**EX. NO: 10 Generics, Enum, Annotations and Collections**

**(03/4/2017-9/4/2017)**

**Note: Part I and Part 2 should be recorded in your observation as directed by your mentor**

**Part-I (Who should do Part-I?)**

***Anyone who wants to clear java, hope everyone wants to…….***

1. Create an enumeration of Apple types say Jonathan, GoldenDel, RedDel, Winesap, Cortland. Demonstrate the use of methods values(), valueof() and also display all the apple types using both the ways 1. for each and 2. switch. Do demonstrate that given 2 apple types, your program is capable of finding if they are same type or different.

2. Demonstrate with the same scenario above that “enum is a class type” by substantiating with needed code.

3. Reflection is the feature that enables information about a class to be obtained at run time. The reflection API is contained in the java.lang.reflect package. Create annotation Myanno with values str() of type string and val() of type integer. Apply Retention policy to RUNTIME. Using which annotate a method myMethod(), which obtains the annotation for this method and display the values of the members i.e First, get a Class object that represents this class. Secondly, get a Method object that represents this method. Next, get the annotation for this class. Finally, display the values.

4. Write a program to demonstrate Collection Interfaces: Collection, Set, List and Queue

5. Write a program to demonstrate Collection Classes: Array List, Hash Set and Tree Set

6. Write a program to demonstrate power of Accessing via Iterators

7. Write a program to demonstrate Map Interfaces: and Map Classes: Abstract Map, Hash Map, Tree Map

Part-II: MCQ /Objective (Who should do Part-II?)

*Who will not love fun? If you love fun go ahead …and you can take this fun back home and continue there as well……………….*

1. Can we declare constructer as private ? if yes where you used it in your project ?

2. Where do you use collections in your project and Which collection we should prefer while sorting.

3. How hashmap works?

4. What is the output for the below code ?

public class NameBean {

private String str;

NameBean(String str ){

this.str = str;

}

public String toString() {

return str;

}

}

import java.util.HashSet;

public class CollClient {

public static void main(String ... sss) {

HashSet myMap = new HashSet();

String s1 = new String("das");

String s2 = new String("das");

NameBean s3 = new NameBean("abcdef");

NameBean s4 = new NameBean("abcdef");

myMap.add(s1);

myMap.add(s2);

myMap.add(s3);

myMap.add(s4);

System.out.println(myMap);

}

}

options

A)das abcdef abcdef

B)das das abcdef abcdef

C)das abcdef

D)abcdef abcdef

5. What is the output for the below code ?

public class Test {

public static void main(String argv[]){

ArrayList list = new ArrayList();

ArrayList listStr = list;

ArrayList listBuf = list;

listStr.add(0, "Hello");

StringBuffer buff = listBuf.get(0);

System.out.println(buff.toString());

}

}

options

A)Hello

B)Compile error

C)java.lang.ClassCastException

D)null

6. What is the output for the below code ?

import java.util.LinkedList;

import java.util.Queue;

public class Test {

public static void main(String... args) {

Queue q = new LinkedList();

q.add("newyork");

q.add("ca");

q.add("texas");

show(q);

}

public static void show(Queue q) {

q.add(new Integer(11));

while (!q.isEmpty ( ) )

System.out.print(q.poll() + " ");

}

}

options

A)Compile error : Integer can't add

B)newyork ca texas 11

C)newyork ca texas

D)newyork ca

7. Synchronized resizable-array implementation of the List interface is \_\_\_\_\_\_\_\_\_\_\_\_\_?

options

A)Vector

B)ArrayList

C)Hashtable

D)HashMap

**Part-III (Who should do Part-III?)**

***If you are a person who loves to challenge yourself, train yourself till you tire and in short for those who aspire to become extra intellect, this is for you***

1. With respect to Bank application, you would have modified Bank Account as an Abstract class and Savings Account and Current Account inheriting from them. Now refine your Bank Activity class to include generics which can work with Savings Account / Current Account. Also apply appropriate collections (List/Map) to define the Accounts as against use of array of Accounts objects.

2. With respect to Bank application, Create a Thread class called as Account Operation and override its run() method which calls accountOperation(int choice) method where choice can be credit or debit operation in which the operations are performed on the synchronized BankAccount object to ensure that when credit happens, debit is not allowed and vice versa.

3. Explore Type annotation of Java 8.