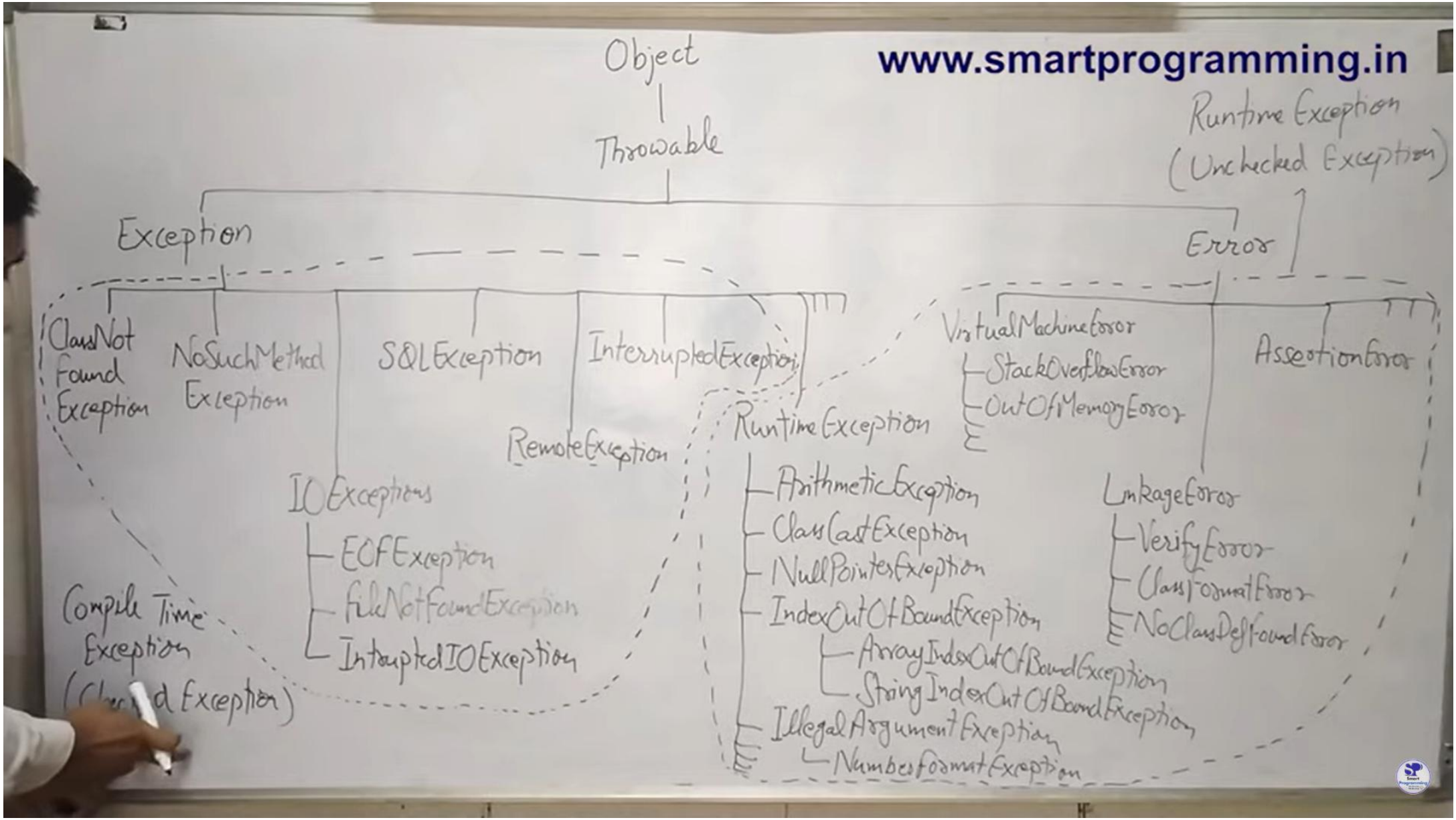
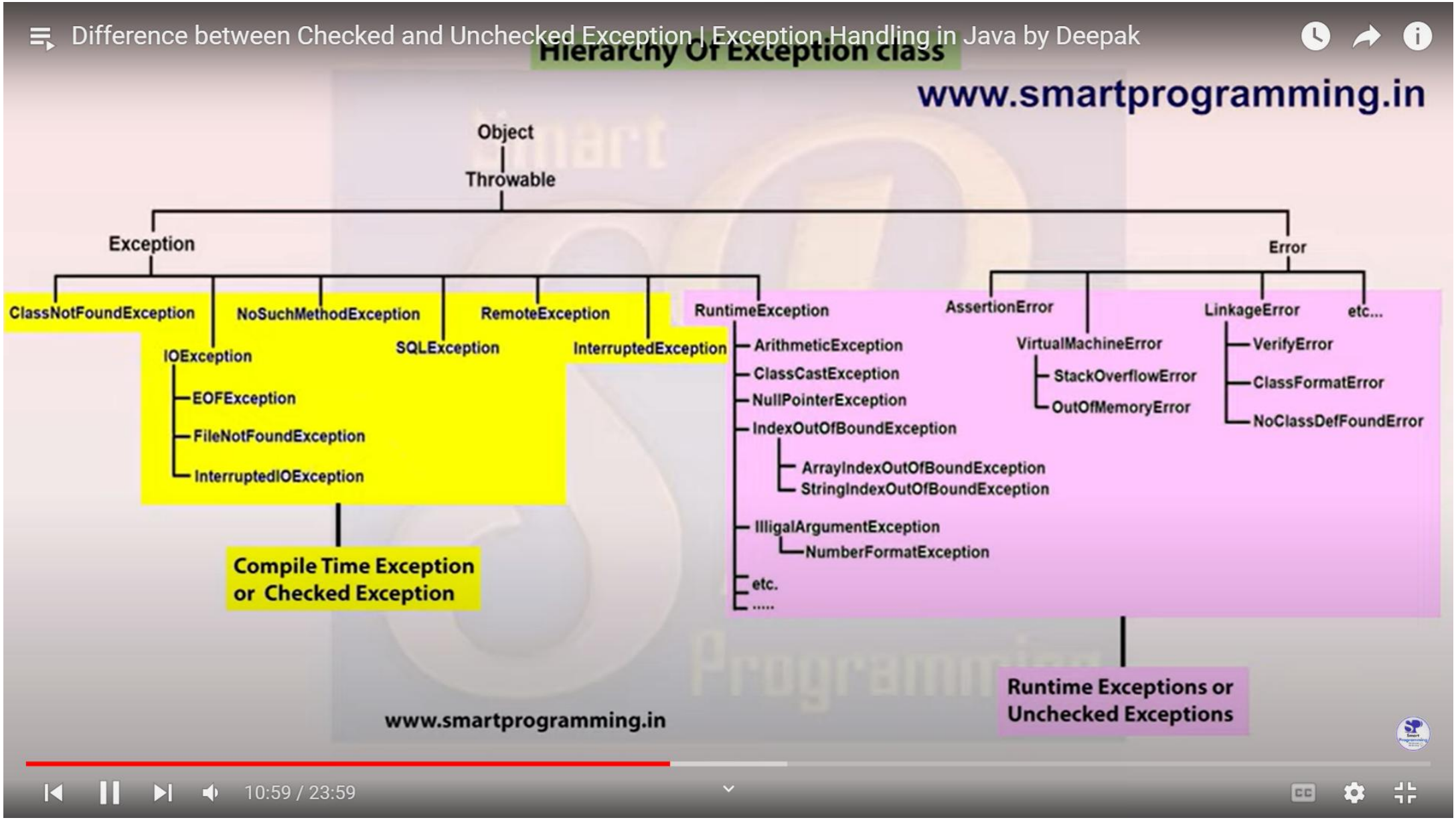
Difference between Exception & Error

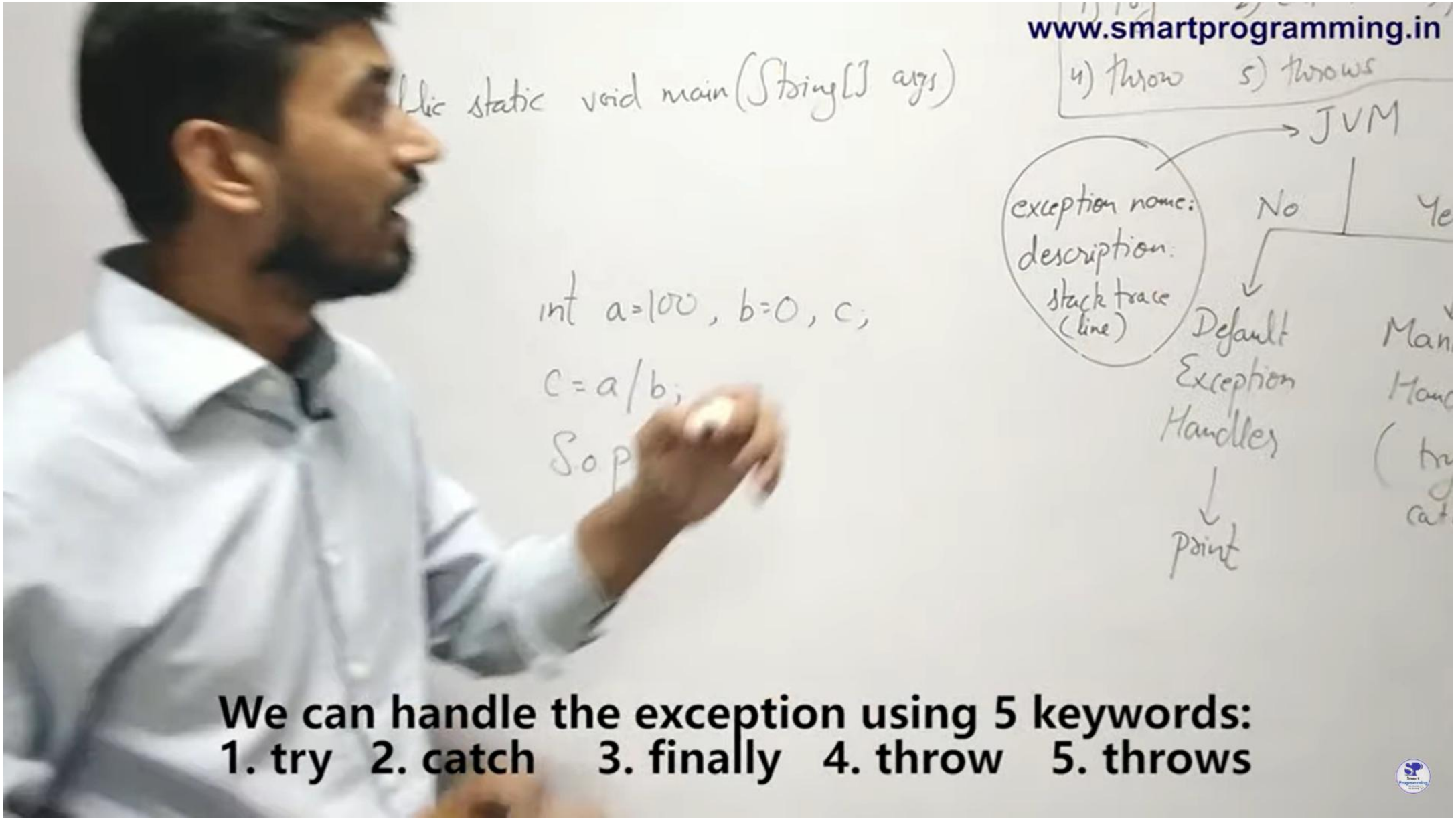
Exception	Error
1. Exception occurs because of our programs	1. Error occurs because of lack of system resources.
2. Exceptions are recoverable i.e. programmer can handle them using try-catch block	2. Errors are not recoverable i.e. programmer can handle them to their level
3. Exceptions are of two types : ■ Compile Time Exceptions or Checked Exceptions ■ Runtime Exceptions or Unchecked Exceptions	3. Errors are only of one type : ■ Runtime Exceptions or Unchecked Exceptions

