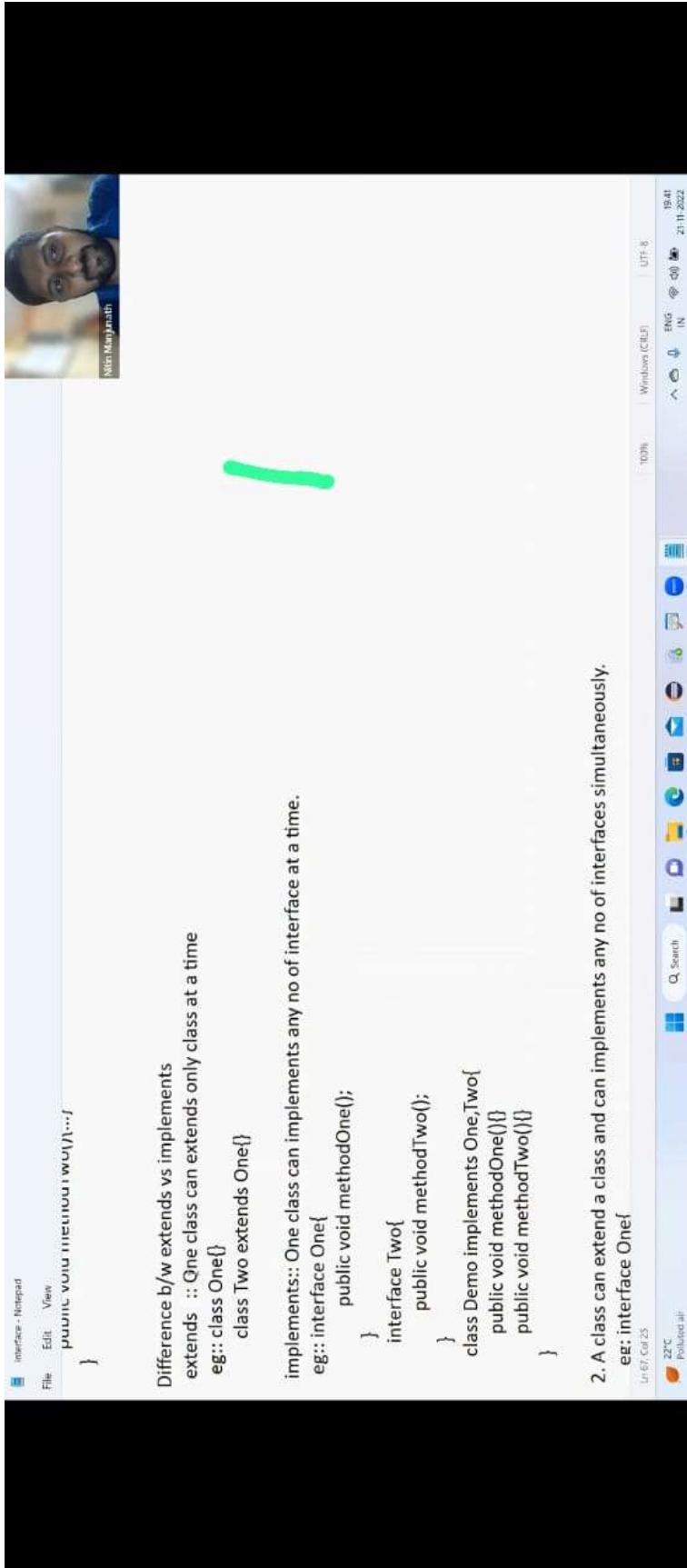


Java Interfaces Part3



Nitin Manjrekar

Difference b/w extends vs implements
extends :: One class can extends only class at a time
eg:: class One{}
class Two extends One{}

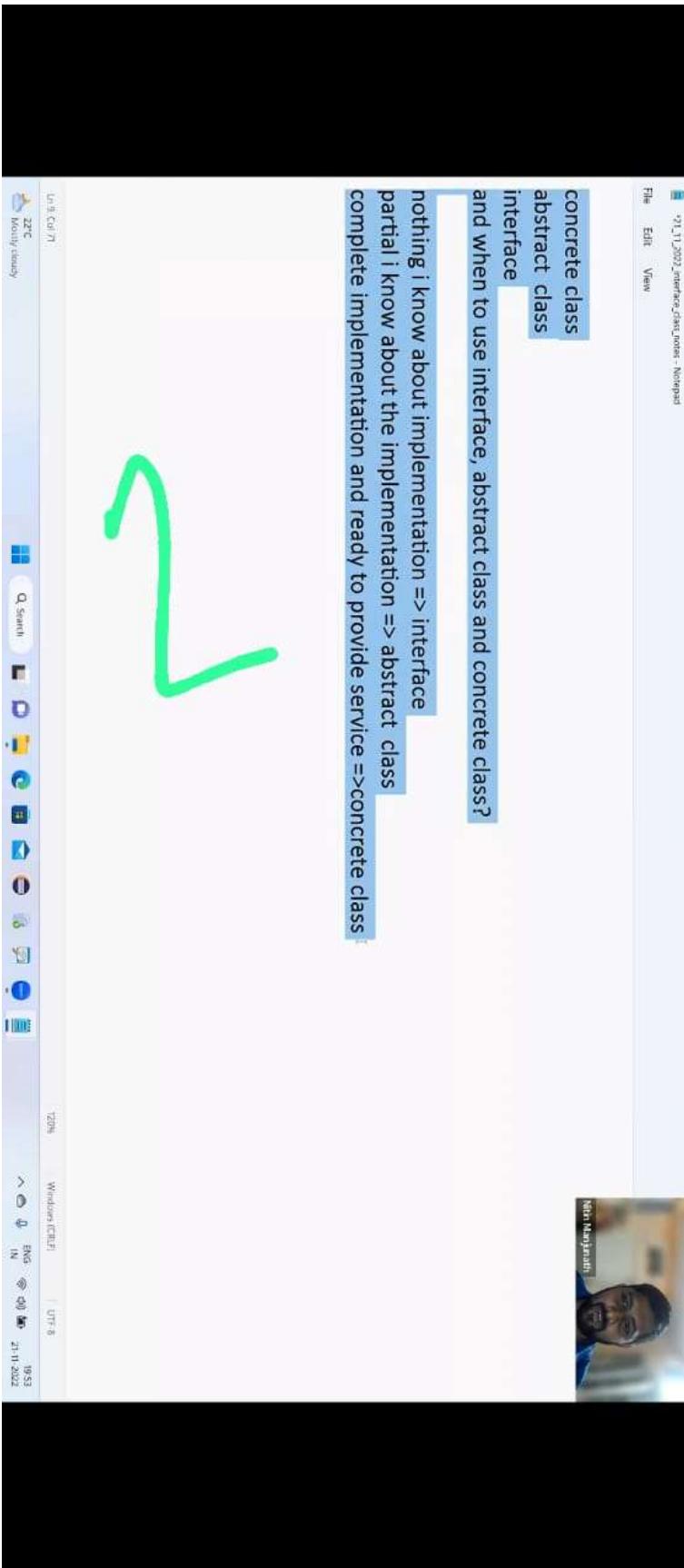
implements:: One class can implements any no of interface at a time.
eg:: interface One{
 public void methodOne();
}
interface Two{
 public void methodTwo();
}
class Demo implements One,Two{
 public void methodOne(){
 }
 public void methodTwo(){
 }
}

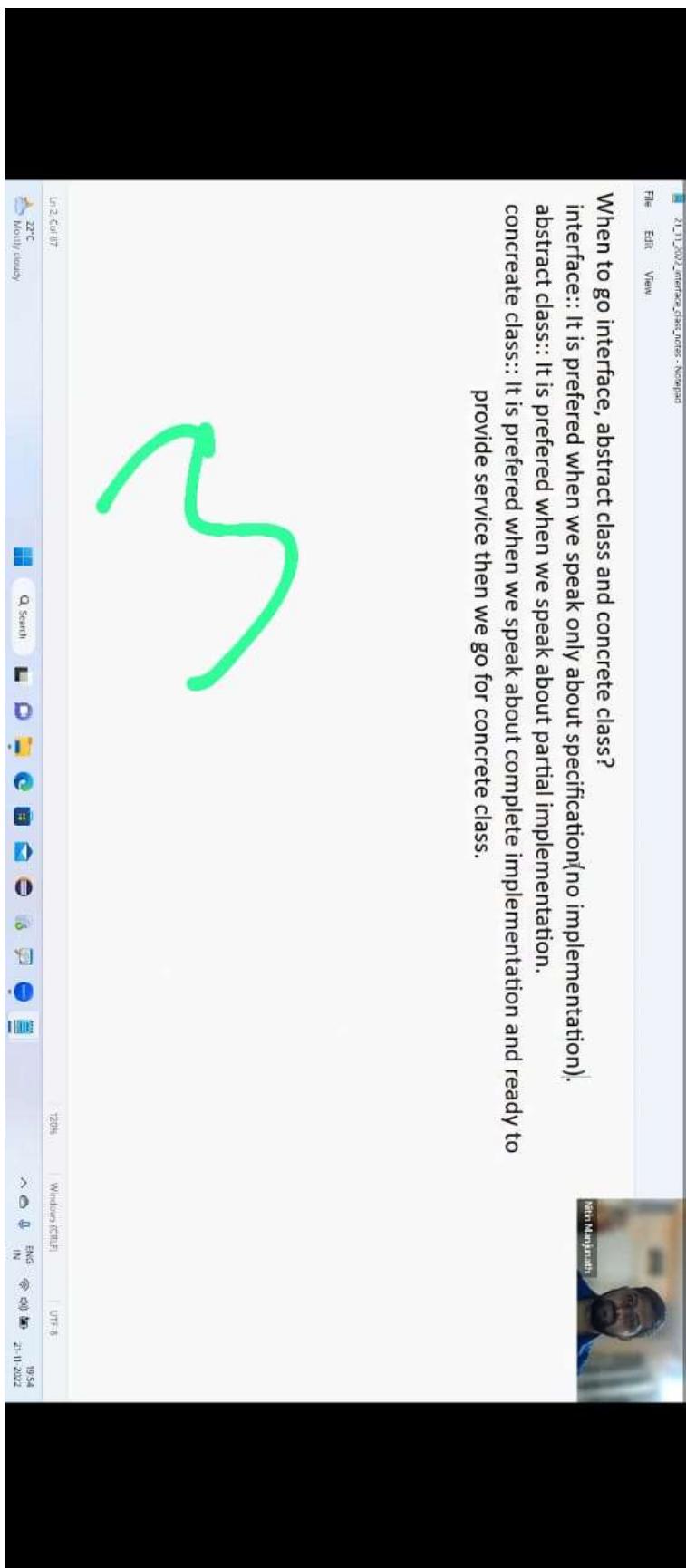
2. A class can extend a class and can implements any no of interfaces simultaneously.

