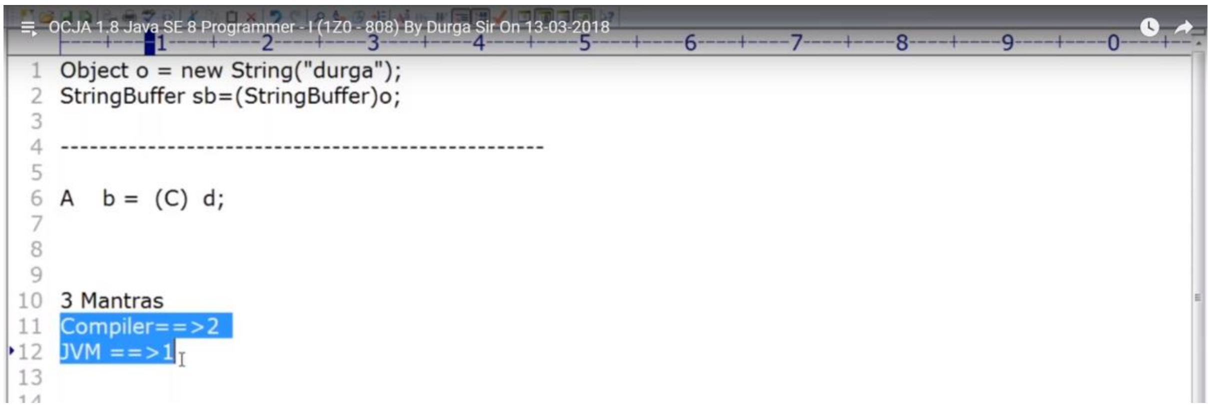
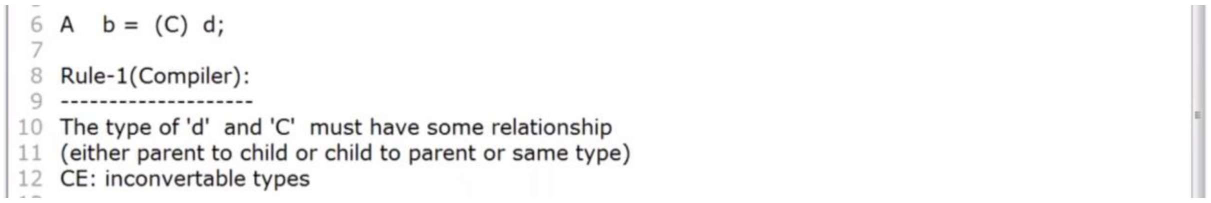
Type Casting


OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 13-03-2018

```
Object o = new String("durga");
StringBuffer sb=(StringBuffer)o;

---------------------------------------------------

A   b = (C) d;



3 Mantras
Compiler==>2
JVM ==>1
```

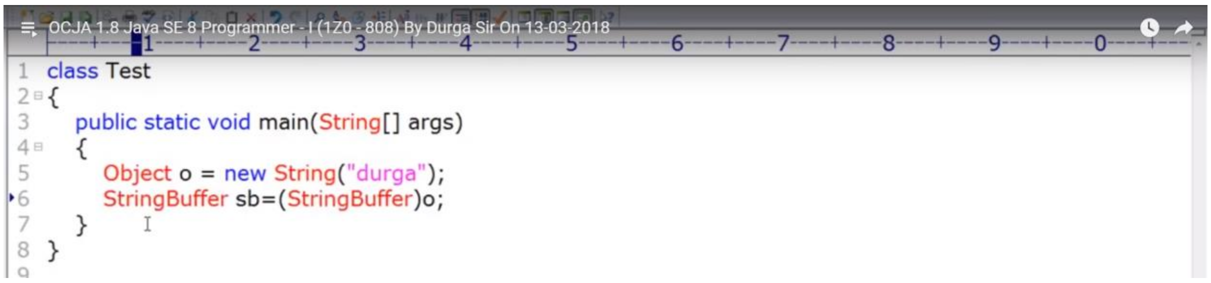
Rule 1- Compiler



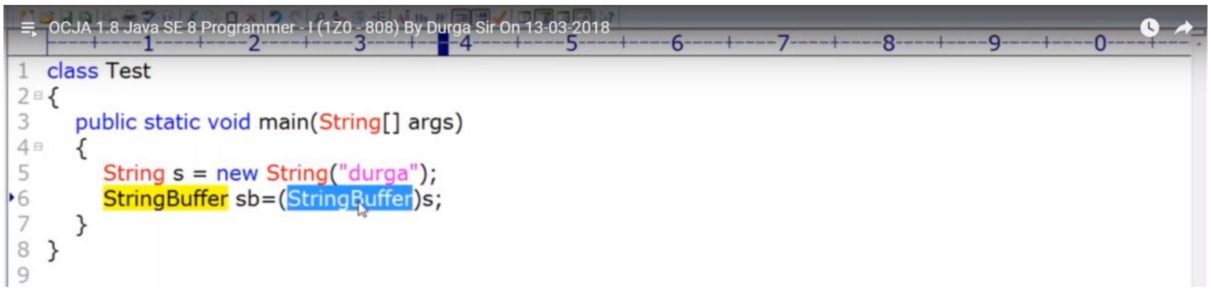
```
A   b = (C) d;

Rule-1(Compiler):
--------------------
The type of 'd'  and 'C'  must have some relationship
(either parent to child or child to parent or same type)
CE: inconvertable types
```

