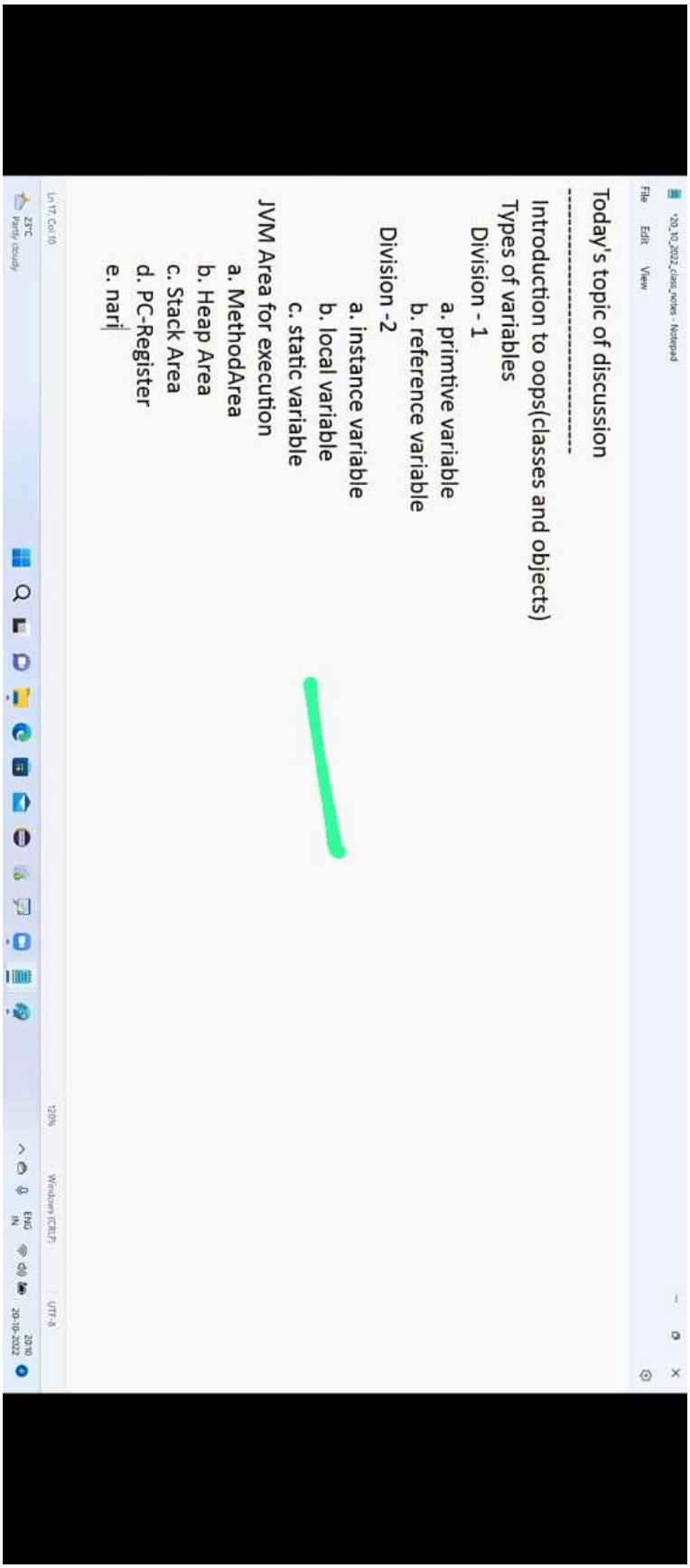
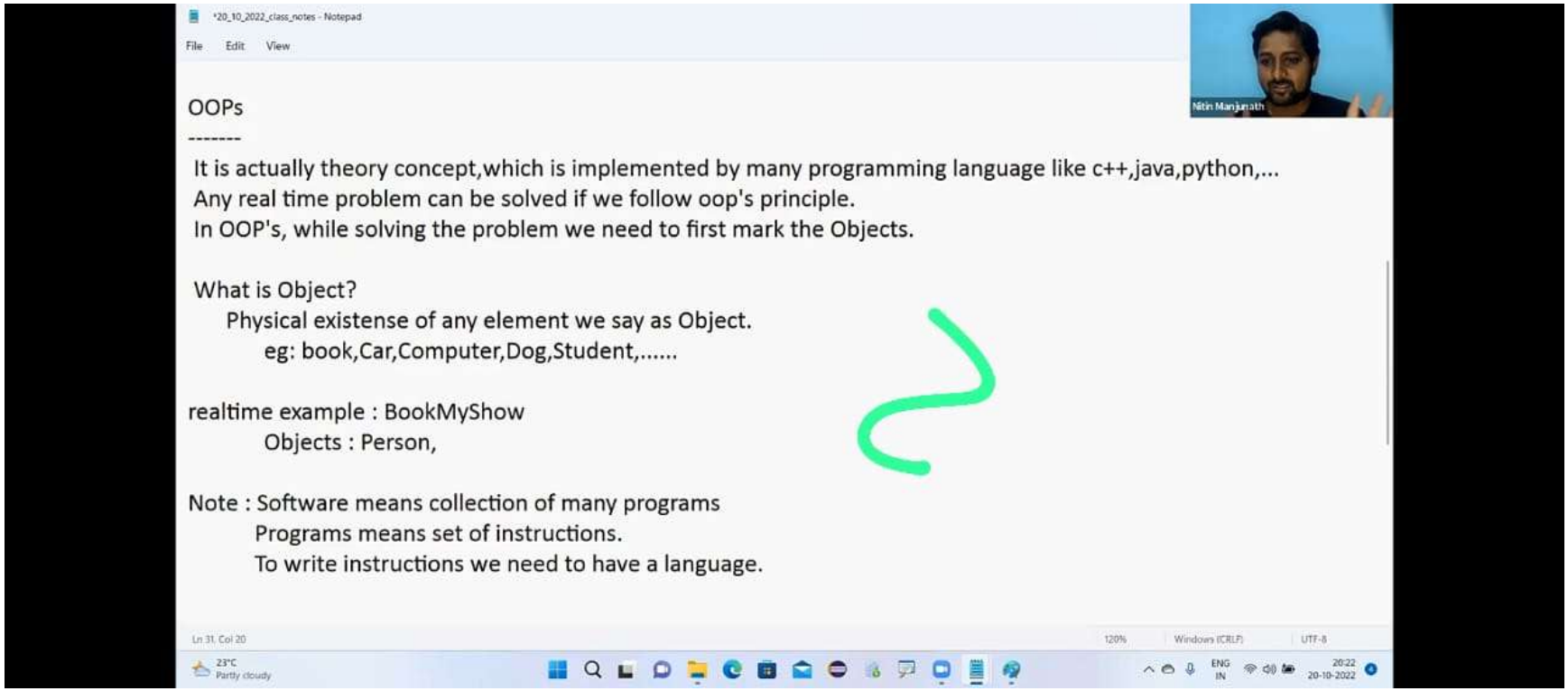
Class_And_Variables




*20_10_2022_class_notes - Notepad
File Edit View
Nitin Manjunath
OOPs

It is actually theory concept,which is implemented by many programming language like c++,java,python,...
Any real time problem can be solved if we follow oop's principle.
In OOP's, while solving the problem we need to first mark the Objects.
What is Object?
Physical existense of any element we say as Object.
eg: book,Car,Computer,Dog,Student,.....
realtime example : BookMyShow
Objects : Person,
Note : Software means collection of many programs
Programs means set of instructions.
To write instructions we need to have a language.
Ln 31, Col 20
120%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
20:22
20-10-2022

*20_10_2022_class_notes - Notepad

File Edit View

OOPs

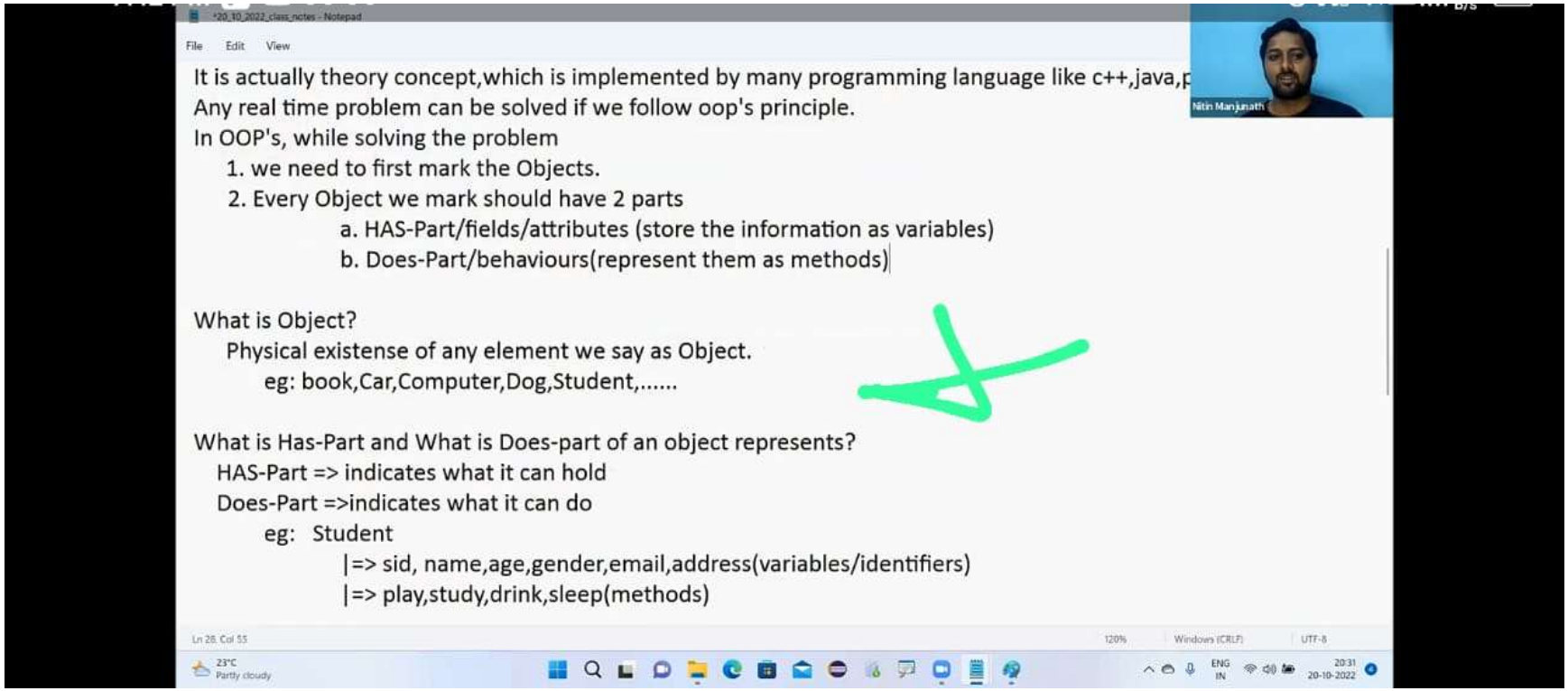
It is actually theory concept,which is implemented by many programming language like c++,java,python,...
Any real time problem can be solved if we follow oop's principle.
In OOP's, while solving the problem
1. we need to first mark the Objects.
2. Every Object we mark should have 2 parts
    a. HAS-Part/fields/attributes (store the information as variables)
    b. Does-Part/behaviours(represent them as methods)

What is Object?
Physical existense of any element we say as Object.
eg: book,Car,Computer,Dog,Student,.....

