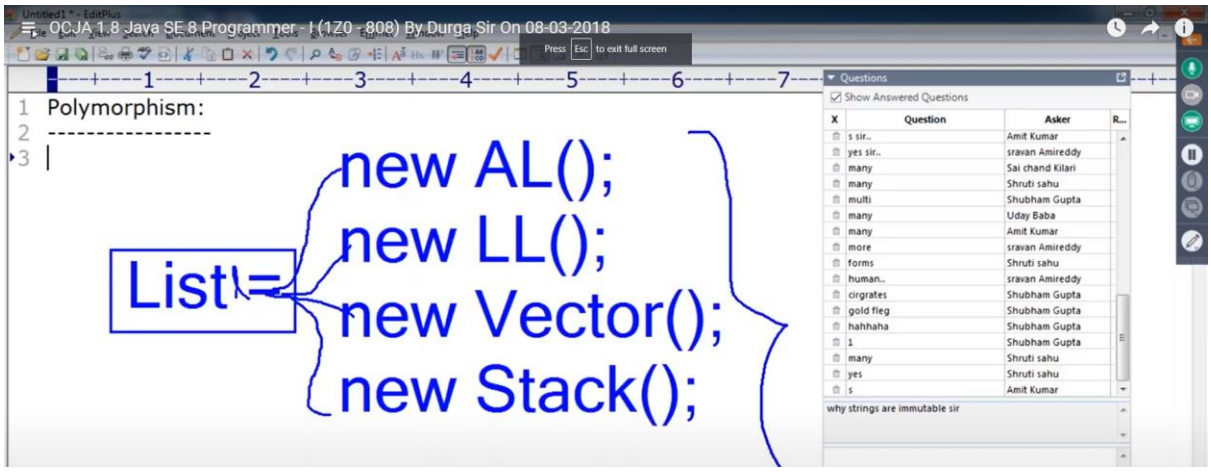
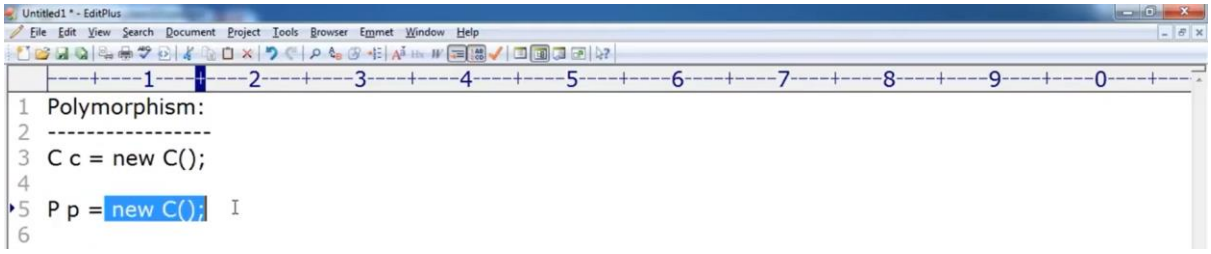
Same ref can hold different object types - polymorphism



Parent ref can hold child object – Eg of polymorphism

But can call only parent call methods and not child class methods. Why are we taking tis ref

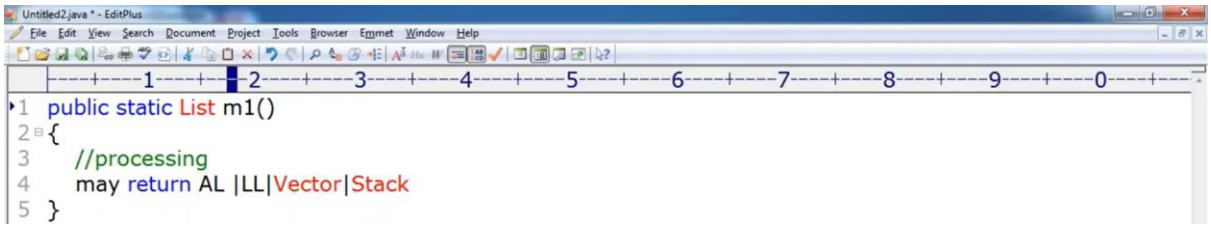


Whereas in line 3 you can call both parent and child methods. What is the biggest advantage