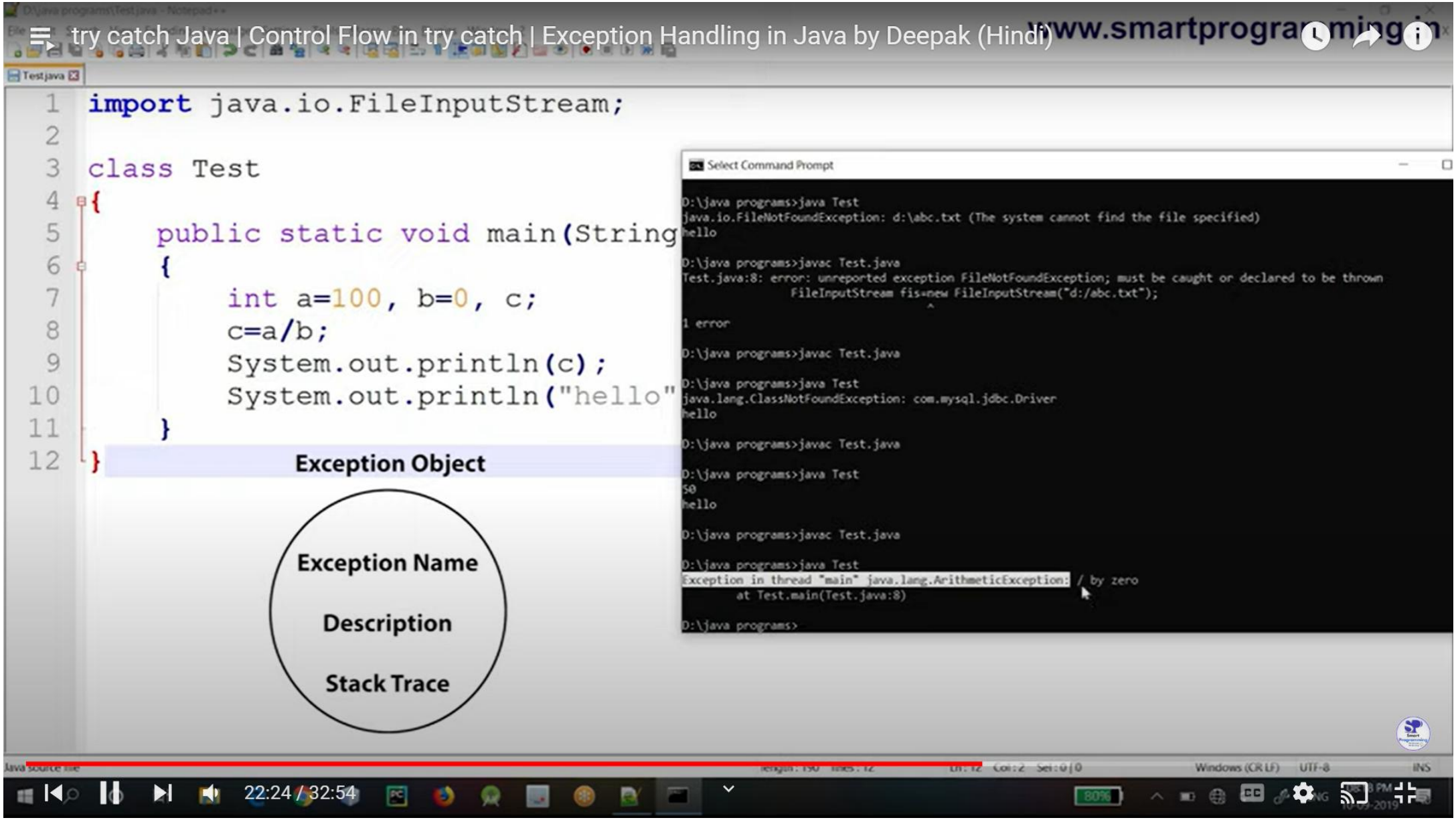

www.smartprogramming.in
Object
Throwable
Exception
Error
Runtime Exception
(Unchecked Exception)
ClassNot Found Exception
NoSuchMethod Exception
SQLException
InterruptedException
IOExceptions
EOFException
FileNotFoundException
InterruptedIOException
RemoteException
RuntimeException
ArithmeticException
ClassCastException
NullPointerException
IndexOutOfBoundException
ArrayIndexOutOfBoundException
StringIndexOutOfBoundException
IllegalArgumentException
NumberFormatException
VirtualMachineError
StackOverflowError
OutOfMemoryError
AssertionError
LinkageError
VerifyError
ClassFormatError
NoClassDefFoundError
Compile Time Exception
(Checked Exception)

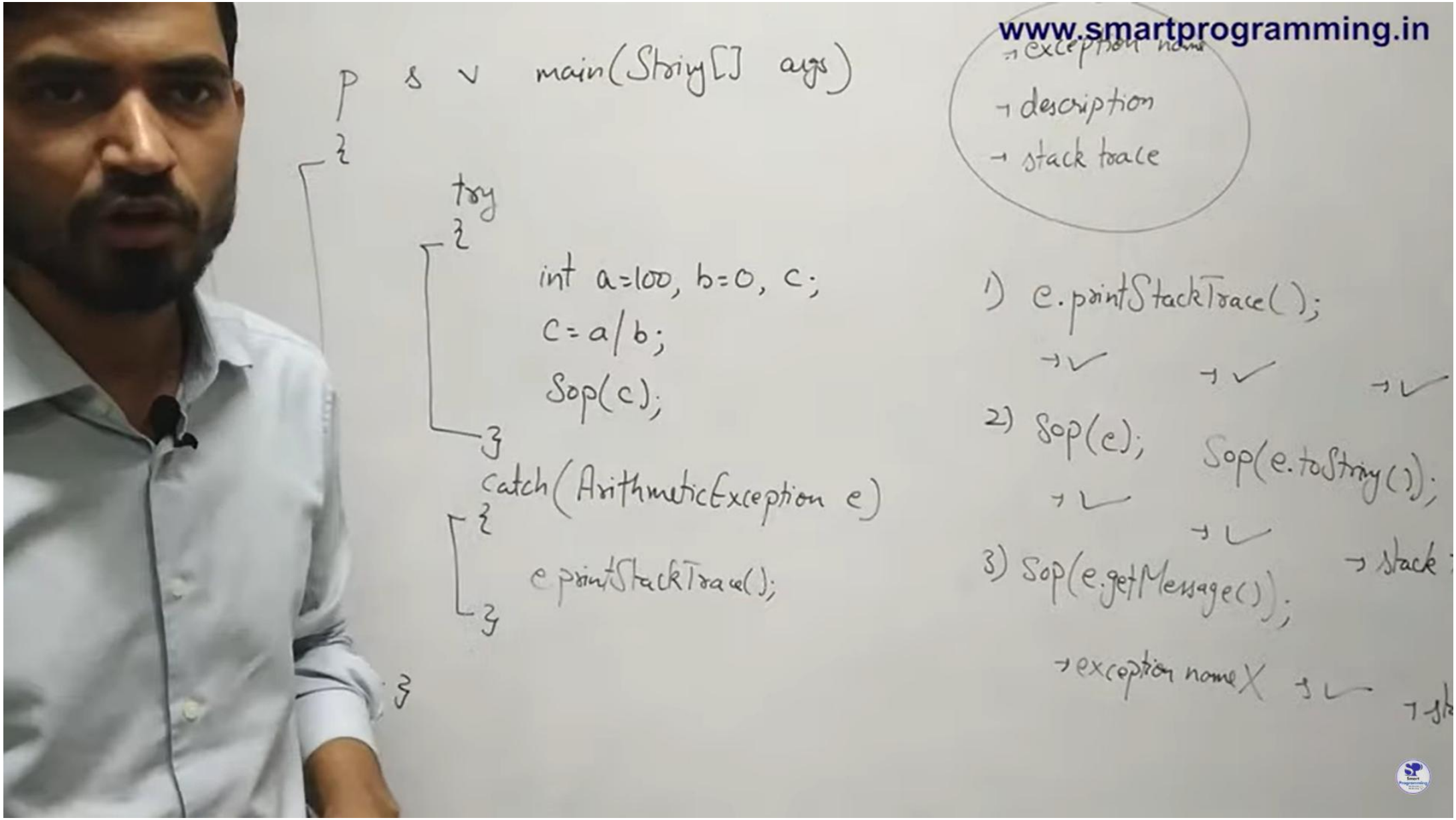

Difference between Checked and Unchecked Exception | Exception Handling in Java by Deepak
Hierarchy Of Exception class
www.smartprogramming.in
Object
Throwable
Exception
Error
ClassNotFoundException
NoSuchMethodException
RemoteException
IOException
SQLException
InterruptedException
EOFException
FileNotFoundException
InterruptedIOException
Compile Time Exception
or Checked Exception
RuntimeException
ArithmeticException
ClassCastException
NullPointerException
IndexOutOfBoundException
ArrayIndexOutOfBoundException
StringIndexOutOfBoundException
IlligalArgumentException
NumberFormatException
etc.
.....
AssertionError
VirtualMachineError
StackOverflowError
OutOfMemoryError
LinkageError
etc...
VerifyError
ClassFormatError
NoClassDefFoundError
Runtime Exceptions or
Unchecked Exceptions
www.smartprogramming.in
10:59 / 23:59