New versions of java throws Incompatible not inconvertible (old versions)

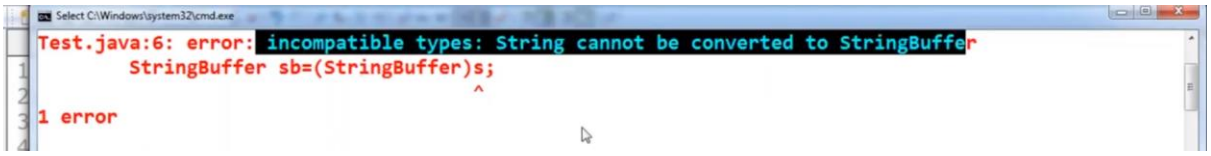
Compile will not throw an error below


OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 13-03-2018

```
class Test
{
   public static void main(String[] args)
   {
      Object o = new String("durga");
      StringBuffer sb=(StringBuffer)o;
   }
}
```

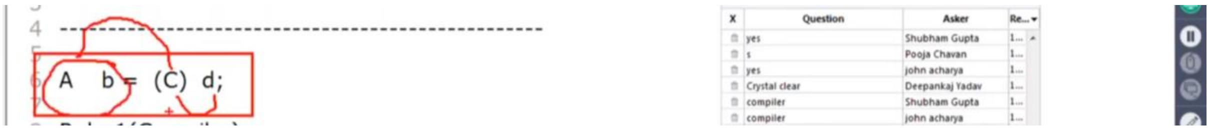

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 13-03-2018

```
class Test
{
   public static void main(String[] args)
   {
      String s = new String("durga");
      StringBuffer sb=(StringBuffer)s;
   }
}
```



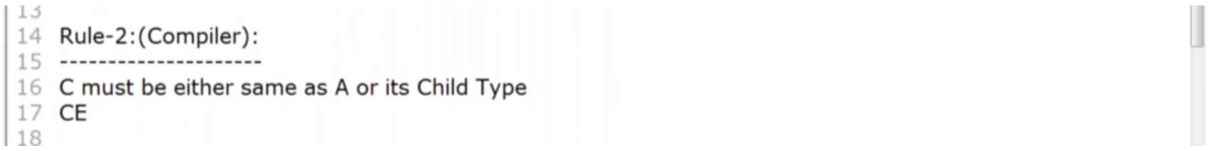
```
Test.java:6: error: incompatible types: String cannot be converted to StringBuffer
        StringBuffer sb=(StringBuffer)s;
                                      ^
1 error
```

Rule2:



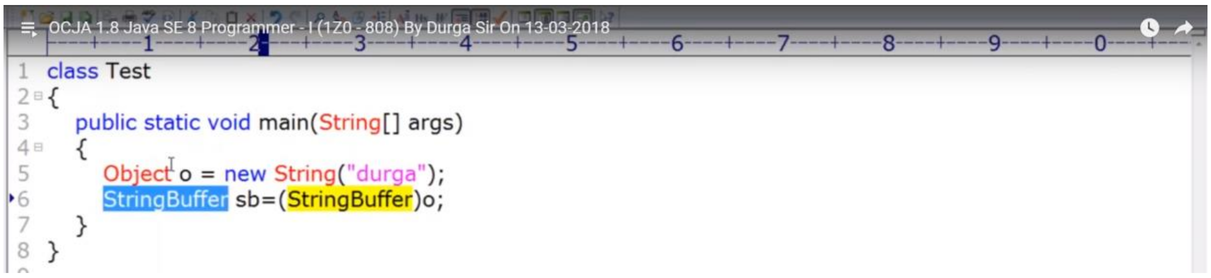
```
-----------------------------------------------
A  b = (C) d;
```

X	Question	Asker	Re...
	yes	Shubham Gupta	1...
	s	Pooja Chavan	1...
	yes	john acharya	1...
	Crystal clear	Deepankaj Yadav	1...
	compiler	Shubham Gupta	1...
	compiler	john acharya	1...



```
Rule-2:(Compiler):
---------------------
C must be either same as A or its Child Type
CE
```

No Error String Buffer is of type String buffer



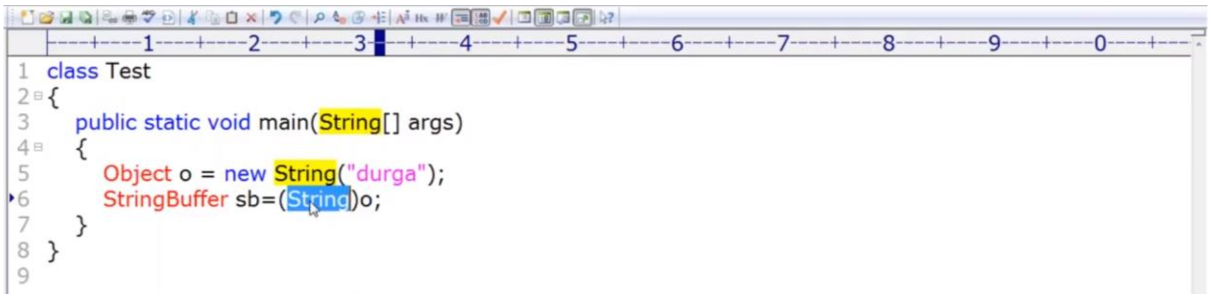
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 13-03-2018

```
class Test
{
   public static void main(String[] args)
   {
      Object o = new String("durga");
      StringBuffer sb=(StringBuffer)o;
   }
}
```

