Feel free to contact santhosh_arena@yahoo.co.in

SCJP java Programmer certification Exam And Training. This site is entirely independent of Sun Microsystems Inc, The Sun Certified Java Programmers Exam (SCJP) is the internationally recognized java certification exam. I am offering free mock Exam preparation questions, practice tests, tutorials, certification faq and sample code. This page contains a very detailed tutorial organized by topic. At the end of the tutorial you can learn from your mistakes and apply your concepts on the free mock exams java certification practice tests.

Chapter 6 - Objects and Classes (Part 2)

Constructors and Sub-classing

- Constructors are not inherited as normal methods, they have to be defined in the class itself.
    - If you define no constructors at all, then the compiler provides a default constructor with no arguments. Even if, you define one constructor, this default is not provided.
    - We can't compile a sub-class if the immediate super-class doesn't have a no argument default constructor, and sub-class constructors are not calling super or this explicitly (and expect the compiler to insert an implicit super() call)
    - A constructor can call other overloaded constructors by '**this (arguments)**'. If you use this, it must be the first statement in the constructor. This construct can be used only from within a constructor.
    - A constructor can't call the same constructor from within. Compiler will say '***recursive constructor invocation***'
    - A constructor can call the parent class constructor explicitly by using '**super (arguments)'**. If you do this, it must be first the statement in the constructor. This construct can be used only from within a constructor.
    - Obviously, we can't use both this and super in the same constructor. **If compiler sees a this or super, it won't insert a default call to super().**
    - Constructors can't have a return type. A method with a class name, but with a return type is not considered a constructor, but just a method by compiler. Expect trick questions using this.
    - **Constructor body can have an empty return statement**. Though void cannot be specified with the constructor signature, empty return statement is acceptable.
    - Only modifiers that a constructor can have are the accessibility modifiers.

- Constructors cannot be overridden, since they are not inherited.
- Initializers are used in initialization of objects and classes and to define constants in interfaces. These initializers are :

    1. Static and Instance variable initializer expressions.
        - Literals and method calls to initialize variables.
        - Static variables can be initialized only by static method calls.
        - Cannot pass on the checked exceptions. Must catch and handle them.
    2. Static initializer blocks.
        - Used to initialize static variables and load native libraries.
        - Cannot pass on the checked exceptions. Must catch and handle them.
    3. Instance initializer blocks.
        - Used to factor out code that is common to all the constructors.
        - Also useful with anonymous classes since they cannot have constructors.
        - All constructors must declare the uncaught checked exceptions, if any.
        - Instance Initializers in anonymous classes can throw any exception.

- In all the initializers, forward referencing of variables is not allowed. Forward referencing of methods is allowed.
- Order of code execution (when creating an object) is a bit tricky.

    1. static variables initialization.
    2. static initializer block execution. (in the order of declaration, if multiple blocks found)
    3. constructor header (super or this - implicit or explicit)
    4. instance variables initialization / instance initializer block(s) execution

5. rest of the code in the constructor