**FACILITATOR’S MANUAL**

|  |
| --- |
| Facilitator’s manual is a guideline to facilitator. Guideline for which all topics /sub-topics to be covered and their sequence. When to go recap or hands-on and with which assignment (mapping of lab assignments with topics)  Basically WHAT–WHEN-HOW  Here, Whole session will be in multiple iteration of 3 steps;  1. What to facilitate, 2. Relevant LAB assignments, 3. Recap and leanings from LAB  Also, there are TIPS (extract from facilitator’s learning) – objective of TIPS is to incorporate best practice and individual’s innovation in facilitating a particular topic. It is desirable that new tips should continue to add/update in this manual.  At last, this is not a rulebook, so it is upto facilitator to follow it or use his/her own style |

**COLLECTION & GENERICS**

**Objective -**  To understand collection API and its interfaces and classes, how and when to use them? To know Shorting, searching and shuffling algorithm; use of iterator, Collections class, etc.

**ROUND 1**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * Collection API * Collection framework, Collection hierarchy (interfaces of Collection) * List (array list) |
| LAB assignment | **LAB 9.1\***  *Create a list ref with ArrayList; add some values into it and print it; see the behavior*  **LAB 9.2**  *Create a list ref with ArrayList; use its various methods (use at least 10 methods, except iterator). Aim of this assignment is to get aware with various method of List and how and when to use them.* |
| Recap (learning from the LAB assignment) | Knowledge of ArrayList and its various properties / behavior |

\*TIPS TIME – Use some List’s methods while teaching; it will give a head start to participant on how to understand method signature from API doc and practice it.

#TIPS TIME – For initial session, ask participant to ignore Generics and syntax while declaring/using List, Set, Iterator, etc; as it will be cover in later half of the session.

**ROUND 2**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * Vector * Linked List |
| LAB assignment | **LAB 9.3**  *Do the similar task (as Lab 9.1) for Linked list – specially use the methods which are available in LinkedList but not in ArrayList.* |
| Recap (learning from the LAB assignment) | Difference between ArrayList , Vector and LinkedList. Understanding of when to use which one. |

**ROUND 3**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * Set * HashSet and other implementing classes of Set * LinkedhashSet |
| LAB assignment | **LAB 9.4**  *Do the similar task (as Lab 9.1) for HashSet and other implementing classes of Set.*  **LAB 9.4 A**  *Create a LinkedHasSet and add 4 different intereger values (say 10, 14, 12 and 8) and print this LinkedHasSet*  *Now change the LinkedHasSet into normal HashSet and execute the program again and compare the output in both the scenerio* |
| Recap (learning from the LAB assignment) | Difference between List and Set  How and when to use HashSet  Difference between HashSet and LinkedHashSet  LinkedHashSet is use to get insertion ordered set |

**ROUND 4**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * SortedSet * TreeSet * Comparable interface |
| LAB assignment | **LAB 9.5**  *Do the similar task (as Lab 9.1) for TreeSet; try to analyze difference between HashSet and TreeSet*  **LAB 9.5 A**  *Create a SortedSet, create two objects of StringBuffer and add them in sorted set; print size of sorted set*  **LAB 9.5 B**  *Create a class (Say AnyClass) and in main class (Having p.s.v.main method) create a sorted set and then create two objects of AnyClass and add them in sorted set, print size of set*  *Case1-*  *Now, implement comparable in AnyClass, and execute the main class again*  *Provide any implementation to compareTo method (preferable return 0)*  *Case -2*  *Now, create a subclass (say AnySubClass) which extends AnyClass; now in main method, create one object of AnyClass and one object of AnySubClass, add them to sorted set and execute the program.*  *Case3-*  Now change the compareTo of AnyClass, and instead of returning 0, return 1 and then return -1; print the sorted set of main method in all three scenario. |
| Recap (learning from the LAB assignment) | What is Sorted Set and Tree Set  Importance of comparable in sorted set  Only comparable object of same type can be added in a sorted set |