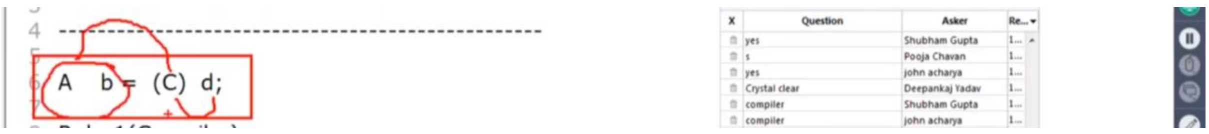
Here Rule 1 Passes

Rule 2 String cannot be converted to StringBuffer

CE- Incompatible Types



```
class Test
{
   public static void main(String[] args)
   {
      Object o = new String("durga");
      StringBuffer sb=(String)o;
   }
}
```



```
A  b = (C) d;
```

X	Question	Asker	Re...
	yes	Shubham Gupta	1...
	s	Pooja Chavan	1...
	yes	john acharya	1...
	Crystal clear	Deepankaj Yadav	1...
	compiler	Shubham Gupta	1...
	compiler	john acharya	1...

Rule 3: Run time Object of d should be the same type as C or derived type of C

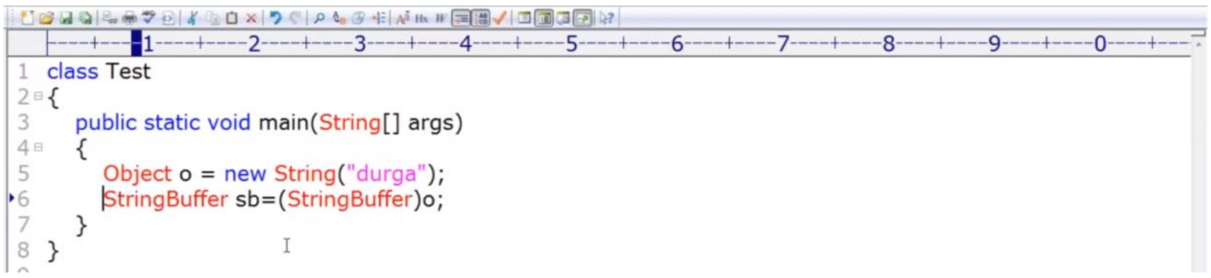
CCE – Class cast Exception

Rule 1- StringBuffer is a child of object – Pass

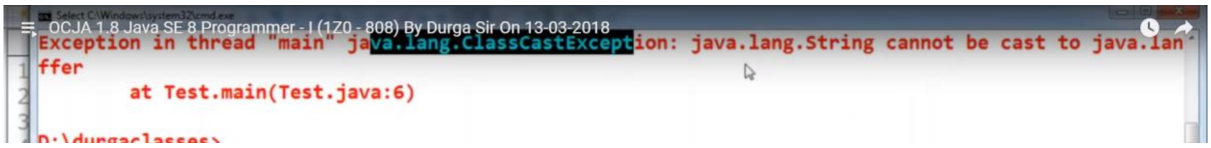
Rule 2: String Buffer is same type as String Buffer - Pass

Rule 3: Run Time Object of O is of type String and has no relationship with StringBuffer – Fail

RE- Class Case Exception

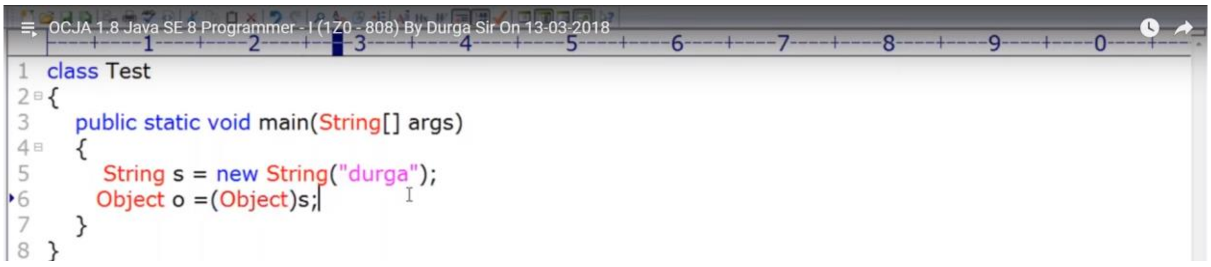


```java
class Test
{
   public static void main(String[] args)
   {
      Object o = new String("durga");
      StringBuffer sb=(StringBuffer)o;
   }
}
```



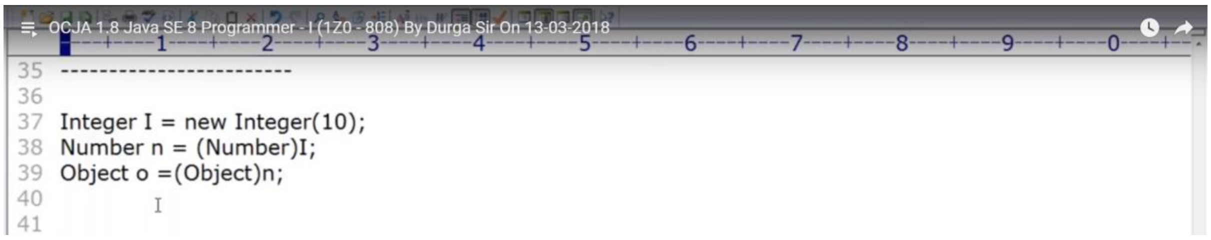
```
Exception in thread "main" java.lang.ClassCastException: java.lang.String cannot be cast to java.lan
ffer
        at Test.main(Test.java:6)

D:\durgaclasses>
```