eg: interface One{
 }

File Edit View
private void methodOne(){...;}

10% Windows (C:\UF) UTF-8
22°C 21/11/2022





When to go interface, abstract class and concrete class?

- interface:: It is preferred when we speak only about specification(no implementation).
- abstract class:: It is preferred when we speak about partial implementation.
- concrete class:: It is preferred when we speak about complete implementation and ready to provide service then we go for concrete class.

Difference b/w interface and abstract class?
interface => 100% abstraction

private,static,strictfp,synchronized,native methods not possible
public static final

abstract => not 100% abstraction

`private,static,strictfp,synchronized,native` methods not possible
need not be public static final

22°C
Cloudy

卷之二

T20% Windows (CRLF) UTR-5

27.11.2022_interface_class.mnw - Notepad

File Edit View

Difference b/w interface and abstract class?

interface => 100% abstraction

public abstract

private,static,strictfp,synchronized,native methods not possible

public static final

variable initialization should be at the time of declaration

abstract class => not 100% abstraction

need not be public and abstract

private,static,strictfp,synchronized,native methods possible

need not be public static final

variable initialization can be at any place

static block,instance block and construc



Ln 21 Col 51
22°C
mostly cloudy



125% Windows (CEP) UTF-8

EN

IN

21.11.1959

21_11_2022_interface_class_notes - Notepad

abstract class:: it is preferred when we speak about partial implementation.
concrete class:: it is preferred when we speak about complete implementation and ready to provide service then we go for concrete class.

Difference b/w interface and abstract class?
interface => 100% abstraction

public abstract
private static st

private,static,strictfp,synchronized,native methods not possible
public static final

variable initialization should be at the time of declaration

`abstract class` => not 100% abstraction

need not be public and abstract

private, static, strictfp, synchronized, native methods possible

variable initialization can be at any place

static block,instance block and constructor



21-11-2022_interface_class_notes - Notepad
File Edit View

concreate class:: It is prefered when we speak about complete implementation and ready to provide service then we go for concrete class.



Difference b/w interface and abstract class?
Interface => 100% abstraction

public abstract
private,static,strictfp,synchronized,native methods not possible
public static final

variable initialization should be at the time of declaration.

No need of constructor,instance block and static block

abstract class => not 100% abstraction

need not be public and abstract.
private,static,strictfp,synchronized,native methods possible.
need not be public static final.

variable initialization can be at any place.

We can have static block,instance block and constructor.

ctrl+shift + ~ clear file headings command and used to initialize static variables
Ln 26 Col 1
22°C
mostly cloudy



100% Windows (CET) UTF-8

21-11-2022

File Edit View
Difference b/w interface and abstract class?
www.Manjusha.in

Difference b/w interface and abstract class?
Interface:: If we dont know anything about implementation just we have requirement specification then we should go for interface.

Abstract class: If we are talking about implementation but not completely then we should go for abstract class.

Interface: Every method present inside the interface is always public and abstract whether we are declaring or defining it.

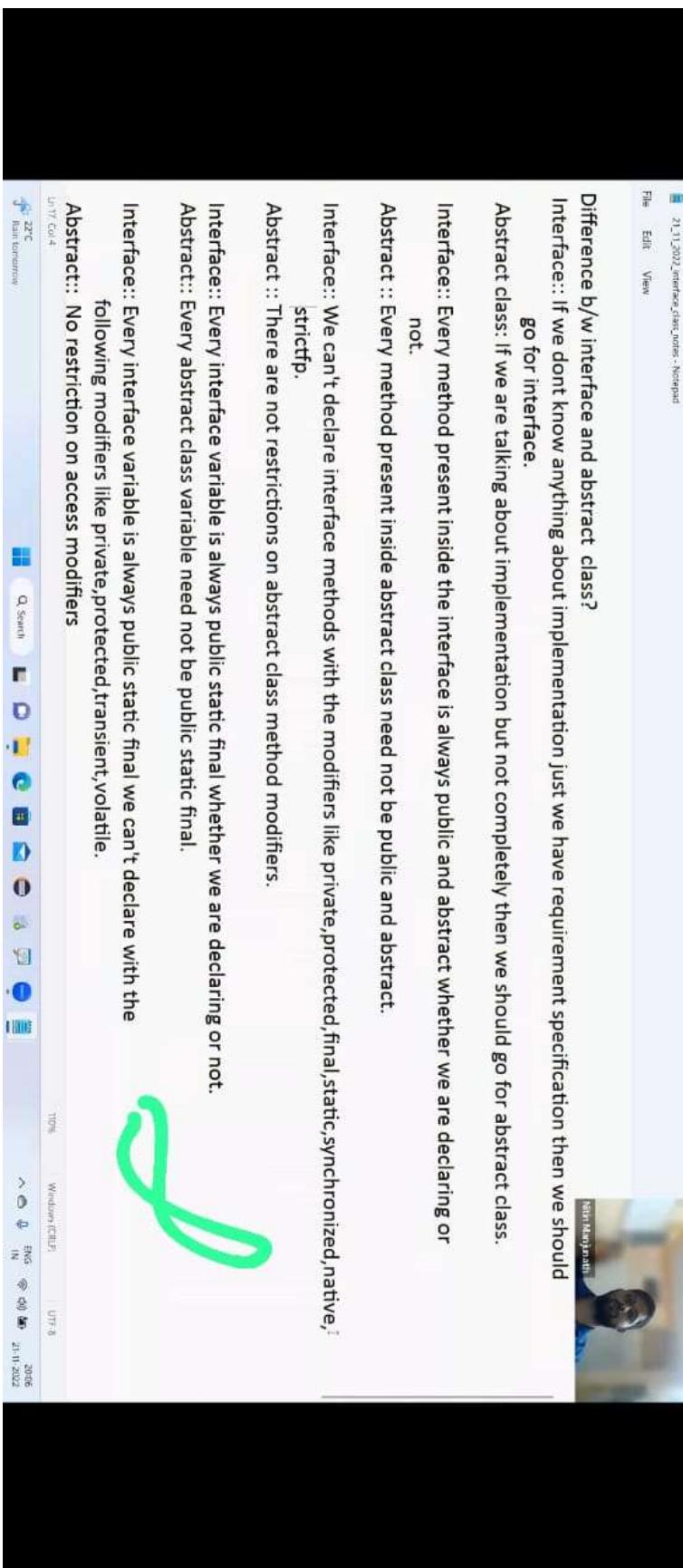
Abstract :: Every method present inside abstract class need not be public and abstract.

Interface: We can't declare interface methods with the modifiers like private,protected,final,static,synchronized,native;

Abstract :: There are not restrictions on abstract class method modifiers.

Abstract: Every abstract class variable need not be public static final.

Abstract: No restriction on access modifiers following modifiers like private, protected, etc.



File Edit View

21-11-2022_interface_class_notes - Notepad

Interface:: We can't declare interface methods with the modifiers like private,protected,final,static,synchronized.

Abstract :: There are not restrictions on abstract class method modifiers.

Interface:: Every interface variable is always public static final whether we are declaring or not.

Abstract:: Every abstract class variable need not be public static final.

Interface:: Every interface variable is always public static final we can't declare with the following modifiers like private,protected,transient,volatile.

Abstract:: No restriction on access modifiers

Interface:: For every interface variable compulsorily we should perform initialisation at the time of declaration, otherwise we get compile time error.

Abstract:: Not required to perform initialisation for abstract class variables at the time of declaration.

Interface:: Inside interface we can't write static and instance block.

Abstract :: Inside abstract class we can write static and instance block.

Interface:: Inside interface we can't write constructor.



Ur 28 Cof 10 22°C, mostly cloudy Q Smart Windows (CE) 10% Windows (CE) 10% 21-11-2022

following modifiers like private,protected,transient,volatile.

Abstract:: No restriction on access modifiers

Interface:: For every interface variable compulsorily we should perform initialisation at the time of declaration, otherwise we get compile time error.

Abstract:: Not required to perform initialisation for abstract class variables at the time of declaration.

Interface:: Inside interface we can't write static and instance block.

Abstract :: Inside abstract class we can write static and instance block.

Interface:: Inside interface we can't write constructor.

Abstract :: Inside abstract class we can write constructor.