Difference Between Checked Exception and Unchecked Exception

Checked Exception / Compile Time Exception	Unchecked Exception / Runtime Exception
1. Checked Exceptions are the exceptions that are checked and handled at compile time.	1. Unchecked Exceptions are the exceptions that are not checked at compiled time.
2. The program gives a compilation error if a method throws a checked exception.	2. The program compiles fine because the compiler is not able to check the exception.
3. If some code within a method throws a checked exception, then the method must either handle the exception or it must specify the exception using throws keyword.	3. A method is not forced by compiler to declare the unchecked exceptions thrown by its implementation. Generally, such methods almost always do not declare them, as well.
4. A checked exceptions occur when the chances of failure are too high.	4. Unchecked exception occurs mostly due to programming mistakes.
5. They are direct subclass of Exception class but do not inherit from RuntimeException.	5. They are direct subclass of RuntimeException class.


www.smartprogramming.in
4) throw 5) throws
JVM
exception name:
description:
stack trace
(line)
No
Default
Exception
Handler
print
int a=100, b=0, c;
c=a/b;
We can handle the exception using 5 keywords:
1. try 2. catch 3. finally 4. throw 5. throws


try catch Java | Control Flow in try catch | Exception Handling in Java by Deepak (Hindi)
www.smartprogramming.in
Testjava
1 import java.io.FileInputStream;
2
3 class Test
4 {
5 public static void main(String
6 {
7 int a=100, b=0, c;
8 c=a/b;
9 System.out.println(c);
10 System.out.println("hello"
11 }
12 }
Exception Object
Exception Name
Description
Stack Trace
Select Command Prompt
D:\java programs>java Test
java.io.FileNotFoundException: d:\abc.txt (The system cannot find the file specified)
hello
D:\java programs>javac Test.java
Test.java:8: error: unreported exception FileNotFoundException; must be caught or declared to be thrown
FileInputStream fis=new FileInputStream("d:/abc.txt");
^
1 error
D:\java programs>javac Test.java
D:\java programs>java Test
java.lang.ClassNotFoundException: com.mysql.jdbc.Driver
hello
D:\java programs>javac Test.java
D:\java programs>java Test
50
hello
D:\java programs>javac Test.java
D:\java programs>java Test
Exception in thread "main" java.lang.ArithmeticException: / by zero
at Test.main(Test.java:8)
D:\java programs>
22:24 / 32:54


www.smartprogramming.in
psvm main(String[] args)
{
try
{
int a=100, b=0, c;
c=a/b;
Sop(c);
}
catch(ArithmeticException e)
{
e.printStackTrace();
}
}
→exception name
→description
→stack trace
1) e.printStackTrace();
2) Sop(e);
Sop(e.toString());
3) Sop(e.getMessage());
→exception name X