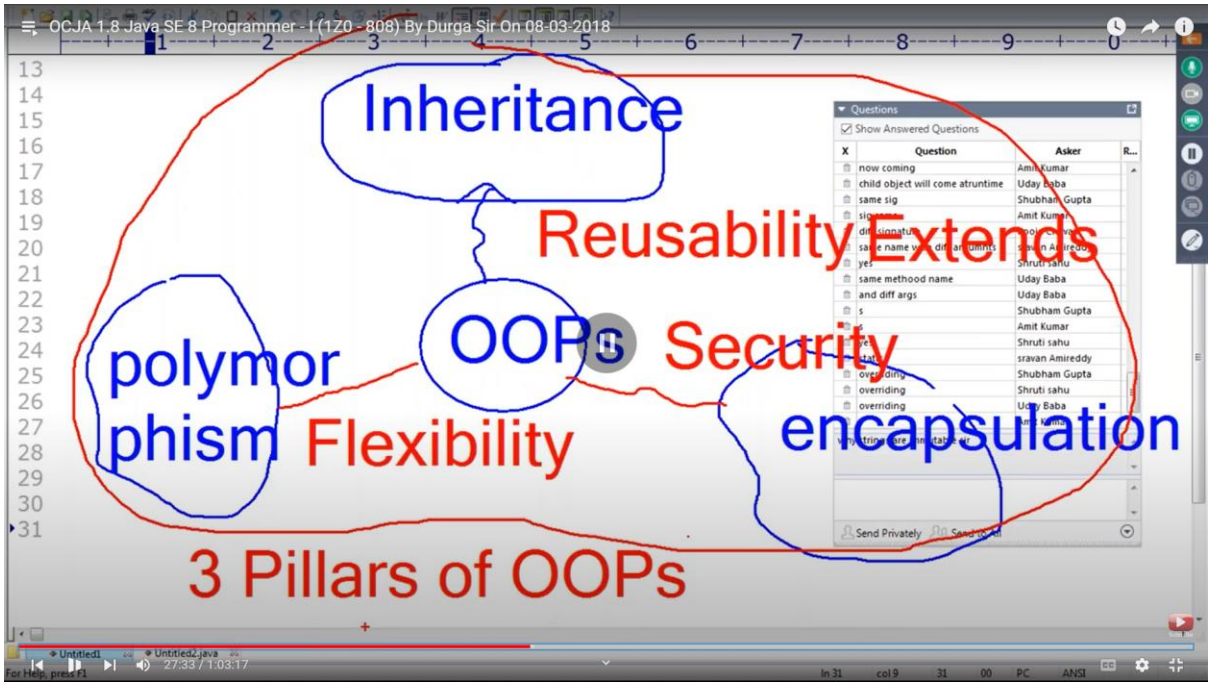
Polymorphism is the advantage.

If you return type is AL or LL or Vector , define the return type as List Parent ref.