Note:

static block => .class file loading happens and used to initialize static variables.

instance block => during the creation of an object,just before the constructor call used for initialization instance variable.

constructor => during the creation of an object, used for initialization instance variable.



Nitin Manohar

```
1 //During the child class object creation, only Child class Object will be created
2 // but no the parent class object(still constructor of parent is called to bring the
3 // properties of parent to child)
4
5 class Parent
6 {
7     Parent()
8     {
9         System.out.println("Parent class constructor");
10    }
11 }
12 class Child extends Parent
13 {
14     Child()
15     {
16         System.out.println("child class constructor");
17     }
18 }
19 public class Test
20 {
21     public static void main(String[] args)
22 }
```

java (1.java) Test.java

For Help press F1

22°C Mostly cloudy

File Edit View Search Document Project Tools Browser Format Window Help

Directory Client Functions

D:\ New Volume

1.DH SRECCEDIN CORALACTIVE Encapsulation enterprisebatch FieldPrograms Front end videos gitHub android native

NetBeans - (1.java)

Erkhan - DVTest.java

File Edit View Search Document Project Tools Browser Ethernet Window Help

Directory Object Functions

D: New Volume S E C O N D E B I N C o m - h a p t i c a E n u m e r a t i o n s e n t e r p r i s e B a s e e n t e r p r i s e F i l e s P r o g r a m s F r o n t e n d v i d e o s g r a f i c s r e m o t e

1 System.out.println(this.hashCode());
2 }
3 }
4 }
5 }
6 }
7 }
8 }
9 }
10 class Child extends Parent
11 {
12 Child()
13 {
14 System.out.println("child class constructor");
15 }
16 }
17 }
18 public class Test
19 {
20 public static void main(String[] args)
21 {
22 Child c = new Child();
23 System.out.println(c.hashCode());
24 }
25 }
26 }

Java (Java) Test.java For Help (javac) 22°C Mozilla Firefox In col. 40 26 PC ANSI 21.11.2017

12

The screenshot shows a Java IDE interface with the following details:

- File Bar:** File Edit View Search Document Project Tools Browser Ethernet Window Help
- Recent Projects:** Test.java
- Recent Files:** Test.java
- Search Bar:** Q. Search
- Code Area:**

```
// Can abstract class be instantiated/object be created? ans. NO
// Can abstract class contains constructor? ans. Yes

class Parent
{
    Parent()
    {
        System.out.println("Parent class constructor");
        System.out.println(this.hashCode());
    }
}

class Child extends Parent
{
    Child()
    {
        System.out.println("Child class constructor");
        System.out.println(this.hashCode());
    }
}
```
- Code Navigator:** Shows the current file "Test.java" and its package "java".
- Output Area:** Shows the output of the code execution.
- Bottom Status Bar:** Shows the current line (ln 3), column (col 55), and row (row 31). It also includes icons for PC, ANSI, and a date/time stamp (27-Nov-2022).
- Right Panel:** Shows a small portrait of a person named Nitin Mangalath.

QUESTION

Can abstract class be instantiated/object be created? ans. No

```
1 // Can abstract class be instantiated/object be created? ans. No
2 // Can abstract class contains constructor? ans. Yes
3
4
5 abstract class Person
6 {
7     String name;
8     Integer age;
9     Float height;
10
11    Person(String name, Integer age,Float height){
12        this.name = name;
13        this.age = age;
14        this.height = height;
15    }
16
17
18 class Student extends Person
19 {
20     Integer sid;
21     Float marks;
22 }
```

ANSWER

14



```
1 Person<String> p = new Person("Rishabh", 20, 180);
2 System.out.println(p);
3
4 class Person {
5     String name;
6     Integer age;
7     Float height;
8
9     public Person(String name, Integer age, Float height) {
10        this.name = name;
11        this.age = age;
12        this.height = height;
13    }
14
15    @Override
16    public String toString() {
17        return "Person{" +
18            "name=" + name +
19            ", age=" + age +
20            ", height=" + height +
21            '}';
22    }
23 }
24
25 public class Main {
26     public static void main(String[] args) {
27         Person p = new Person("Rishabh", 20, 180);
28         System.out.println(p);
29     }
30 }
```

File Edit View Insert Project John Sonwane Runner Window Help

D:\New Volume

10 Person(String name, Integer age, Float height){

11 this.name = name;

12 this.age = age;

13 this.height = height;

14 }

15 }

16 }

17 }

18 class Student extends Person

19 {

20 Integer sid;

21 Float marks;

22 String courseName;

23 }

24

25 Student(String name, Integer age, Float height,

26 Integer sid, Float marks, String courseName){

27 super(name, age, height);

28 this.sid = sid;

29 this.marks = marks;

30 }

31

32

33 (1.java)

34 ● TestJava

35 22°C

Mon 11 Nov 2022

Q Search

Non-Delimited

```
File Edit View  
interface - Notepad  
  
this.age=age;  
this.height=height;  
this.weight=weight;  
}  
  
class Student extends Person{  
    int rollno;  
    int marks;  
  
    Student(String name,int age,int height,int weight,int rollno,int marks){  
        super(name,age,height,weight,rollno);  
        this.rollno=rollno;  
        this.marks=marks;  
    }  
}
```

16



3:04 PM

```
File Edit View
public static void main(String[] args) {
    Child c = new Child();
    System.out.println(c.hashCode());
}
```

Why abstract class can contain constructor whereas interface does not contain constructor?

abstract class => it is used to perform initialization of the object.

it is used to provide the value for the instance variable.

it is used to contain instance variables which are required for child object to perform initialization for those instance variables.

interface => every variable is always static, public and final, there is no chance of existing instance variable inside the class.

so we should perform initialization at the time of declaration.

so constructor is not required for interface.

eg#1.

```
abstract class Person{}
```

10% Windows (CEP) | UTC+5

22°C mostly cloudy

The screenshot shows a Java code editor with the following code:

```
1 // Can abstract class be instantiated/object be created? ans. No
2 // Can abstract class contains constructor? ans. Yes
3 // Can interface object be instantiated? ans. No
4 // Can interface contains constructor? ans. No instance variables, so
5 // so constructor not required.
6
7
8 abstract class Person
9 {
10     String name;
11     Integer age;
12     Float height;
13
14     // To initialize the instance variables
15     Person(String name, Integer age, Float height){
16         this.name = name;
17         this.age = age;
18         this.height = height;
19     }
20 }
21
22 //>>> Student extends Person
```

A large green question mark is drawn over the code area.

21_11_2022_interface_class_notes - Notepad

Why abstract class can contain constructor where as interface doesnot contain constructor?
abstract class => it is used to perform initialization of the object.

it is used to contain instance variable which are required for child object to perform initialisation for those instance variables.

interface => every variable is always static,public and final their is no chance of existing instance variable inside the class.
so we should perform initialisation at the time of declaration.
so constructor is not required for interface.

```
abstract class Person{  
    String name;  
    int age;  
    int height;  
    int weight;  
  
    Person(String name,int age,int height,int weight){  
        super();  
        this.name=name;
```

v9

Univ. Col 1

110% Windows (C:\UFI)

```
*21.11.2022_interface_class.mnw - Notepad
File Edit View
this.rollno=rollno;
this.marks=marks;
}
}

Question1:
Can reference be created for abstract class?
Person p =new Student("sachin",49.5,6f,71,10,100);
Can reference be created for interface?
ISample sample = null;
```

Note:-Every method present inside the interface is abstract, but in abstract class also we take only abstract methods then what is the need of interface concept?

20

```
class SampleImpl implements ISample{  
    }  
  
ISample sample = new SampleImpl();  
  
}  
  
abstract class Sample{  
    }  
  
class SampleApp extends Sample{  
    }  
  
Sample sample = new SampleApp();  
}
```

58

21

File Edit View
we can replace interface with abstract class, but it is not a good programming practise.

interface => performance high

abstract class => performance low

二四

1

```
    class Object{  
        Object(){  
            ...;  
        }  
    }  
  
    class SampleImpl extends Object implements ISample{  
        SampleImpl(){  
            ...;  
        }  
    }  
}
```

-



21°C
Münster

卷之三


```
        }
    }
}

abstract class Sample extends Object{
    Sample(){
        super();
    }
}

class SampleApp extends Sample{
    SampleApp(){
        super();
    }
}

Sample sample = new SampleApp();//3 secs
```

24

```
21.11.2022 Interface class Notes - Nithin  
File Edit View  
abstract class => performance low  
class Object{  
    Object(){  
        ....  
    }  
}  
abstract class Sample extends Object{  
    Sample(){  
        super();  
    }  
}  
class SampleApp extends Sample{  
    SampleApp(){  
        super();  
    }  
}  
Sample sample = new SampleApp(); // 3 secs
```

//Logical conclusion => If everything is abstract then recommended to go for "Interface".