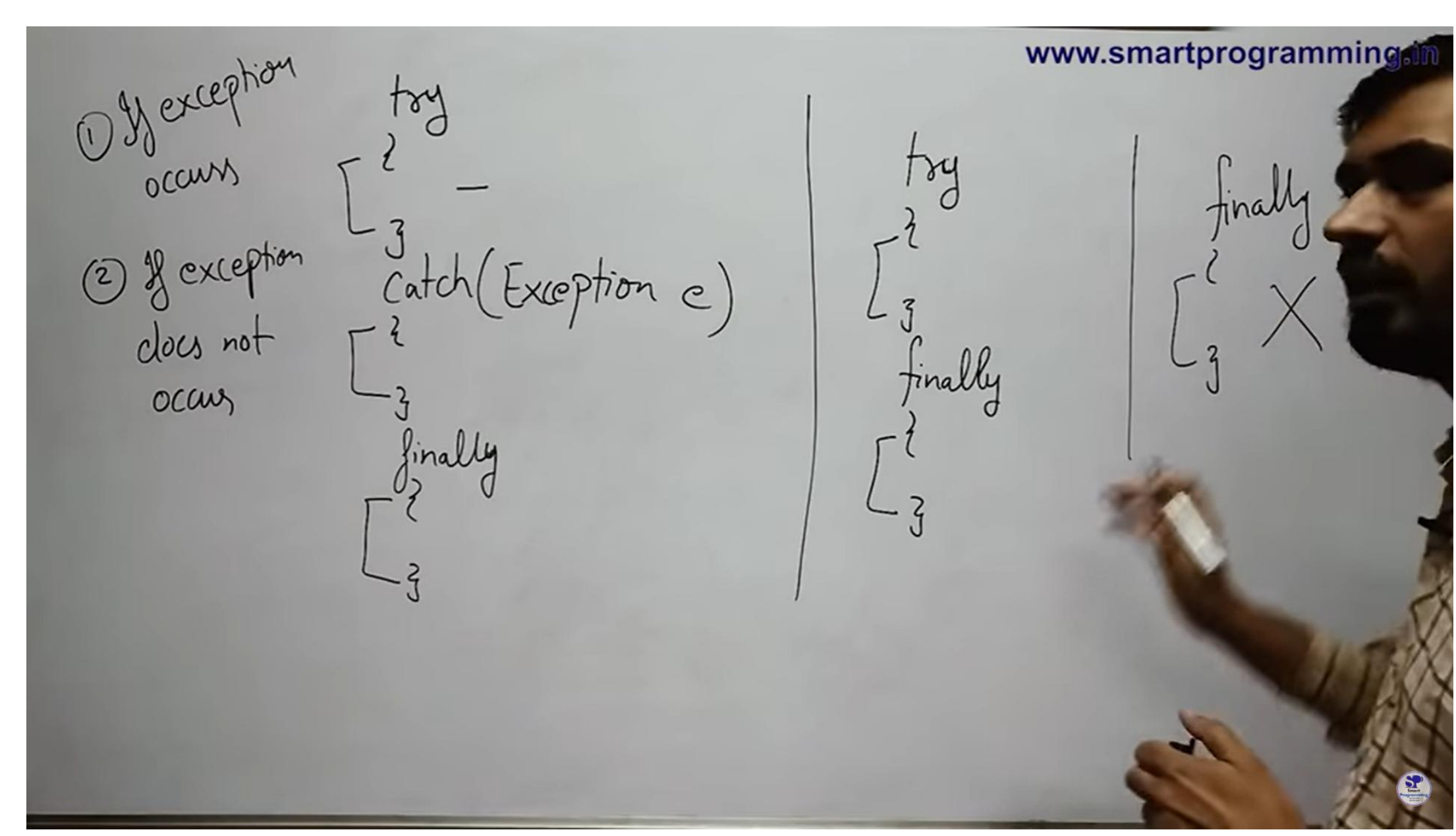
① If exception occurs

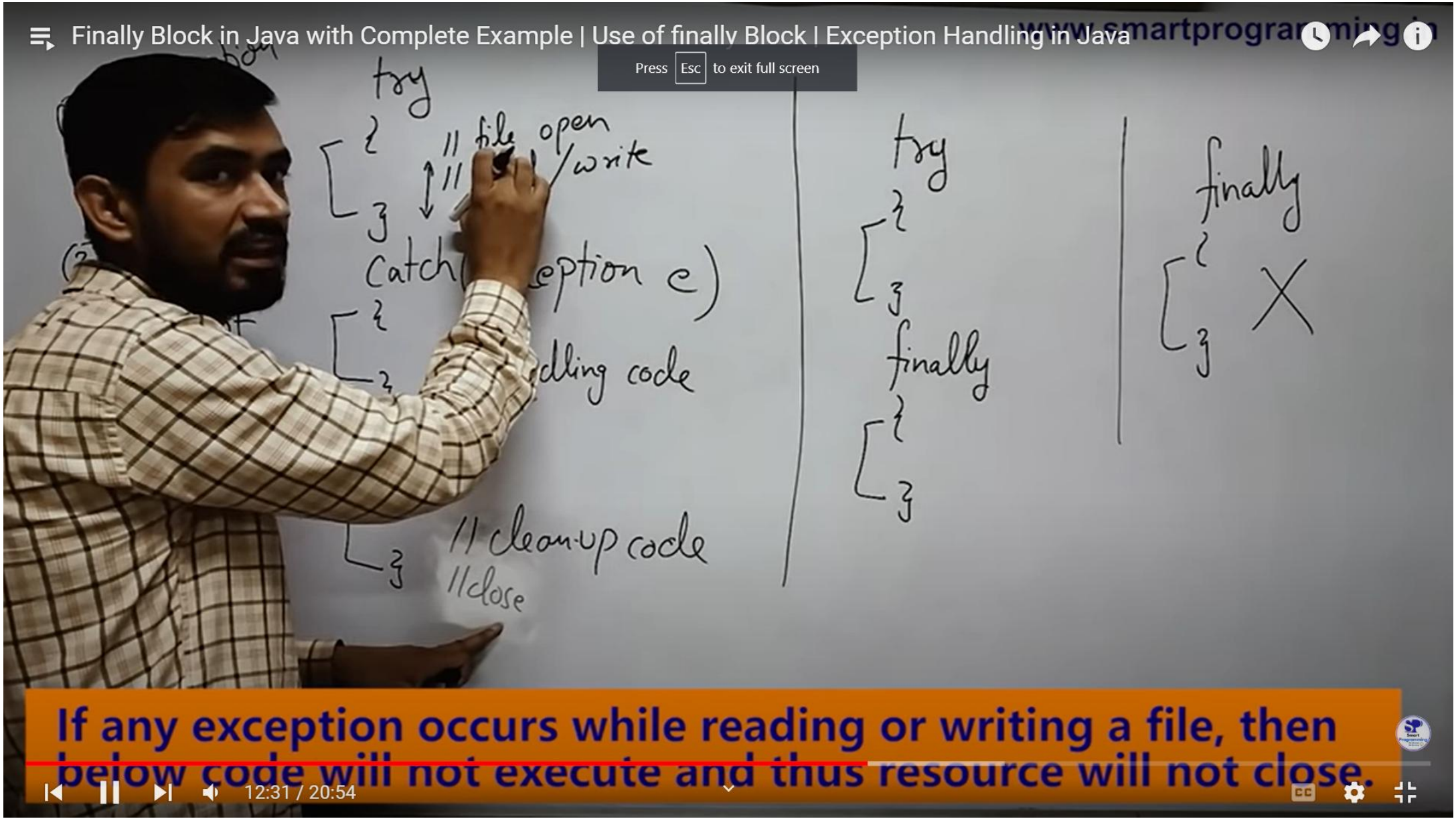
② If exception does not occurs

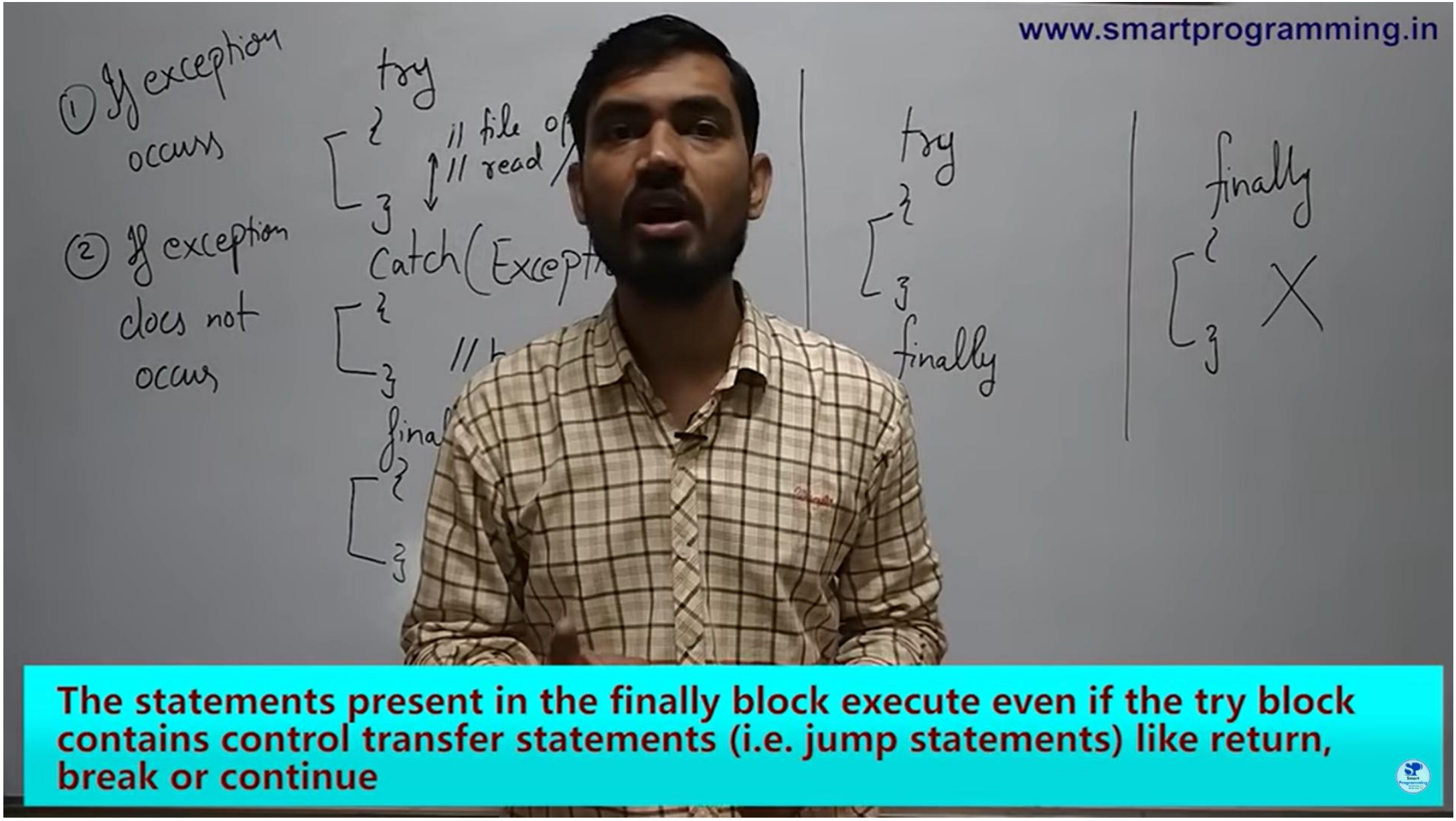
```
try
{
    -
}
Catch(Exception e)
{
}
finally
{
}
```