But I line no 12 cant access child class methods cos its parent rref

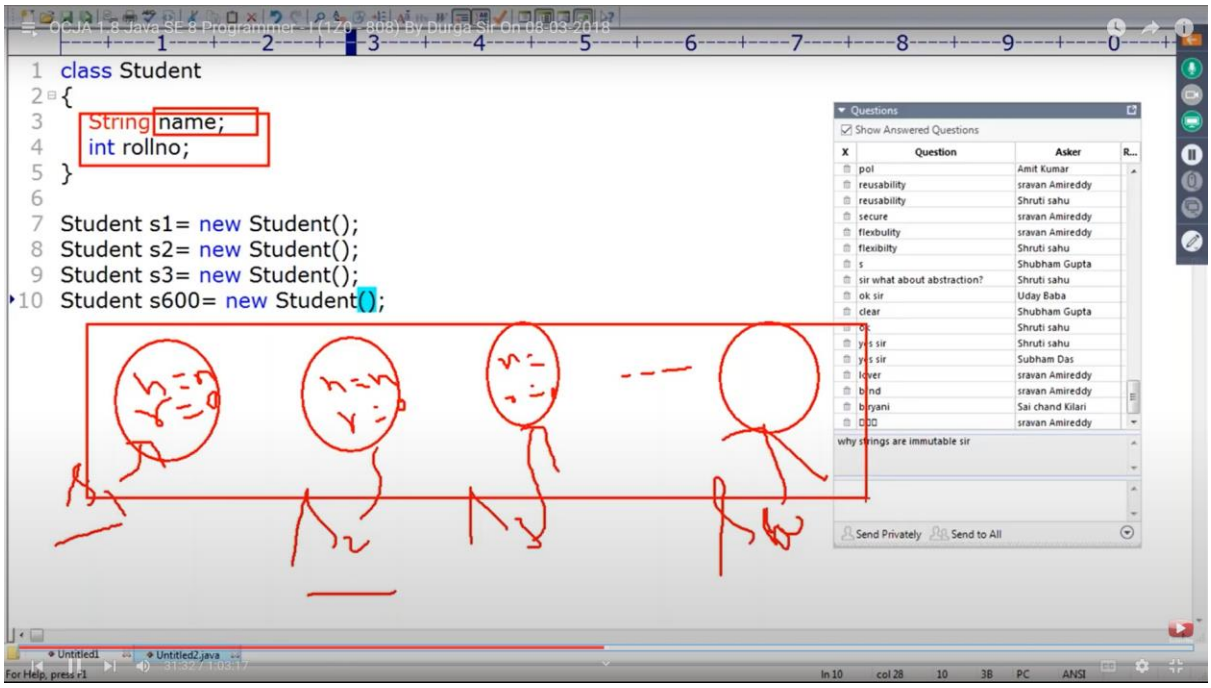


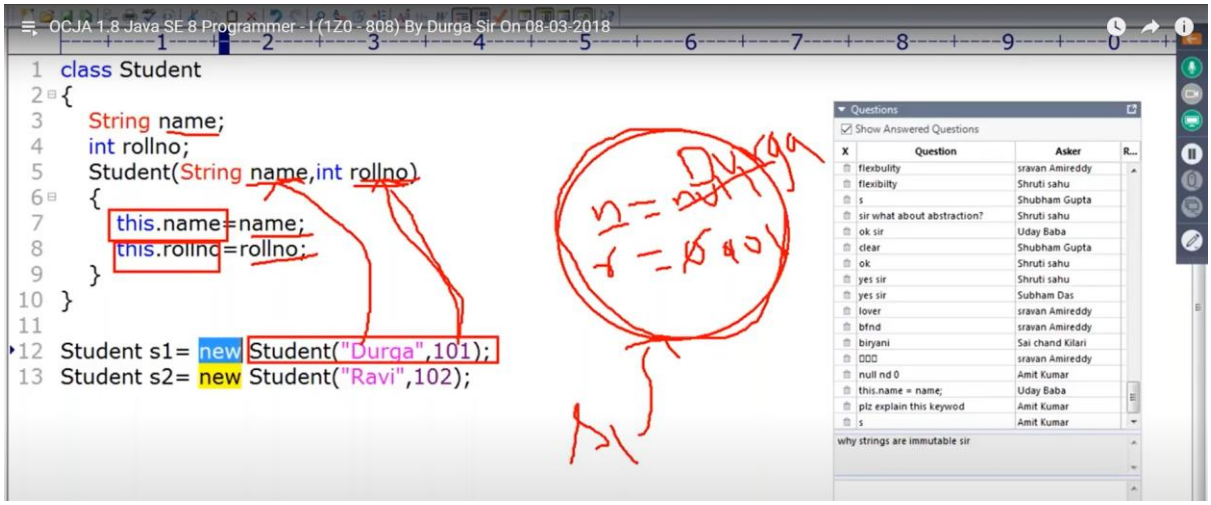
C c = new C(); AL l = new AL();	P p = new C(); List l = new AL();
1. If we know exact runtime type of object	1. If we dont know exact runtime type of object
2. By using AL reference we can hold only AL object	2. By using this Parent referece we can hold any type of child object
3. By using child reference we can call both parent and child class methods	3. BY using Parent reference we can call only parent class methods but not child specific mthodes, b'z we don't know which child object will come at runtime.



Constructor

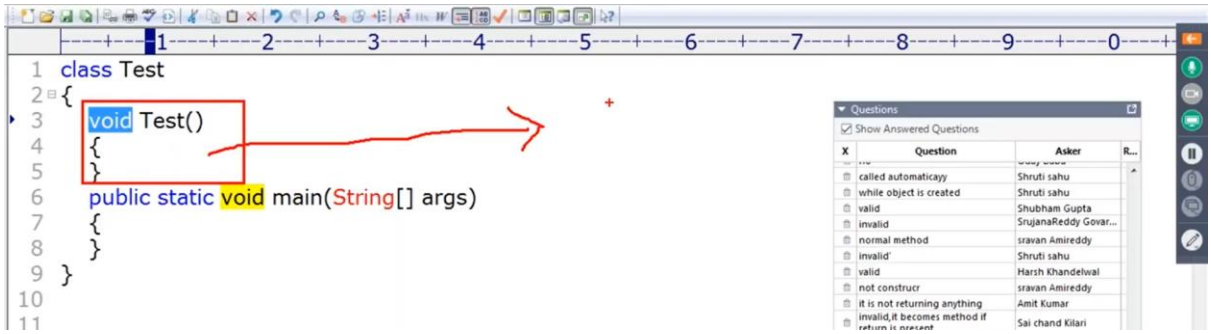
Once an object is created we need initialize. Constructor does it