2S



Unlocked 20°C
mostly cloudy



100% Windows (CEP) UTF-8

^ < > Esc F1 21.11.2022

21.11.2022_interface_class_notes - Notepad

File Edit View

}

=> we can replace interface concept with abstract class, but it is not a good programming practise.



eg#1

```
interface X{  
    ...  
}  
class Test implements X{  
    ...  
}  
Test t=new Test();
```

- i. performance is high.
- ii. While implementing X we can extends one more class, through which we can bring reusability.

eg#2:

Ln 124 Col 5 100% Windows (CEP) UTT-8
20°C Mostly cloudy

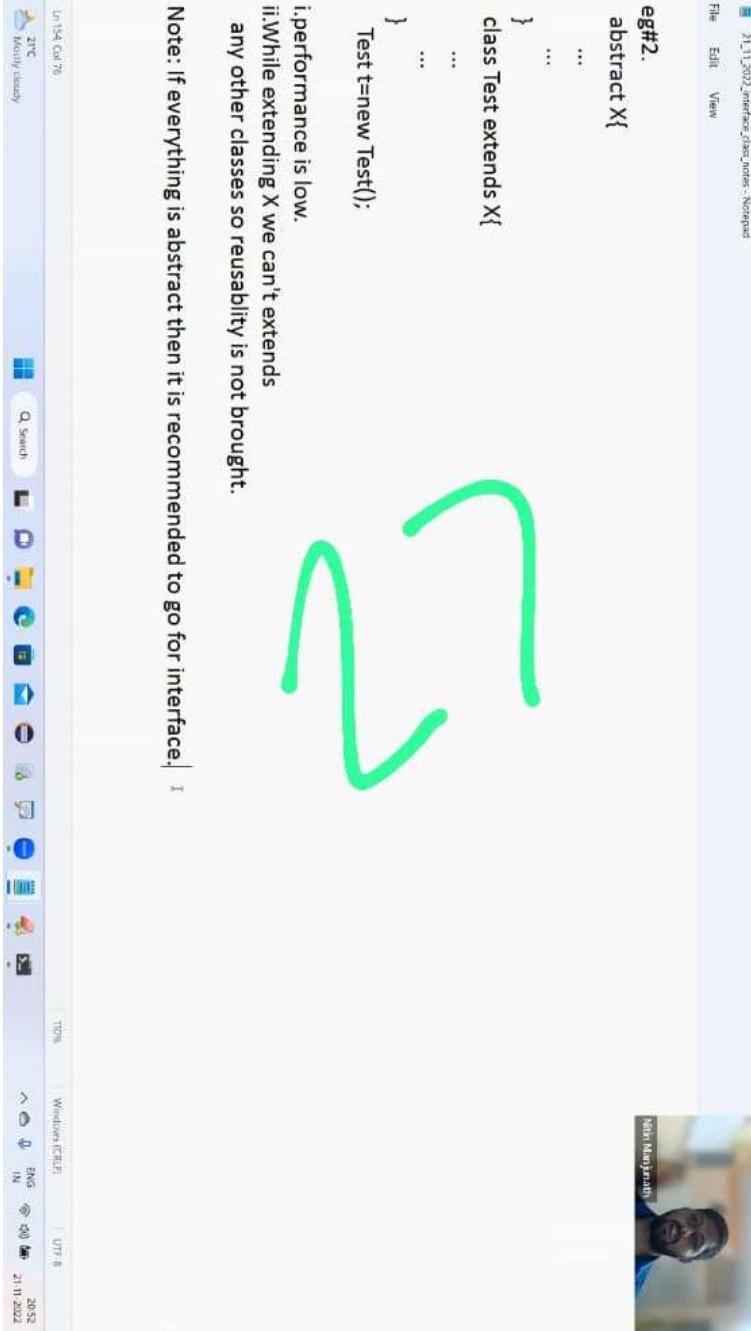
Q. Search

ENGLISH 21.11.2022

```
File Edit View  
21.11.2022 mifree_class_jnotes - Notepad  
  
eg#2.  
abstract X{  
    ...  
}  
class Test extends X{  
    ...  
}  
Test t=new Test();
```

- i.performance is low.
- ii.While extending X we can't extends any other classes so reusability is not brought.

Note: If everything is abstract then it is recommended to go for interface.



D:\>javap java.util.Collection

Compiled from "Collection.java"

```
public interface java.util.Collection<E> extends java.lang.Iterable<E> {
    public abstract int size();
    public abstract boolean isEmpty();
    public abstract boolean contains(java.lang.Object t);
    public abstract java.util.Iterator<E> iterator();
    public abstract java.lang.Object[] toArray();
    public abstract <T> T[] toArray(T[]);
    public abstract boolean add(E);
    public abstract boolean remove(java.lang.Object);
    public abstract boolean containsAll(java.util.Collection<?>);
    public abstract boolean addAll(java.util.Collection<? extends E>);
    public abstract boolean removeAll(java.util.Collection<?>);
    public abstract boolean removeIf(java.util.function.Predicat<? super E>);
    public abstract boolean retainAll(java.util.Collection<?>);
    public abstract void clear();
    public abstract boolean equals(java.lang.Object);
    public abstract int hashCode();
    public java.util.Spliterator<E> spliterator();
    public java.util.stream.Stream<E> stream();
    public java.util.stream.Stream<E> parallelStream();
}
```

D:\>



Command Prompt

```
D:\>javap java.util.ArrayList
Error: class not found: java.util.ArrayList

D:\>javap java.util.ArrayList
Compiled from "ArrayList.java"
public class java.util.ArrayList<E> extends java.util.AbstractList<E> implements java.util.List<E>, java.util.RandomAccess, j
transient java.lang.Object[] elementData;
public java.util.ArrayList(int);
public java.util.ArrayList();
public java.util.ArrayList(Collection<? extends E>);

public void trimToSize();
public void ensureCapacity(int);
public int size();
public boolean isEmpty();
public boolean contains(java.lang.Object);
public int indexOf(java.lang.Object);
public int lastIndexOf(java.lang.Object);
public java.lang.Object clone();
public java.lang.Object[] toArray();
public <T> T[] toArray(T[]);
public E elementData(int);
public E get(int);
public E set(int, E);
public boolean add(E);
public void add(int, E);
public E remove(int);
public boolean remove(java.lang.Object);
public void clear();
```

Nitin Manjrekar



```
D:\>javap java.util.AbstractList
Compiled from "AbstractList.java"
public abstract class java.util.AbstractList<E> extends java.util.AbstractCollection<E> implements java.util.List<E> {
    protected transient int modCount;
    protected java.util.AbstractList();
    public boolean add(E);
    public abstract E get(int);
    public E set(int, E);
    public void add(int, E);
    public E remove(int);
    public int indexOf(java.lang.Object);
    public int lastIndexOf(java.lang.Object);
    public void clear();
    public boolean addAll(int, java.util.Collection<? extends E>);
    public java.util.Iterator<E> iterator();
    public java.util.ListIterator<E> listIterator();
    public java.util.List<E> subList(int, int);
    public boolean equals(java.lang.Object);
    public int hashCode();
    protected void removeRange(int, int);
}
```



D:\>



20°C
Mostly cloudy

ENGLISH IN 21.11.2022

Command prompt

```
public static java.lang.String toOctalString(int);
public static java.lang.String toBinaryString(int);
static int formatUnsignedInt(int, int, char[], int, int);
public static java.lang.String toString(int);
public static java.lang.String toUnsignedString(int);

static void getChars(int, int, char[]);
static int stringSize(int);
public static int parseInt(java.lang.String, int) throws java.lang.NumberFormatException;
public static int parseSignedInt(java.lang.String, int) throws java.lang.NumberFormatException;
public static int parseUnsignedInt(java.lang.String, int) throws java.lang.NumberFormatException;
public static java.lang.Integer valueOf(java.lang.String, int) throws java.lang.NumberFormatException;
public static java.lang.Integer valueOf(java.lang.String);
public static java.lang.Integer valueOf(int);
public java.lang.Integer(int);
public java.lang.Integer(java.lang.String) throws java.lang.NumberFormatException;
public byte byteValue();
public short shortValue();
public int intValue();
public long longValue();
public float floatValue();
public double doubleValue();
public java.lang.String toString();
public int hashCode();
public static int hashCode(int);
public boolean equals(java.lang.Object);
public static java.lang.Integer getInteger(java.lang.String, int);
public static java.lang.Integer decode(java.lang.String, int);
public static java.lang.Integer getInteger(java.lang.String) throws java.lang.NumberFormatException;
public static int compareToInt(java.lang.Integer);
public static int compare(int, int);
public static int compareUnsigned(int, int);
```

Nitin Mehta



Wrapper class - Notepad

File Edit View

Wrapper class

Purpose

1. To wrap primitives into object form so that we can handle primitives also just like objects.
2. To define several utility functions which are required for the primitives.

Constructors

Almost all the Wrapper class have 2 constructors

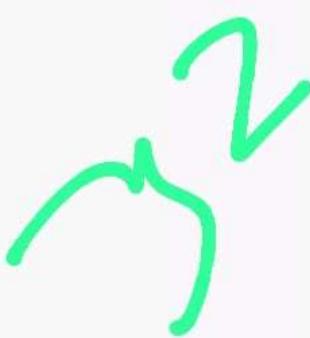
- a. one taking primitive type.
- b. one taking String type.