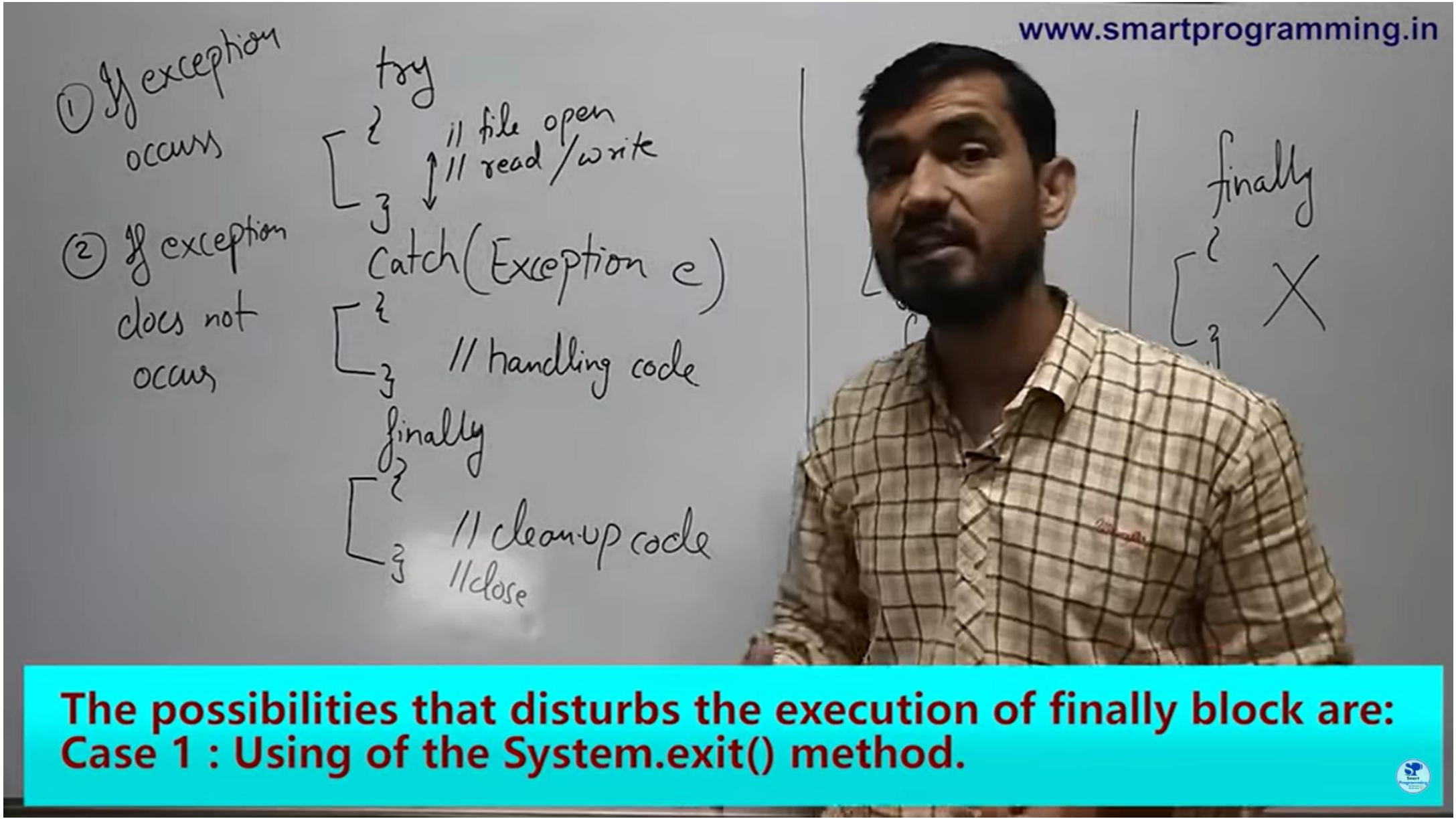
```
try
{
}
finally
{
}
```

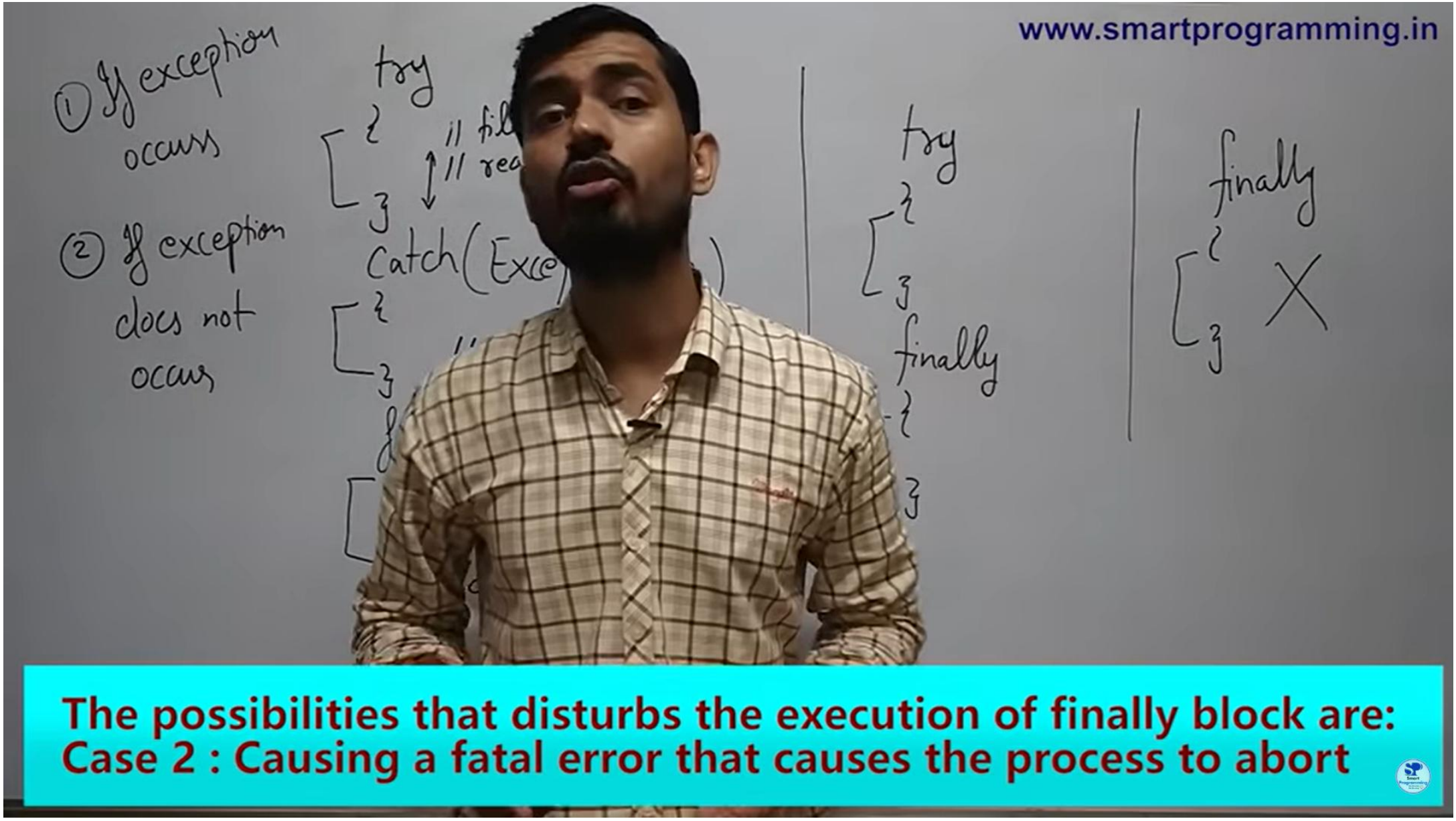
```
finally
{
}
```

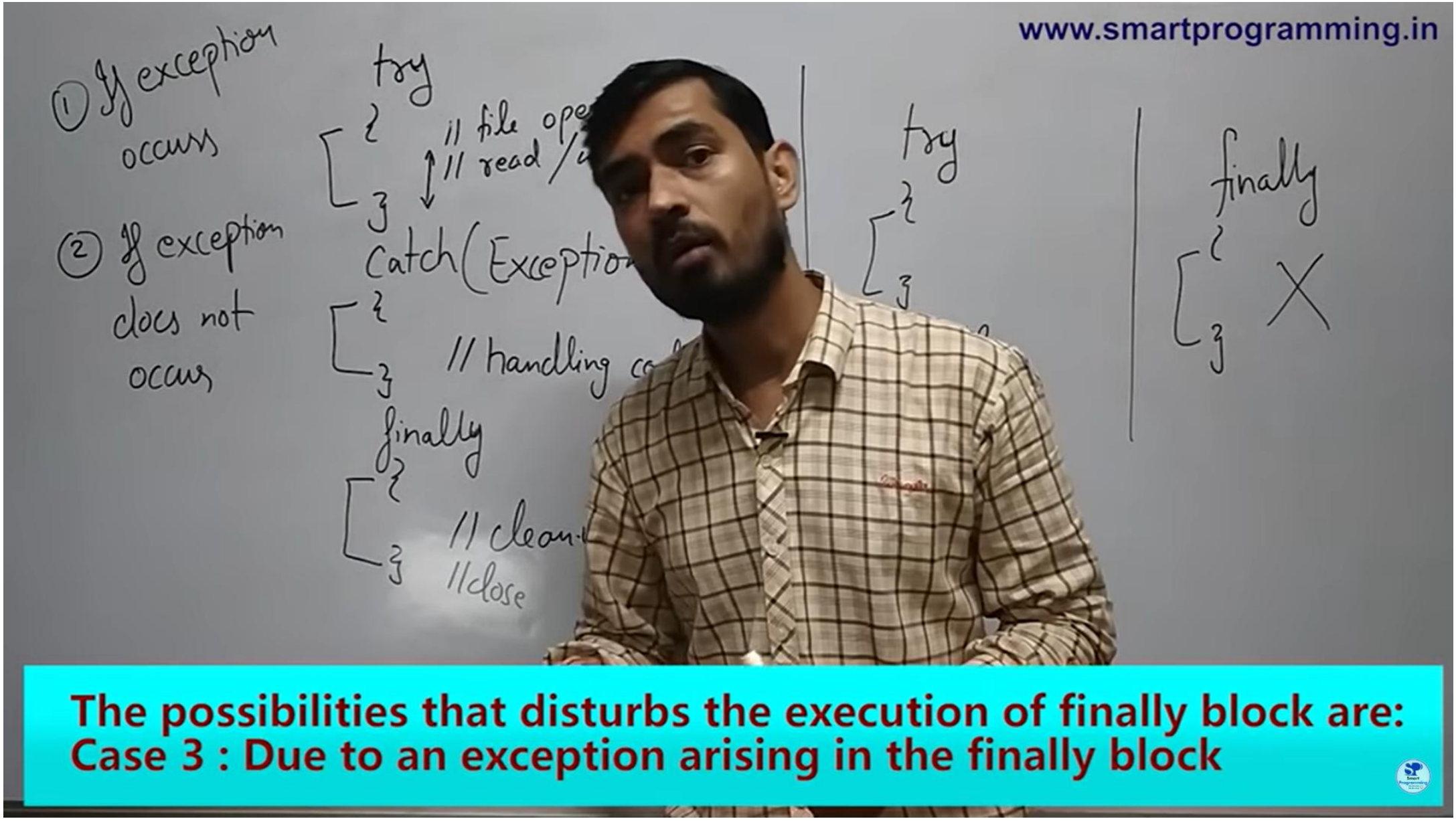



Finally Block in Java with Complete Example | Use of finally Block | Exception Handling in Java
www.smartprogramming.in
Press Esc to exit full screen
try
{
// file open /write
}
catch(exception e)
{
dling code
}
// clean-up code
//close
try
{
}
finally
{
}
finally
{
X
}
If any exception occurs while reading or writing a file, then below code will not execute and thus resource will not close.
12:31 / 20:54

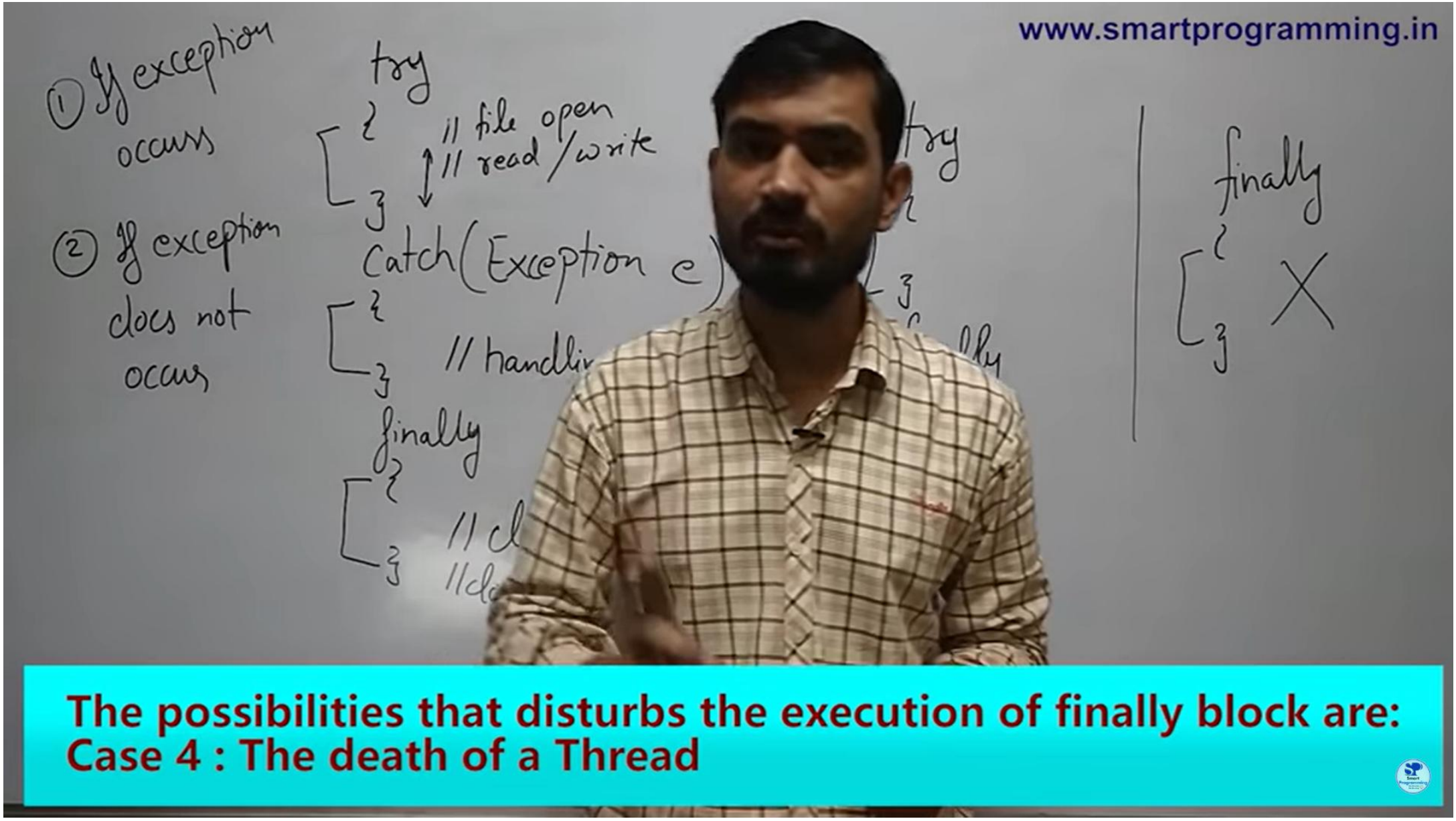

www.smartprogramming.in
① If exception occurs
② If exception does not occur
try
{
// file op
// read
}
Catch(Except
{
// h
}
final
{
}
try
{
}
finally
finally
{
X
}
The statements present in the finally block execute even if the try block contains control transfer statements (i.e. jump statements) like return, break or continue