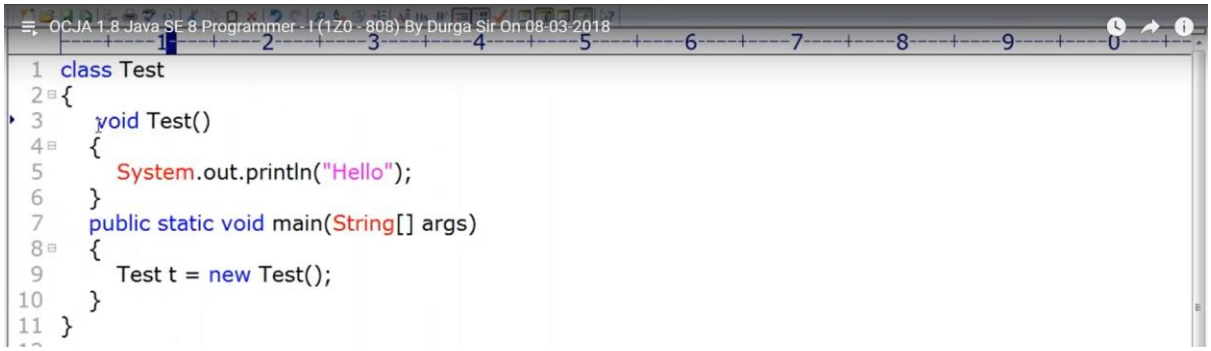


First object will be created with default value then constructor will be executed.

U cant specify a return type for cons. If u specified then it will be considered as a method not cons

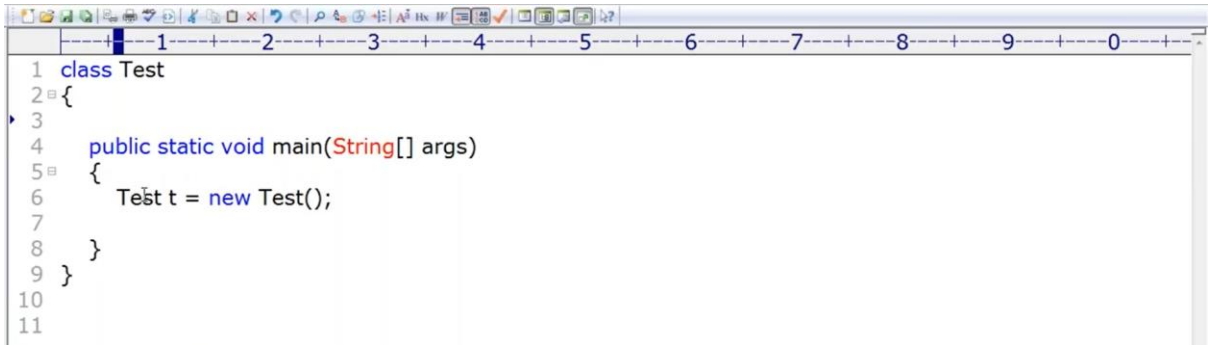


The below method will not be executed cos its not a constructor. Not recommended



Only accepted modifiers for constructors are

Who generates the default constructor. JVM can't do as it's an interpreter which executes code line by line how can it do. Compiler generates the default constructor for every class including abstract class



If our class has any constructor then the compiler will not create the default constructor.

```
Default Constructor:
---------------------
compiler

compiler will always generates default constructor for every class.


Every class contains constructor it may be explicit constructor provided by programmer or default
constructor generated by compiler but not both simulataneously
```

If the class is public then constructor by default is public, If class is default then constructor would be default. Only if the class is public or default then this rule applies

Else by default its default only

```
Default Constructors:
-------------------------

prototype of DC:
--------------------

1. no-arg constructor
2. same class modifier
3. only one line : super()
```

You should specify the const as public

```
public class Test
{
    public Test()
    {
        super();
    }
}
```

Now the constructor is default

```
final class Test
{
    Test()
    {
        super();
    }
}
```

When compile generates the default const it just creates 1 line Super();

In this case the compiler generates the default as Test below is method

```
1  class Test
2  {
3      void Test()
4      {
5      }
6
7
8  }
```

The programmer checks if there are any const. Yes True

The first line of the constructor should be either super() of this(). Here in line 8 there is no this() or super . So compiler will place the line super() in line 8

```
1  class Test
2  {
3      public Test()
4      {
5          this(10);
6      }
7      public Test(int i )
8      {
9      }
10 }
```