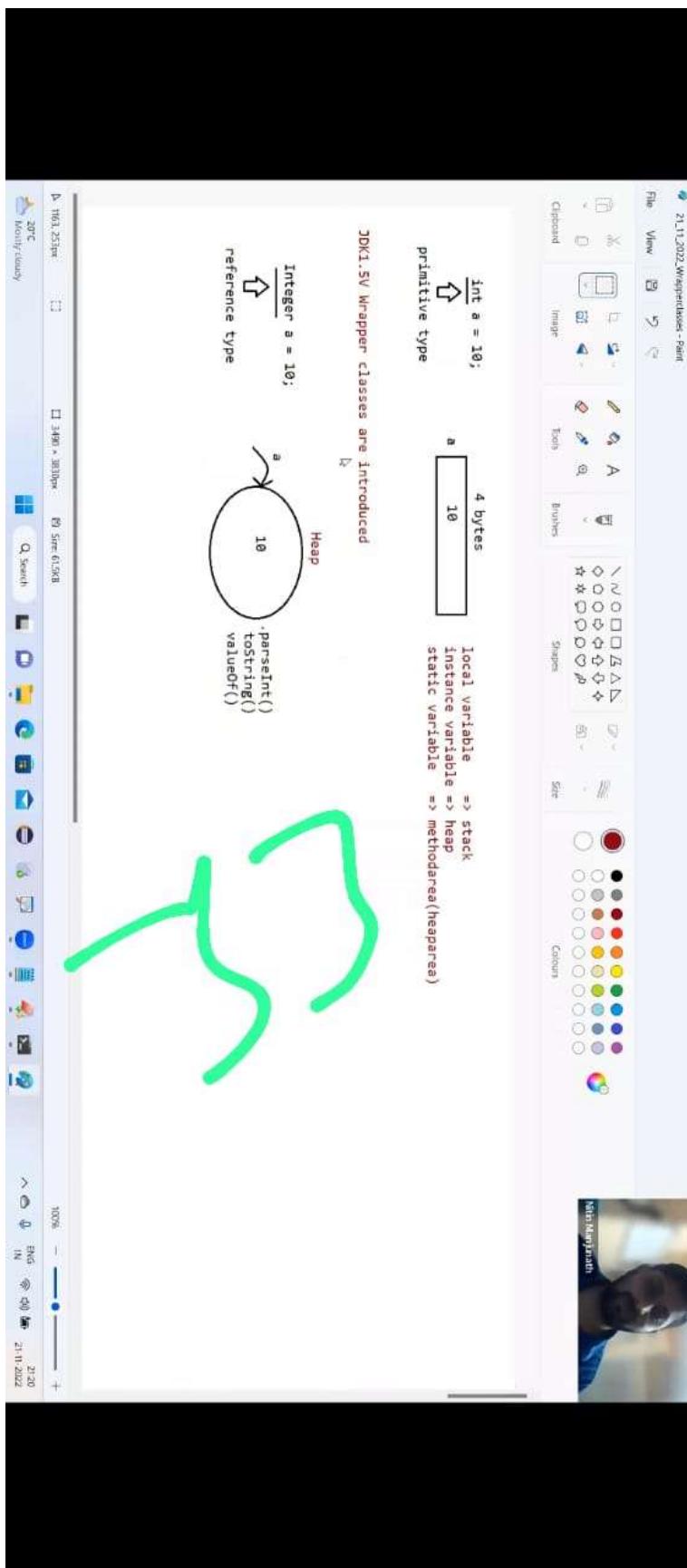
eg: Integer i=new Integer(10);
Integer i=new Integer("10");

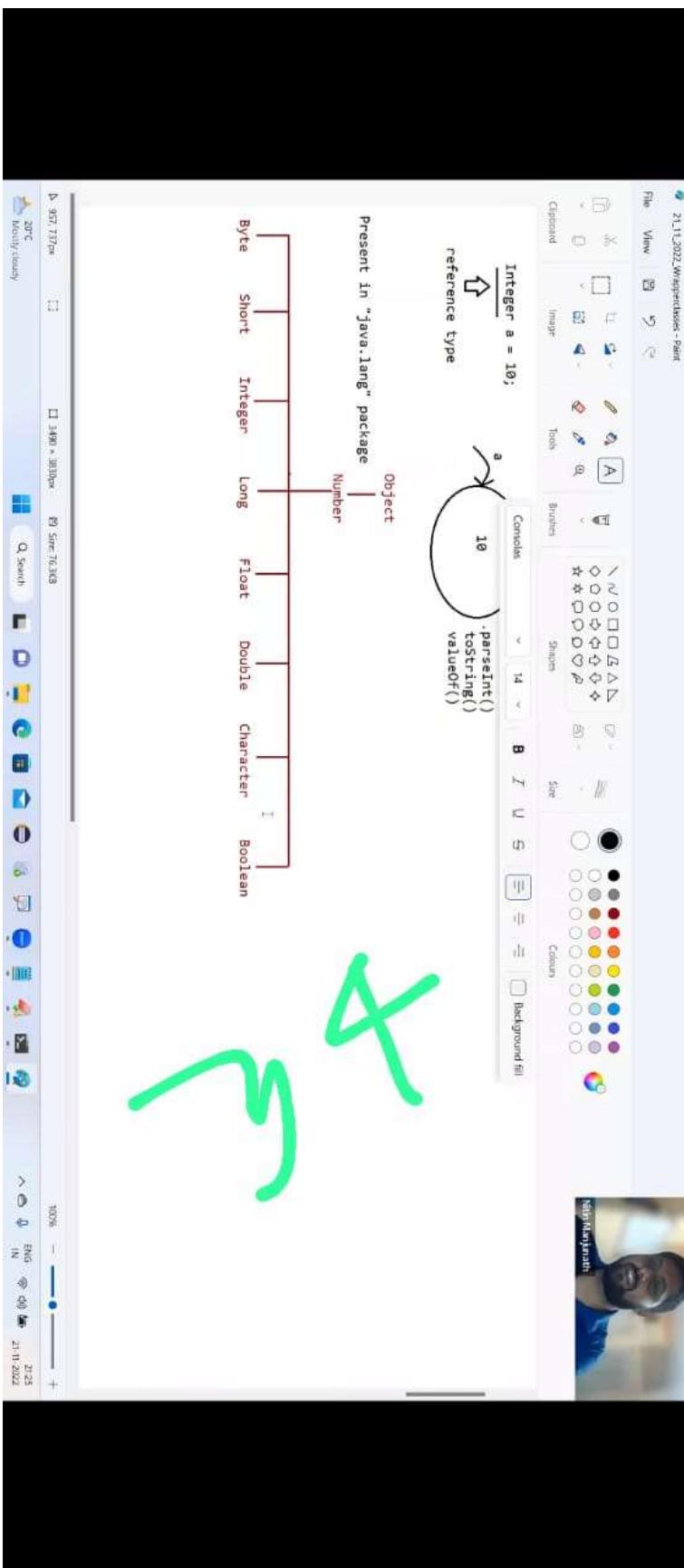
Double d=new Double(10.5);
Double d=new Double("10.5");

Note: If String argument is not properly defined then it would result in RunTimeException called "NumberFormatException".

eg:: Integer i=new Integer("ten");//RE:NumberFormatException



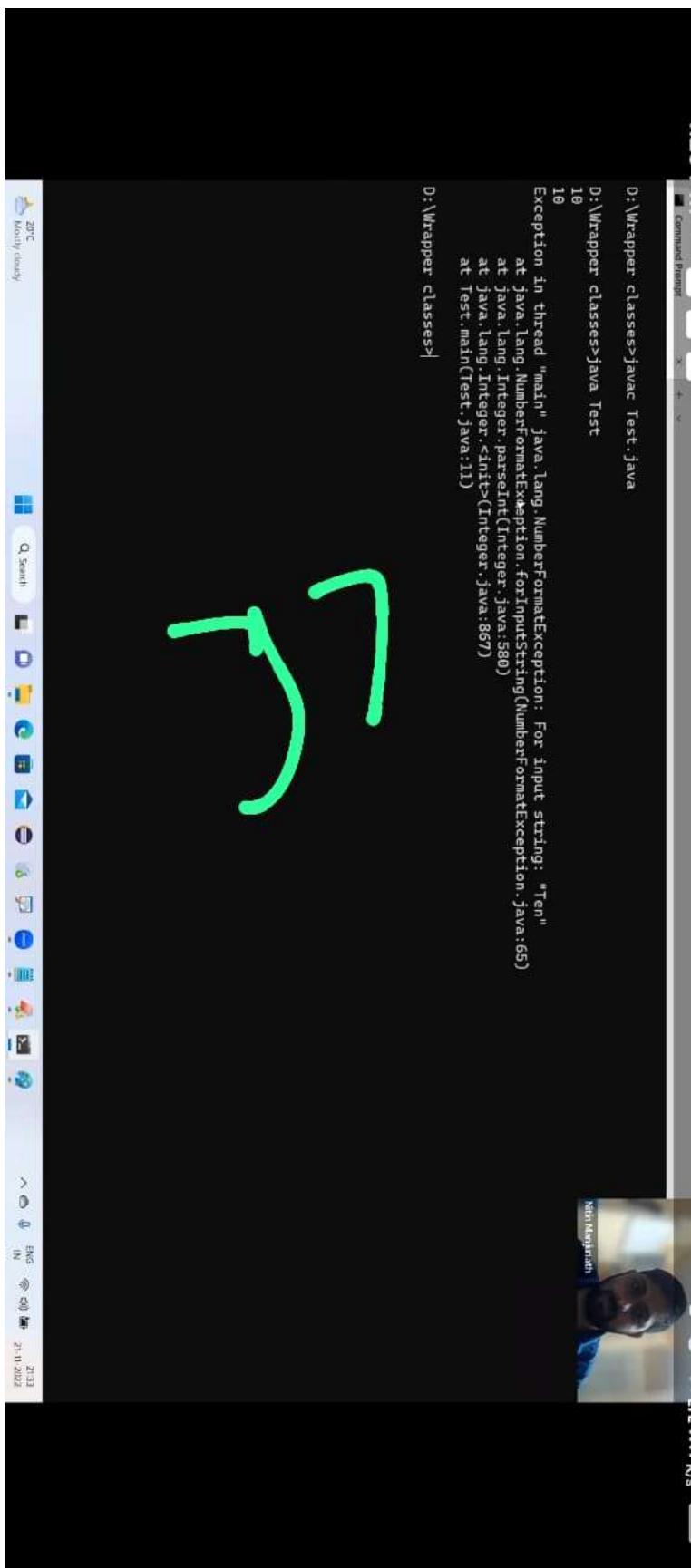




A portrait photograph of Nalin Manjrekar, a man with dark hair and a beard, wearing a light-colored shirt.

```
3
4 {
5     Integer i1 = new Integer(10);
6     System.out.println(i1);//toString()
7
8     Integer i2 = new Integer("10");
9     System.out.println(i2);//toString()
10
11    Integer i3 = new Integer("Ten");
12    System.out.println(i3);//toString()
13
14
15
16 }
17 }
```