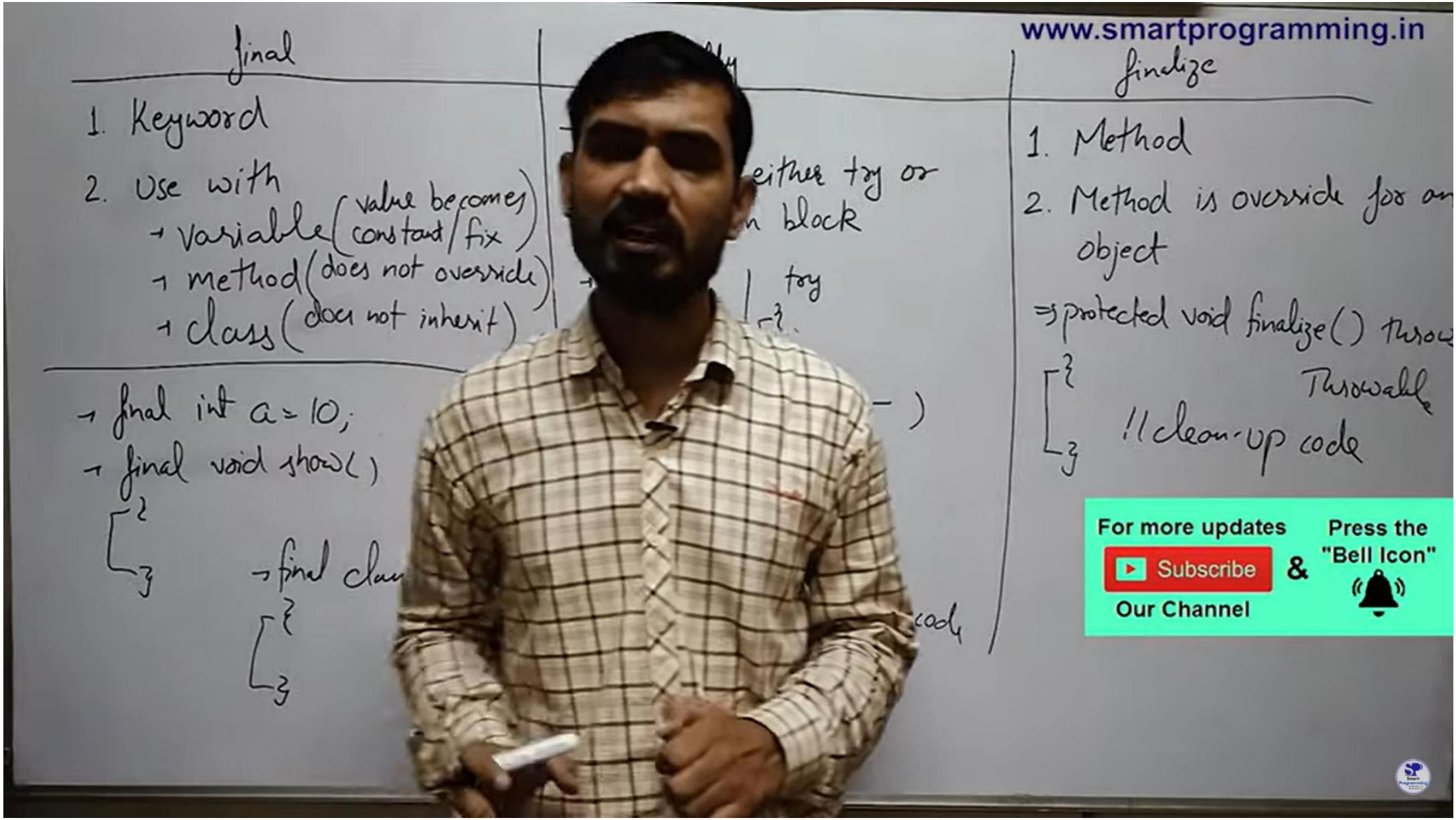

www.smartprogramming.in
① If exception occurs
② If exception does not occurs
try
{
// file open
// read / write
}
Catch(Exception e)
{
// Handling code
}
finally
{
// clean-up code
//close
}
finally
{
X
}
The possibilities that disturbs the execution of finally block are:
Case 1 : Using of the System.exit() method.


www.smartprogramming.in
① If exception occurs
② If exception does not occur
try
Catch(Exce
finally
X
The possibilities that disturbs the execution of finally block are:
Case 2 : Causing a fatal error that causes the process to abort



The possibilities that disturbs the execution of finally block are:
Case 3 : Due to an exception arising in the finally block


www.smartprogramming.in
① If exception occurs
② If exception does not occur
try
{
// file open
// read/write
}
Catch(Exception e)
{
// Handlin
}
finally
{
// cl
//cl
try
{
}
finally
{
X
}
The possibilities that disturbs the execution of finally block are:
Case 4 : The death of a Thread


www.smartprogramming.in
final
1. Keyword
2. Use with
→ variable (value becomes constant/fix)
→ method (does not override)
→ class (does not inherit)
→ final int a = 10;
→ final void show()
finalize
1. Method
2. Method is override for an object
=> protected void finalize() throws Throwable
!! clean-up code
either try or
n block
toy
For more updates
Subscribe
Our Channel
&
Press the "Bell Icon"

class Test

JVM

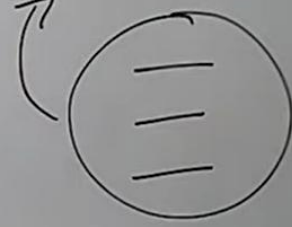
```
{
    p s v m(-)
    {
        int a=100, b=0, c;
        c=a/b;
        Sop(c);
    }
}
```

```
Exception in thread "main"
    java.lang.ArithmeticException: / by zero
- at Test.main(Test.java:6)
```

throw new ExceptionClassName("—");

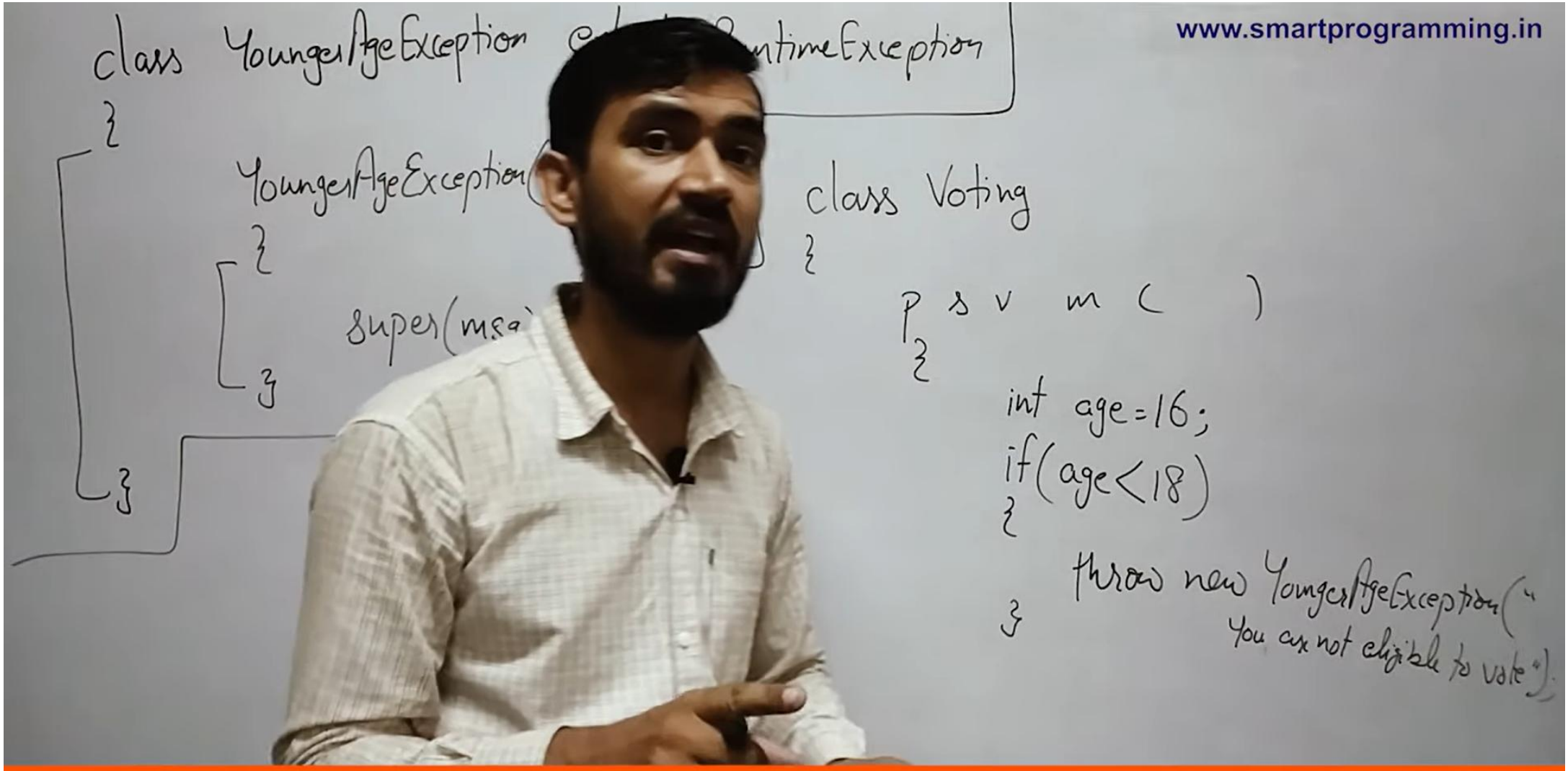
class Test

JVM



```
{
    p s v m( )
    {
        throw new ArithmeticException();
    }
}
```

→ used for custom exception / User defined exception



If the age is below 18, then programmer creates an exception object manually using throw keyword.

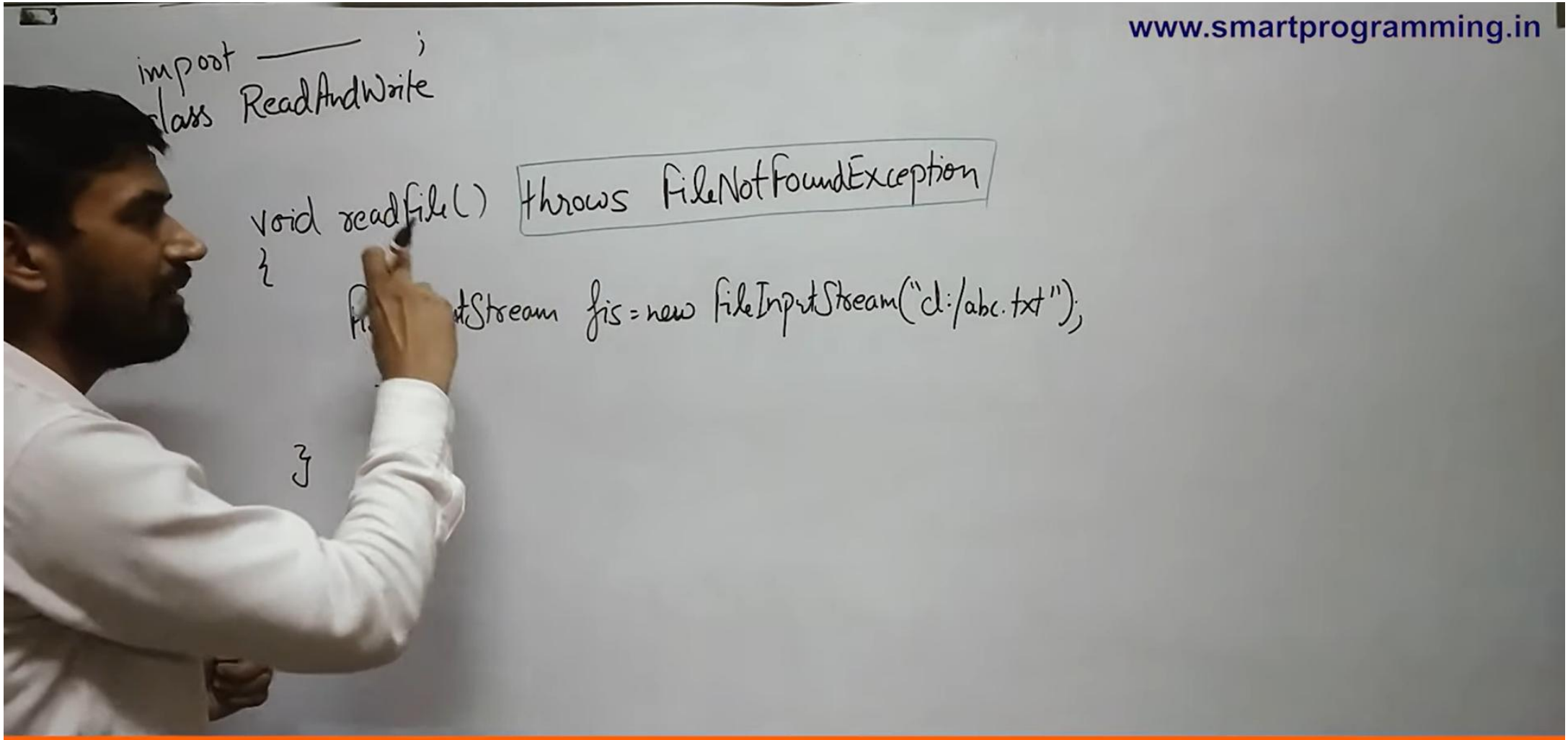
Important Points to Note

1. keywords working :
   - try : In try block we write statements that can throw exception i.e. it mentains risky code
   - catch : It mentains exception handling code i.e. alternative way for exception
   - finally : It mentains clean up code i.e. closing the resources
   - throw : It creates exception object manually (by programmer) and handover to JVM

