

abstract class vs abstract method:

- 1. If the class contains atleast one abstract method then compulsory we should declare class as abstract.
- 2. An abstract class can contains zero number of abstract methods also

what happen if we create instance of another class in abstract class?

I

abstract vs final:

- 1. final method abstract method
- 2. final class abstract class
- 3. final class can contain abstract methods
- 4. abstract class can contain final methods

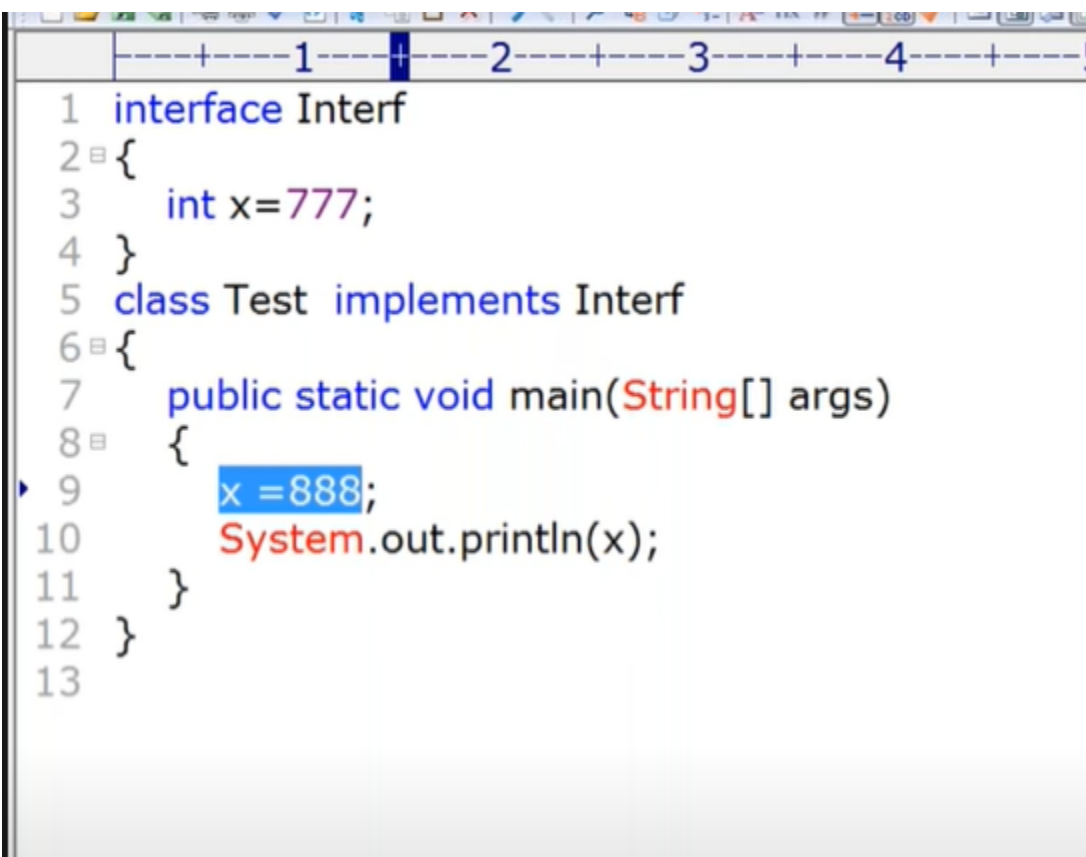


Video31:

```
1 package pack1;
2 public class A
3 {
4     protected void m1()
5     {
6         System.out.println("Hello it is protected method");
7     }
8 }
9 class C extends A
10 {
11     public static void main(String[] args)
12     {
13         A a = new A();
14         a.m1();
15
16         C c = new C();
17         c.m1();
18
19         A a1 = new C();
20         a1.m1();
21     }
22 }
```

Protected member inside the package everywhere valid

```
1 interface Interf
2 {
3     int x=777;
4 }
5 class Test implements Interf
6 {
7     public static void main(String[] args)
8     {
9         int x =888;
10        System.out.println(x);
11    }
12 }
13
```



The screenshot shows the same Java code as the first image, but with a ruler at the top of the editor window. The ruler has markings for 1, 2, 3, and 4. A blue selection box highlights the text 'x = 888;' on line 9. The code is as follows:

```
1 interface Interf
2 {
3     int x=777;
4 }
5 class Test implements Interf
6 {
7     public static void main(String[] args)
8     {
9         x =888;
10        System.out.println(x);
11    }
12 }
13
```

```
1 interface Left
2 {
3     public void m1();
4 }
5 interface Right
6 {
7     public int m1();
8 }
9 class Test implements Left, Right
10 {
11     public void m1()
12     {
13     }
14     public int m1()
15     {
16         return 10;
17     }
18 }
19
```

45:51 / 59:23

```

1 interface Left
2 {
3     public void m1();
4 }
5 interface Right
6 {
7     public int m1();
8 }
9 abstract class Test implements Left, Right
10 {
11     public void m1()
12     {
13     }
14 }
15 class SubTest extends Test
16 {
17     public int m1() {
18     {
19         return 10;
20     }
21 }
22

```

```

1 class Test implements Cloneable
2 {
3     public static void main(String[] args) throws Exception
4     {
5         Test t = new Test();
6         Test t1 = (Test)t.clone();
7     }
8 }
9

```

Marker interface cloneable

```
1 interface X
2 {
3     m1();
4     m2();
5     default m3()
6     {
7         dummy implementation
8     }
9
10 }
11 class Test1 implements X
12 {
13     m1(){}
14     m2(){}
15 }
16 class Test2 implements X
17 {
18     m1(){}
19     m2(){}
20 }
21
```