All Rule Passed



```java
class Test
{
   public static void main(String[] args)
   {
      String s = new String("durga");
     Object o =(Object)s;
   }
}
```

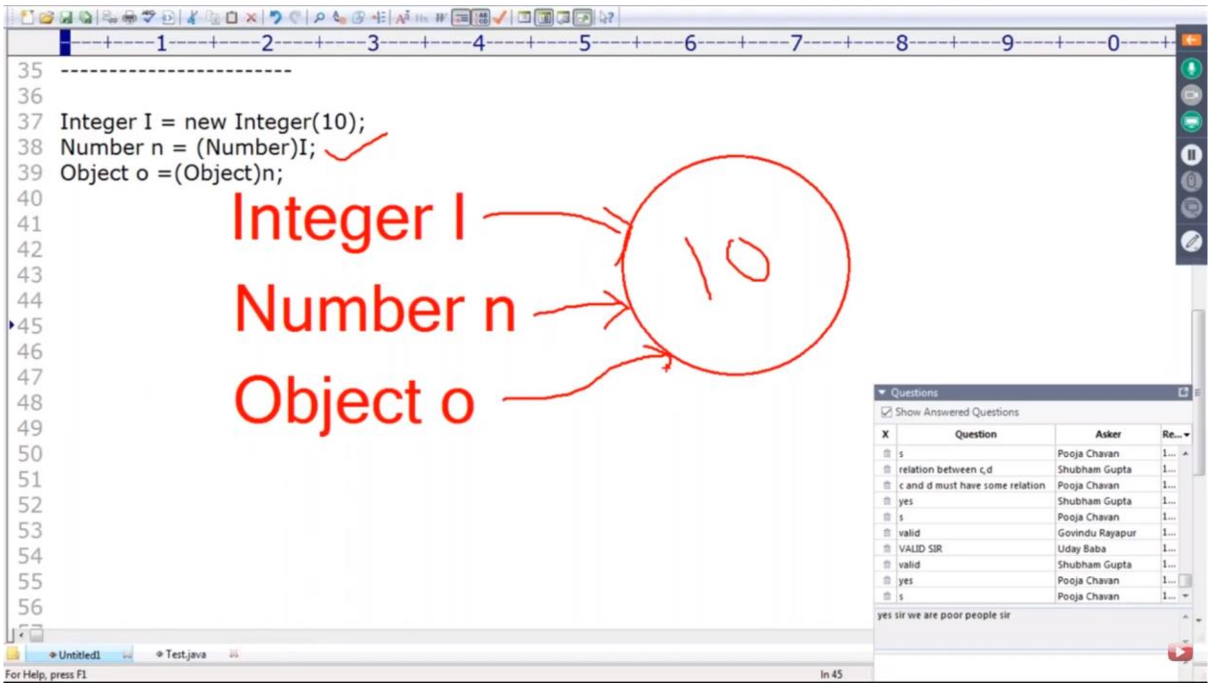
Valid



OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 13-03-2018

```
------------------------

Integer I = new Integer(10);
Number n = (Number)I;
Object o =(Object)n;
```

Here when we are typecasting we are not creating new objects we are creating new references only



```
------------------------

Integer I = new Integer(10);
Number n = (Number)I;
Object o =(Object)n;
```

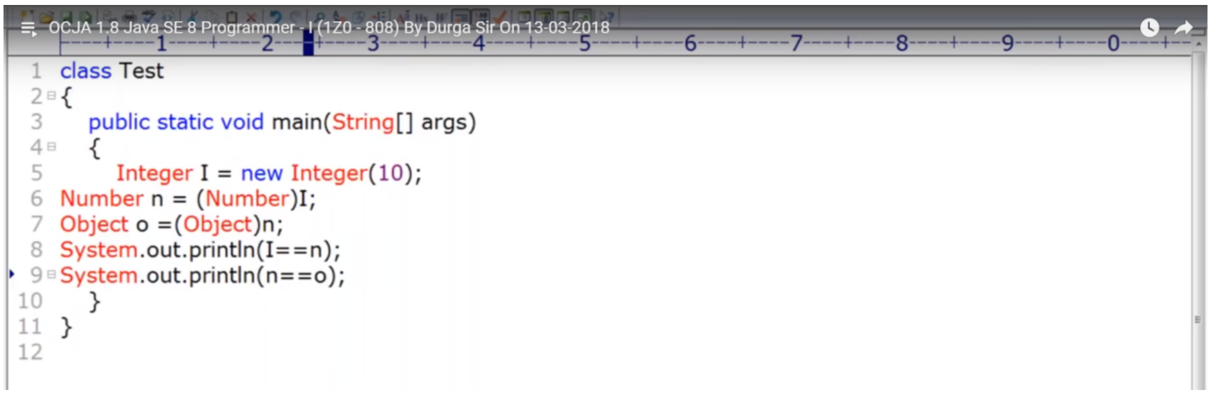
Questions

☑ Show Answered Questions

X	Question	Asker	Re...
	s	Pooja Chavan	1...
	relation between c,d	Shubham Gupta	1...
	c and d must have some relation	Pooja Chavan	1...
	yes	Shubham Gupta	1...
	s	Pooja Chavan	1...
	valid	Govindu Rayapur	1...
	VALID SIR	Uday Baba	1...
	valid	Shubham Gupta	1...
	yes	Pooja Chavan	1...
	s	Pooja Chavan	1...