The first line inside any constructor should be this or super

```
13 --------------------
14
15 1. no-arg constructor
16 2. same class modifier
17 3. only one line : super()
18
19
20 super() or this()
21 ===>super()
22
23
```

Class is

```
class Test
{
    Test()
    {
    }
}
```

The default constructor does not contain any code, so compiler places the code Super() inside it

If you place both super or this, then CE error. Cant place both of them

```
class Test
{
    public Test()
    {
        super();
        this();
    }

}
```

```
D:\durgaclasses>javac Test.java
Test.java:6: error: call to this must be first statement in constructor
                this();
                    ^
1 error
```

Invalid . First line should be either Super or this

```
class Test
{
    public Test()
    {
        System.out.println("constructor");
        super();
    }
}
```

```
class Test
{

    Test()
    {
        super(); or this(10);
    }
```

Calling super class constructor inside a method.

```
class Test
{
    public void m1()
    {
        super();
        System.out.println("Hello");
    }
}
```

U can call a const from another const

```
2. properly
---------------------------------------

super.x
this.x
super.m1()
this.m1()
```

keywor

super()
this()

constructor calls|

Cant call super class const inside method

```
class Test
{

    Test()
    {
        super(); or this(10);
    }
    public void m1()
    {
        super();
        System.out.println("Hello");
    }
}
```

```
D:\durgaclasses>javac Test.java
Test.java:6: error: call to super must be first statement in constructor
                super();
                     ^

1 error
```

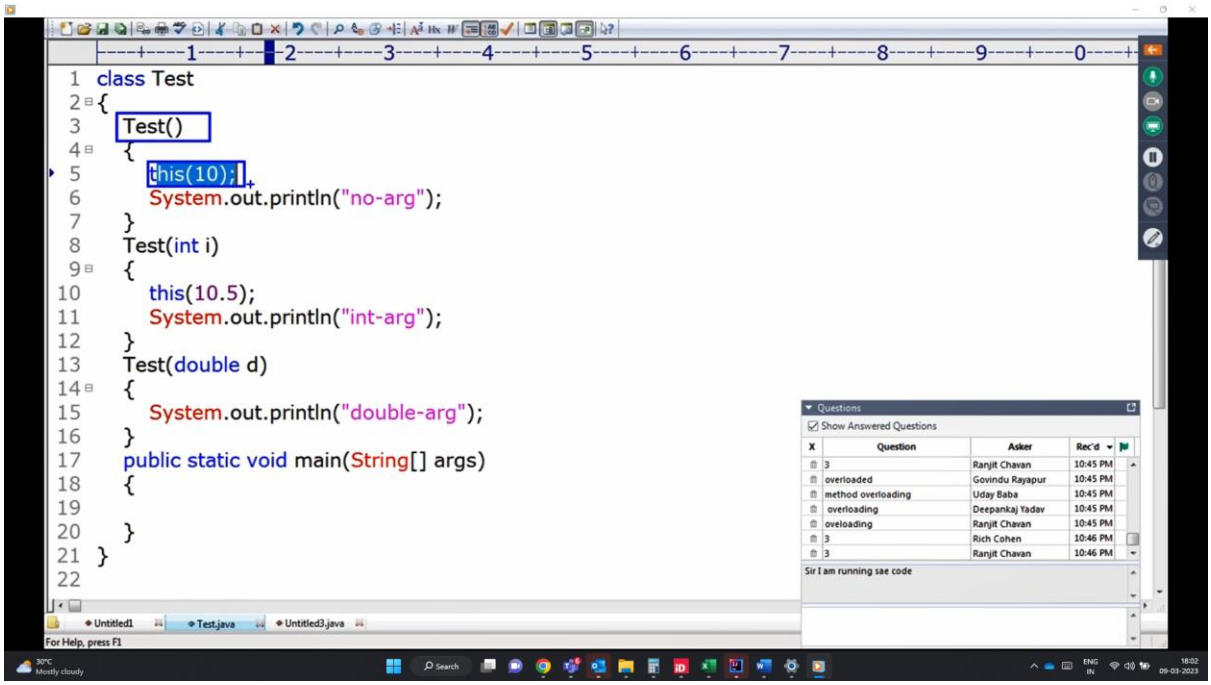
We can take super() or this() only in constructors as first line only
only one but not both simulataneously

super,this	super(), this()
keywords to refer super class and current class instance memebers	constructor calls to call super class and current class constructors
We can use any number of times no restrictions	Only once inside constructor at first line only
Inside methods,blocks,constructors... except static area	only inside constructors

Cant use non static variable super cannot be referrenced from Static context

```
class Test
{
    public static void main(String[] args)
    {
        System.out.println(super.hashCode());
    }
}
```

Constructor overloading possible



Here line 14 there is no super or this. So compiler will place Super() there.