What is Has-Part and What is Does-part of an object represents?
HAS-Part => indicates what it can hold
Does-Part =>indicates what it can do
eg: Student

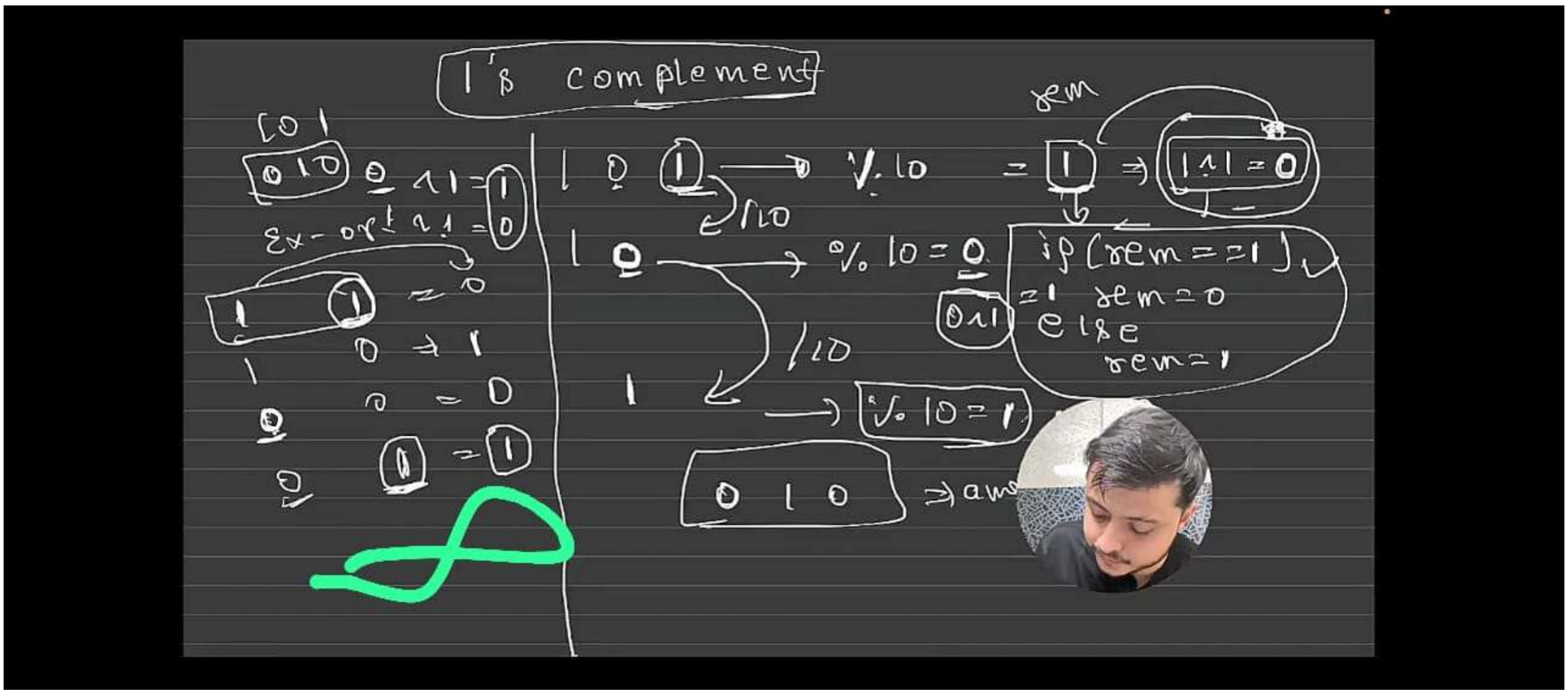
Ln 28, Col 55 120% Windows (CRLF) UTF-8

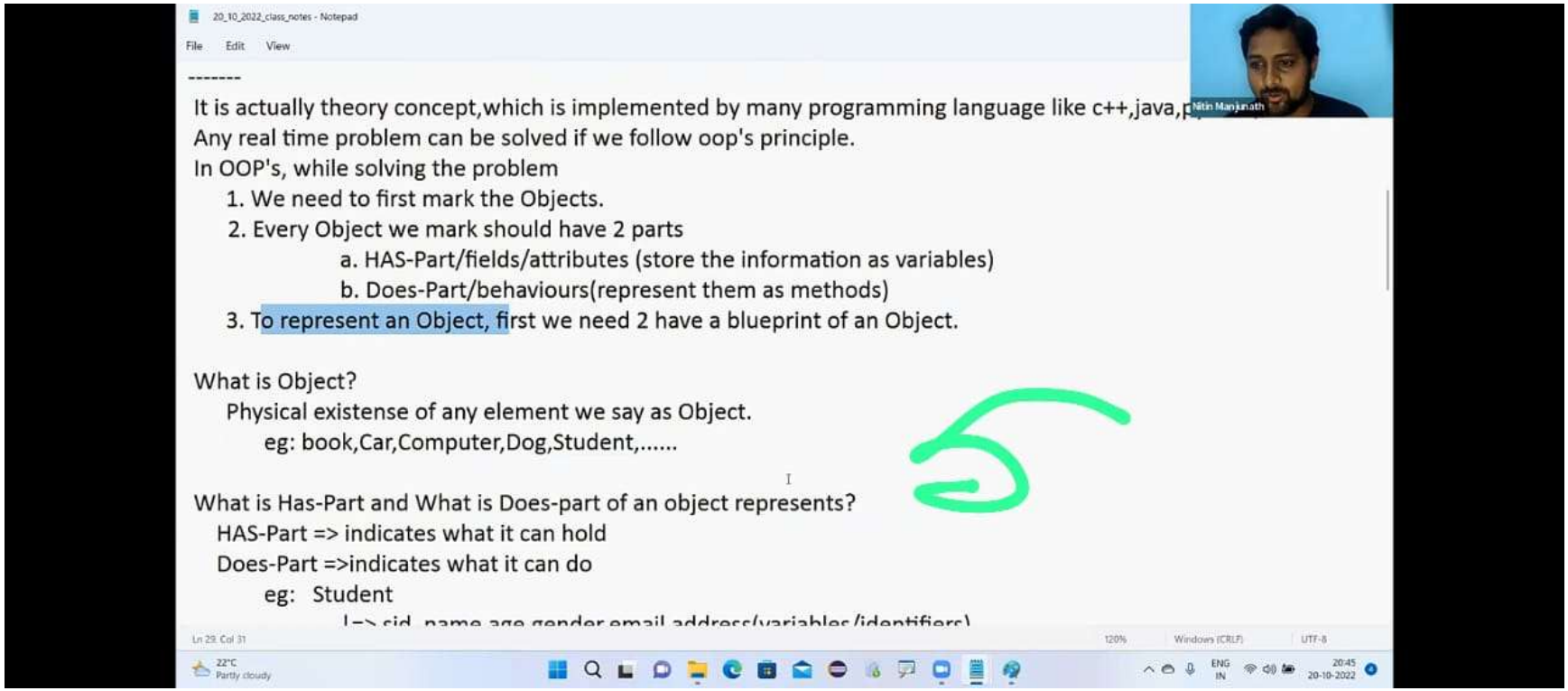
23°C Partly cloudy ENG IN 20:31 20-10-2022


*20_10_2022_class_notes - Notepad
File Edit View
It is actually theory concept,which is implemented by many programming language like c++,java,p
Any real time problem can be solved if we follow oop's principle.
In OOP's, while solving the problem
1. we need to first mark the Objects.
2. Every Object we mark should have 2 parts
a. HAS-Part/fields/attributes (store the information as variables)
b. Does-Part/behaviours(represent them as methods)
What is Object?
Physical existense of any element we say as Object.
eg: book,Car,Computer,Dog,Student,.....
What is Has-Part and What is Does-part of an object represents?
HAS-Part => indicates what it can hold
Does-Part =>indicates what it can do
eg: Student
|=> sid, name,age,gender,email,address(variables/identifiers)
|=> play,study,drink,sleep(methods)
Nitin Manjunath
Ln 28, Col 55
120%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
20:31
20-10-2022

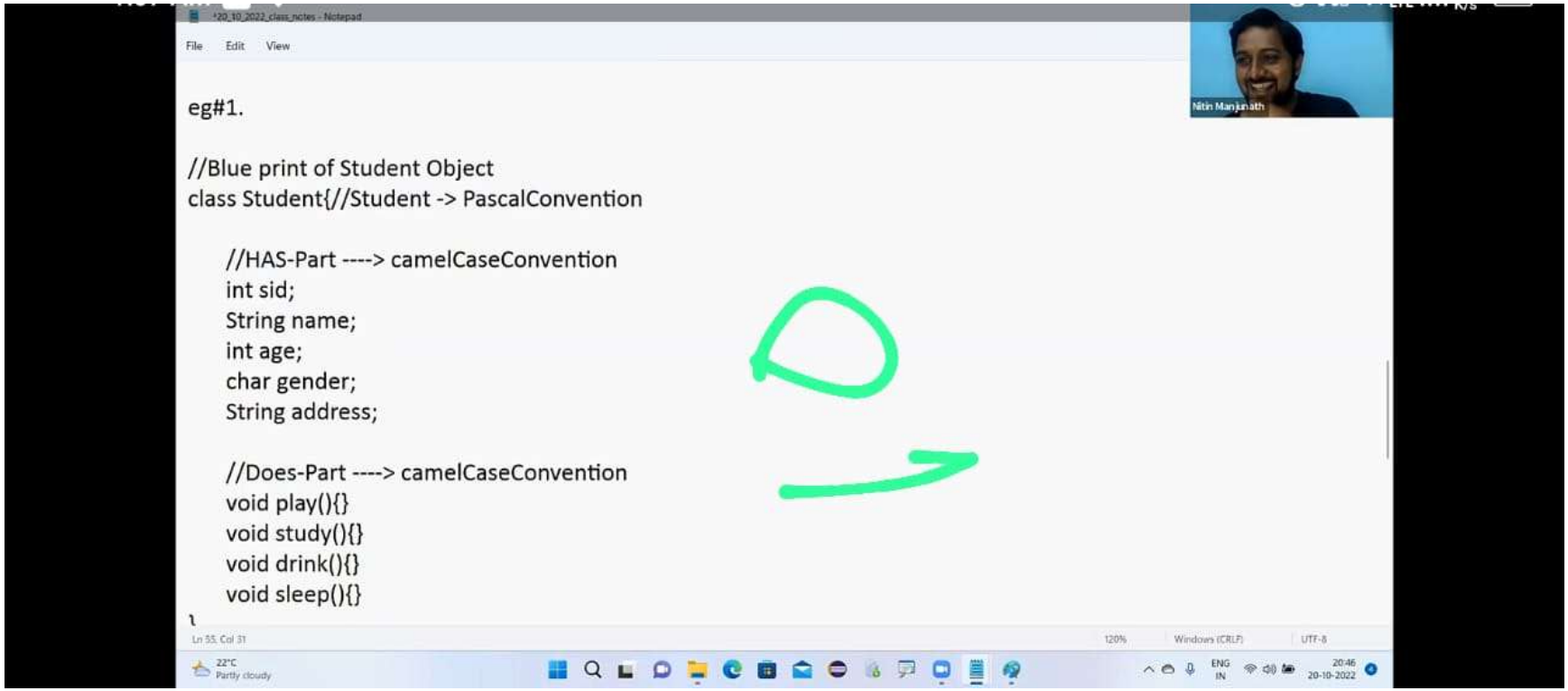
Resizing
100%

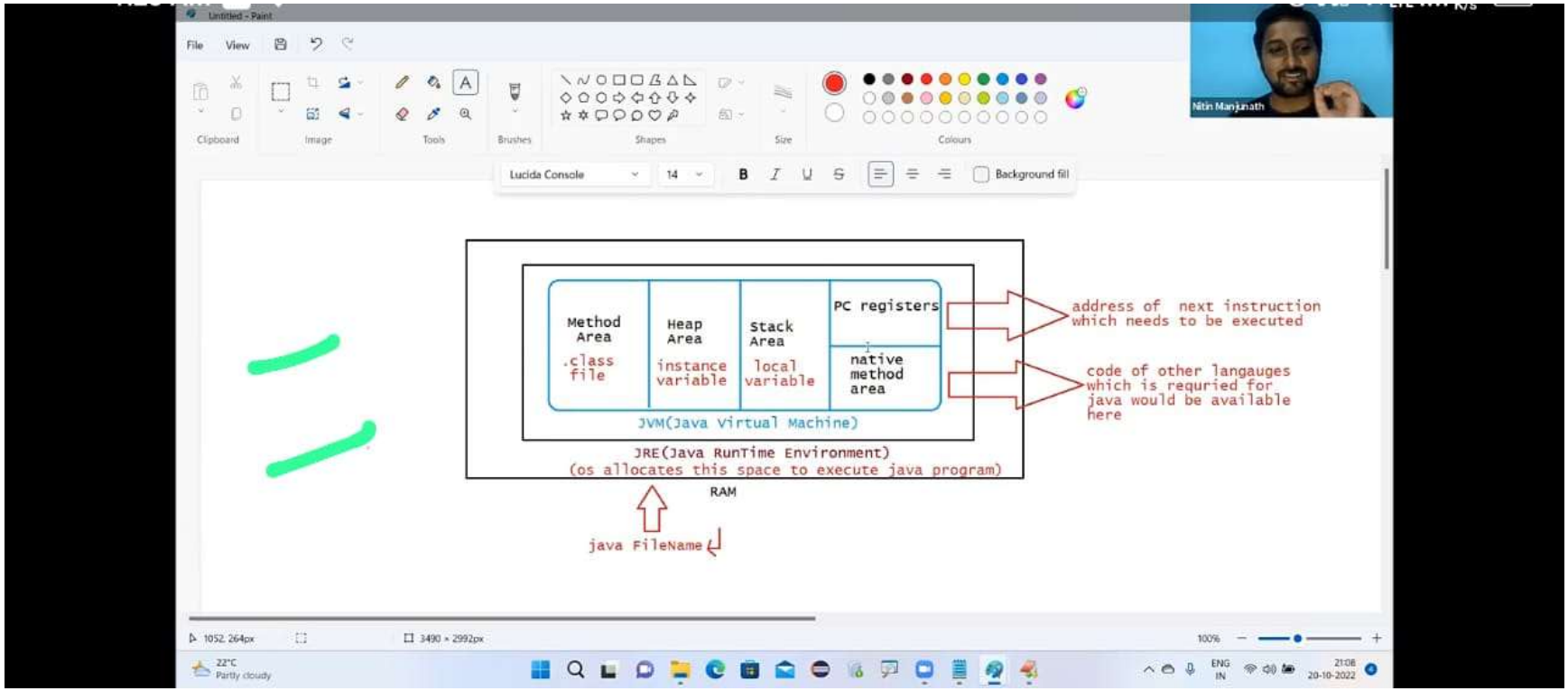
Binary to Decimal
(i) [1010] ⇒
1 0 1 0
1×2³ + 0×2² + 1×2¹ + 0×2⁰
8 0 2 0
(ii) 101
Binary
Decimal
0 1 2 3 4 5 6 7 8 9
Octal