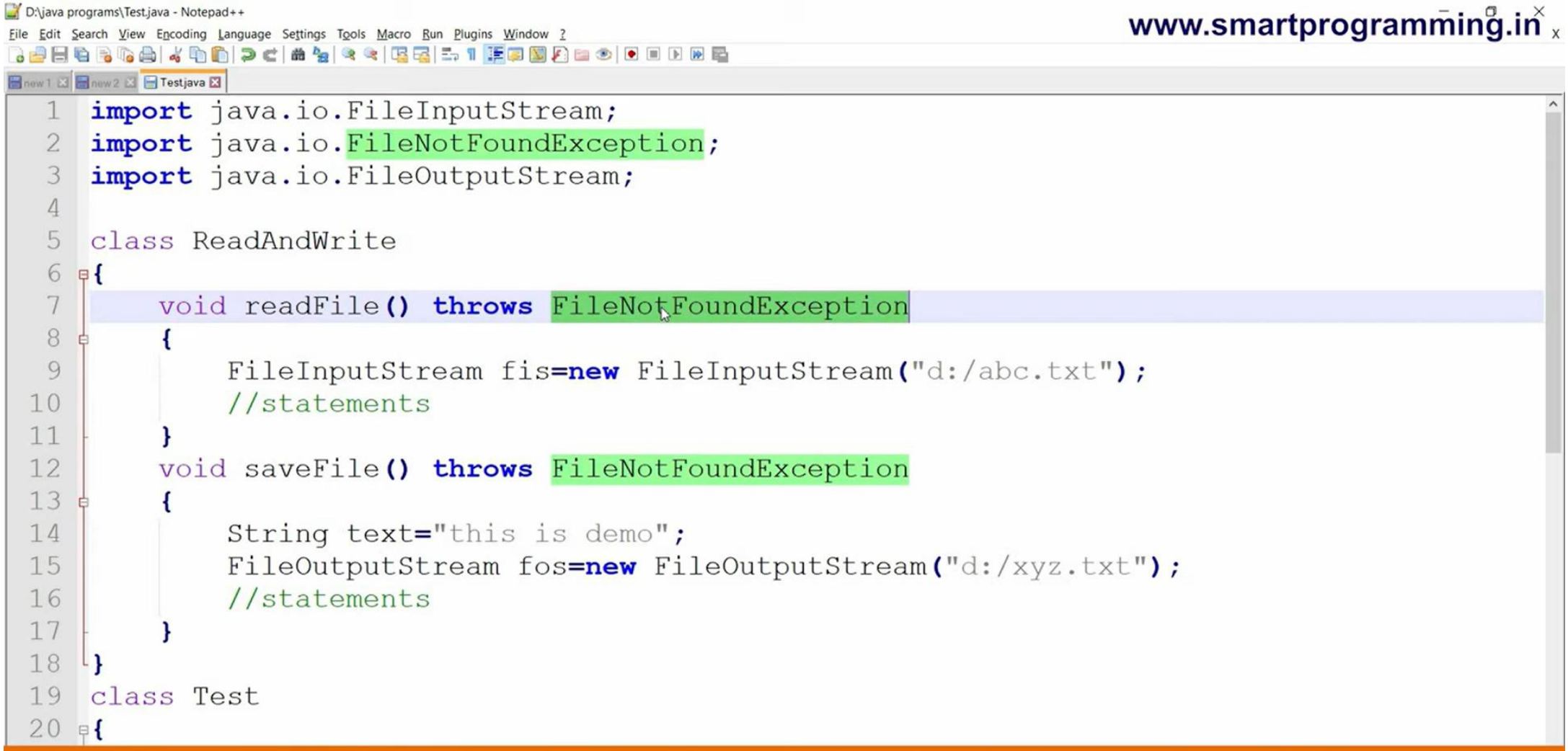
2. We can throw either checked or unchecked exception but throw is best for customized exception

3. We can only throw class that comes in throwable child class

4. We cannot write any statement after throw, otherwise it will provide unrechable statement error.



"throws" keyword is used to declare an exception. It gives an information to the caller method that there may occur an exception so it is better for the caller method to provide the exception handling code so that normal flow can be maintained.



throws keyword is used to declare only for the checked exceptions. If there occurs any unchecked exception such as NullPointerException, it is programmers fault that he is not performing check up before the code being used.

Important Points to Note

1. keywords working :
   - try : In try block we write statements that can throw exception i.e. it mentains risky code
   - catch : It mentains exception handling code i.e. alternative way for exception
   - finally : It mentains clean up code i.e. closing the resources
   - throw : It creates exception object manually (by programmer) and handover to JVM
   - throws : It is used to declare the exception. It gives an information to the caller method that there may occur an exception so it is better for the caller method to provide the exception handling code so that normal flow can be maintained.

2. If we call a method that declares an exception, we must either caught the exception using try catch block or declare the exception using throws keyword or say
   If there is any checked exception, we will get compile time error saying "unreported exception XXX must be caught or declared to be thrown". To prevent this compile time error we can handle the exception in two ways:
   - By using try catch
   - By using throws keyword

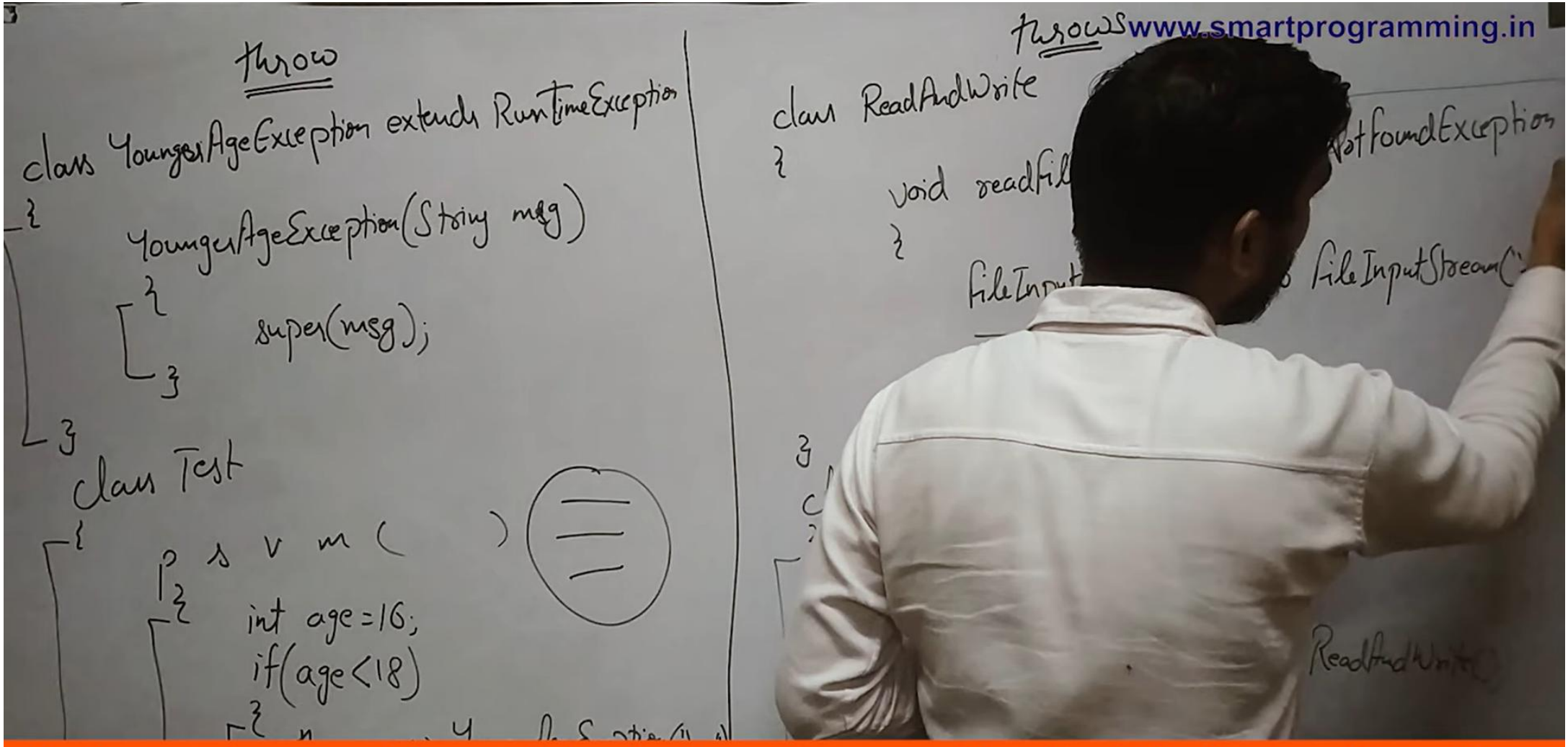
3. throws keyword used to declare the checked exceptions only. If there occurs any unchecked exception such as NullPointerException, it is programmers fault that he is not performing check up before the code being used.