Stack over flow error when recursive call without stoping condition. Run Time error

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 09-03-2018

```
class Test
{
    public static void m1()
    {
        m2();
    }
    public static void m2()
    {
        m1();
    }
    public static void main(String[] args)
    {
        m1();
    }
}
```

Here the code will execute because we are not calling either m1 or m2

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 12-03-2018

```
class Test
{
    public static void m1()
    {
        m2();
    }
    public static void m2()
    {
        m1();
    }
    public static void main(String[] args)
    {
        System.out.println("Hello");
    }
}
```

```
D:\durgaclasses>javac Test.java

D:\durgaclasses>java Test
Hello
```

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 12-03-2018

```
class Test
{
    public static void m1()
    {
        m2();
    }
    public static void m2()
    {
        m1();
    }
    public static void main(String[] args)
    {
        System.out.println("Hello");
        m1();
    }
}
```

Questions

Show Answered Questions

X	Question	Asker	R...
	some one needs req	Govindu Rayapur	
	nooo	Govindu Rayapur	
	no	sateesh y	
	Haha	Deepankaj Yadav	
	accidently may happens	Govindu Rayapur	
	hahahhaha	Shubham Gupta	
	Sir got it...	Deepankaj Yadav	
	yes, got it	sateesh y	
	petrol tank will explode now	Shubham Gupta	

sir in factory method not every time object will be created?

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 12-03-2018

```
D:\durgaclasses>javac Test.java

D:\durgaclasses>java Test
Hello
Exception in thread "main" java.lang.StackOverflowError
        at Test.m2(Test.java:9)
        at Test.m1(Test.java:5)
        at Test.m2(Test.java:9)
        at Test.m1(Test.java:5)
        at Test.m2(Test.java:9)
        at Test.m1(Test.java:5)
        at Test.m2(Test.java:9)
        at Test.m1(Test.java:5)
        at Test.m2(Test.java:9)
        at Test.m1(Test.java:5)
```

CE. Compiler creates default constructor so wrt constructor compiler plays a role.

```
class Test
{
    Test()
    {
        this(10);
    }
    Test(int i)
    {
        this();
    }
    public static void main(String[] args)
    {
        System.out.println("Hello Don't Sleep!!!!");

    }
}
```

```
D:\durgaclasses>javac Test.java
Test.java:7: error: recursive constructor invocation
        Test(int i)
        ^
1 error
```

Parent constuctor by default is not available to child class. Inheitence is not available for const

Line 18 will not call P constructor. This is CE

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 12-03-2018

```
1  class P
2  {
3     P()
4     {
5     }
6  }
7  class C extends P
8  {
9     C(int i)
10    {
11    }
12 }
13 class Test
14 {
15    public static void main(String[] args)
16    {
17       C c= new C(10);
18       C c = new C();
19    }
20 }
21
```

Select C:\Windows\system32\cmd.exe

```
2 errors

D:\durgaclasses>javac Test.java
Test.java:18: error: constructor C in class C cannot be applied to given types;
        C c1 = new C();
               ^
  required: int
  found: no arguments
  reason: actual and formal argument lists differ in length
1 error
```

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 12-03-2018

```
8  super() or this()===>super()
9
10 overloading concept
11 ---------------------
12 Inheritance concept :
13 -------------------
14 Overriding concept
15
16
```