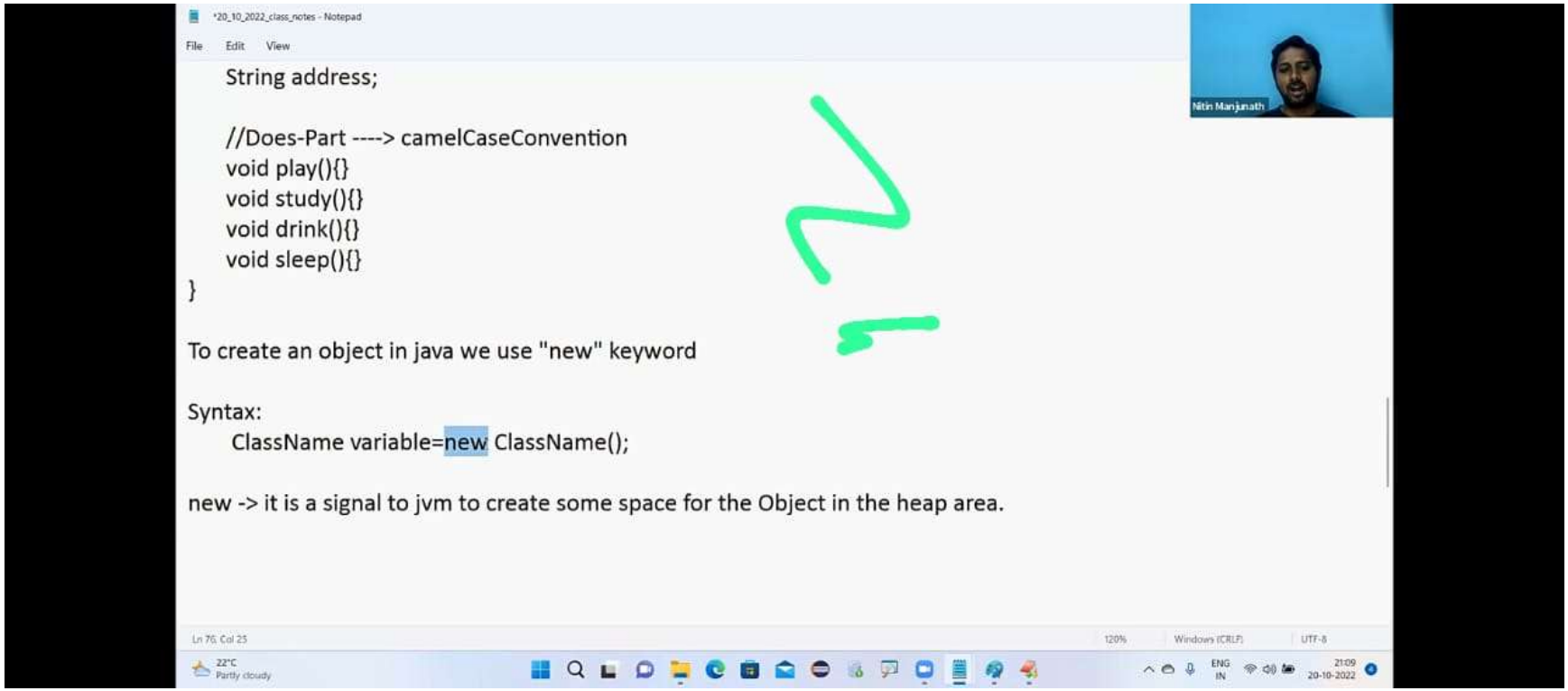

1's complement
rem
if (rem == 1)
rem = 0
else
rem = 1
% 10 = 0
% 10 = 1
/10
0 1 0
Ex-or

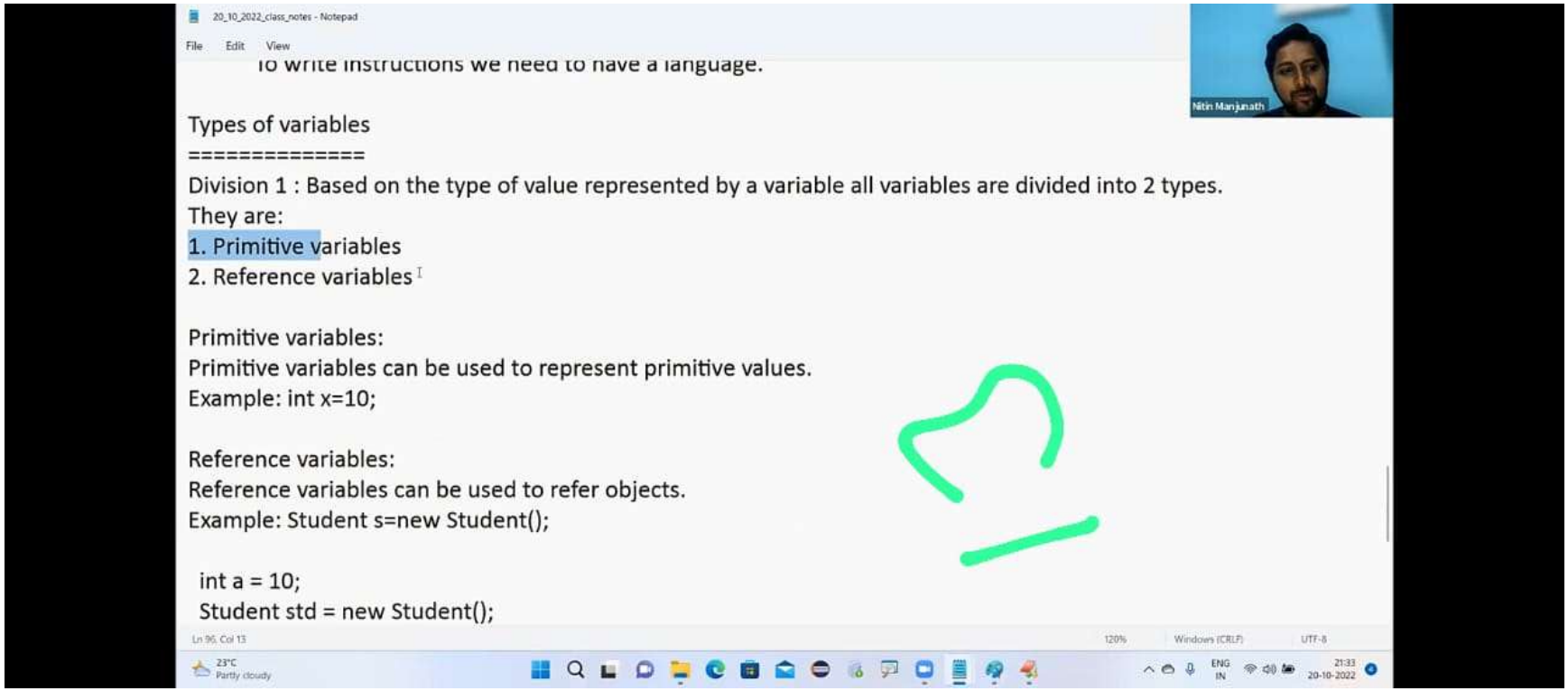

20_10_2022_class_notes - Notepad
File Edit View

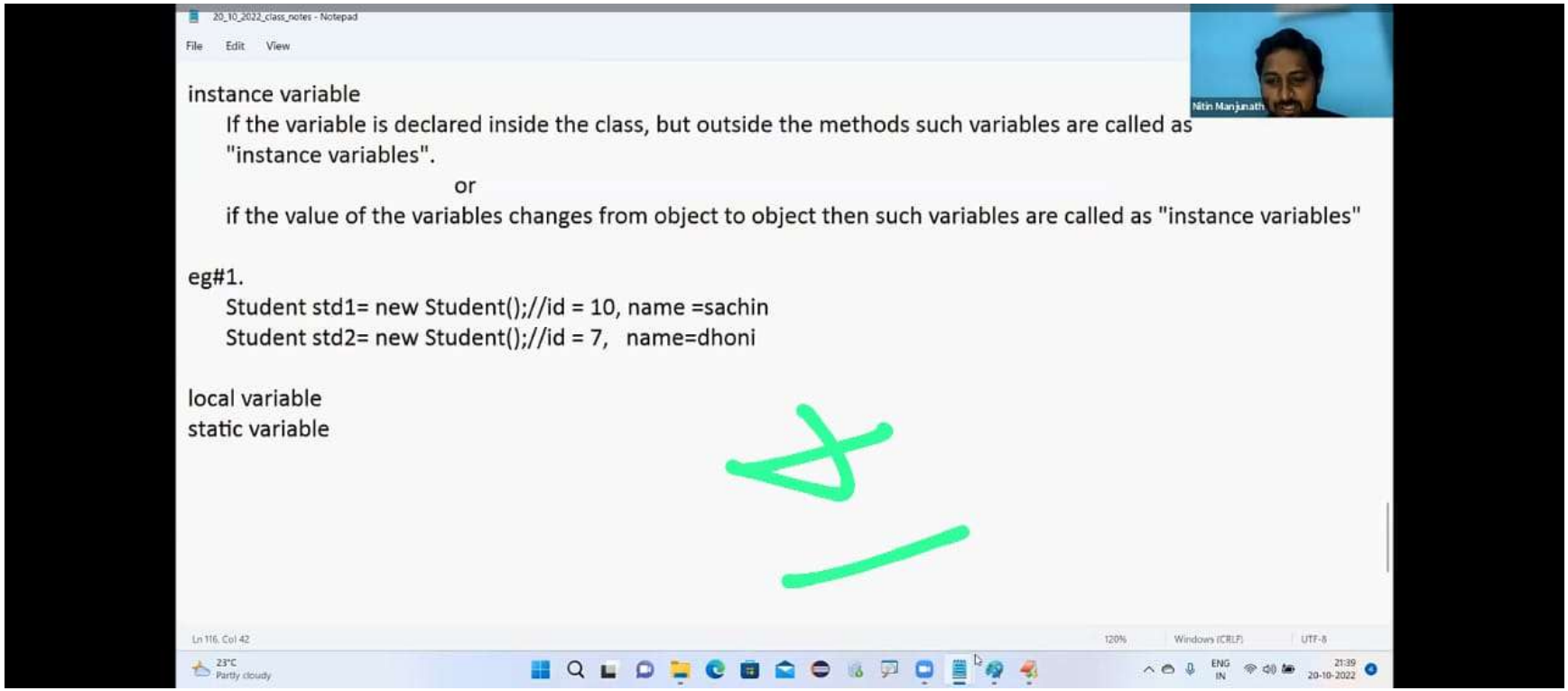
It is actually theory concept,which is implemented by many programming language like c++,java,
Nitin Manjunath
Any real time problem can be solved if we follow oop's principle.
In OOP's, while solving the problem
1. We need to first mark the Objects.
2. Every Object we mark should have 2 parts
a. HAS-Part/fields/attributes (store the information as variables)
b. Does-Part/behaviours(represent them as methods)
3. To represent an Object, first we need 2 have a blueprint of an Object.
What is Object?
Physical existense of any element we say as Object.
eg: book,Car,Computer,Dog,Student,.....
What is Has-Part and What is Does-part of an object represents?
HAS-Part => indicates what it can hold
Does-Part =>indicates what it can do
eg: Student
Ln 29, Col 31
120%
Windows (CRLF)
UTF-8
22°C
Partly cloudy
ENG
IN
20:45
20-10-2022

