Core java Test

**Name- venkateshwarlu Egurla (Shashi)**

1. 0120120120 (C)

2. 9 (B)

3. 720 (c)

4.0123456 (D)

5. 2 (c)

6. 27 9 (B)

7. 2 (B)

8. compilation error (c)

9.X is larger (B)

10. Truetrue ( c)

11. 6

12. ac

13.class (B)

14.851 (d)

15. \*

16. 0 0 (B)

17.02468

18. World world (D)

19. none

20. 1 1 (A)

1. Write a program to reverse particular string ?

**class** Exam {

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

String word = "Hello world 1 ";

*reverseWord*(word);

}

**public** **static** **void** reverseWord(String s) {

String[] words = s.split(" ");

**for** (**int** i = 0; i <words.length; i++) {

**if** (i==0) {

System.***out***.print(**new** StringBuilder(words[i]).reverse() + " ");

} **else** **if**(i==2) { System.***out***.print(" ");}

**else** {

System.***out***.print(words[i] + " ");

}

}

}

}

**Output**

**olleH world**

**class** Exam {

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

String word = "What is the meaning of life 4";

*reverseWord*(word);

}

**public** **static** **void** reverseWord(String s) {

String[] words = s.split(" ");

**for** (**int** i = 0; i <words.length; i++) {

**if** (i==3) {

System.***out***.print(**new** StringBuilder(words[i]).reverse() + " ");

} **else** **if**(i==6) { System.***out***.print(" ");}

**else** {

System.***out***.print(words[i] + " ");

}

}

}

}

Out put

**What is the gninaem of life**

Write a program to write employee information using serialization and collection frame works ?

**import** java.io.Serializable;

**public** **class** Employeeinfo **implements** Serializable {

String empname;

String empid;

**int** empsal;

**public** Employeeinfo(String empname, String empid, **int** empsal) {

**super**();

**this**.empname = empname;

**this**.empid = empid;

**this**.empsal = empsal;

}

@Override

**public** String toString() {

**return** "Employeeinfo [empname=" + empname + ", empid=" + empid + ", empsal=" + empsal + "]";

}

}

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.ObjectOutputStream;

import java.util.ArrayList;

public class Empserializedrive {

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

ArrayList<Employeeinfo> a1=new ArrayList<Employeeinfo>();

Employeeinfo e1 = new Employeeinfo("Shashi","117z1a", 25000);

Employeeinfo e2 = new Employeeinfo ("tarun","10z423", 53000);

Employeeinfo e3 = new Employeeinfo ("bharath","a14033", 15000);

a1.add(e1);

a1.add(e2);

a1.add(e3);

FileOutputStream fos = new FileOutputStream("E:/sample/employee.ser");

ObjectOutputStream oos= new ObjectOutputStream(fos);

oos.writeObject(a1);

System.out.println("object serilazed");

oos.close();

fos.close();

}

}

Output **object serilazed**

import java.io.FileInputStream;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.util.ArrayList;

public class EmpDeSerialization {

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

ArrayList<Employeeinfo> a1=new ArrayList<Employeeinfo>();

FileInputStream fis = new FileInputStream("E:/sample/employee.ser");

ObjectInputStream ois = new ObjectInputStream(fis);

a1 = (ArrayList<Employeeinfo>) ois.readObject();

ois.close();

fis.close();

System.out.println(a1);

}

}

**Output**

**[Employeeinfo [empname=Shashi, empid=117z1a, empsal=25000], Employeeinfo[empname=tarun, empid=10z423, empsal=53000], Employeeinfo[empname=bharath, empid=a14033, empsal=15000]]**

3. write a program to demonstrate interthread communication.

**package** queuedemo;

**public** **class** InterThreadDemo {

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

**final** Customer c = **new** Customer();

**new** Thread()

{

**public** **void** run()

{

c.withdraw(1500);

}

}.start();

**new** Thread()

{

**public** **void** run()

{

c.deposit(1000);

}

}.start();

**new** Thread()

{

**public** **void** run()

{

c.deposit(1000);

}

}.start();

}

}

**class** Customer

{

**int** amount = 1000;

**synchronized** **void** withdraw(**int** amount)

{

System.***out***.println("Balance avaliable " + **this**.amount);

System.***out***.println("Going to withdraw." + amount);

**if** (**this**.amount < amount)

{

System.***out***.println("Insufficient Balance waiting for deposit.");

**try**

{

wait();

} **catch** (Exception e)

{

System.***out***.println("Interruption Occured");

}

}

**this**.amount -= amount;

System.***out***.println("Detected amaount: " + amount);

System.***out***.println("Balance amount : " + **this**.amount);

}

**synchronized** **void** deposit(**int** amount)

{

System.***out***.println("Going to deposit " + amount);

**this**.amount += amount;

System.***out***.println("Available Balance " + **this**.amount);

System.***out***.println("Transaction completed.");

notify();

}

}

**Output**

**Balance avaliable 1000**

**Going to withdraw.1500**

**Insufficient Balance waiting for deposit.**

**Going to deposit 1000**

**Available Balance 2000**

**Transaction completed.**

**Detected amaount: 1500**

**Balance amount : 500**

**Going to deposit 1000**

**Available Balance 1500**

**Transaction completed.**