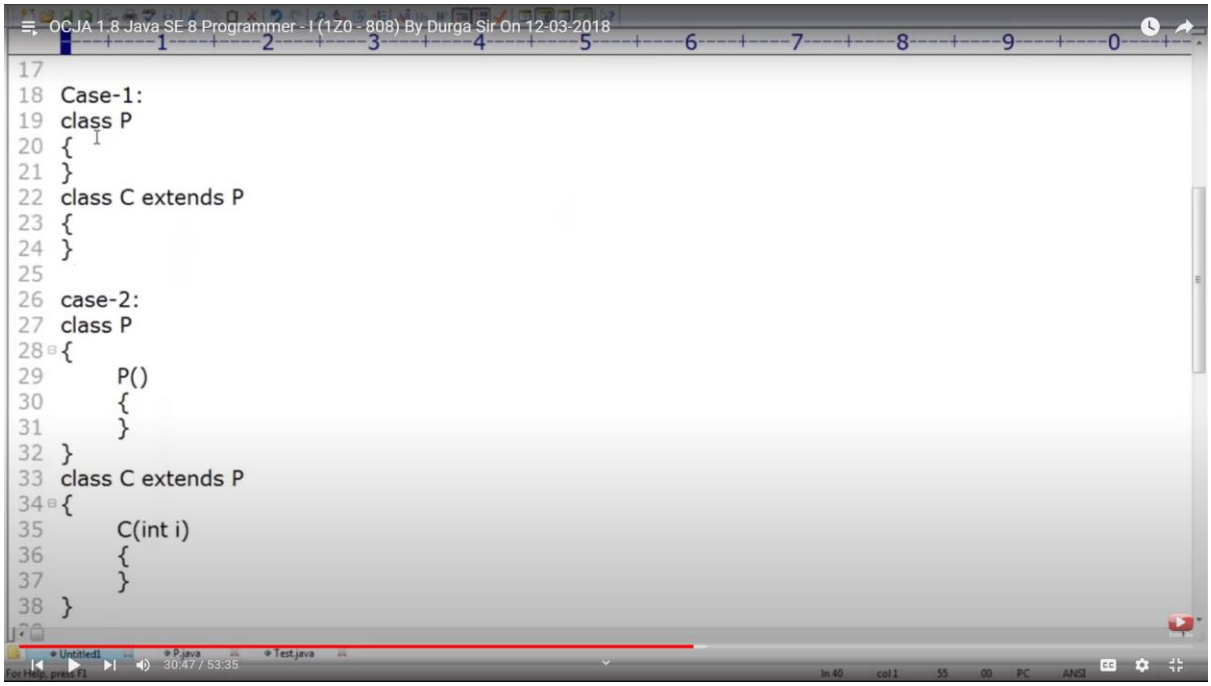
Questions

Show Answered Questions

X	Question	Asker	R...
	s	Pooja Chavan	

In case 1 – The compiler will place Super in both line 20 and 23 , which is valid only

In case 2 – Same as Case 1 No Impact



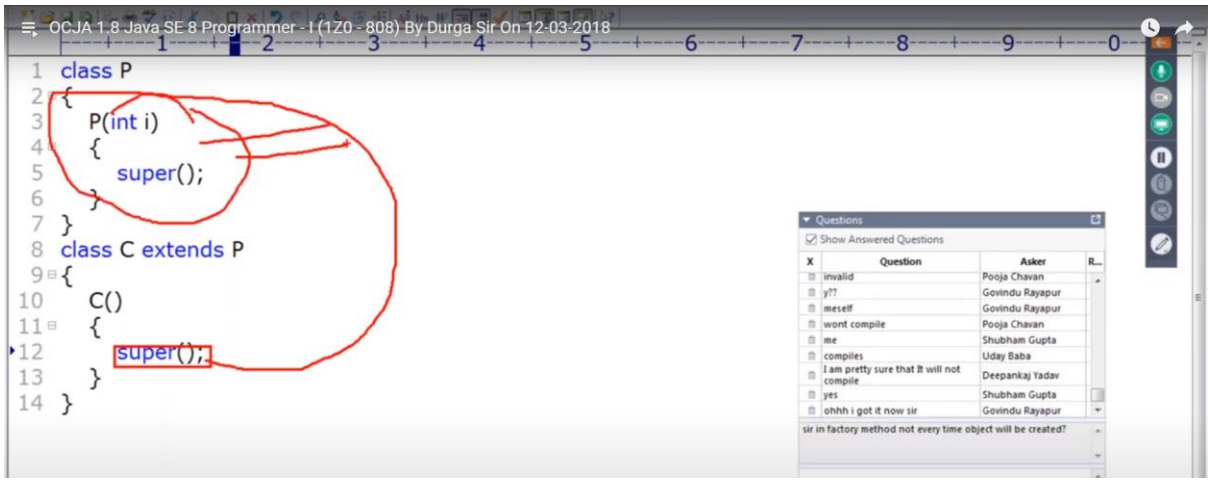
Case 3:

InValid Constructor

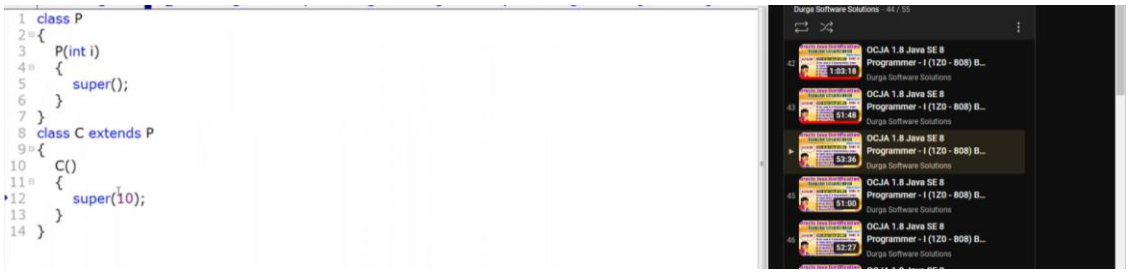
Here line 4 there is no Super or this line so compiler will place it. Since there is user defined constructor available , Compiler would not create a default constructor.