yes sir we are poor people sir

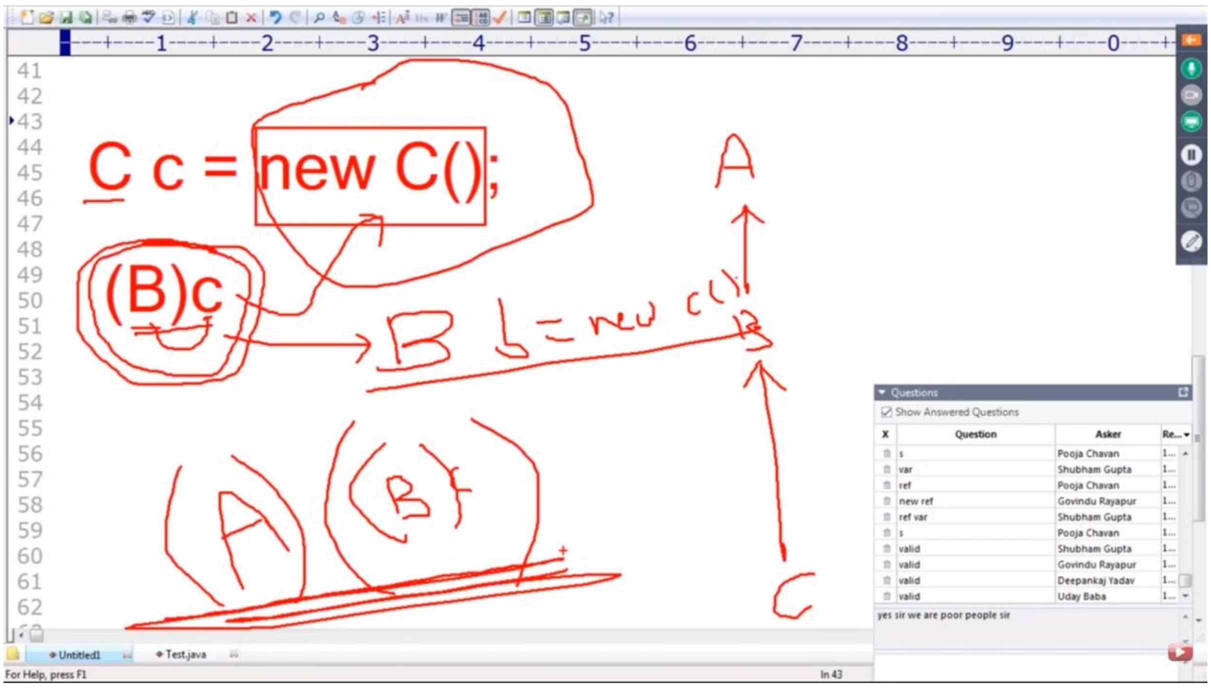
Check if all Refs variables points to same object reference



OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 13-03-2018

```
class Test
{
   public static void main(String[] args)
   {
      Integer I = new Integer(10);
Number n = (Number)I;
Object o =(Object)n;
System.out.println(I==n);
System.out.println(n==o);
   }
}
```

Here finally run time object is C . But the final reference variable is A



```
C c = new C();
(B)c
```

Questions

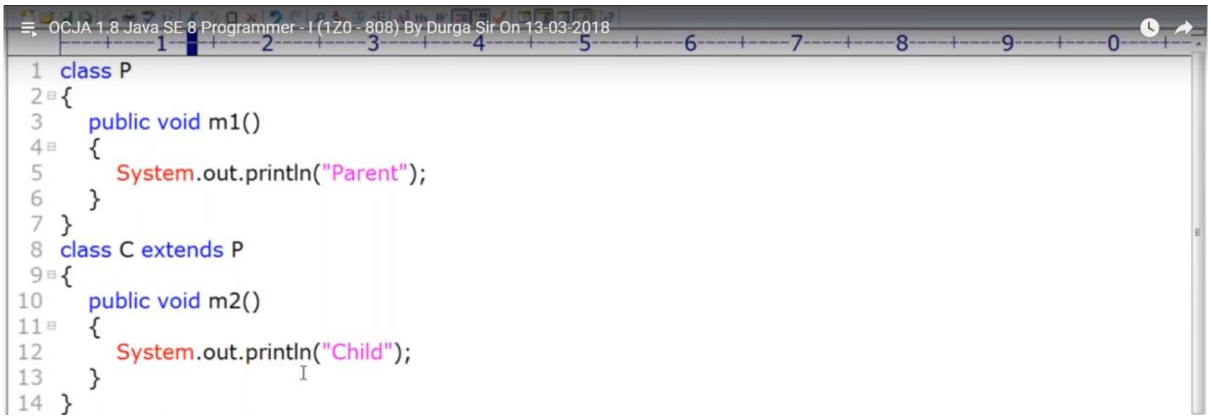
Show Answered Questions

X	Question	Asker	Re...
	s	Pooja Chavan	1...
	var	Shubham Gupta	1...
	ref	Pooja Chavan	1...
	new ref	Govindu Rayapur	1...
	ref var	Shubham Gupta	1...
	s	Pooja Chavan	1...
	valid	Shubham Gupta	1...
	valid	Govindu Rayapur	1...
	valid	Deepankaj Yadav	1...
	valid	Uday Baba	1...

