**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 up to facilitator to follow it or use his/her own style |

**Java Util package (important classes of Util package, apart from Collection API)**

**Objective -**  To understand classes like Scanner, Date, Calendar, RegEx and String tokenize. Working knowledge (when and how) of these classes.

**ROUND 1**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * What is scanner object, how to use it |
| LAB assignment | **LAB 6.1**  *Write a program, where system will take input values from console, system will ask for user’s birth place, user’s full name and age (one by one) and receive the user inputs using various scanner methods like next(), nextLine() and nextInt().*  *This program claims to tell the real age of user; it counts the length (number of character) in user’s birth place (say x) and adds this to user’s age and will display it as a message below -*  *“Hi USERNAME, you are from USER’S BIRTH PLACE and your actual age is (USER’S AGE + X)….. Ha ha ha”*  **LAB 6.2**  *Write a program where user will enter many numerical values from consol (Using Scanner), and output will be sorted list of these value (ASC and DESC)*  **LAB 6.3**  *Create a calculator, where user will select the arithmetic operation, then user will ask for values on which the operation is required to perform. Now user will provide number and operation will be performed.*  *Write this program with the help of scanner and switch; also for mathematic operation use the previously written MyMath class.* |
| Recap (learning from the LAB assignment) | How to use Scanner  Some important method of scanner  Revision of some core concepts of java |

**ROUND 2**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | **StringTokenizer and its methods**   * StringTokenizer overview and its uses * Various method of StringTokenizer |
| LAB assignment | **LAB 6.4**  *Rewrite Lab 6.2 – this time user will enter many number at one go (separated by either space or comma or any special character), and output will be sorted list. Do it with the help of StringTokenizer.*  **LAB 6.5**  *Modify Lab 6.1; this time – system will validate the data entered by user (Name, age and birth place). Basically now system will validate the data, and will not allow invalid info. (NAME and BIRTH PLACE must be text and AGE must be numbers only).*  *Do this using various methods like charAt, split, etc of string and isLetter, isDigit, etc. of Stringclass.*  *Aim of this LAB is to understand – how to validate various value entered from user and use of various methods of string and other wrapper class*  **LAB 6.6**  *Create calculator, where user will directly enter numbers and arithmetic operations (of any combination) and system will produce correct result. Here the system should handle any unwanted value entered by user (by validation)* |
| Recap (learning from the LAB assignment) | How and when to use StringTokenizer  Basic way to validate a form (without RegEx) |

**DATE –**

**ROUND 3**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | **Date**   * Date class \* (java.util.Date) * Constructors of date class |
| LAB assignment | **LAB 6.7**  *Create 2 date objects using two constructors of date (use one for each object), and run its methods (after, before, compare, getTime, setTime).* |
| Recap (learning from the LAB assignment) | How to create date object and its various methods |

*\*****TIPS TIME*** *– tell about java.sql.Date object as well, and difference between them. As it is very common mistake (in fresher) to instantiate sql date object instead of util date.*

**ROUND 4**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | **Date**   * Simple Date format * Date parsing * SQL date * Conversion from SQL date to UTIL date and vice-versa |
| LAB assignment | **LAB 6.8**  *Create a date object, print it on console; now create a SDF object, and format the date object, print it. Check the display of date is changed as format. Do the same for few more formats.*  **LAB 6.9**  *Create a SDF object, and parse and String (of given format) into date object.*  **LAB 6.10**  *Practice Java UTIL date and Java SQL date – create both UTIL date and SQL date and print/display it on console, check what is difference between them. Create a SQL date from UTIL date and vice versa.* |
| Recap (learning from the LAB assignment) | How and when to use SDF  How to apply and use various format  How to Parse String into date |

**CALENDAR –**

**ROUND 5**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | **Calendar**   * Calendar class * Various methods of calendar class |
| LAB assignment | **LAB 6.11**  *Create a calendar object using Calendar.getInstance(), and practice/execute at least 15 methods of calendar class (Ref to calendar API on oracle website )* |
| Recap (learning from the LAB assignment) | How to instantiate a calendar object and Various methods of calendar class  How to set values and get values |

**ROUND 6**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | **Calendar**   * Gregorian Calendar class * Various methods of Gregorian calendar class |
| LAB assignment | **LAB 6.11**  *Create a Gregorian calendar object and practice/execute at least 15 methods of calendar class (Ref to Gregorian calendar API on oracle website )*  **LAB 6.12**  *Print the days of weeks, using “DateFormatSymbols” class.*  **LAB 6.13**  *Write a program where system will ask for user’s date of birth (take input with the help of scanner) and then as a response system will tell – user was born of which day and in current year user’s birthday will fall on which day.* |
| Recap (learning from the LAB assignment) | Understanding of Gregorian Calendar class  How and when to use it |

**REG EX –**

**ROUND 7**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | **RegEx**   * regEx overview * pattern & matches |
| LAB assignment | **LAB 6.13**  *Rewrite Lab 6.2 – this time use RegEx to validate the data entered by user*  **LAB 6.14**  *Write a program, where user will fill a registration form (with all basic info – Name, password, confirm password, data of birth, birth place, phone number and email id); Validate the form and save the info in a user object (POJO) and then store this object in a user-stack (array).*  *System will check the form at the time user will ask to save it. System should show appropriate message in case of invalid data. Upon successful submission, system will display the data of the entire user entered so far in user stack (array).*  **LAB 6.15**  *Modify the lab 6.X2.*  *Now, system should check the data entered by the user, at that field only; and if it is invalid, it should not allow user to enter the data for other field. In other word, instead of at the time of submitting the form, now data is being validating at the time of entry only.* |
| Recap (learning from the LAB assignment) | How to use RegEx to validate data  More practice on how to use java bean (POJO class) to store data in an object; and how to user array to store such objects.  Validating date and email Id, will further sharpen skill on RegEx |

**ANNOTATION –**

**ROUND 8**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | **Annotation**   * Use of annotation * How to rite Annotation, how to use it for class comment * How to use it as java doc |
| LAB assignment | **LAB 6.16**  *Create an annotation type element; use it as class level comment for a class.*  **LAB 6.17**  *Convert the annotation type element of Lab X.1 to appear in Javadoc-generated documentation* |
| Recap (learning from the LAB assignment) | How to annotation type element  How to covert it for javadoc type elements |

**ROUND 9**

|  |  |
| --- | --- |
| Topics to be facilitated (teach) | * **@Deprecated -** Annotation for - deprecate a method * **@Override -** Annotation for override, its advantage * **@SuppressWarnings -** Annotation for warning suppress, for deprecated and unchecked |
| LAB assignment | **LAB 6.18**  *Use @Deprecated annotation on any method of a class; now call this method from any other class.*  *See what happen (system will show warning message – that method is deprecated)*  **LAB 6.19**  *Create subclass of any class – use @override annotation on any method which subclass is overriding. Try to change the method signature (change parameter load).*  *See what happen (system will not allow to overload the method, as @override annotation is there, so it can only be overrided)*  **LAB 6.19**  *In Lab X.3 use @SuppressWarnings("deprecation") annotation in the method of calling class (which is calling the @deprecated method of Lab X.1).*  *See what happen (system will remove the deprecated warning message)* |
| Recap (learning from the LAB assignment) | How and why to @deprecated, @override and @ SuppressWarnings annotation.  How to mark a method deprecated |