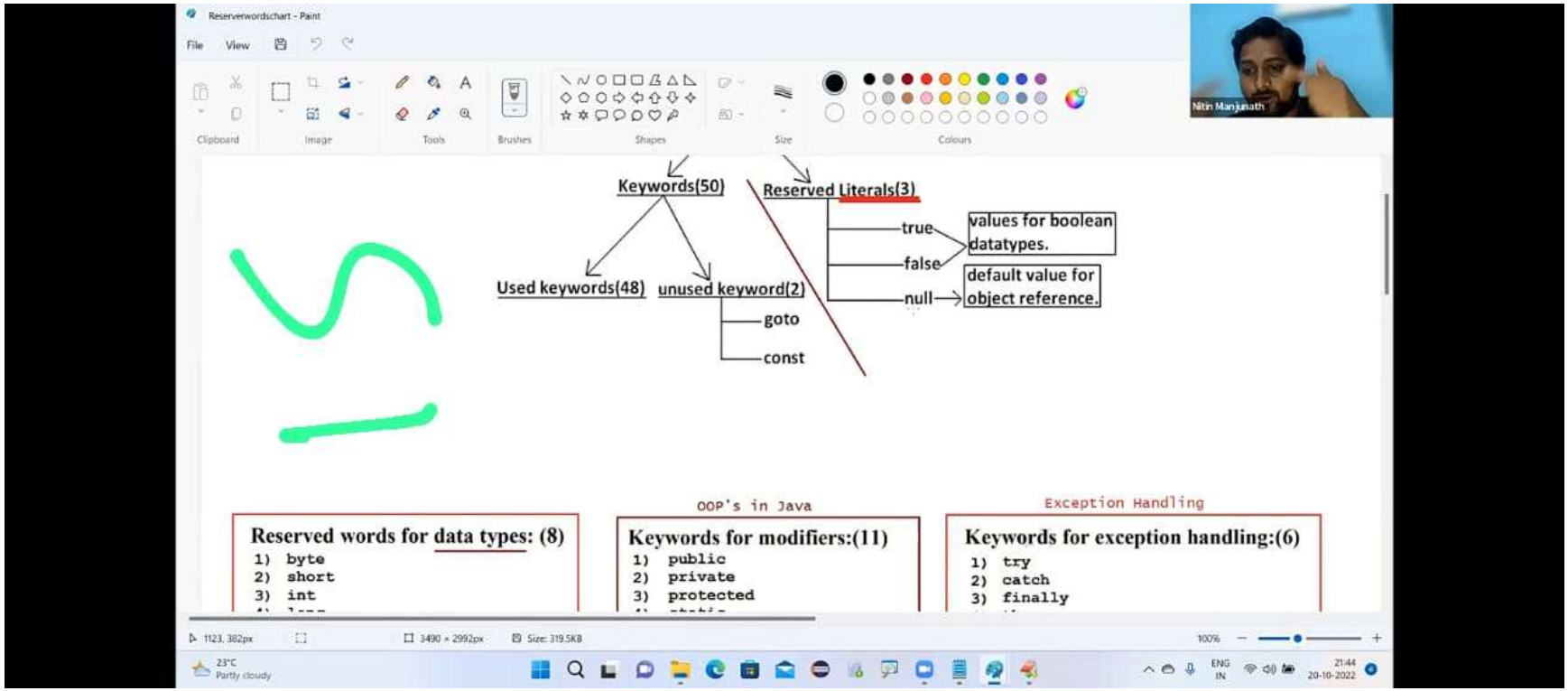

*20_10_2022_class_notes - Notepad
File Edit View
eg#1.
//Blue print of Student Object
class Student{//Student -> PascalConvention
//HAS-Part ----> camelCaseConvention
int sid;
String name;
int age;
char gender;
String address;
//Does-Part ----> camelCaseConvention
void play(){}
void study(){}
void drink(){}
void sleep(){}
}
Nitin Manjunath
Ln 55, Col 31
120%
Windows (CRLF)
UTF-8
22°C
Partly cloudy
ENG
IN
20:46
20-10-2022

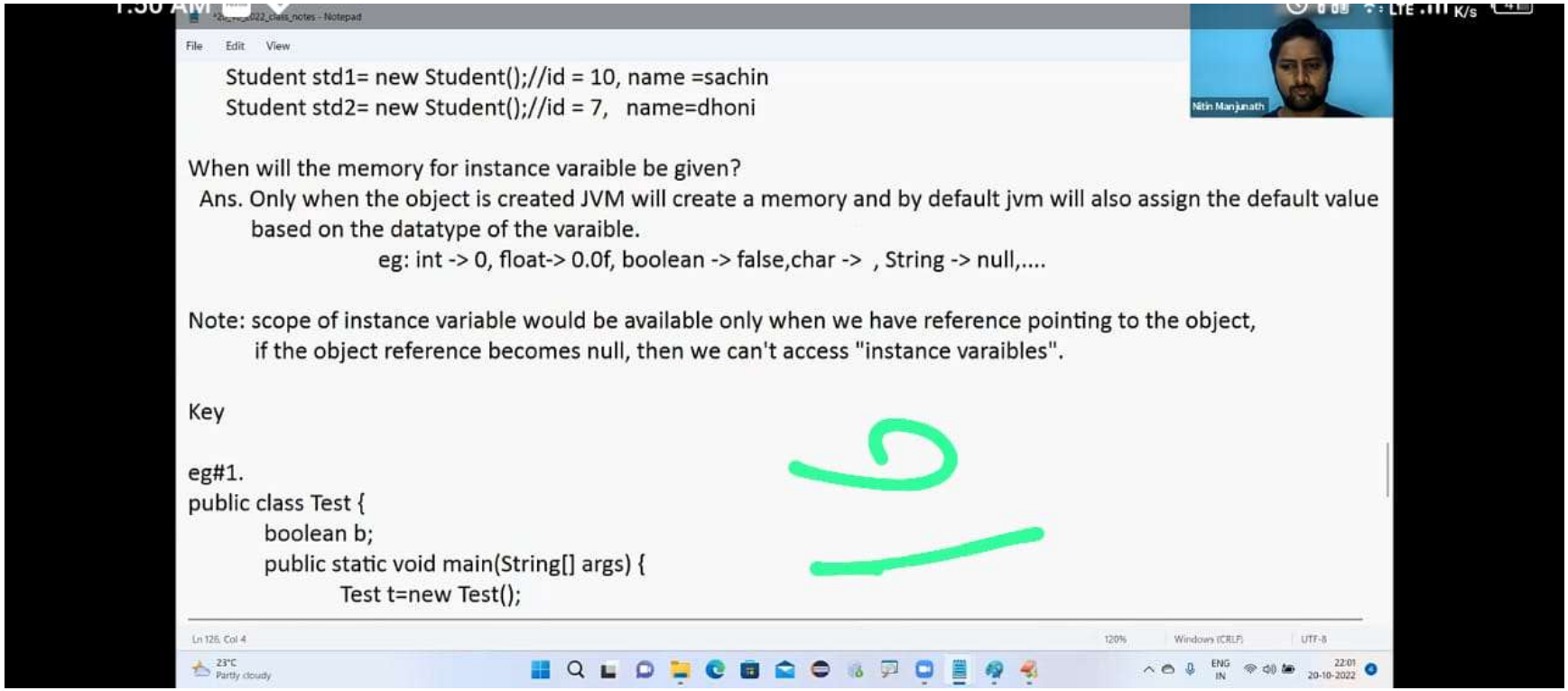

Untitled - Paint
File
View
Clipboard
Image
Tools
Brushes
Shapes
Size
Colours
Nitin Manjunath
Lucida Console
14
Background fill
Method Area
.class file
Heap Area
instance variable
Stack Area
local variable
PC registers
native method area
JVM(Java Virtual Machine)
JRE(Java RunTime Environment)
(os allocates this space to execute java program)
RAM
java FileName
address of next instruction which needs to be executed
code of other langauges which is requried for java would be available here
1052, 264px
3490 × 2992px
100%
22°C
Partly cloudy
ENG
IN
21:08
20-10-2022

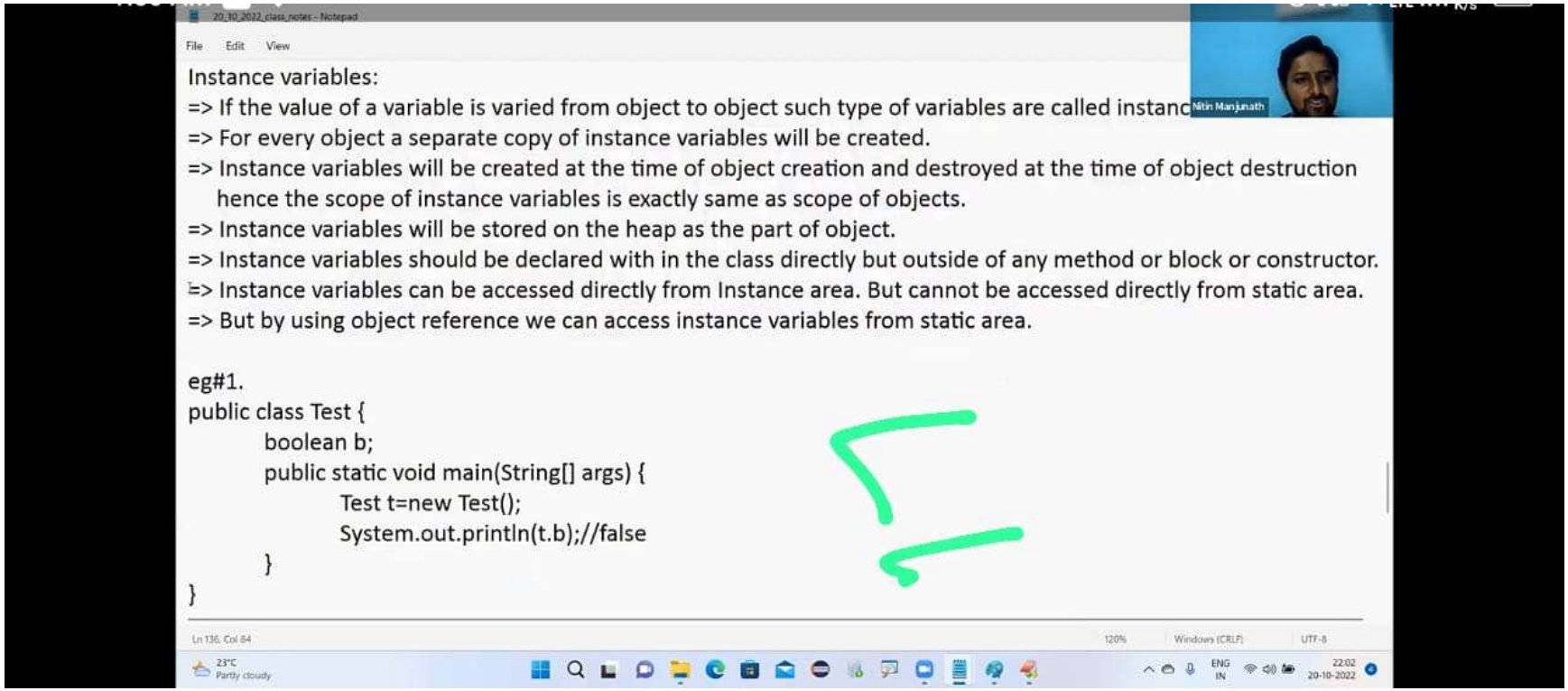

*20_10_2022_class_notes - Notepad
File Edit View
String address;
//Does-Part ----> camelCaseConvention
void play(){}
void study(){}
void drink(){}
void sleep(){}
}
To create an object in java we use "new" keyword
Syntax:
ClassName variable=new ClassName();
new -> it is a signal to jvm to create some space for the Object in the heap area.
Nitin Manjunath
Ln 76, Col 25
120%
Windows (CRLF)
UTF-8
22°C
Partly cloudy
ENG
IN
21:09
20-10-2022