ପ୍ରକାଶନ
କମିଶନ



21.11.2022 interface class notes - (Completed)

File Edit View

```
Double d=new Double("10.5");

Note: If String argument is not properly defined then it would result in RunTimeException called
      "NumberFormatException".
e.g.: Integer i=new Integer("ten");//RE:NumberFormatException
```

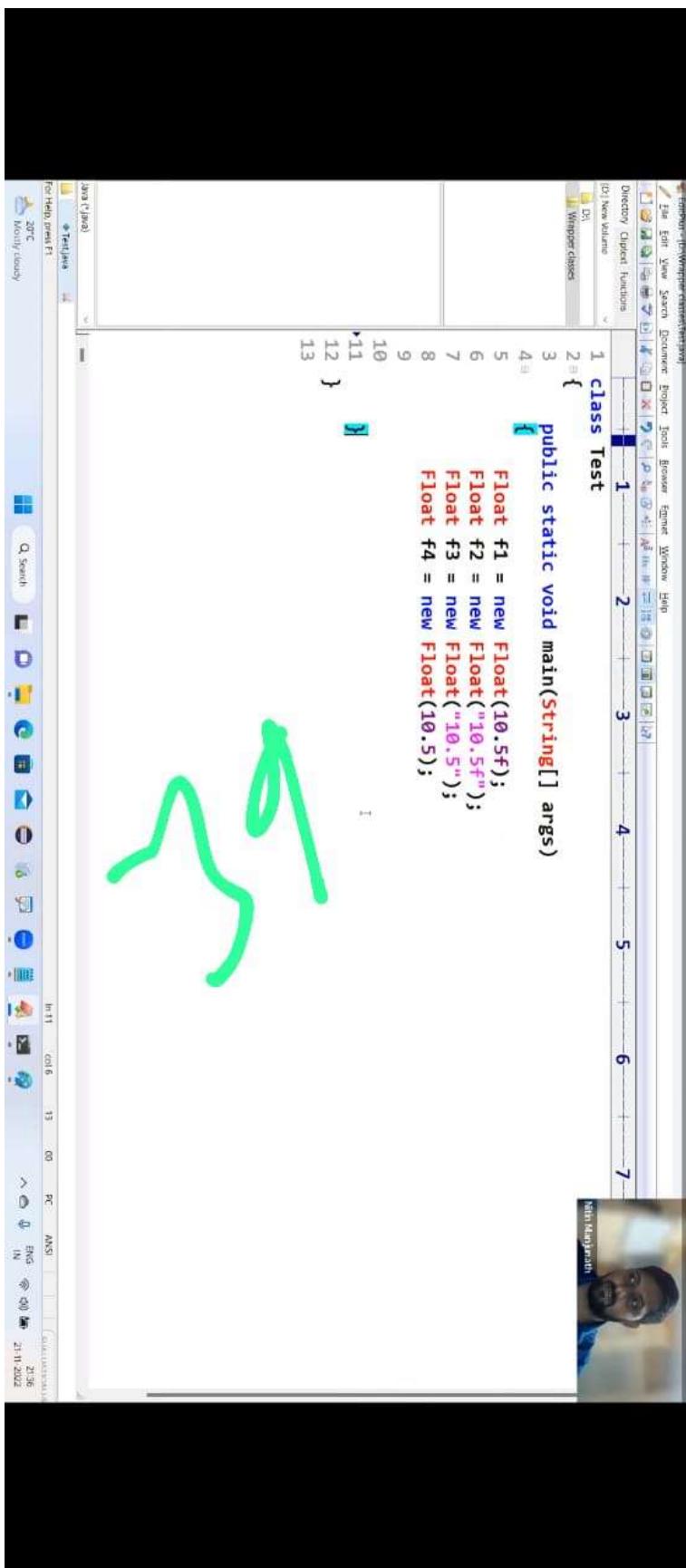
Wrapper class and its associated constructor

- Byte => byte and String
- Short => short and String
- Integer => int and String
- Long => long and String
- **Float => float ,String and double
- Double => double and String
- **Character=> character
- ***Boolean => boolean and String

Nitin Manjrekar



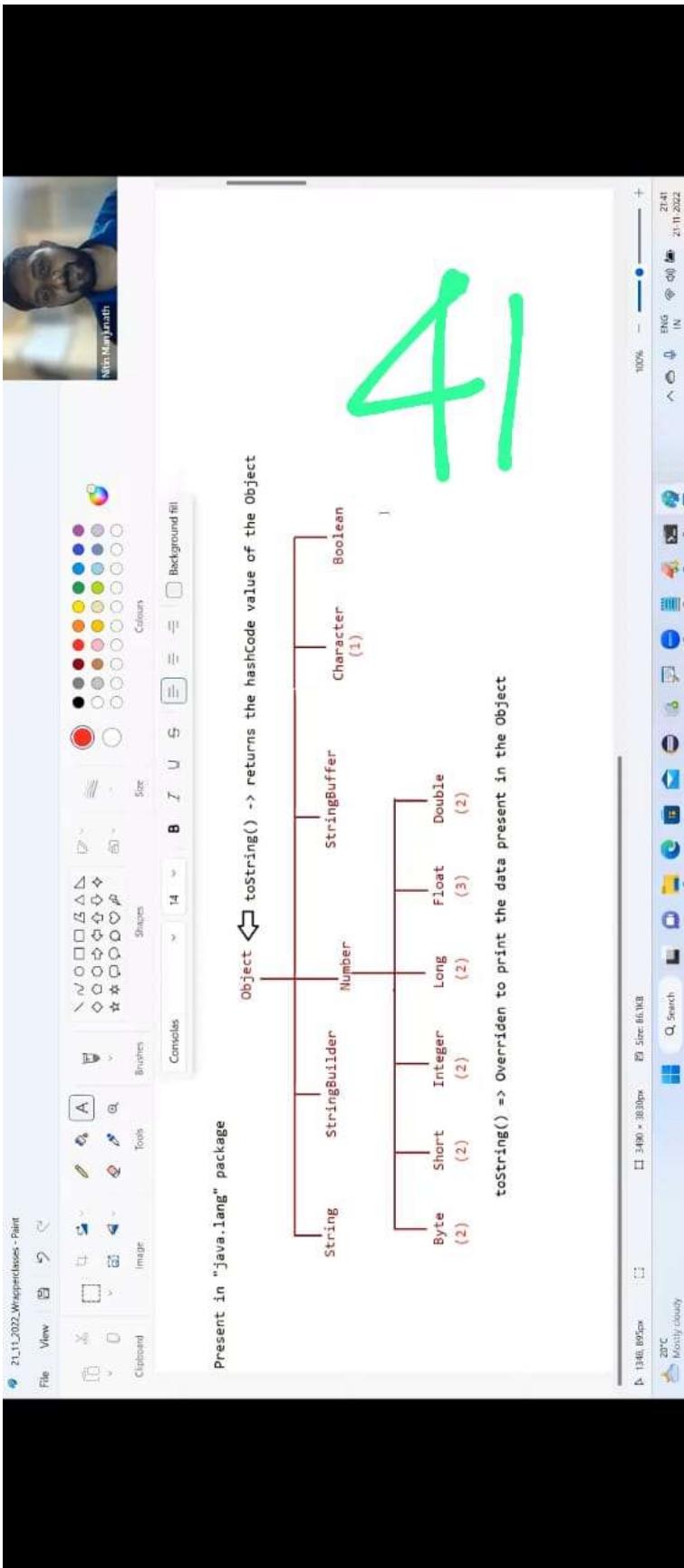
1175_Go12 10% Windows (CEP) UTF-8 ↻ ENG IN 21.11.2022



40

```
D:\Wrapper\classes>javac Test.java
D:\Wrapper\classes>java Test
10
10
Exception in thread "main" java.lang.NumberFormatException: For input string: "ten"
        at java.lang.NumberFormatException.forInputString(NumberFormatException.java:65)
        at java.lang.Integer.parseInt(Integer.java:580)
        at java.lang.Integer.<init>(Integer.java:867)
        at Test.main(Test.java:11)

D:\Wrapper\classes>javac Test.java
D:\Wrapper\classes>java Test
D:\Wrapper\classes>
```



Character c1 =new Character("a");

Note: Some messages have been simplified; recompile with -Xdiags:verbose to get full output
1 error

D:\Wrapper\classes>javap java.lang.Boolean

Compiled from "Boolean.java"

```
public final class java.lang.Boolean implements java.io.Serializable, java.lang.Comparable<java.lang.Boolean> {
    public static final java.lang.Boolean TRUE;
    public static final java.lang.Boolean FALSE;
    public static final java.lang.Class<java.lang.Boolean> TYPE;
    public java.lang.Boolean(boolean);
    public java.lang.String toString();
    public static boolean parseBoolean(java.lang.String);
    public boolean booleanValue();
    public static java.lang.Boolean valueOf(java.lang.String);
    public static java.lang.Boolean valueOf(boolean);
    public static java.lang.String toString(boolean);
    public java.lang.String hashCode();
    public int hashCode();
    public static int hashCode(boolean);
    public static boolean equals(java.lang.Object);
    public boolean getBoolean(java.lang.String);
    public int compareTo(java.lang.Boolean);
    public static int compare(boolean, boolean);
    public static boolean logicalAnd(boolean, boolean);
    public static boolean logicalOr(boolean, boolean);
    public static boolean logicalXor(boolean, boolean);
    public int compareTo(java.lang.Object);
    static {};
}
```