yes sir we are poor people sir

Untitled1 Test.java

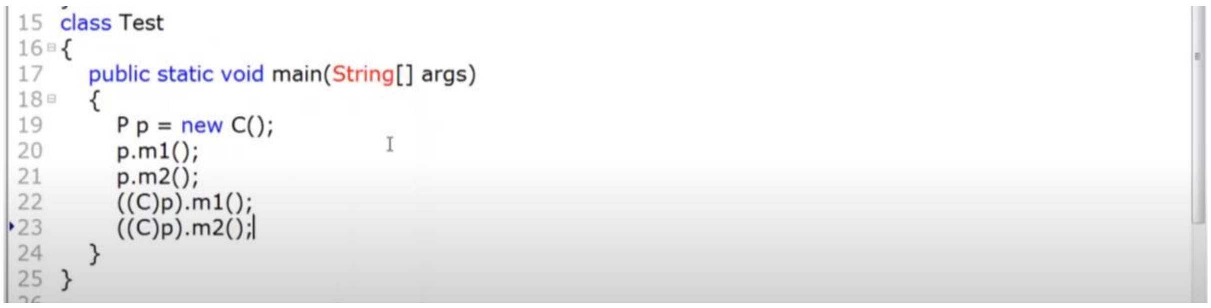
For Help, press F1 ln 43


OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 13-03-2018

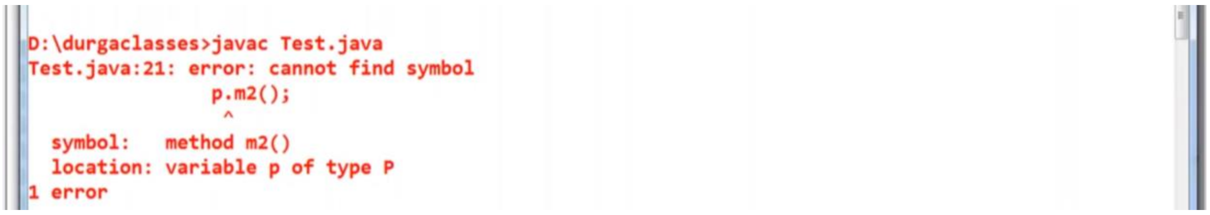
```
1  class P
2  {
3     public void m1()
4     {
5        System.out.println("Parent");
6     }
7  }
8  class C extends P
9  {
10    public void m2()
11    {
12       System.out.println("Child");
13    }
14 }
```

Here P.m2() is invalid , Parent ref cant child methods

Line 22 and 23 are valid bcos run time object C is converted to C reference . Hence it can call both parent and child methods

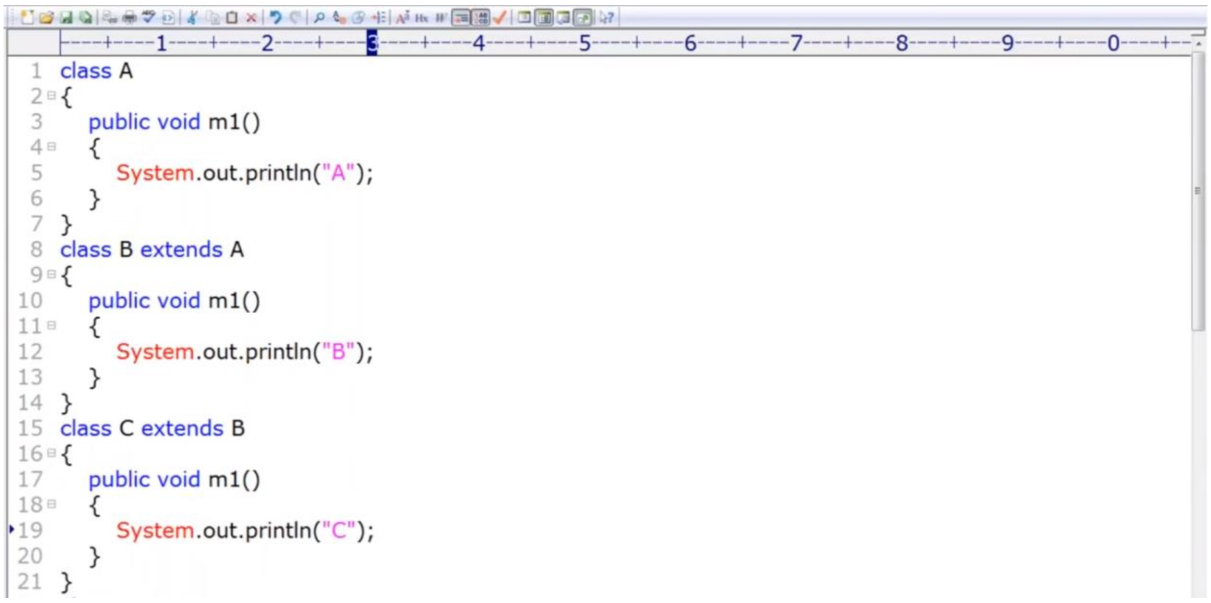


```
15 class Test
16 {
17    public static void main(String[] args)
18    {
19       P p = new C();
20       p.m1();
21       p.m2();
22       ((C)p).m1();
23       ((C)p).m2();
24    }
25 }
```



