Java – Learn From Home

Assignment – Chapter 6

Concept: Collections Framework

Objective: At the end of the assignment, participants will be able to:

- Create and manage collections

Problems:

Exercise 1:

Write a java program to find the Duplicate of Elements HashSet using **For-Loop**.

Guided Solution:

Step 1: import the Collection package java.util.*
Step 2: Create a class named FindingDuplicate.
Step 3: write a main method inside the FindingDuplicate class.
Step 4: Create a String array with 4 String elements.

E.g:

```
String name[] = {
        new String("Sang"),
        new String("Shin"),
        new String("Boston"),
        new String("Shin")
    };
```

TLS Copyright © 2016, Tech Mahindra. All rights reserved.

Step 5: Create an instance of HashSet and assign it to the Set interface variable.

E.g.: Set s = new HashSet();

Step 6: Create a for loop to iterate the String array.

E.g.:
for (int i=0; i<name.length; i++){}

Step 7: write an if condition inside the for loop with a condition to add element comparing with the String Array.

E.g: if (!s.add(name[i]))

Step 8: If the condition of the if statement is valid print the Duplicated element.

E.g: System.out.println("Duplicate detected: "+name[i]);

Step 9: Come out of the for loop and print the distinct elements.

System.out.println(s.size()+" distinct words detected: "+s);

Step 10: Compile the class using "javac FindingDuplicate.java" and Execute as "java FindingDuplicate"

Exercise 2:

Sort the Given List Elements. ["Smith", "Jones", "Smith", "Brown", "Able”] and Reverse the sorted List Elements.

Note: Duplicate elements should be allowed in the collection

TLS Copyright © 2016, Tech Mahindra. All rights reserved.

Guided Solution:

Step 1: Import the Collection package java.util.*
Step 2: Create a class named SortingAndReversing
Step 3: Write a main method inside the SortingAndReversing class.
Step 4: Create a List Variable and instantiate them using ArrayList Class.

E.g: List c = new ArrayList();

Step 5: Add all the elements given in the Questions using add() method.

```
c.add("Smith");
c.add("Jones");
c.add("Smith");
c.add("Brown");
c.add("Able");
```

Step 6: Call the method sort using Collections class and pass the List object inside the sort() method argument.

e.g.: Collections.sort(c);

Step 7: Print the O/P of the sorted List using the System.out.println(c).

Step 8: call the method sort using Collections class and pass the List object inside the sort() method argument.

e.g.: Collections.reverse(c);

Step 9: Print the O/P of the sorted List using the System.out.println(c).

Step 10: Compile the class using "javac SortingAndReversing.java" and Execute as "java SortingAndReversing".

TLS Copyright © 2016, Tech Mahindra. All rights reserved.

Exercise 3:

Add a number of elements to a HashSet, then prints the set.

Guided Solution:

Step 1: Import the Collection package java.util.*
Step 2: Create a class named CollectExample
Step 3: write a main method.
Step 4: Create an instance of HashSet and assign it to the Collection interface variable.
Step 5: use the Collection interface add() method to add elements into the HashSet collection
Step 6: Iterate the collection using Iterator interface
Step 7: Create Iterator variable and use the iterator function of Collection interface to assign the Iterator instance. e.g: (Iterator i = c.iterator();)
Step 8: Use the hasNext() method of the iterator interface and loop them using while loop.

e.g. while (i.hasNext())

Step 9: Use the System.out.println() method to print the collection elements inside the while loop using the next() method of the iterator interface.

e.g. (i.next())

Step 10: Compile the class using "javac CollectExample.java" and Execute as "java CollectExample".

TLS Copyright © 2016, Tech Mahindra. All rights reserved.

Exercise 4:

Create five employee objects, add them to a TreeSet collection, and then print the set.

Guided Solution:

Step 1: Create a Bean class named Employee.
Step 2: Create an Argument and No-Argument Constructors in the employee class.
Step 3: Create private member variables in Employee class

e.g.:
```
private int empID;
private String empName;
private int empSalary;
```

Step 4: Create getters and setter for the member variables creates in step #3.
Step 5: Create a separate notepad file and define a class named CreateEmployee.
Step 6: write main method inside the CreateEmployee class.
Step 7: Create 5 Employee class object.(each objects arg-constructor takes the corresponding employee details like (empID,empName, empSalary)

e.g.: Employee emp1 = new Employee(101,"Jonathan",25000);

Step 8: Create a Collection interface variable and assign the instance of TreeSet class.

e.g. Collection empStore = new TreeSet();

TLS Copyright © 2016, Tech Mahindra. All rights reserved.

Step 9: Add all the 5 employee objects inside the Collection object using add() Method.

e.g.:
empStore.add(emp1);empStore.add(emp2); etc..

Step 10: Create Iterator variable and use the iterator function of Collection interface to assign the Iterator instance.

e.g.: (Iterator i = c.iterator();)

Step 11: Use the hasNext() method of the iterator interface and loop them using while loop.

e.g. while (i.hasNext())

Step 12: get the Employee collection elements using the i.next() method inside the while loop.

e.g.: Employee tmp = (Employee) i.next();

Step 13: Use the System.out.println() method to print the all 5 Employee details inside the while loop.

System.out.println(tmp.empID + " "+ tmp.empName + " " + tmp.empSalary);

Step 14: Compile the class using "javac CreateEmployee.java" and Execute as "java CreateEmployee".

Exercise 5:

Add a number of elements to a HashMap, then prints the values of the map using FOR EACH LOOP and Generics.

TLS Copyright © 2016, Tech Mahindra. All rights reserved.

Guided Solution:

Step 1: Import the Collection package java.util.*;

Step 2: Create a class named MapExample

Step 3: Write a main method inside the MapExample class.

Step 4: Create an instance of HashMap and assign it to the Map interface with Generics referring the Integer as KEY and String as Value.

e.g:

```
Map<Integer,String> m = new HashMap<Integer,String>();
```

Step 5: Use the Collection interface put() method to add key and element into the HashMap collection

e.g.: m.put(new Integer(1), "Smith");m.put(new Integer(2), "Roshan"); etc...

Step 6: Use a keySet on Map interface object to derive the list of Keys in the collection object. (Use Generics)

e.g.:

```
Collection<Integer> c = m.keySet<Integer>();
```

Step 7: Use the For Each loop to iterate the Collection of Keys.

e.g.:

```
for(int key : c){
}
```

Step 8: Print the Map Collection Key and Value by using the get(key) method of Map object inside the FOR Each loop.

e.g.:

```
for(int key : c){
System.out.print( "Key:" + key + "Value"+ m.get(key));
```

TLS Copyright © 2016, Tech Mahindra. All rights reserved.

```
}
```

Step 9: Compile the class using "javac MapExample.java" and Execute as "java MapExample"

TLS Copyright © 2016, Tech Mahindra. All rights reserved.