Constructors

Subject : _____ Date

→ used to initialize the object & is invoked at the time of object creation

⇒ if no constructor is present, then compiler automatically creates a default constructor

→ two basic rules:

* constructor name must be same as class name
* must have no explicit return type

→ two types of constructors:

* default constructor [only action is to call the constructor using super() call]
* parameterized constructor

* Constructor Overloading

a class can have any number of constructors that differ in the number of parameters in the list and their type [argument list]

* Java Constructor	Java Method
→ used to initialize the state of an object.	used to expose behavior of an object.
→ must not have return type	must have return type.
→ invoked implicitly	invoked explicitly.
→ compiler provides def const if no constructor present	method is not provided by compiler in any case.
→ constr name must be same as class name.	method name may or may not be same as class name.

Scanned by CamScanner

Subject : _____ Date

* Java Copy Constructor:

→ there is no copy constructor in java as one like in C++. But there are many ways to copy the values of one object into another.

(i) By constructor: Class Constructor (Class Constructor cc)
{ id = cc.id; name = cc.name }

(ii) By assigning the values of one object into another
Class1 c1 = new Class1(1, "A");
Class2 c2 = new Class2();
c2.id = c1.id; c2.name = c1.name;

(iii) By clone() method of object class.

* Does constructor return any value?

→ Yes, that is current class instance. Still we cannot use return type yet it returns a value.

* what happens if you keep return type for a constructor?

It will be treated as a normal method. But compiler gives a warning saying that method has a constructor name.

* Constructor can perform other tasks instead of initialization, like object creation, thread starting, etc.

Subject: ---------- Date

* Constructor Chaining:

→ occurs when a class inherits another class.

→ An implied super() is included in each subclass constructors which does not contain this() or explicit super() call as its first statement.

→ The super() statement or this() or explicit super() call should be the first statement in constructor because parent class is initialized before child class.

→ The explicit super allows parameter values to be passed to the constructor of its superclass.

```
subClass(sno, name, id)
{ super(10, "A");
  this.id = id;
}
```

* Local constructor chaning:

→ this() constructor is used to implement local chaning of constructors.

```
Class Constructor(id, name)
{ .... }

Class Constructor(sno)
{ this(10, "A");
  this.sno = sno;
}
```

* both this() and super() calls cannot occur in the same constructor.

* Disadvantage: not good in terms of encapsulation, because when creation of instance depends upon structure of constructor and if any change in constructor of superclass will require changes in all places where its object gets created.

TATA CONSULTANCY SERVICES
Experience certainty.

→ Standard way is to use factory design pattern.

Subject : _____ Date

* If a class contains only non-default constructors, then its subclasses will not include an implicit super() call, and will be flagged as compile-time error.
⇒ the subclasses must explicitly call super() with right arguments to match appropriate constructor of superclass.

* What is the use of private constructors?
→ used to restrict the instantiation of a class.
(e) when a class needs to prevent other classes from creating its objects.
→ objects for classes which has only private constructors can be created only within its own classes.
and this occurs when the class only contains static members.
→ Use: Singleton pattern. this ensures only one instance of a class exist at any point of time.

```
class MyClass
{   private static MyClass object = null;
    private MyClass() { ... }
    public static MyClass getObject()
    {   if( object == null ) { object = new MyClass(); }
        return object;
    }
}
```