In Line 10 , The constructor has no Super or This, so compiler will place Super(). But the super class has no default constructor it has only 1 user defined constructor expecting an argument so CE

```
D:\durgaclasses>javac P.java
P.java:10: error: constructor P in class P cannot be applied to given types;
        {
        ^
  required: int
  found: no arguments
  reason: actual and formal argument lists differ in length
1 error
```



Solution:1



Solution:2

Always write no arg constructor also

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 12-03-2018

```
class P
{
    P(int i)
    {
        super();
    }
    P()
    {
    }
}
class C extends P
{
    C()
    {
        super();
    }
}
```

Case 4:

If parent class constructor throws any checked exception then child class also should throw the same checked exception or its parent

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 12-03-2018

```
import java.io.*;
class P
{
    P() throws IOException
    {
    }
}
class C extends P
{

}
```

```
D:\durgaclasses>javac P.java

D:\durgaclasses>javac P.java
P.java:8: error: unreported exception IOException in default constructor
class C extends P
^
1 error
```

If a method throws a checked exception then the caller should also handle the exception . Either we have a try catch or throw the exception (throws Exception) or its parent (Throws Throwable)

```

m1() throws Exception
{
}

m2() throws Exception,
{

    m1();

 }
```

Since the Child class constructor is calling the Super constructor it should handle the exception or throw the same or parent exception

```
import java.io.*;
class P
{
    P() throws IOException
    {
    }
}
class C extends P
{

    C()
    {
        super();
    }

}
```

Questions

Show Answered Questions

X	Question	Asker
	yes	Govindu Rayapur
	yes	Shubham Gupta
	s sir	Uday Baba
	caller will handle	Shubham Gupta
	try	Uday Baba
	super()	Govindu Rayapur
	P	Uday Baba
	child class const	Deepankaj Yadav
	child	Pooja Chavan

sir in factory method not every time object will be created?

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 12-03-2018

```
D:\durgaclasses>javac P.java

D:\durgaclasses>javac P.java
P.java:8: error: unreported exception IOException in default constructor
class C extends P
^
1 error
```

```
import java.io.*;
class P
{
    P() throws IOException
    {
    }
}
class C extends P
{

    C()
    {
        try
        {
            super();
        }
        catch (IOException e)
        {
        }

    }
}
```

Untitled1 | P.java | Test.java

For Help, press F1 | ln 17 | col 29 | 22 | 29 | PC | ANSI

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 12-03-2018

```
1  import java.io.*;
2  class P
3  {
4      P() throws IOException
5      {
6      }
7  }
8  class C extends P
9  {
10
11     C() throws IOException
12     {
13         super();
14     }
15 }
```

Parent of IOException -> Exception

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 12-03-2018

```
1  import java.io.*;
2  class P
3  {
4      P() throws IOException
5      {
6      }
7  }
8  class C extends P
9  {
10
11     C() throws Exception
12     {
13         super();
14     }
15 }
```

1.Name of the constructor and Class name should be same

2. We cant declare return type for constructor even Void Also, this will be treated as method

3. We can use only Public Default Private and Protected modifiers for constructor

4. Compiler will always generate default constructor only when there are no user defined constructor else it will not generate default constructor

5. Modifier of default constructor is the modifier of the class (either public or Default)

6. First line inside every constructor should be either Super() or This()

7.If no Super or This is written in the constructor, Compiler will always place Super() in the first line of the constructor

8. Overloading of constructor is applicable for Constructor

9. Inheritance and Overriding is not applicable for Constructors

10. Concrete class and abstract class both can have constructor and default constructors

11. Interfaces cannot contain constructor, because instance variable cannot be there inside interface only static blocks

12. Recursive constructor call in interface is CE handled and throws CE error

13. If Parent class throws some checked exception then child class should throw the same un checked exception or its parent

14. 13. If Parent class throws some unchecked exception then child class can throw or may not throw the unchecked exception