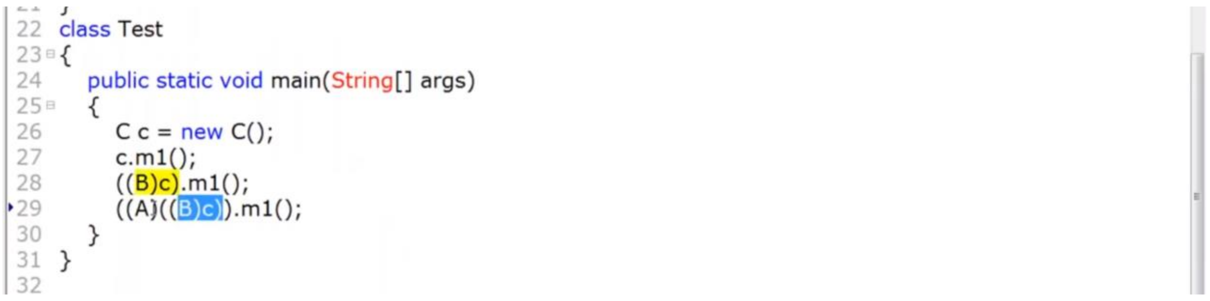
```
D:\durgaclasses>javac Test.java
Test.java:21: error: cannot find symbol
                p.m2();
                 ^
  symbol:   method m2()
  location: variable p of type P
1 error
```

Overriding – Method resolution is always taken care by JVM Runtime



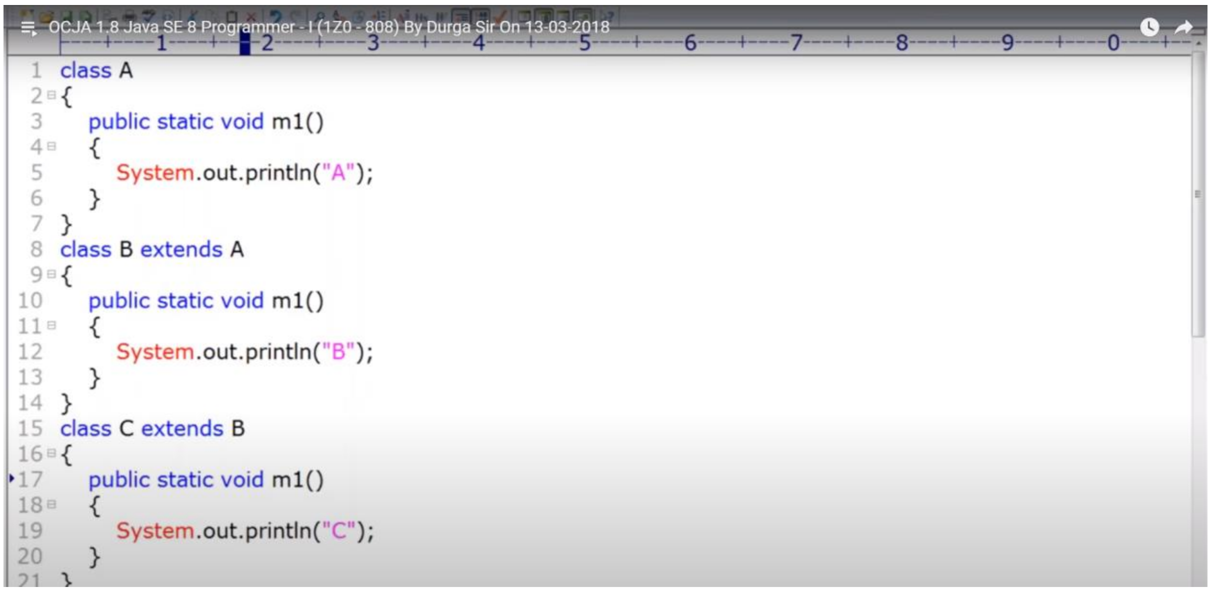
```
class A
{
   public void m1()
   {
      System.out.println("A");
   }
}
class B extends A
{
   public void m1()
   {
      System.out.println("B");
   }
}
class C extends B
{
   public void m1()
   {
      System.out.println("C");
   }
}
```

Object is internally of type c. Ref variable changes . In Overriding JVM resolves method call. Always C class method will be called . JVM only gives pref to child class methods



```
class Test
{
   public static void main(String[] args)
   {
      C c = new C();
      c.m1();
      ((B)c).m1();
      ((A)((B)c)).m1();
   }
}
```

