value objects in a bucket. Bucket is a Linked List implementation.
3. Repeat the above for each of the key value pairs

REAL TIME EXAMPLE

Storing Customer details like phone number and their name in telephone directory form.

ADDITIONAL INFORMATION

Refer to the differences between HashMap and HashTable.

Hashmap internally using Hashing. Hashing is the process of converting an object into an integer value. The integer value helps in indexing and faster searches.

5. Generics

DEFINITION/WHAT

Generics force the java programmer to store a specific type of objects in a collection

NEED/WHY

- ✓ Need 1: Type Safety- We can hold only a single type of objects in generics.
- ✓ Need 2:Type casting is not required
- ✓ Need 3: Compile-Time TypeChecking

IMPLEMENTATION/HOW IT WORKS

use <> to specify parameter types in generic class creation. To create objects of a generic class, we use the following syntax.

// To create an instance of generic class

BaseType <Type> obj = new BaseType <Type>()

REAL TIME EXAMPLE

You have to use arraylist to save the marks of a student scored in all subjects.

Programmer by mistake adds string “Fail” if the mark is less than 70.

The below mentioned code will prevent this issue

```
ArrayList <Intger> markList = new ArrayList< Intger > ();
```

6.Iterator Interface

Java Iterator is an interface that is practiced in order to iterate over a collection of Java object components entirety one by one.

NEED/WHY

- ✓ Need 1: Java Iterator, can be used for both read and remove operations.
- ✓ Java Iterator only preserves the iteration in the forward direction,while ListIterator preserves iteration in both direction

IMPLEMENTATION/HOW IT WORKS

Make an instance of the Iterator interface from the collection of objects they desire to traverse over.The following methods are provided in Iterator interface

1)hasNext() 2)next() 3)remove()

REAL TIME EXAMPLE

Can be used to print the each Customer details stored in the list which is retrieved from Database

ADDITIONAL INFORMATION

Enhanced Forloop can be used as an alternate to iterator to iterate in Collection