What is Static Block?

At the time of java – class loading, the corresponding native libraries should be loaded hence we have to define this activity inside static block.

class Test {

static {

System.loadLibrary(“native library path”)

}

}

After loading every database driver class, we have to register driver class with driver manager, but inside database driver class there is a static block to perform this activity and we are not responsible to register explicitly

class DBDriver {

static {

Register this driver with driver manger

}

}

What is the execution order of static block, when there are many number of static blocks?

Within a class we declare any number of static blocks, but all these static blocks will be executed from top to bottom

From 1.7v onwards main method is mandatory to start a program execution. Hence from 1.7v onwards without writing main method it in not possible to print statement to the console.

What is the order of static execution?

1. Identification of static member from top to bottom

2. Execution if static variable assignment and static blocks from top to bottom

3. Execution of main method

Refer the StaticTest.java for details explaination.

What is Read-direct and Read-indirect?

Read-direct: If you are trying to read a variable inside a static block

Read-indirect: If you are trying to read a variable within a method

**class** Test {

**static int** *i*=10;

**static** {

*m1*();

System.***out***.println(*i*); *//direct read*

}

**public static void** m1() {

System.***out***.println(*i*); *//indirect read*

}

}

If the variable is just identified by the JVM and original value not yet assigned then the variable is said to be in Read-indirectly and write only state(RIWO)

In a variable is in RIWO state then we can’t perform direct read, but we can perform indirect read.

If we are trying to read directly we will get compile time error saying “Illegal forward reference”

Refer IllegalForwardTest.java for more details

**public class** IllegalForwardTest {

**static** {

*m1*();

System.***out***.println(**"Inside static block: "**+ *j*); *//illegal forward*

}

**public static void** m1() {

System.***out***.println(**"Inside static block: "**+ *j*);//indirect read

}

**static int** *j* = 10;

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

}

}

What is the order of static block in parent-child relation?

Whenever we are executing child class, the following sequence of event will be executed

1. Identification of static member from parent to child

2. Execution of static variable assignment and static blocks from parent to child

3. Execution of only child class main method.

Whenever we loading child class automatically parent class will be loaded. But the opposite is not true.

Discuss instance control flow and its order

Whenever we are executing a java class first static control flow will be executed, in the static control flow if we are creating an object the following sequence of events will be exeucted.

1. Identification of instance member from top to bottom

2. Execution of instance variable assignment and instance block from top to bottom

3. Execution of constructor

Refer InstanceVariableTest.java for more details

Static control flow is one-time activity which will be performed at the time of class loading but instance control flow is not one time activity and it will be performed for every object creation.

So object creation is most costly operation if there is no specific requirement then it is not recommended to create object.

Discuss about the instance control flow in parent to child relationship

1. Identification of instance member from parent to child

2. Execution of instance variable assignment and instance blocks of parent class

3. Execution of parent constructor

4. Execution of instance variable assignments and instance blocks in child class

5. Execution of child constructor.