Method Hiding- Method Resolution is always taken care by Reference Type

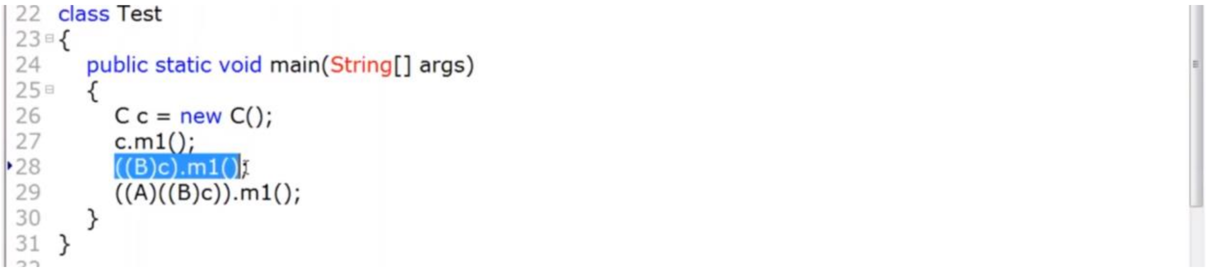

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 13-03-2018

```
1  class A
2  {
3     public static void m1()
4     {
5        System.out.println("A");
6     }
7  }
8  class B extends A
9  {
10    public static void m1()
11    {
12       System.out.println("B");
13    }
14 }
15 class C extends B
16 {
17    public static void m1()
18    {
19       System.out.println("C");
20    }
21 }
```

Line 27- C class method – Refernce variable is of type C

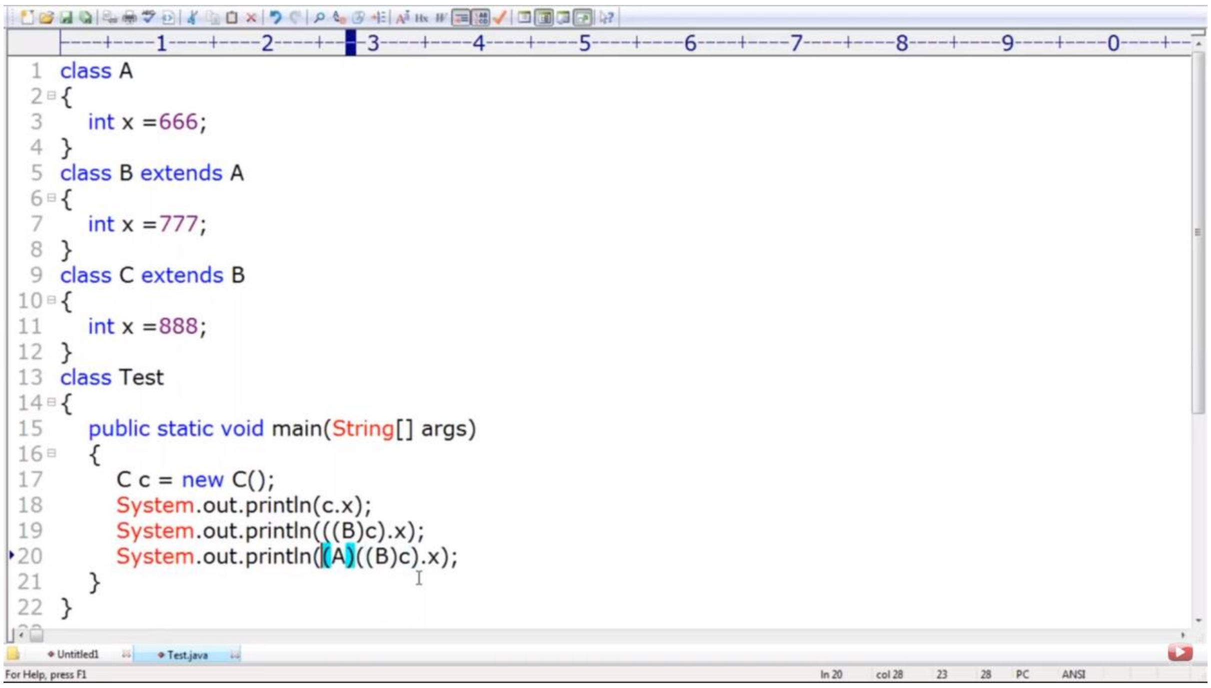
Line 28- Refernce variable is o type B . B method will be called

Line 29- Ref variable is of type A , A method will be called

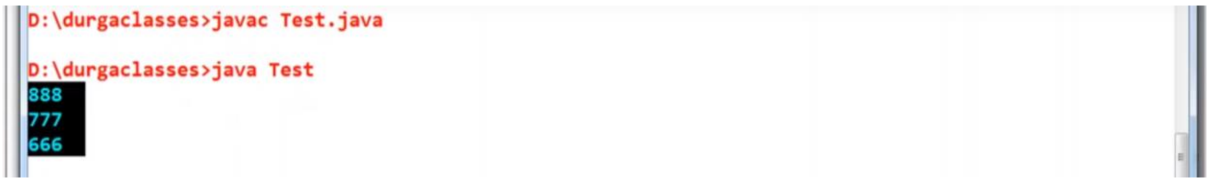


```
22 class Test
23 {
24    public static void main(String[] args)
25    {
26       C c = new C();
27       c.m1();
28       ((B)c).m1();
29       ((A)((B)c)).m1();
30    }
31 }
```

Overriding concepts is not applicable to instance variables only by methods. Variable assignment is taken care by Compiler->



```
class A
{
   int x =666;
}
class B extends A
{
   int x =777;
}
class C extends B
{
   int x =888;
}
class Test
{
   public static void main(String[] args)
   {
      C c = new C();
      System.out.println(c.x);
      System.out.println(((B)c).x);
      System.out.println((A)((B)c).x);
   }
}
```



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

D:\durgaclasses>java Test
888
777
666
```