**ROUND 5 (need to add it in PPT)**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * Iterator |
| LAB assignment | **LAB 9.6**  *Create a List or Set; add multiple values into it; print it with the help of Iterator.* |
| Recap (learning from the LAB assignment) | What is Iterator and how to use it |

**ROUND 6 (need to add it in PPT)**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * Enumerator * For loop for collection - for(items : collection) |
| LAB assignment | **LAB 9.7**  *Create a List or Set; add multiple values into it; print it with the help of enumerator and for loop (collection).*  **LAB 9.8**  *Create a List or Set; add multiple String values (say names) into it; Now with the help of iterator, iterate it and append another String (say Capgemini) as surname with all the names.*  *Try to do the same with the help of enumerator and for loop.* |
| Recap (learning from the LAB assignment) | Difference between iterator, for loop and enumerator.  Iterator iterate objects and may use to manipulate it; while enumerator iterate elements and for loop value of object, and they can be use only to print the value (cannot modify object) |

**ROUND 7**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * Map * HashMap and other implementing classes of Map   (HashTable, LinkedMap, etc.) and difference between them   * Sorted Map |
| LAB assignment | **LAB 9.9**  *Do the similar task (as Lab 9.1) for HashMap and other implementing classes of Map.* |
| Recap (learning from the LAB assignment) | Understanding of Map, HashMap other implementing class of and Map; and how and when to use them. Their behavior. |

**ROUND 8 (need to add it in PPT)**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * Generics * Its benefit |
| LAB assignment | **LAB 9.10**  *Create an ArrayList (without using generics) and add many values (say one integer value and all other Strings). Now in iterator cast the object value as String to print values of this ArrayList. Run the code.*  “You will find a runtime exception.”  *Now convert the same ArrayList as Generic one (Define Strings as generics values); try to do same activity as earlier.*  “You will find that system is not allowing it”  *Proceed with all String values and use iterator to iterate the list and print the values.*  “You will see no explicit casting is required” |
| Recap (learning from the LAB assignment) | How to use generics and why  Benefit of generics as it convert runtime error into compile time error; also coding effort is less (do not need to cast every time) |

**ROUND 9 (need to add it in PPT)**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * Wildcard |
| LAB assignment | **LAB 9.11**  *Exercise on wildcard* |
| Recap (learning from the LAB assignment) | What is wildcard in generics, it ‘s benefit |

**ROUND 10 (need to add it in PPT)**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * Overriding equals() and hashCode() method and its purpose and rules * Overriding toString() method |
| LAB assignment | **LAB 9.12**  *Write a program; where end user will fill a user registration form. Create a employee POJO class (with all required properties like –email id, name, date of birth, etc – here email id must be unique); use scanner API for user interaction.*  *Now user may add/create (register) many employee; employee data must be store in a collection class.*  *System must not allow duplicate entry; to achieve this override equals method.*  *Override toString method to display customized value of employee.*  **LAB 9.12 A**  *Write a class (say MyName ) having one global variable String name and one parameterized constructor which set value of name. Now in main class (in main method) create a Set (HashSet) and then create two objects of MyClass with same value for name field.*  *Case 1- Now check the size of set*  Case 2- Now Override hashCode and equals method of MyName for name field and execute main class again  Case 3 – Remove hashCode and only equal method and execute it; then remove equals and keep only hasCode method and check the output |
| Recap (learning from the LAB assignment) | How and why to override equals and hashCode method  Rules for equals and hashCode  How and why to override toString method  Use of collection objects to store data (like a runtime database) |

**ROUND 11**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * Collections class * Abstract classes of Collection framework   Algorithms in collection   * Sorting * Shuffling * Searching * Swapping |
| LAB assignment | **LAB 9.13**  *Write a class, create a List and use various method of CollectionsClass* |
| Recap (learning from the LAB assignment) | Collections class and its usage |

**ROUND 12**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * Variable Arguments   How to decide/pass number of arguments dynamically (Object…obj) |
| LAB assignment | **LAB 9.14**  *Write a class, which have a method which adds numbers passed in its argument. Then use it for different number of arguments* |
| Recap (learning from the LAB assignment) | How to use variable argument |