Vimal Raj Viswaraj

M1135

Answers

Summer Camp JAVA Subjective Written Test

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Questions** | **Topics Covered** | **Marks** |
| 1 | Write a program to demonstrate Last In First Out concept demonstration | Collection | 10 |
| 2 | Write a program to read all the lines from the text file. While printing, print only the alternate lines and amend “#” to every line that is printed | Files I/O Streams | 10 |
| 3 | Write a program to demonstrate constructor, encapsulation and Inheritance . | Constructor, Encapsulation & Inheritance | 10 |
| 4 | Write a program that will remove a given characters from the string (String = “REST ASSURED” , remove characters “ST” )  Write a program to compare two strings using JAVA Program | String function | 5 |
| 5 | Write a program to remove duplicate values from the has set that has the key value pair as below  Key 1: TestVal1  Key 2: TestVal2  Key 3: TestVal1  Key 4: TestVal2  Key 5: TestVal2  Key 6: TestVal3 | Collection | 5 |
| 6 | Write a program to illustrate method overloading and Interface | Method Overloading | 10 |

Answers:

1)

**import** java.util.LinkedList;

/\*

\* @Title: LastInFirstOut.java

\* @Description: Demonstrates Last In First Out concept using LinkedList

\* @Author: Vimal Raj Viswaraj

\* @EmployeeId: M1135

\*/

**public** **class** LastInFirstOut {

**public** **static** **void** main(String[] args) {

// LinkedList object is created

LinkedList<String> list = **new** LinkedList<String>();

// Data's are posted inside the list using add()

list.add("I'm");

list.add("writing");

list.add("exam");

list.add("on");

list.add("exam");

System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*Last In First Out\*\*\*\*\*\*\*\*\*\*\*\*");

// Displays the complete list

System.***out***.println("List : " + list);

// Displays the last element

System.***out***.println("Last IN string is (Tail) : " + list.getLast());

// Removes the last element

list.removeLast();

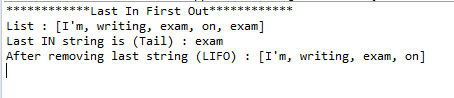
// Displays the updated list

System.***out***.println("After removing last string (LIFO) : " + list);

}

}

Output:



2)

import java.io.BufferedReader;

import java.io.FileInputStream;

import java.io.IOException;

import java.io.InputStreamReader;

/\*

\* @Title: PrintingAlternateLinesInFiles.java

\* @Description: program to read all the lines from the text file.

\* While printing, print only the alternate lines and amend “#” to every line that is printed

\* @Author: Vimal Raj Viswaraj

\* @EmployeeId: M1135

\*/

public class PrintingAlternateLinesInFiles {

public static void main(String[] args) throws IOException {

// FileInputStream object is created

FileInputStream fstream = new FileInputStream("./SupportDocuments/Sales.txt");

// BufferedReader object is created and input file path is assigned

BufferedReader br = new BufferedReader(new InputStreamReader(fstream));

// Local variables assigned

String strLine;

int n = 1;

// Try block starts

try {

// Reads line by line

while ((strLine = br.readLine()) != null) {

// Prints only even lines

if (n % 2 == 0) {

// StringBuilder object is created a

StringBuilder obj = new StringBuilder(strLine);

// appends # to the end of each line

obj.append("#");

System.out.println(obj);

}

n++;

}

}

//Catch block

catch (IOException e) {

e.printStackTrace();

}

//Finally block

finally {

//closing the file stream

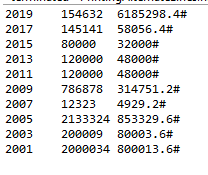
fstream.close();

}

}

}

Output



3)

/\*

\* @Title: EncapsulatedClass.java

\* @Description: Demonstrates Encapsulation

\* @Author: Vimal Raj Viswaraj

\* @EmployeeId: M1135

\*/

**public** **class** EncapsulatedClass {

// Private variables declared

**private** String userName;

**private** **int** empId;

// Getter methods

**public** String getUserName() {

**return** userName;

}

// Setter methods

**public** **void** setUserName(String userName) {

**this**.userName = userName;

}

// Getter methods

**public** **int** getEmpId() {

**return** empId;

}

// Setter methods

**public** **void** setEmpId(**int** empId) {

**this**.empId = empId;

}

}

**import** java.util.Random;

/\*

\* @Title: InheritedClass.java

\* @Description: Demonstrates Inheritance and Constructor

\* @Author: Vimal Raj Viswaraj

\* @EmployeeId: M1135

\*/

**public** **class** InheritedClass **extends** EncapsulatedClass {

// Constructor method

**public** InheritedClass() {

Random rand = **new** Random();

// Returns random number

System.***out***.println("Employee Record created : " + rand.nextInt());

}

}

**import** java.util.Scanner;

/\*

\* @Title: EncapsulationAndInheritance.java

\* @Description: Program to demonstrate constructor, encapsulation and Inheritance .