D:\Wrapper\classes>

20°C Mostly cloudy

Nitin Mangrulkar

ENGLISH IN 21/11/2022 23:41

42

Editor - [D:\WhatsApp\classes\Untitled]

File Edit View Search Document Project Tools Browser Errors Window Help

Directory Options Functions

[D:] New Volume D:\ Writing classes

```
1 {  
2   Boolean b1=new Boolean(true);  
3   System.out.println(b1);//true  
4   Boolean b2=new Boolean(false);  
5   System.out.println(b2);//false  
6  
7  
8  
9  
10  
11  
12   Boolean b3=new Boolean(true);  
13   System.out.println(b3);//false  
14  
15   Boolean b4=new Boolean(false);  
16   System.out.println(b4);//false  
17  
18   Boolean b5=new Boolean(true);  
19   System.out.println(b5);  
20  
21  
22  
23 }  
24
```

Java (Java)

For Help press F1

20°C
mostly cloudy

Q. Search

113 col. 9 24 53 PC ANSI □ (Automatic)
Eng ⇄ zh-CN IN 21.11.2022

83

20°C
Mostly cloudy

Q. Search

Min Max Min Max

Eng ⌂ 21.11.2022

44

```
D:\Wrapper\classes>javac Test.java
Test.java:12: error: cannot find symbol
    Boolean b3=new Boolean("True");
                           ^
symbol:   variable True
location: class Test
Test.java:15: error: cannot find symbol
    Boolean b4=new Boolean(False);
                           ^
symbol:   variable False
location: class Test
Test.java:18: error: cannot find symbol
    Boolean b5=new Boolean(TRUE);
                           ^
symbol:   variable TRUE
location: class Test
3 errors

D:\Wrapper\classes>
```

Editor - (D:\Wdpt\class\Java\src\java)

File Edit View Search Document Project Icons Browser Egret Window Help

Directory Classes Functions D:\ New Volume Writing classes

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         //String input => case is not important, content is not important
6         //String input => case insensitive of true is treated as true other
7         // cases treated as false.
8         Boolean b1=new Boolean("true");//true
9         Boolean b2=new Boolean("True");//true
10        Boolean b3=new Boolean("false");//false
11        Boolean b4=new Boolean("False");//false
12        Boolean b5=new Boolean("nitin");//false
13        Boolean b6=new Boolean("TRUE");//true
14
15
16        System.out.println(b1);
17        System.out.println(b2);
18        System.out.println(b3);
19        System.out.println(b4);
20        System.out.println(b5);
21        System.out.println(b6);
```

Java (1.java)

For Help press F1

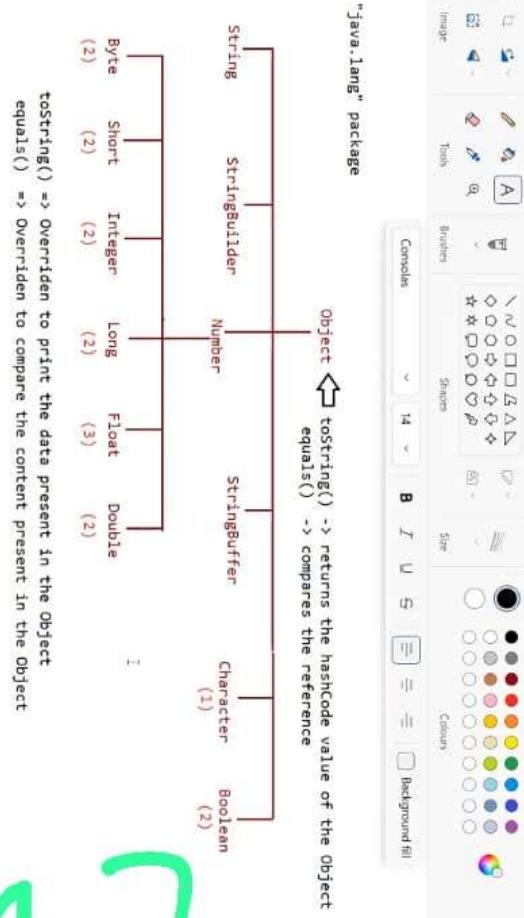
20°C
mostly cloudy

45

46

```
D:\Wrapper\classes>javac Test.java
D:\Wrapper\classes>java Test
true
true
false
false
true
true

D:\Wrapper\classes>
```



`toString()` => Overridden to print the data present in the Object
`equals()` => Overridden to compare the content present in the Object

Diagram illustrating the inheritance hierarchy of Java's Comparable interface:

```

graph TD
    Object --> Comparable
    Comparable --> String
    Comparable --> Integer
    Comparable --> Long
    Comparable --> Float
    Comparable --> Double
    Comparable --> Short
    Comparable --> Character
    Comparable --> Boolean
    Comparable --> Number
    Comparable --> StringBuffer

```

Note: The Comparable interface is part of the `Object` class.

toString() -> returns the hashCode value of the Object

equals() -> compares the reference

Annotations in the diagram:

- `(2)`: Used for `String`, `Integer`, `Long`, `Float`, `Double`, `Short`, `Character`, and `Boolean`.
- `(3)`: Used for `Number`.

Toolbars and color palette are visible on the right side of the slide.

48

The screenshot shows a video conference interface with a participant named Nithi Mayurath. In the foreground, a Java code editor displays the following code:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         Boolean b1 =new Boolean("yes");//false
6         Boolean b2 =new Boolean("no");//false
7         System.out.println(b1);
8         System.out.println(b2);
9
10        System.out.println(b1.equals(b2));//false.equals(false)-> true
11
12    }
13
14 }
15
```

```
*21.11.2022_Inference_Class.mines - Notepad  
File Edit View  
  
System.out.println(b2);  
  
System.out.println(b1.equals(b2));//false.equals(false)-> true  
System.out.println(b1 == b2);//false  
  
}  
  
}  
  
Note: In case of Boolean constructor, boolean value be treated as true w.r.t to case insensitive part of "true", for all others it would be treated as "false".  
  
Note:  
If we are passing String argument then case is not important and content is not important.  
If the content is case insensitive String of true then it is treated as true in all other cases it is treated as false.  
  
Note: In case of Wrapper class,toString() is overridden to print the data.  
In case of Wrapper class>equals() is overridden to check the content.  
Just like String class, Wrapper classes are also treated as "immutable class".
```

98

Erkhan - (D:\Work\src\classes\test.java)