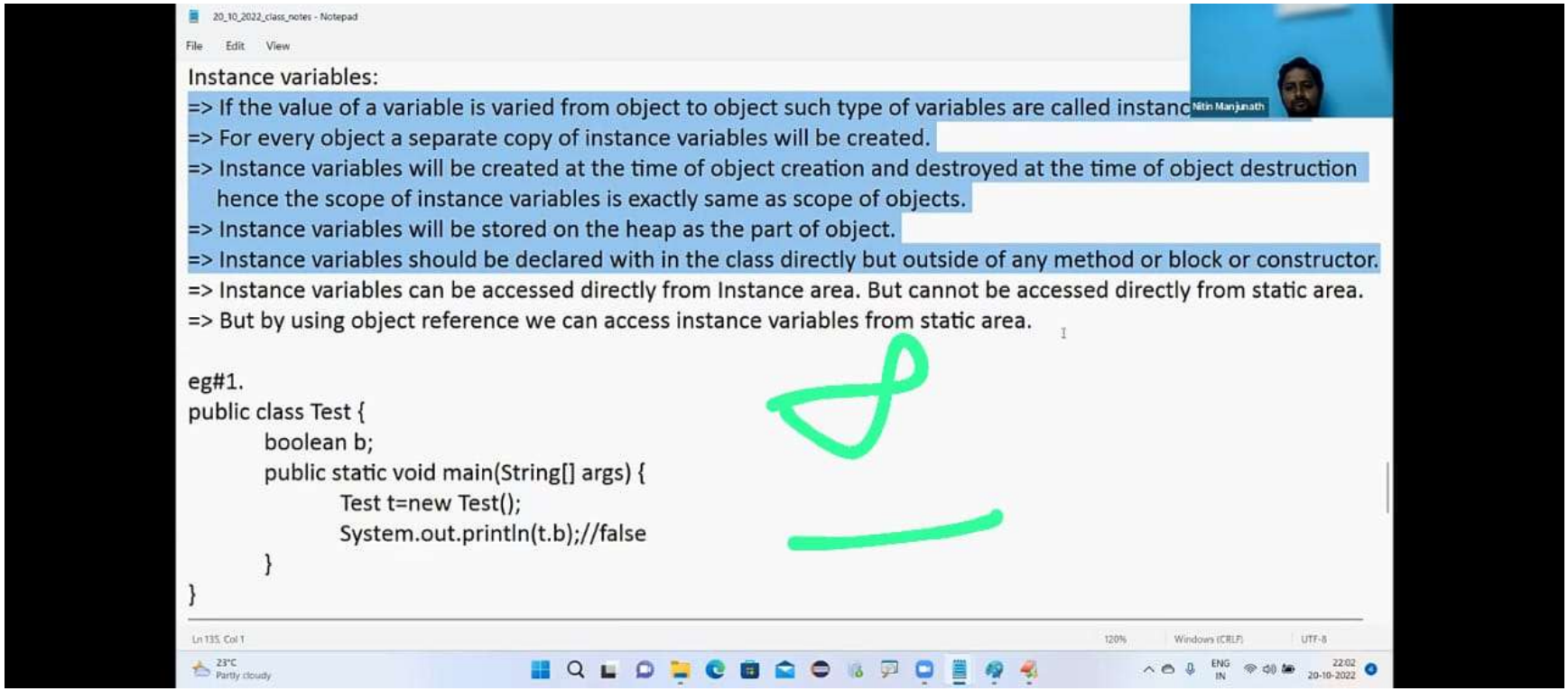

20_10_2022_class_notes - Notepad
File Edit View
To write instructions we need to have a language.
Nitin Manjunath
Types of variables
==========
Division 1 : Based on the type of value represented by a variable all variables are divided into 2 types.
They are:
1. Primitive variables
2. Reference variables
Primitive variables:
Primitive variables can be used to represent primitive values.
Example: int x=10;
Reference variables:
Reference variables can be used to refer objects.
Example: Student s=new Student();
int a = 10;
Student std = new Student();
Ln 96, Col 13
120%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
21:33
20-10-2022


20_10_2022_class_notes - Notepad
File Edit View
Nitin Manjunath
instance variable
If the variable is declared inside the class, but outside the methods such variables are called as "instance variables".
or
if the value of the variables changes from object to object then such variables are called as "instance variables"
eg#1.
Student std1= new Student();//id = 10, name =sachin
Student std2= new Student();//id = 7, name=dhoni
local variable
static variable
Ln 116, Col 42
120%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
21:39
20-10-2022


Reserverwordschart - Paint
File
View
Nitin Manjunath
Keywords(50)
Reserved Literals(3)
Used keywords(48)
unused keyword(2)
goto
const
true
false
null
values for boolean datatypes.
default value for object reference.
OOP's in Java
Exception Handling
Reserved words for data types: (8)
1) byte
2) short
3) int
Keywords for modifiers:(11)
1) public
2) private
3) protected
Keywords for exception handling:(6)
1) try
2) catch
3) finally


File Edit View
Student std1= new Student();//id = 10, name =sachin
Student std2= new Student();//id = 7, name=dhoni
Nitin Manjunath
When will the memory for instance varaible be given?
Ans. Only when the object is created JVM will create a memory and by default jvm will also assign the default value based on the datatype of the varaible.
eg: int -> 0, float-> 0.0f, boolean -> false,char -> , String -> null,....
Note: scope of instance variable would be available only when we have reference pointing to the object, if the object reference becomes null, then we can't access "instance varaibles".
Key
eg#1.
public class Test {
boolean b;
public static void main(String[] args) {
Test t=new Test();
Ln 126, Col 4
120%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
22:01
20-10-2022


20_10_2022_class_notes - Notepad
File Edit View
Instance variables:
=> If the value of a variable is varied from object to object such type of variables are called instanc
=> For every object a separate copy of instance variables will be created.
=> Instance variables will be created at the time of object creation and destroyed at the time of object destruction hence the scope of instance variables is exactly same as scope of objects.
=> Instance variables will be stored on the heap as the part of object.
=> Instance variables should be declared with in the class directly but outside of any method or block or constructor.
=> Instance variables can be accessed directly from Instance area. But cannot be accessed directly from static area.
=> But by using object reference we can access instance variables from static area.
eg#1.
public class Test {
boolean b;
public static void main(String[] args) {
Test t=new Test();
System.out.println(t.b);//false
}
}
Nitin Manjunath
Ln 136, Col 84
120%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
22:02
20-10-2022


20_10_2022_class_notes - Notepad
File Edit View
Instance variables:
=> If the value of a variable is varied from object to object such type of variables are called instanc
=> For every object a separate copy of instance variables will be created.
=> Instance variables will be created at the time of object creation and destroyed at the time of object destruction
hence the scope of instance variables is exactly same as scope of objects.
=> Instance variables will be stored on the heap as the part of object.
=> Instance variables should be declared with in the class directly but outside of any method or block or constructor.
=> Instance variables can be accessed directly from Instance area. But cannot be accessed directly from static area.
=> But by using object reference we can access instance variables from static area.
eg#1.
public class Test {
boolean b;
public static void main(String[] args) {
Test t=new Test();
System.out.println(t.b);//false
}
}
Nitin Manjunath
Ln 135, Col 1
120%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
22:02
20-10-2022

20_10_2022_class_notes - Notepad

File Edit View

}

local varaibles

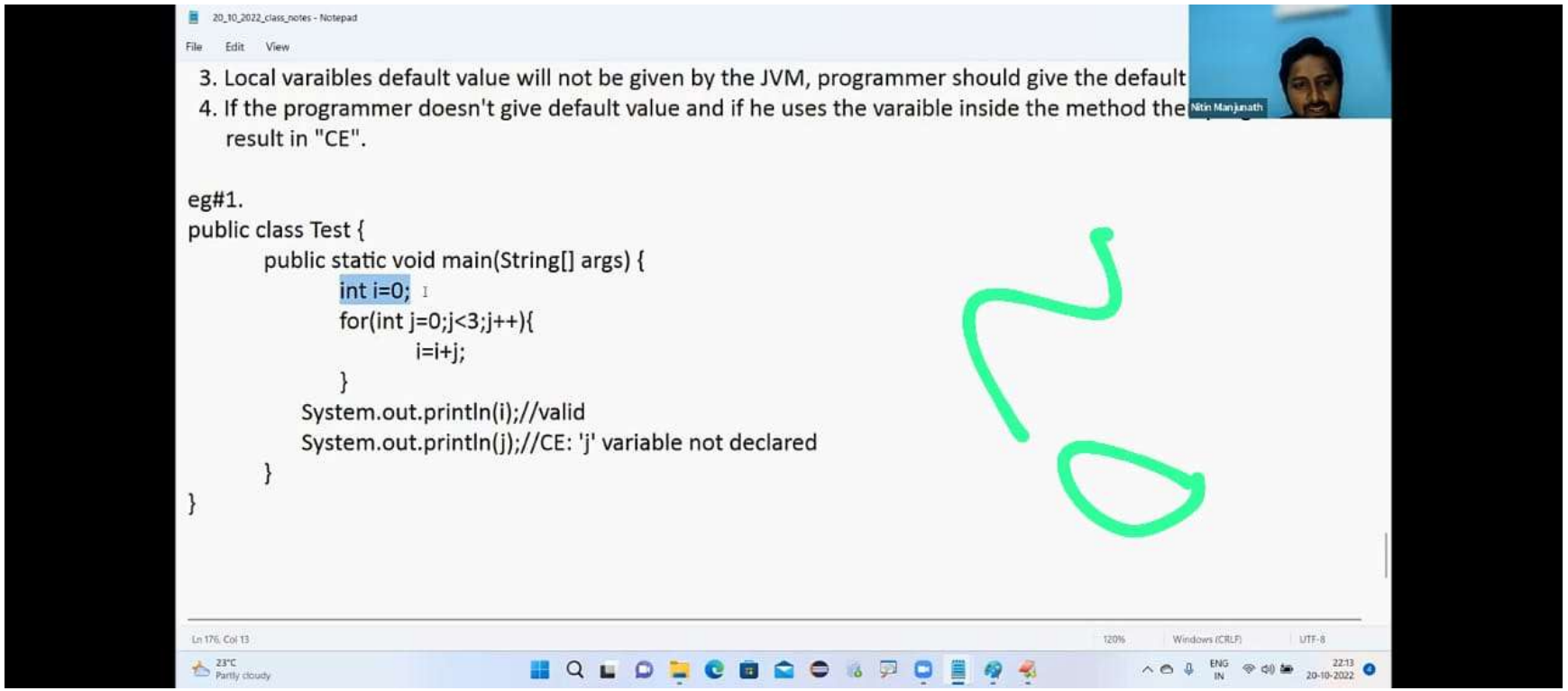
---

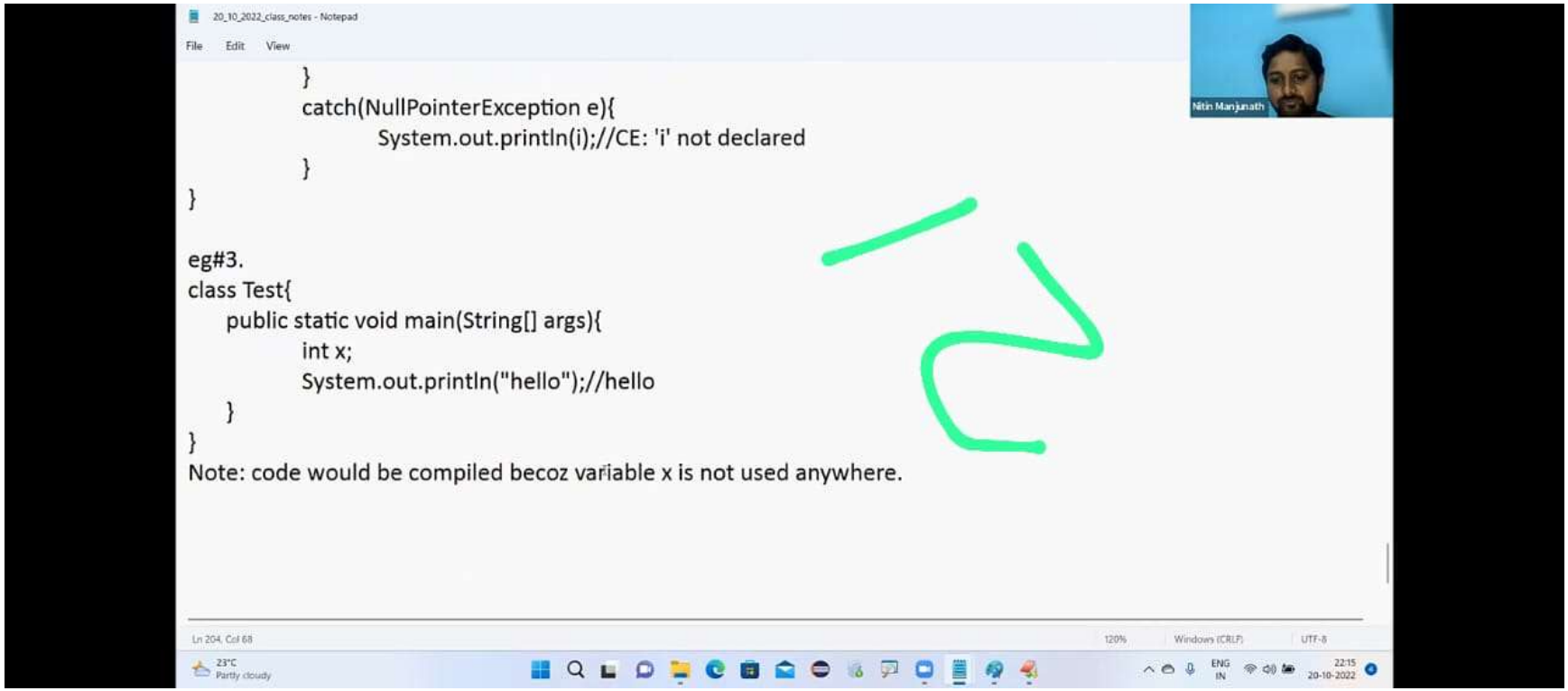
1. Variables which are created inside the method are called local variables and memory for those variables will be given in the stackarea.
2. During the execution of the method the memory for local variables will be given,and after the execution of the method the memory for variables will be taken out from the stack area.
3. Local varaibles default value will not be given by the JVM, programmer should give the default value.
4. If the programmer doesn't give default value and if he uses the varaible inside the method then program would result in "CE".