\* @Author: Vimal Raj Viswaraj

\* @EmployeeId: M1135

\*/

**public** **class** EncapsulationAndInheritance {

**public** **static** **void** main(String[] args) {

// InheritedClass object is created

InheritedClass obj = **new** InheritedClass();

// Scanner class defined

Scanner input = **new** Scanner(System.***in***);

System.***out***.println("Enter Name : ");

// Gets input from user

String name = input.next();

obj.setUserName(name);

System.***out***.println("Enter Employee Id : ");

**int** emp = input.nextInt();

obj.setEmpId(emp);

// Displays the details stored

System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Details\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

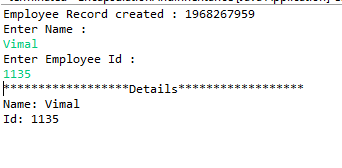
System.***out***.println("Name: " + obj.getUserName());

System.***out***.println("Id: " + obj.getEmpId());

}

}

Output



4)

/\*

\* @Title: StringFunction.java

\* @Description: Demonstrates replaceFirst and Compare methods

\* @Author: Vimal Raj Viswaraj

\* @EmployeeId: M1135

\*/

**public** **class** StringFunction {

**public** **static** **void** main(String[] args) {

/\* Write a program that will remove a given characters from the

string (String = “REST ASSURED” , remove characters “ST” )\*/

//Test string is defined

String testStr = "REST ASSURED";

//Replace the value from the string

System.***out***.println(testStr.replaceFirst("ST", ""));

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*Write a program to compare two strings using JAVA Program \*/

String sampleStr = "Vimal";

String sampleStr1 = "Raj";

// One way of comparing two string is using equals function

**if** (sampleStr.equals(sampleStr1)) {

System.***out***.println("Both string is same!");

}

**else** {

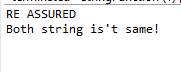
System.***out***.println("Both string is't same!");

}

}

}

Output



5)

**import** java.util.Collection;

**import** java.util.Collections;

**import** java.util.HashMap;

**import** java.util.Iterator;

/\*

\* @Title: KeyValue.java

\* @Description: Write a program to remove duplicate values from the has set that has the key value pair as below

Key 1: TestVal1

Key 2: TestVal2

Key 3: TestVal1

Key 4: TestVal2

Key 5: TestVal2

Key 6: TestVal3

\* @Author: Vimal Raj Viswaraj

\* @EmployeeId: M1135

\*/

**public** **class** KeyValue {

**public** **static** **void** main(String[] args) {

//HashMap object is created

HashMap<String, String> mapValues = **new** HashMap<String, String>();

//Values are put inside the HashMap

mapValues.put("Key 1", "TestVal1");

mapValues.put("Key 2", "TestVal2");

mapValues.put("Key 3", "TestVal1");

mapValues.put("Key 4", "TestVal2");

mapValues.put("Key 5", "TestVal2");

mapValues.put("Key 6", "TestVal3");

//Collection object is defined

Collection<String> list = mapValues.values();

//For loop using Iterator

**for** (Iterator<String> itr = list.iterator(); itr.hasNext();) {

//Checks the condition

**if** (Collections.*frequency*(list, itr.next()) > 1) {

//Removes the duplicate values

itr.remove();

}

}

//Displays the final list

System.***out***.println(mapValues);

}

}

Output



6)

/\*

\* @Title: InterfaceSample.java

\* @Description: Demonstrates Interface

\* @Author: Vimal Raj Viswaraj

\* @EmployeeId: M1135

\*/

**public** **interface** InterfaceSample {

//Declaring all methods as abstract

**abstract** **int** sum(**int** a, **int** b);

**abstract** **float** sum(**float** a, **float** b);

**abstract** **float** sum(**int** a, **float** b);

}

/\*

\* @Title: MethodOverloading.java

\* @Description: Demonstrates Method Overloading and Interface

\* @Author: Vimal Raj Viswaraj

\* @EmployeeId: M1135

\*/

**public** **class** MethodOverloading **implements** InterfaceSample {

@Override

**public** **int** sum(**int** a, **int** b) {

// **TODO** Auto-generated method stub

**return** a + b;

}

@Override

**public** **float** sum(**float** a, **float** b) {

// **TODO** Auto-generated method stub

**return** a + b;

}

@Override

**public** **float** sum(**int** a, **float** b) {

// **TODO** Auto-generated method stub

**return** a + b;

}

**public** **static** **void** main(String[] args) {

//Object for class is created

MethodOverloading obj = **new** MethodOverloading();

//Calling method

**int** sumInt = obj.sum(10, 20);

**float** sumFloat = obj.sum(5.5f, 8.9f);

**float** sumIntFloat = obj.sum(15, 98.78f);

//Displays the values

System.***out***.println("Sum of two integer is: " + sumInt);

System.***out***.println("Sum of two float is: " + sumFloat);

System.***out***.println("Sum of one integer and one float is: " + sumIntFloat);

}

}

Output