File Edit View Search Document Project Icons Browser Egret Window Help

Directory Object Functions

(D) New Volume

D: Wimpel class

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         Integer i1 = new Integer(10);
6         Integer i2 = new Integer(10);
7         System.out.println(i1);//10
8         System.out.println(i2);//10
9         System.out.println(i1.equals(i2));//true
10    }
11 }
12
13
```

SO



In 5 col 9 13 49 PC ANSI

For Help press F1

20°C
Mostly cloudy

Q Search

21-11-2022_interface_class_notes - Notepad

File Edit View

part of "true", for all others it would be treated as "false".

Note:

If we are passing String argument then case is not important and content is not important.
If the content is case insensitive String of true then it is treated as true in all other cases it is treated as false.

Note: In case of Wrapper class, `toString()` is overridden to print the data.

In case of Wrapper class, `equals()` is overridden to check the content.

Just like String class, Wrapper classes are also treated as "Immutable class".

Immutable class

=====

If we create an Object and if we try to make a change, with that change new object will be created and those changes will not reflected in the old copy.

11:20, Oct 1
20°C
Mostly cloudy



I

Ln 260, Col 1

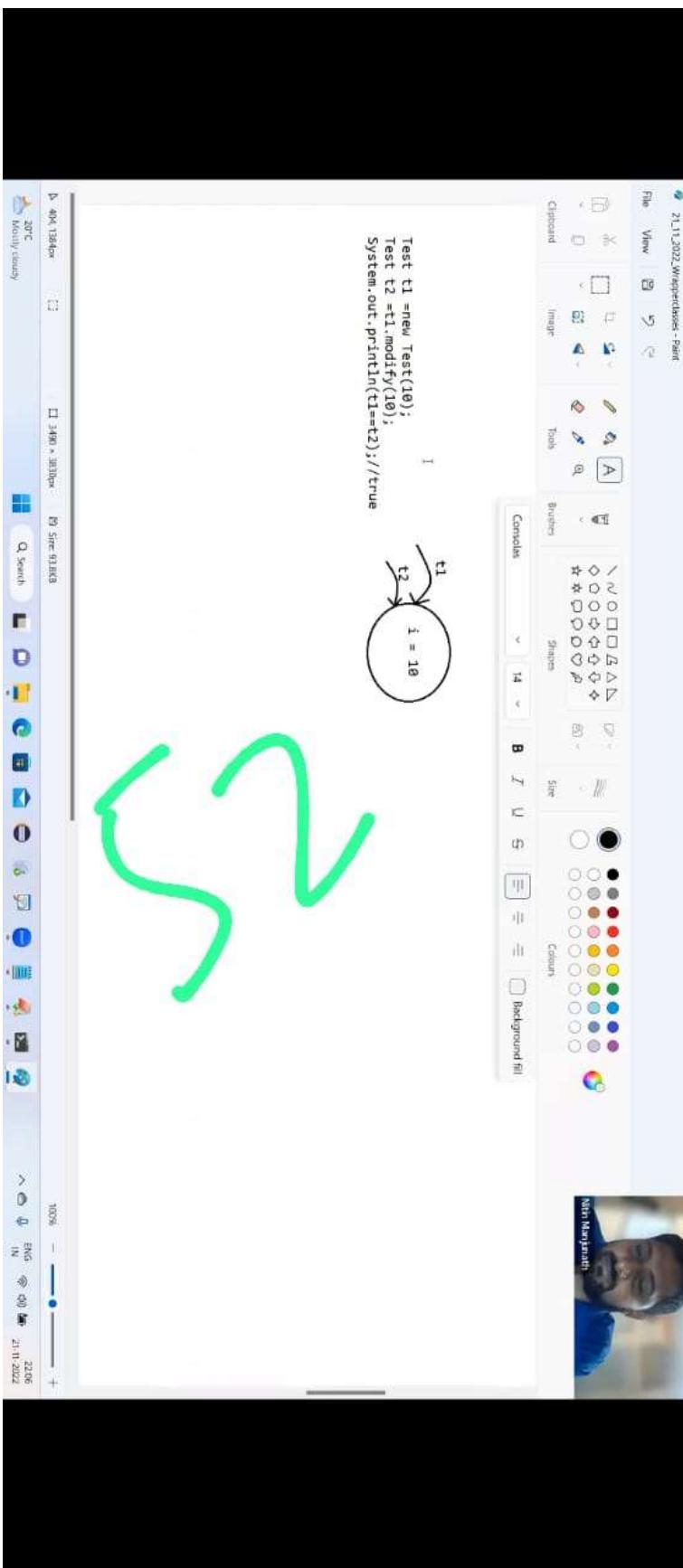
10%

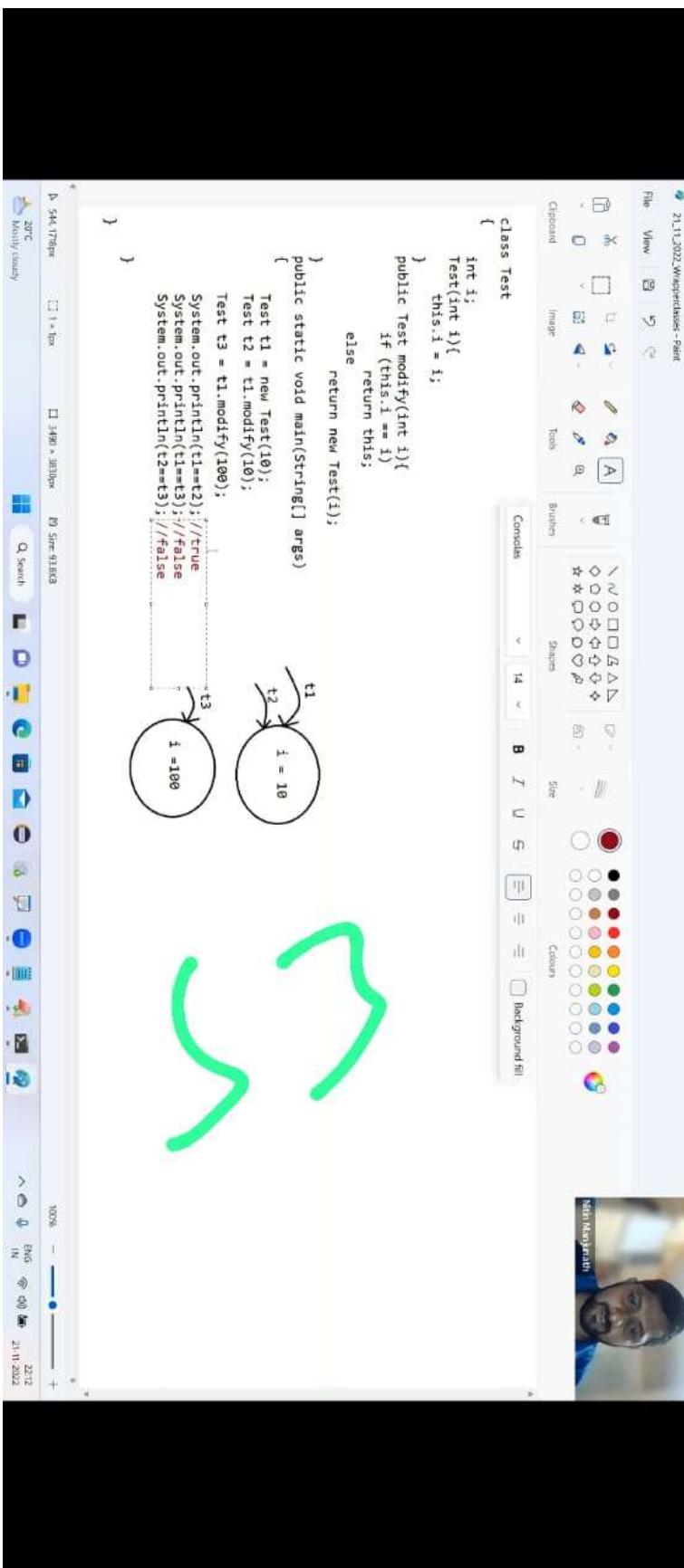
Windows-CE/FF

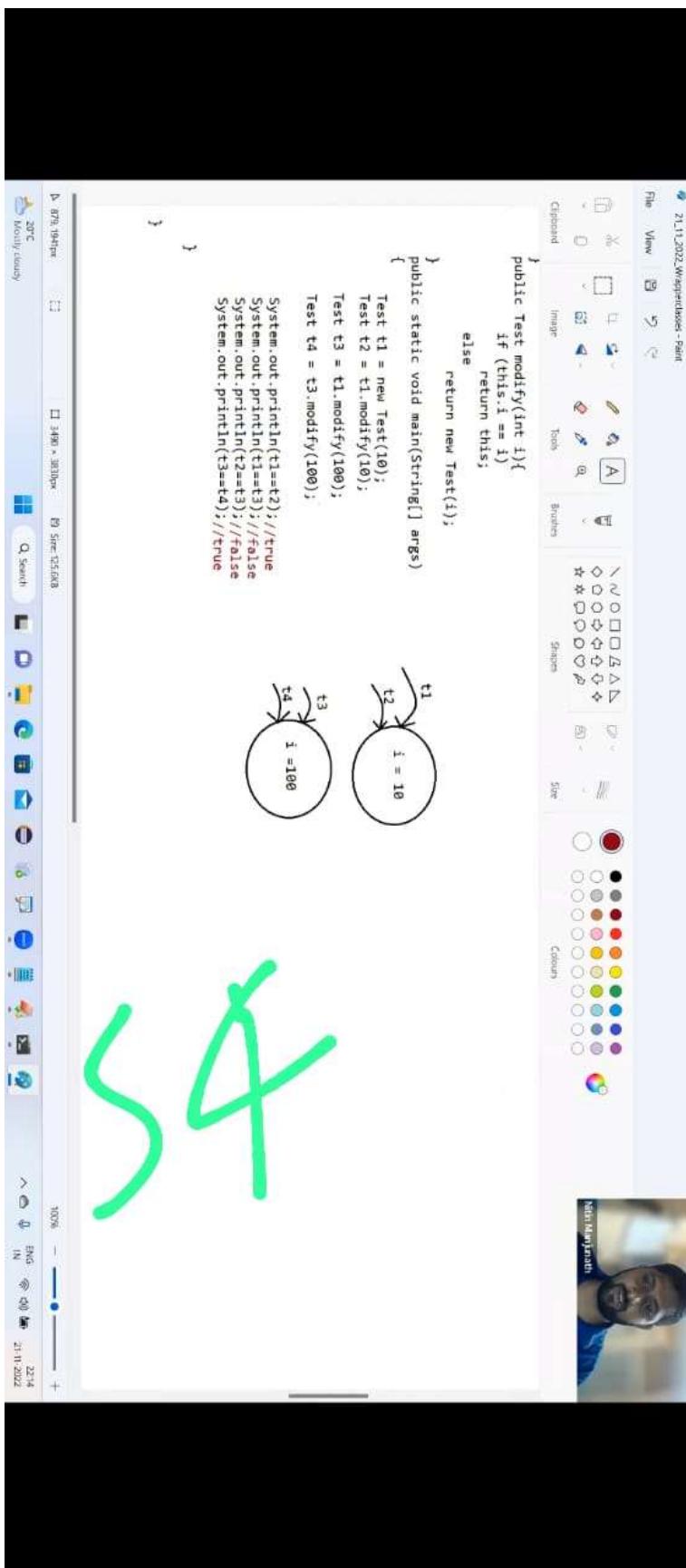
UTF-8

Eng

21-11-2022







File Edit View Search Document Project Tools Browser Editor Window Help

Directory Chroot Functions

D: New Volume Writing classes

```
1 final class Test
2 {
3     int i;
4     Test(int i){
5         this.i = i;
6     }
7     public Test modify(int i){
8         if (this.i == i)
9             return this;
10        else
11            return new Test(i);
12    }
13 }
14 public static void main(String[] args)
15 {
16     Test t1 = new Test(10);
17     Test t2 = t1.modify(10);
18     Test t3 = t1.modify(100);
19     Test t4 = t3.modify(100);
20
21     System.out.println(t1==t2);
22 }
```

Java (javac) 1.8.0_291 Java(TM) SE Runtime Environment 11.0.11+1~11-Ubuntu Java HotSpot(TM) 64-Bit Server VM 21.11-b225

Nitin Kompalli