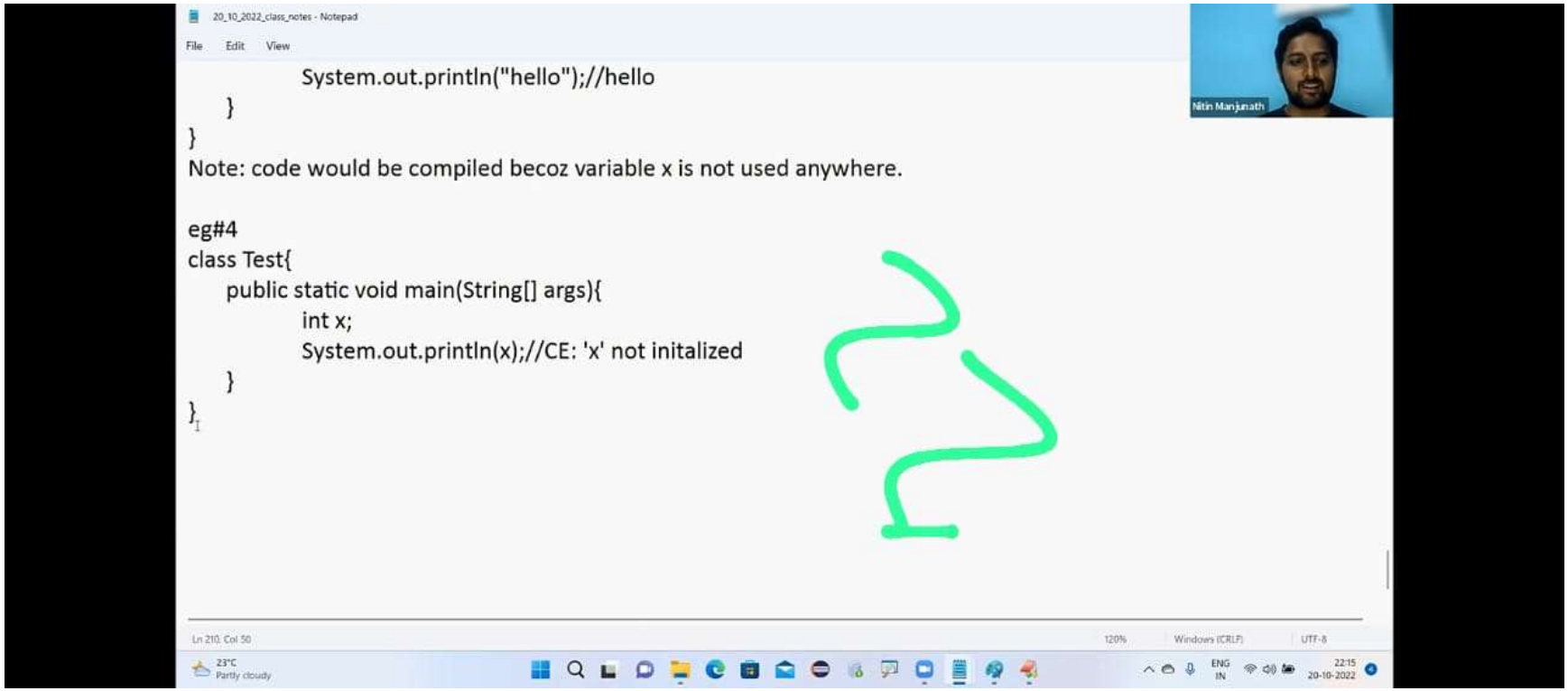
Ln 168, Col 80 120% Windows (CRLF) UTF-8

23°C Partly cloudy

ENG IN 22:11 20-10-2022


20_10_2022_class_notes - Notepad
File Edit View
3. Local varaibles default value will not be given by the JVM, programmer should give the default
4. If the programmer doesn't give default value and if he uses the varaible inside the method the
result in "CE".
eg#1.
public class Test {
public static void main(String[] args) {
int i=0;
for(int j=0;j<3;j++){
i=i+j;
}
System.out.println(i);//valid
System.out.println(j);//CE: 'j' variable not declared
}
}
Nitin Manjunath
Ln 176, Col 13
120%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
22:13
20-10-2022


20_10_2022_class_notes - Notepad
File Edit View
}
catch(NullPointerException e){
System.out.println(i);//CE: 'i' not declared
}
}
eg#3.
class Test{
public static void main(String[] args){
int x;
System.out.println("hello");//hello
}
}
Note: code would be compiled becoz variable x is not used anywhere.
Nitin Manjunath
Ln 204, Col 68
120%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
22:15
20-10-2022


20_10_2022_class_notes - Notepad
File Edit View
System.out.println("hello");//hello
}
}
Note: code would be compiled becoz variable x is not used anywhere.
eg#4
class Test{
public static void main(String[] args){
int x;
System.out.println(x);//CE: 'x' not initalized
}
}
Nitin Manjunath
Ln 210, Col 50
120%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
22:15
20-10-2022

20_10_2022_class_notes - Notepad

File Edit View

Local variables:

=> Some times to meet temporary requirements of the programmer we can declare variables inside a method or block or constructors such type of variables are called local variables or automatic variables or temporary variable or stack variables.

=> Local variables will be stored inside stack.

=> The local variables will be created as part of the block execution in which it is declared and destroyed once that block execution completes. Hence the scope of the local variables is exactly same as scope of the block in which we declared.

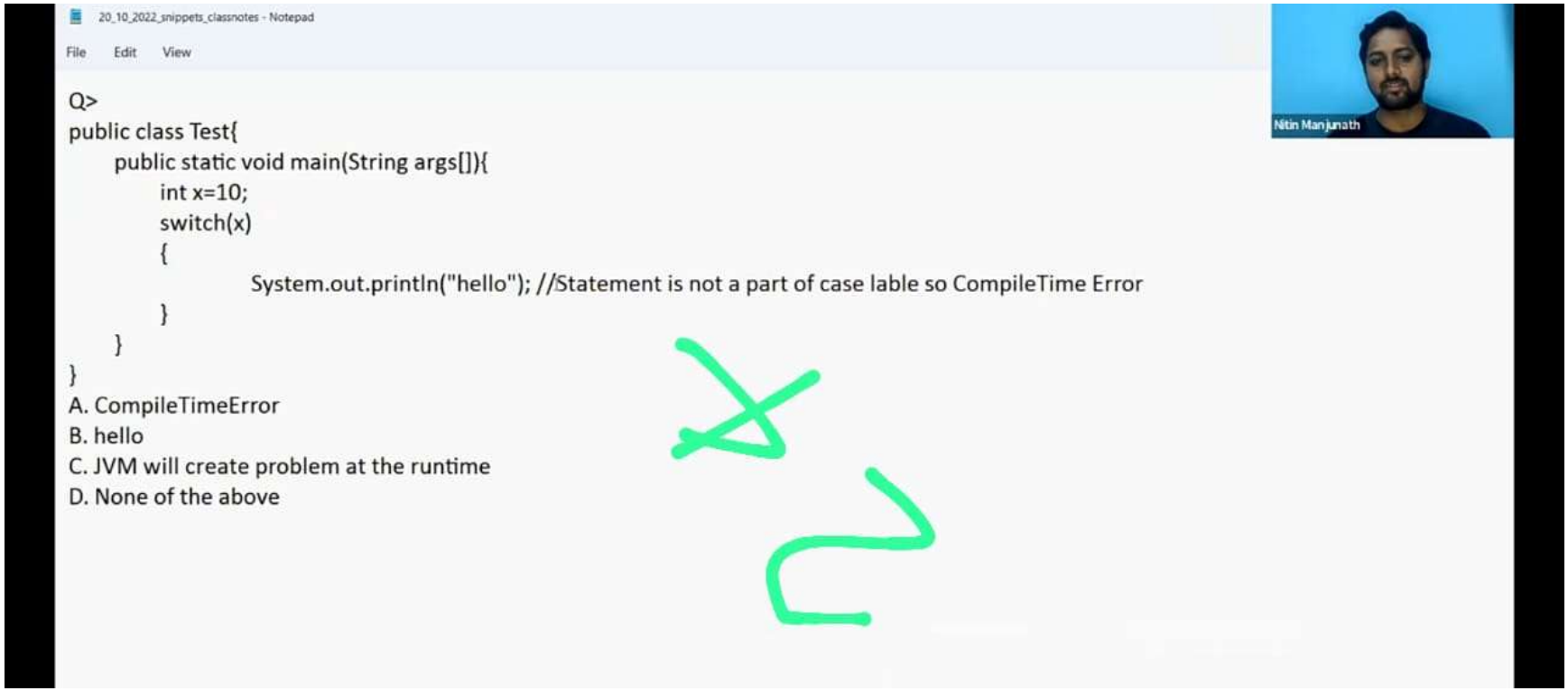
eg#1.

```
public class Test {
        public static void main(String[] args) {
                int i=0;
                for(int j=0;j<3;j++)
                {
                        i=i+j;
                }
```

Nitin Manjunath

Ln 178, Col 7 120% Windows (CRLF) UTF-8

23°C Partly cloudy ENG IN 22:17 20-10-2022


20_10_2022_snippets_classnotes - Notepad
File Edit View
Q>
public class Test{
public static void main(String args[]){
int x=10;
switch(x)
{
System.out.println("hello"); //Statement is not a part of case lable so CompileTime Error
}
}
}
A. CompileTimeError
B. hello
C. JVM will create problem at the runtime
D. None of the above
Nitin Manjunath

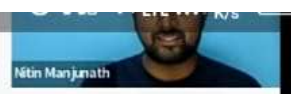
Q>

```
switch(args){
      case label1: stmt-1;
      case label2: stmt-2;
      default : stmt-n
}
```

label in switch should be "Compiletime Constants", meaning the value should be know to compiler.

```
public class Test{
      public static void main(String args[]){
            int x= 10;
            int y = 20;
            switch(x)
            {
                  case 10: System.out.println("hello");
                              break;
                  case y:System.out.println("hiee");
                              break;
            }
      }
}
```

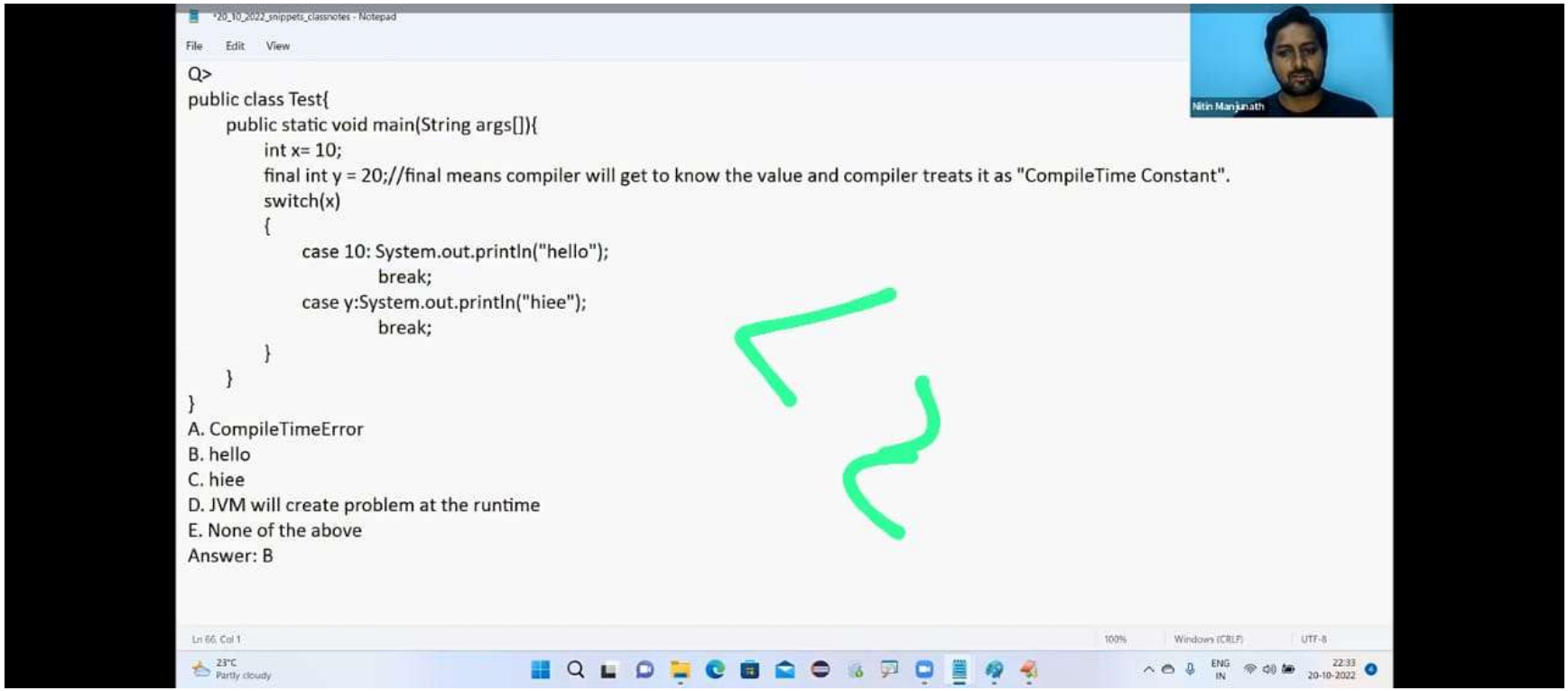
A. CompileTimeError



label in switch should be "Compiletime Constants", meaning the value should be know to compiler.


```
public class Test{
    public static void main(String args[]){
        int x= 10;
        int y = 20;
        switch(x)
        {
            case 10: System.out.println("hello");
                     break;
            case y:System.out.println("hiee"); //CE: 'y' value is not CompileTime Constant
                     break;
        }
    }
}
```

