public class Test {

public static void main(String[] args) {

String s1 = "Hello";

String s2 = s1;

String s3 = new String("Hello");

String s4 = new String("Hello");

String s5 = "Hello";

System.out.println(s1 == s5);

System.out.println(s1 == s2);

System.out.println(s1.equals(s2));

System.out.println(s3 == s4);

System.out.println(s3 == s1);

System.out.println(s3.equals(s4));

}

}

String twister = "zebras zig and zebras zag";

System.out.println(twister.indexOf('z'));

System.out.println(twister.lastIndexOf('z'));

System.out.println(twister.indexOf("zebra",4));

System.out.println(twister.lastIndexOf("zebra",4));

}

Scanner class can be used to read from a string also

Serialization is to put entire object into a file

If a variable is declared as transient, then the value of this vaiarble is not serialized

Class employee{

String aname;

Private transient int id; // will notbe stored into the file

Static int gender;//

Fileoutputstream fos = new fileouputstream(“filename’);

Objectoutputstream oos = new objectooutputstream(fos)

Oos.writeobject(emp);

Opening the file and verifying output wont work as file is not readable

Fileinputstream fos = new fileinputstream(“filename’);

Objectoinputstream oos = new objectinputstream(fos)

Oos.readobject(); //if u print this out, observe whether transient variable value got serialized or not

Similarly obseve for static variazbles

public class Test

{ public static void main(String[] args)

{Set set=new TreeSet();

set.add("2");

set.add(3);

set.add("1");

set.add(6);

set.add("10");

set.add(50);

Iterator it=set.iterator();

while(it.hasNext())

{

System.out.println(it.next()+" ");

}}

}

To check whether a method throws exception or not

'@Test(expected=NullPointerException.class)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Consider the following code  1. interface Demo {  public void method() throws SQLException, RuntimeException;  }class Test implements Demo{  //Insert code here } Which method definition, inserted in above code guarantees that this program will compile? | public void method() throws SQLException, RuntimeException{} | public void method() throws Exception{} | public void method() throws RuntimeException{} | public void method() throws SQLException{} | public void method() throws SQLException, RuntimeException,Exception{} |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Considering following code, select the correct assertions for testing that RuntimeException is thrown? public class Maths {  public int add(int x, int y) throws RuntimeException{  if(x < 0 || y < 0){  throw new RuntimeException("-ve Values are not allowed");  }  return x + y;  } } | try{  assertEquals(15, obj.add(-10, 5));  }catch(RuntimeException re){  assertTrue(false);  } | try{  assertEquals(15, obj.add(-10, 5));  }catch(RuntimeException re){  assertTrue(true);  } | try{  assertEquals(15, obj.add(-10, 5));  }catch(RuntimeException re){  assertFalse(false);  } | try{  assertEquals(15, obj.add(-10, 5));  }catch(RuntimeException re){  assertFalse(true);  } |  |  | 2,3 |

This file is saved in this folder \\arltrgsrv\JEE Cloud Lab 8\for lab 6\core java notes for lab 6.doc

Lambda expression : new feature of java 8

|  |  |
| --- | --- |
| **int fun(int arg);** | **( num) -> num + 10** |

|  |  |
| --- | --- |
| **int fun(int arg0,int arg1);** | **( num1, num2) -> {**  **int min = num1>num2?num2:num1;**  **return min; }** |

|  |  |
| --- | --- |
| **void fun();** | **() -> { }** |

|  |  |  |
| --- | --- | --- |
| **Supplier<String>** | **String get();** | **() -> “Hello World”;** |

Streams API

Reason for this :

The senior most emp in the company ? select min(hiredate) from emp

Count of emps : select count(\*) from emp : output : 15

Display all those emps who belong to dept 20 select \* from emp where deptno=20

Select length(ename) from emp;

Ename length(ename)

King 4

Allen 5

If this situation has to be created in the java code

What is the equivalent of the table in java code ?

What is the equivalent of each row of the table in java code

String[] arr = {"Mouse","Monitor","KeyBoard"};

List<String> list = Arrays.*asList*(arr);

Stream.*of*(list).

filter(e->e.equals("Monitor")==**true**).

forEach(e->System.***out***.println(e));

Class Employee{

Int empid;

String empname ;//all columns of the table become attributes of the class

}

Object of this class becomes a row in the table

Emp e = new Emp();

e.setempname(“king”);

e,setdeptno(10);

e.setjob(“manager”);

one object is equivalent to one table row

many objects stored into a collection

ArrayList is equivalent to the oracle table

ArrayList<Emp> list = new ArrayList<Emp>();

List.add(emp1);

List.add(emp2);

List.add(emp3);

Old way of implementing this is use a for loop or an iterator and within this; apply the logic

To find out .

Newer and more concise of implementing sql like operations in java is the streams api

The main interface is the Stream interface

Many methods in this interface

Step 1. We need to obtain a stream from the array/collection/many values

Step 2: apply different methods on the stream

Multi threading

Every java application has only one thread : called the main thread

i.e the application executes the code in sequential manner ; line by line

multi threading is to make the application perform two or more tasks at the same time within the same application

if u want to print two for loops

but if this output has to alternate ; i.e output of first loop and then second loop and so on

rules how to make a class behave like a Thread

1. Either extend from the Thread class or
2. Implement the Runnable interface ; public void run()

public class MyForm extends Applet implements Runnable{

}

2.Override the run method and write the parallel task to be executed here

3. start the thread by calling start() of the Thread class