# Exam Aug 17

Sunday, August 20, 2023 10:47 PM

```
package jaavaTutorial;
import java.util.HashMap;
public class Exam2023 {
 public static void main(String[] args) {
 /*
 Define a multiple dimensional array and loop usin
Define a class for singleton pattern
Define a two classes with inheritance
Define a example for overloding
Define example for overriding Define a interface ar
via class
Define a class with abstract method and inherit vi
 Define a example for exception handling via try c
 Define a enum with switch example
Define a program to write a text file
Define a Hashmap and loop over it .
 once done create a new git repo and send it on te

*/

//Define a multiple dimensional array and loop usi
// int [][] muld = {
// {1,2,3,4},
// {5,6,7,8},
// {9,10,11,12},
// {13,14,15,16} };
```

g for Each

nd implement

a normal class
atch block

legram

ng for Each

```
// for (int i = 0; i<muld.length;i++) {
//
// int [] obj = muld[i];
// for ( int j = 0; j<obj.length; j++) {
// System.out.println(obj[j]);
// }
// }


//Define a class for singleton pattern

//Define a two classes with inheritance Define a ex
overloding
//calling method for inheritance
// Parent p = new Parent ("dad", "mom");
// p.dispalyParent();
// Child c = new Child ("son", " daughter");
// c.displayChild();
// c.dispalyParent();


//Define example for overloading
 //calling OverLoading2 class
// OverLoading2 or = new OverLoading2("Harry", "Je
// or.displayNA(5, 7);
// or.displayNA("Sam", "NY");

//calling Overriding class
// Overriding2 O = new Overriding2();
// O.exam();
// O.test();
// O.task();
// Overriding3 o = new Overriding3();
```

```
xample for
```

```
rsey");
```

```
//calling Overriding class
// Overriding2 O = new Overriding2();
// O.exam();
// O.test();
// O.task();
// Overriding3 o = new Overriding3();
// o.exam();
// o.test();
// o.task();

//Define a class with abstract method and
inherit via normal class
 //calling AbsMethod2 class
// AbsMethod2 abs = new AbsMethod2 ("My
method");
// abs.tree();
// abs.flower();

//Define a Hashmap and loop over it .
// HashMap <String , Integer> HMap = new
HashMap<String , Integer>();
// HMap.put("Key one", 4);
// HMap.put("Key two", 4);
// HMap.put("Key three", 4);
// HMap.put("Key four", 4);
//
// for (HashMap.Entry<String, Integer> obj :
HMap.entrySet()) {
// System.out.println(obj);//took referance
for loop
// }
```

```
}
 }
 //Define a two classes with inheritance
//class for inheritance
//class Parent {
// String father;
// String mother;
//
// public Parent (String f, String m) {
// this.father= f;
// this.mother = m;
// }
// public void dispalyParent() {
// System.out.println("This is from parent");
// }
//}
//class Child extends Parent {
//
// public Child(String f, String m) {
// super(f, m);
// // TODO Auto-generated constructor stub
// }
// public void displayChild() {
// System.out.println("This is from child");
// }
//}
 //Define example for overriding Define a
interface and implement via class
//class for overloading
```

```
// class OverLoading2 {
// String name;
// String address;
//
// public OverLoading2 (String n, String a) {
// this.name = n;
// this.address = a;
// }
// public void displayNA (String n, String a)
{
// System.out.println("This is display from
String");
//
// }
// public void displayNA (int n, int a) {
// System.out.println("This is display from
Integer");
// }
// }
 //Define example for overriding Define a
interface and implement via class
// class Overriding2 {
// public void exam () {
// System.out.println("this is from exam.");
// }
// public void test() {
// System.out.println("this is from test.");
// }
// public void task() {
// System.out.println("this is from task.");
// }
// }
```

```
// class Overriding3 extends Overriding2{
// public void exam () {
// System.out.println("this is from exam
one.");
// }
// public void test() {
// System.out.println("this is from test
two.");
// }
// public void task() {
// System.out.println("this is from task
three.");
// }
// }
 //Define a class with abstract method and
inherit via normal class
// abstract class AbsMethod {
//
// String forest;
// public AbsMethod (String f) {
// this.forest = f;
//
// }
// abstract void tree();
// abstract void flower();
// }
// class AbsMethod2 extends AbsMethod{
//
// public AbsMethod2(String f) {
// super(f);
// // TODO Auto-generated constructor stub
```

```
// }
//
// @Override
// public void tree() {
// System.out.println("This is a tree.");
//
// }
//
// @Override
// public void flower() {
// System.out.println("This is a flower");
//
// }
//
// }
```