A. CompileTimeError
B. hello
C. hiee
D. JVM will create problem at the runtime
E. None of the above


*20_10_2022_snippets_classnotes - Notepad
File Edit View
Nitin Manjunath
Q>
public class Test{
public static void main(String args[]){
int x= 10;
final int y = 20;//final means compiler will get to know the value and compiler treats it as "CompileTime Constant".
switch(x)
{
case 10: System.out.println("hello");
break;
case y:System.out.println("hiee");
break;
}
}
}
A. CompileTimeError
B. hello
C. hiee
D. JVM will create problem at the runtime
E. None of the above
Answer: B
Ln 66, Col 1
100%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
22:33
20-10-2022

*20_10_2022_snippets_classnotes - Notepad

File Edit View

Answer: B

Q>

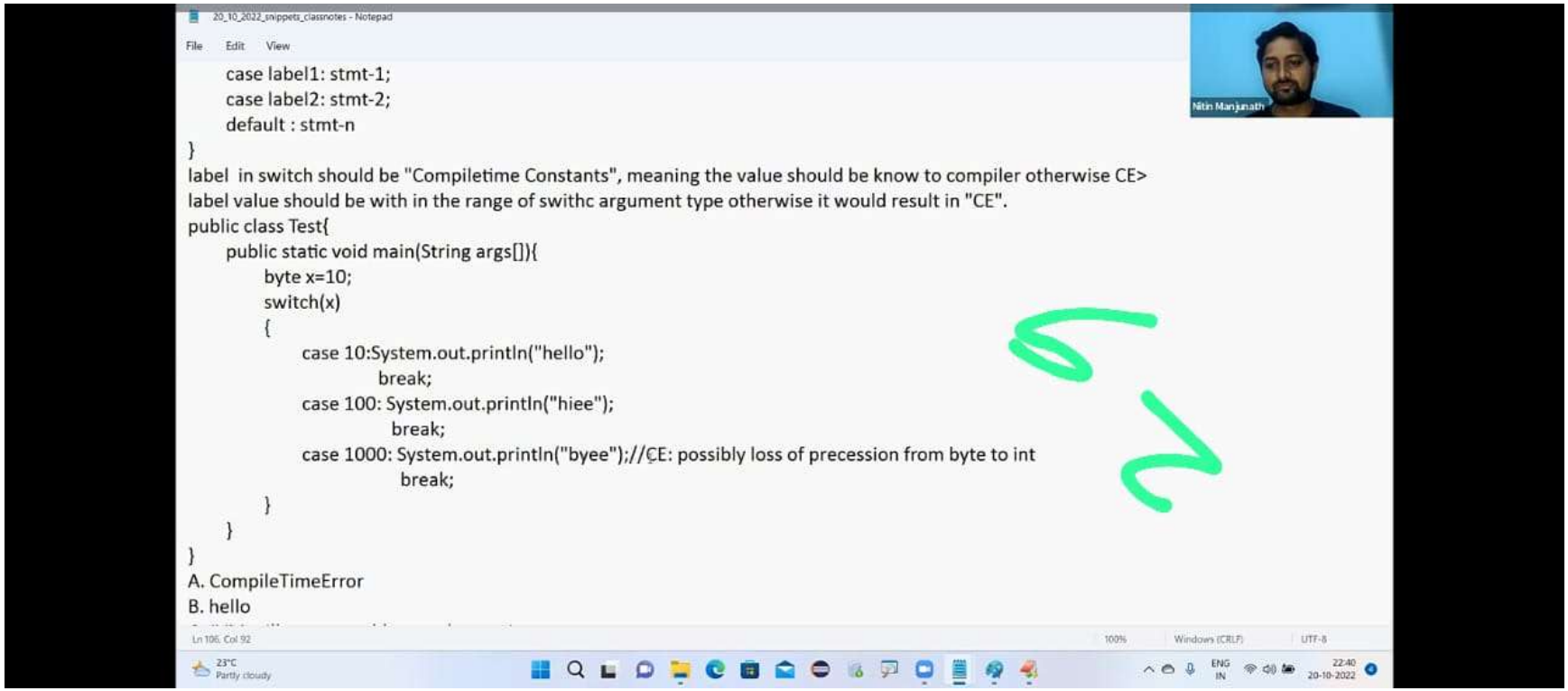
```
public class Test{
    public static void main(String args[]){
        int x=10;
        switch(x+1)
        {
            case 10:
            case 10+20:
            case  10+20+30:
        }
    }
}
```

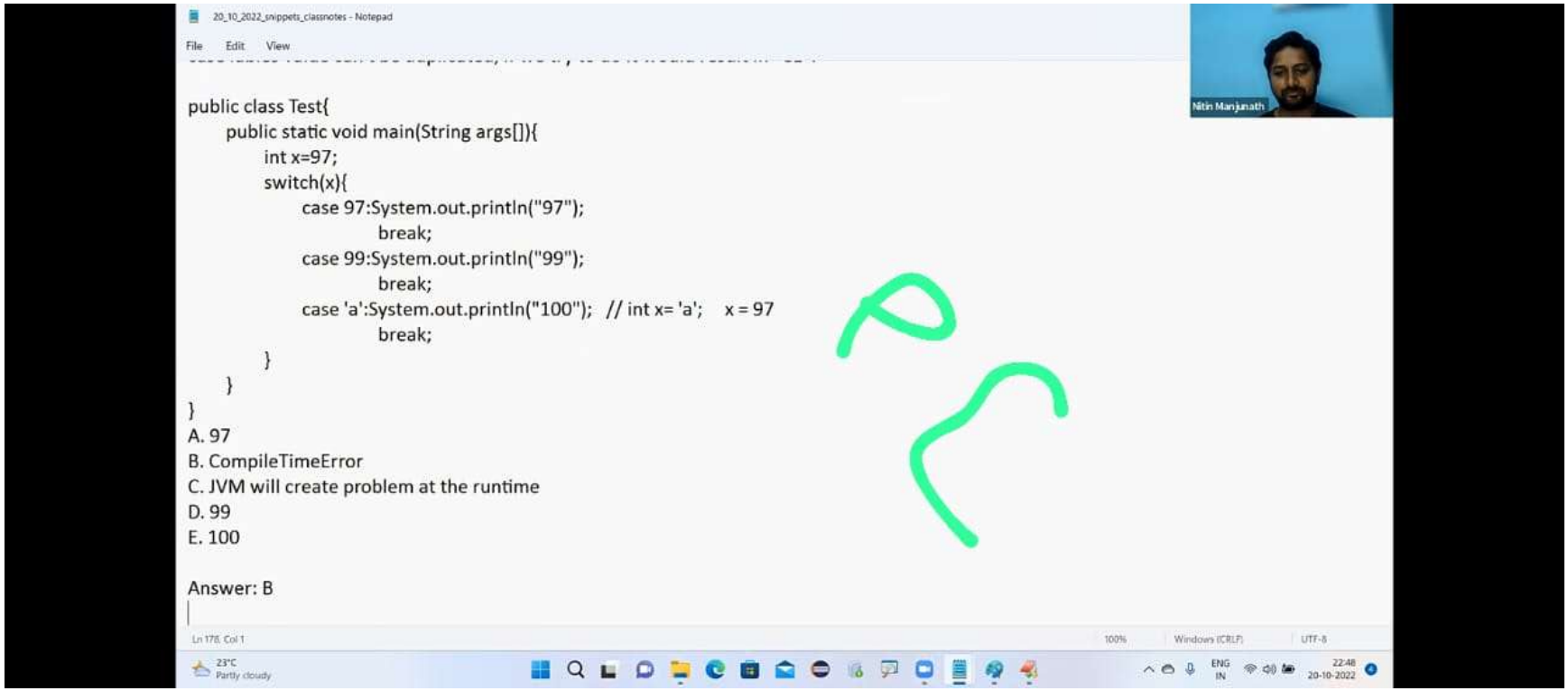
A. CompileTimeError
B. No Output
C. JVM will create problem at the runtime
D. None of the above

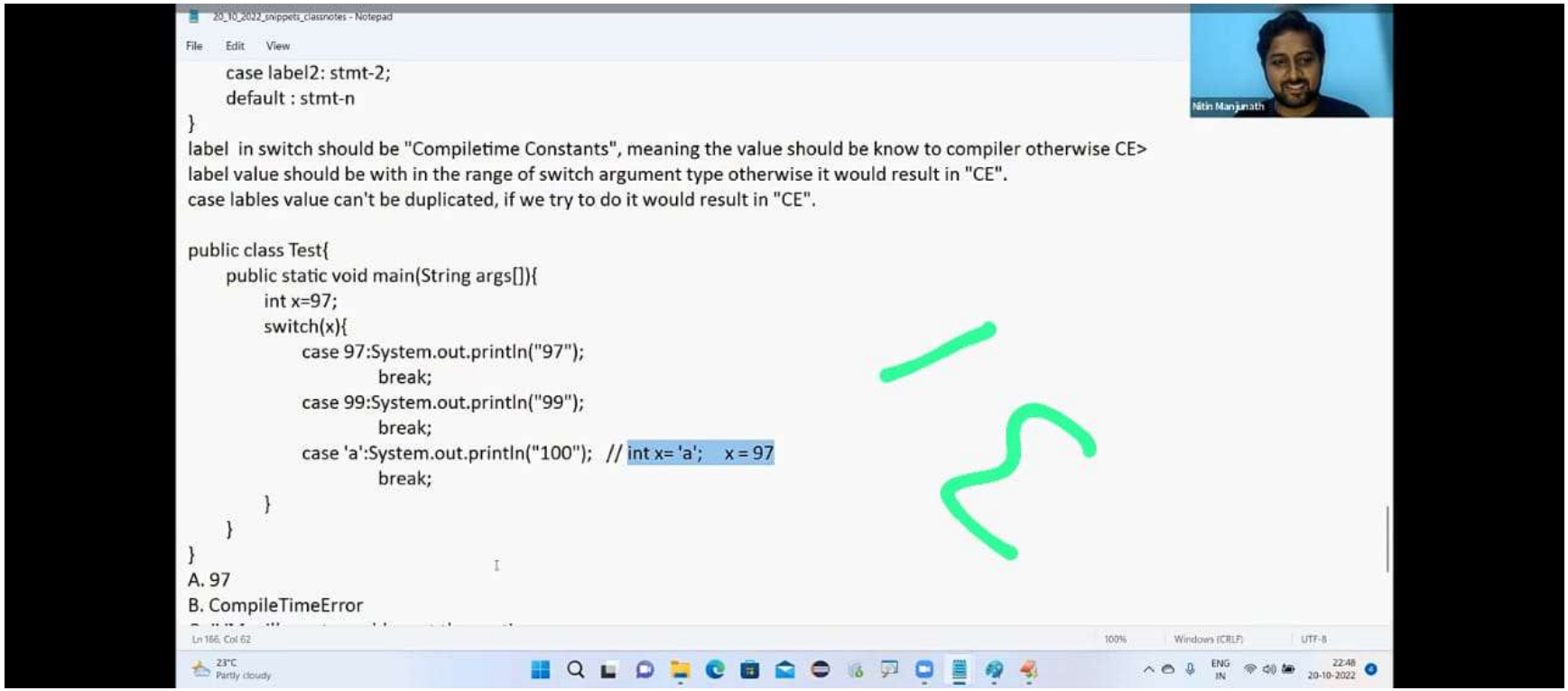
Answer: B

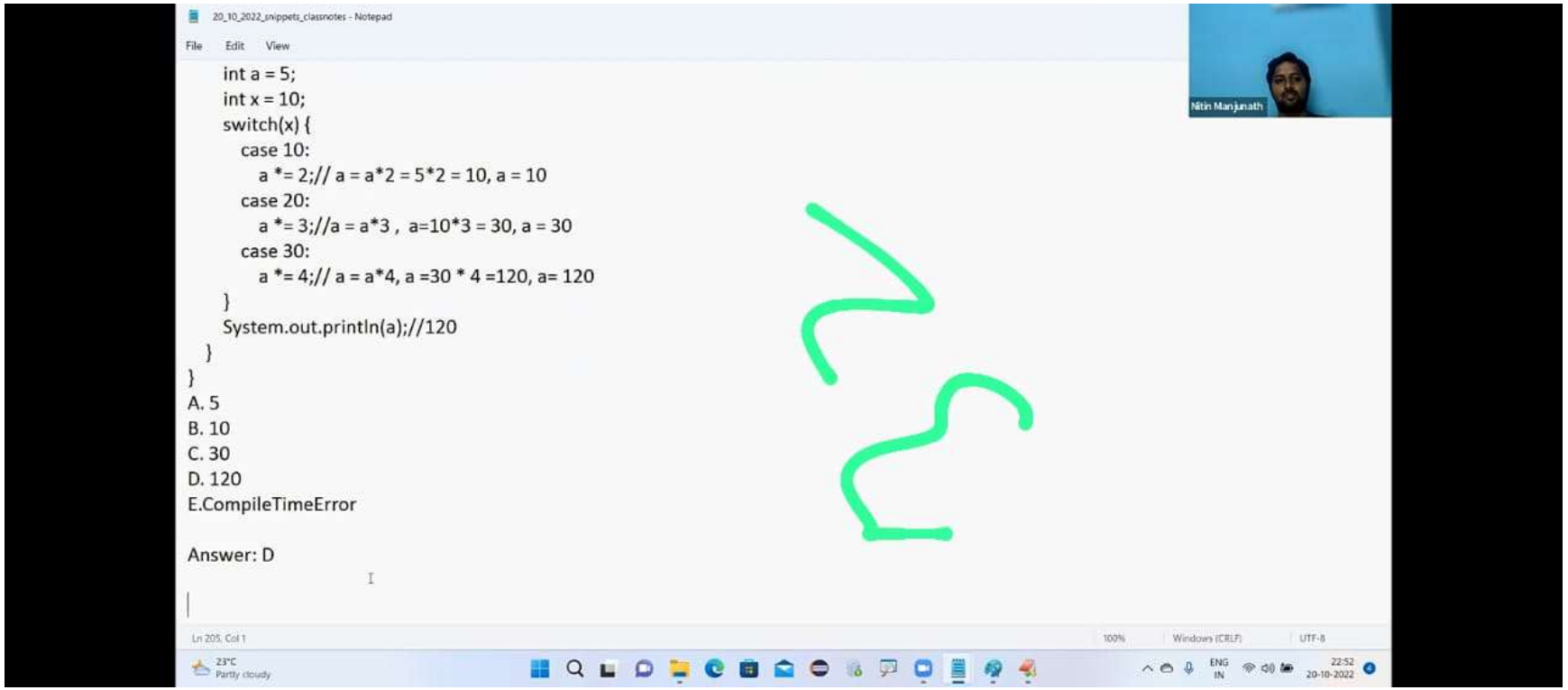
Ln 89, Col 1 100% Windows (CRLF) UTF-8

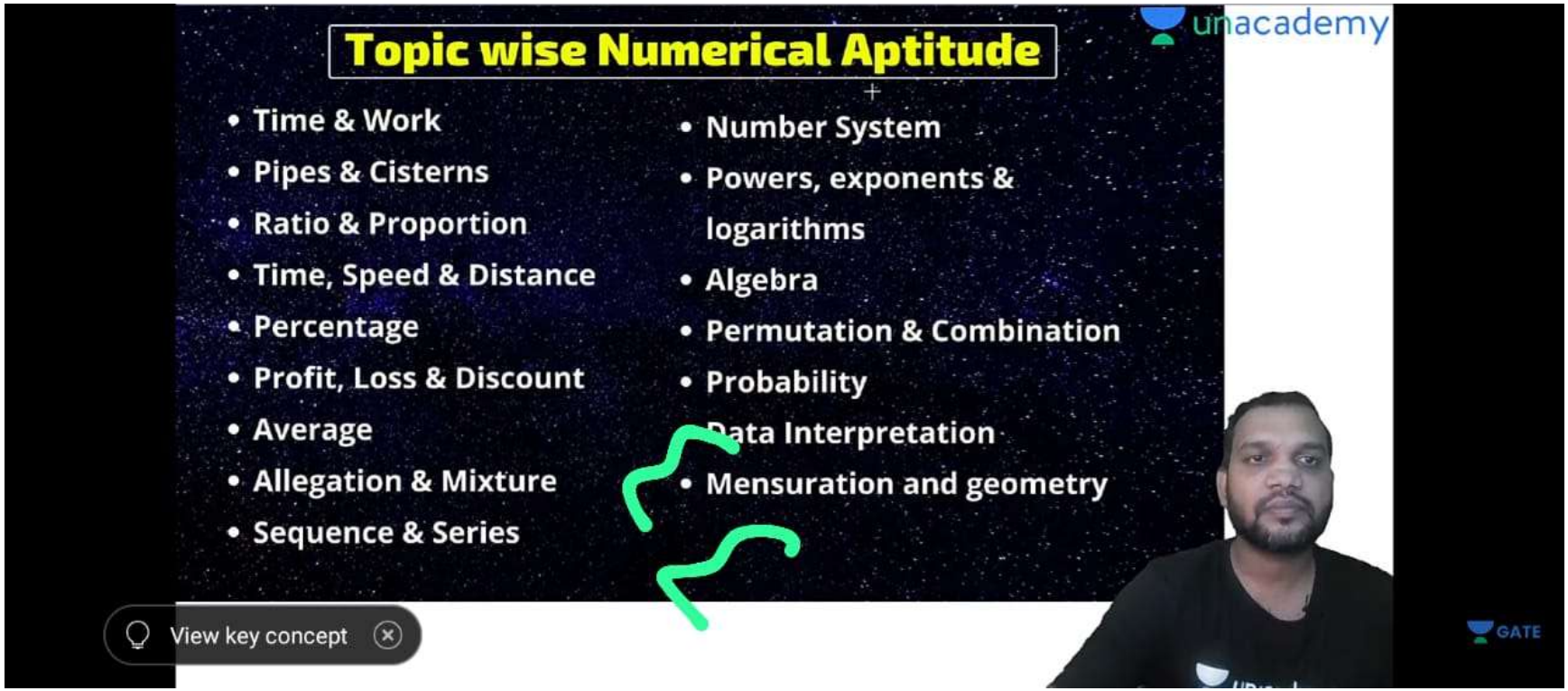
23°C Partly cloudy ENG IN 22:36 20-10-2022

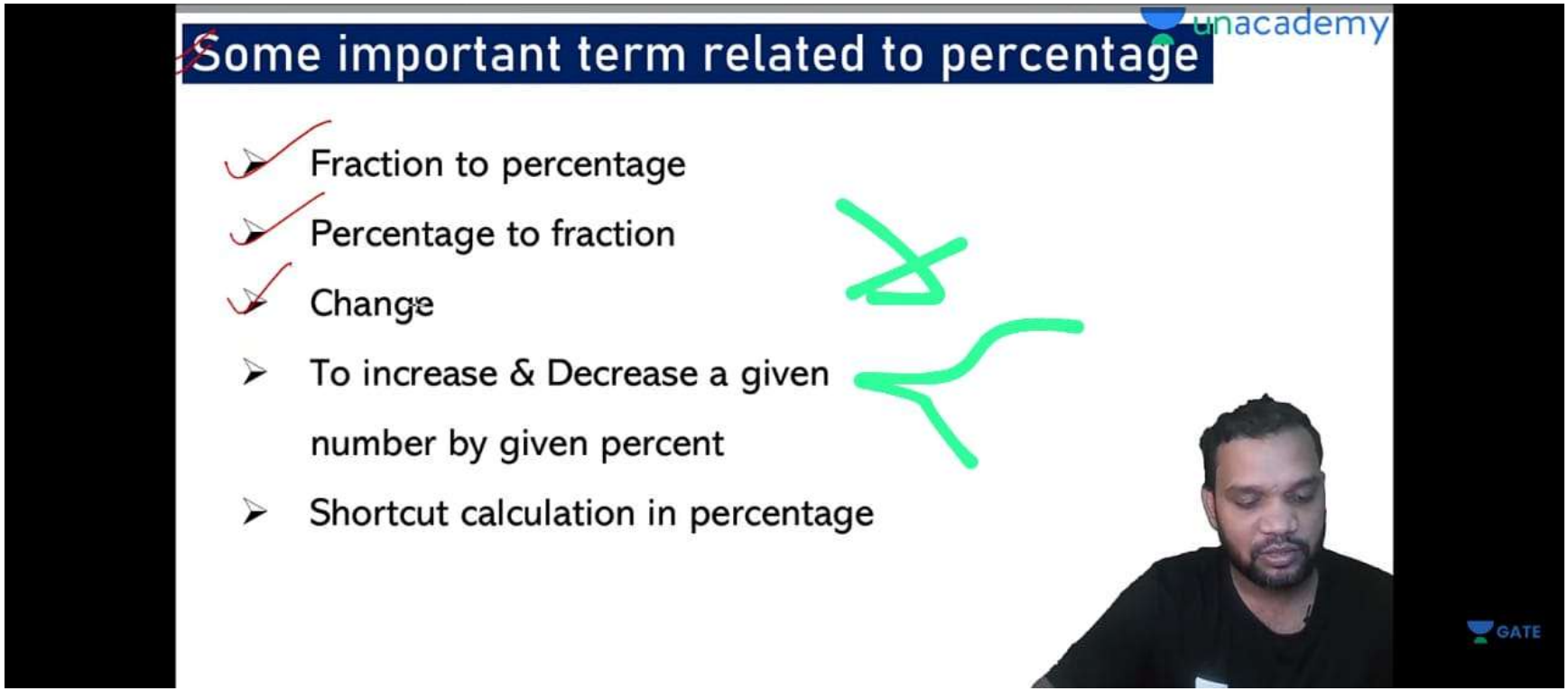

20_10_2022_snippets_classnotes - Notepad
File Edit View
Nitin Manjunath
case label1: stmt-1;
case label2: stmt-2;
default : stmt-n
}
label in switch should be "Compiletime Constants", meaning the value should be know to compiler otherwise CE>
label value should be with in the range of swithc argument type otherwise it would result in "CE".
public class Test{
public static void main(String args[]){
byte x=10;
switch(x)
{
case 10:System.out.println("hello");
break;
case 100: System.out.println("hiee");
break;
case 1000: System.out.println("byee");//CE: possibly loss of precession from byte to int
break;
}
}
}
A. CompileTimeError
B. hello
Ln 106, Col 92
100%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
22:40
20-10-2022


20_10_2022_snippets_classnotes - Notepad
File Edit View
Nitin Manjunath
public class Test{
public static void main(String args[]){
int x=97;
switch(x){
case 97:System.out.println("97");
break;
case 99:System.out.println("99");
break;
case 'a':System.out.println("100"); // int x= 'a'; x = 97
break;
}
}
}
A. 97
B. CompileTimeError
C. JVM will create problem at the runtime
D. 99
E. 100
Answer: B
Ln 178, Col 1
100%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
22:48
20-10-2022


20_10_2022_snippets_classnotes - Notepad
File Edit View
case label2: stmt-2;
default : stmt-n
}
label in switch should be "Compiletime Constants", meaning the value should be know to compiler otherwise CE>
label value should be with in the range of switch argument type otherwise it would result in "CE".
case lables value can't be duplicated, if we try to do it would result in "CE".
public class Test{
public static void main(String args[]){
int x=97;
switch(x){
case 97:System.out.println("97");
break;
case 99:System.out.println("99");
break;
case 'a':System.out.println("100"); // int x= 'a'; x = 97
break;
}
}
}
A. 97
B. CompileTimeError
Nitin Manjunath
Ln 166, Col 62
100%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
22:48
20-10-2022


20_10_2022_snippets_classnotes - Notepad
File Edit View
int a = 5;
int x = 10;
switch(x) {
case 10:
a *= 2;// a = a*2 = 5*2 = 10, a = 10
case 20:
a *= 3;//a = a*3 , a=10*3 = 30, a = 30
case 30:
a *= 4;// a = a*4, a =30 * 4 =120, a= 120
}
System.out.println(a);//120
}
}
A. 5
B. 10
C. 30
D. 120
E.CompileTimeError
Answer: D
Nitin Manjunath
Ln 205, Col 1
100%
Windows (CRLF)
UTF-8
23°C
Partly cloudy
ENG
IN
22:52
20-10-2022


Topic wise Numerical Aptitude
unacademy
• Time & Work
• Pipes & Cisterns
• Ratio & Proportion
• Time, Speed & Distance
• Percentage
• Profit, Loss & Discount
• Average
• Allegation & Mixture
• Sequence & Series
• Number System
• Powers, exponents & logarithms
• Algebra
• Permutation & Combination
• Probability
• Data Interpretation
• Mensuration and geometry
View key concept
GATE


unacademy
Some important term related to percentage
Fraction to percentage
Percentage to fraction
Change
To increase & Decrease a given number by given percent
Shortcut calculation in percentage
GATE