throw

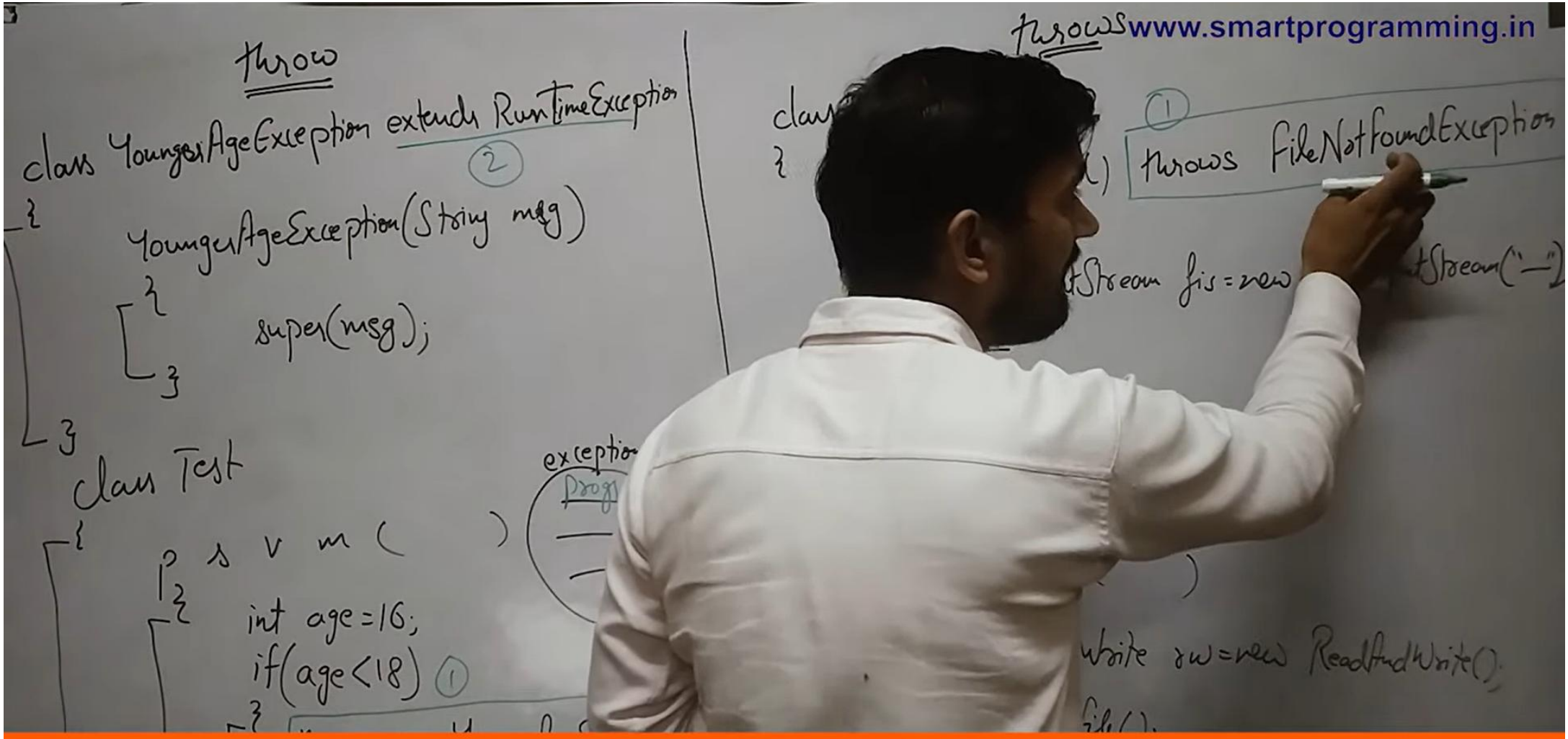
```
class YoungerAgeException extends RuntimeException
{
    YoungerAgeException(String msg)
    {
        super(msg);
    }
}

class Test
{
    p s v m ( )
    {
        int age=16;
        if(age<18)
        {
            throw new YoungerAgeException("...");
        }
    }
}
```

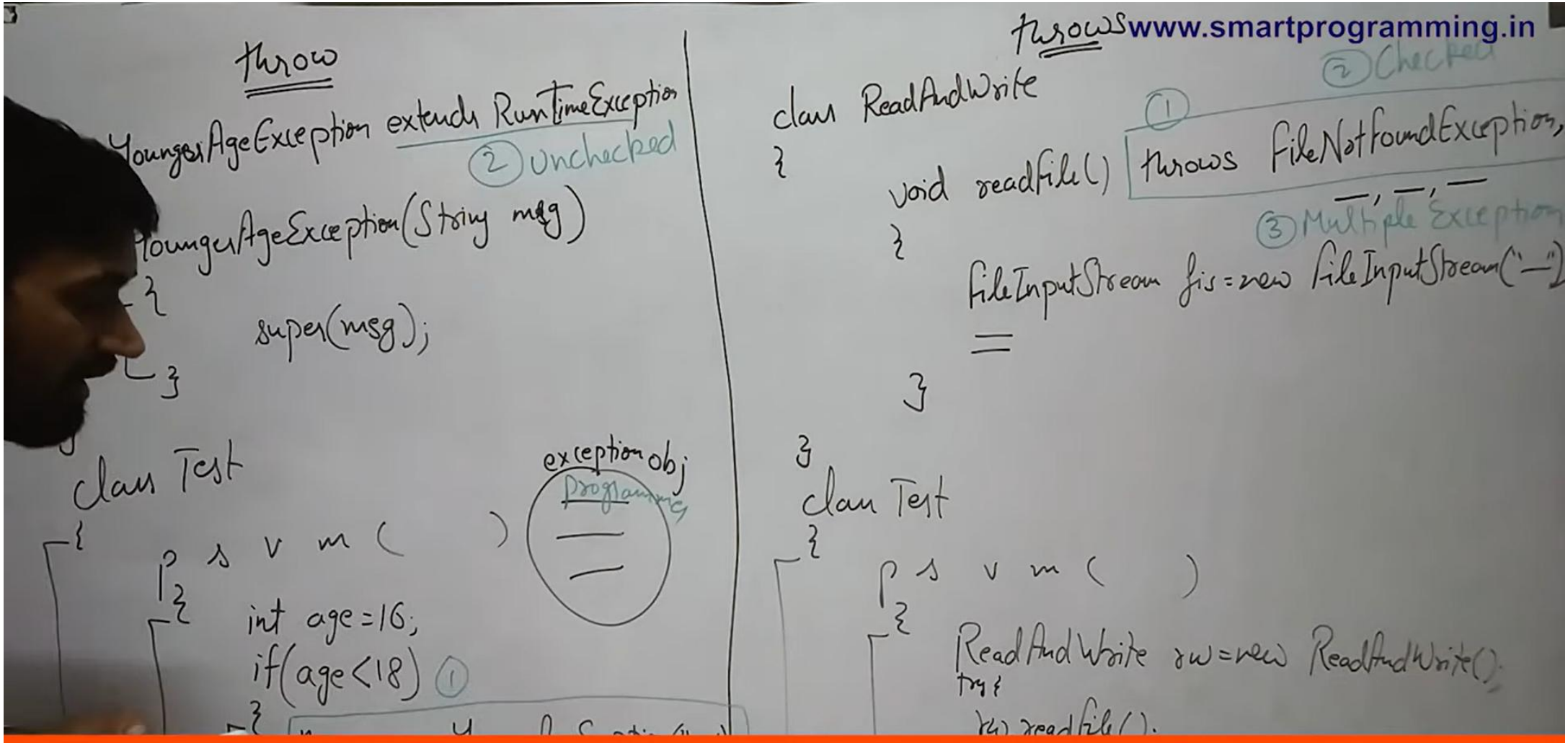
1. throw keyword is used to create an exception object manually i.e. normally method creates an exception object as exception occurs in that method, but when we use throw, programmer is responsible to create an exception object.



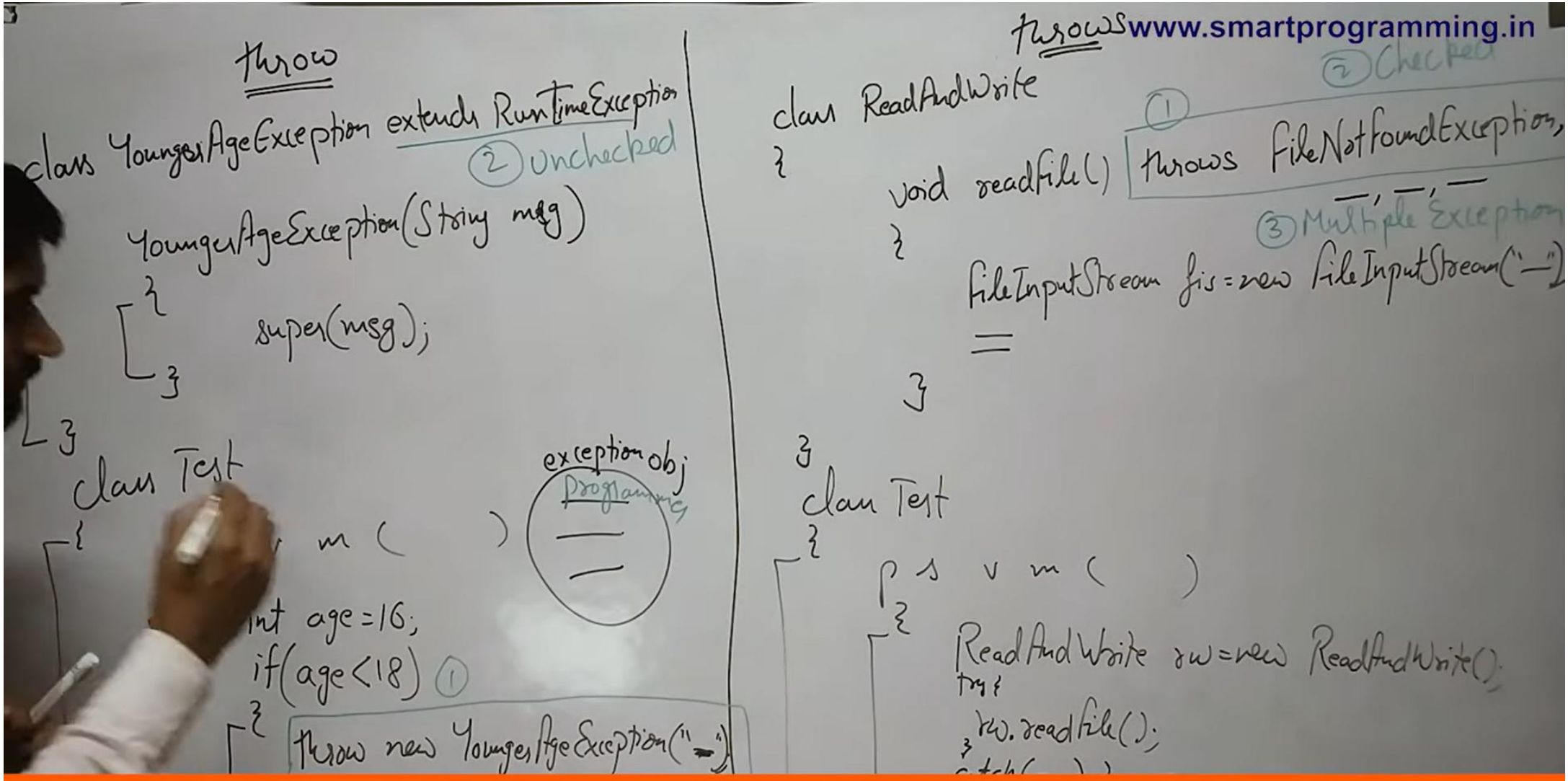
1. throws keyword is used to declare the exception i.e. it indicates the caller method that given exception can occur so we have to handle it either using try catch block or again declare it by using throws keyword



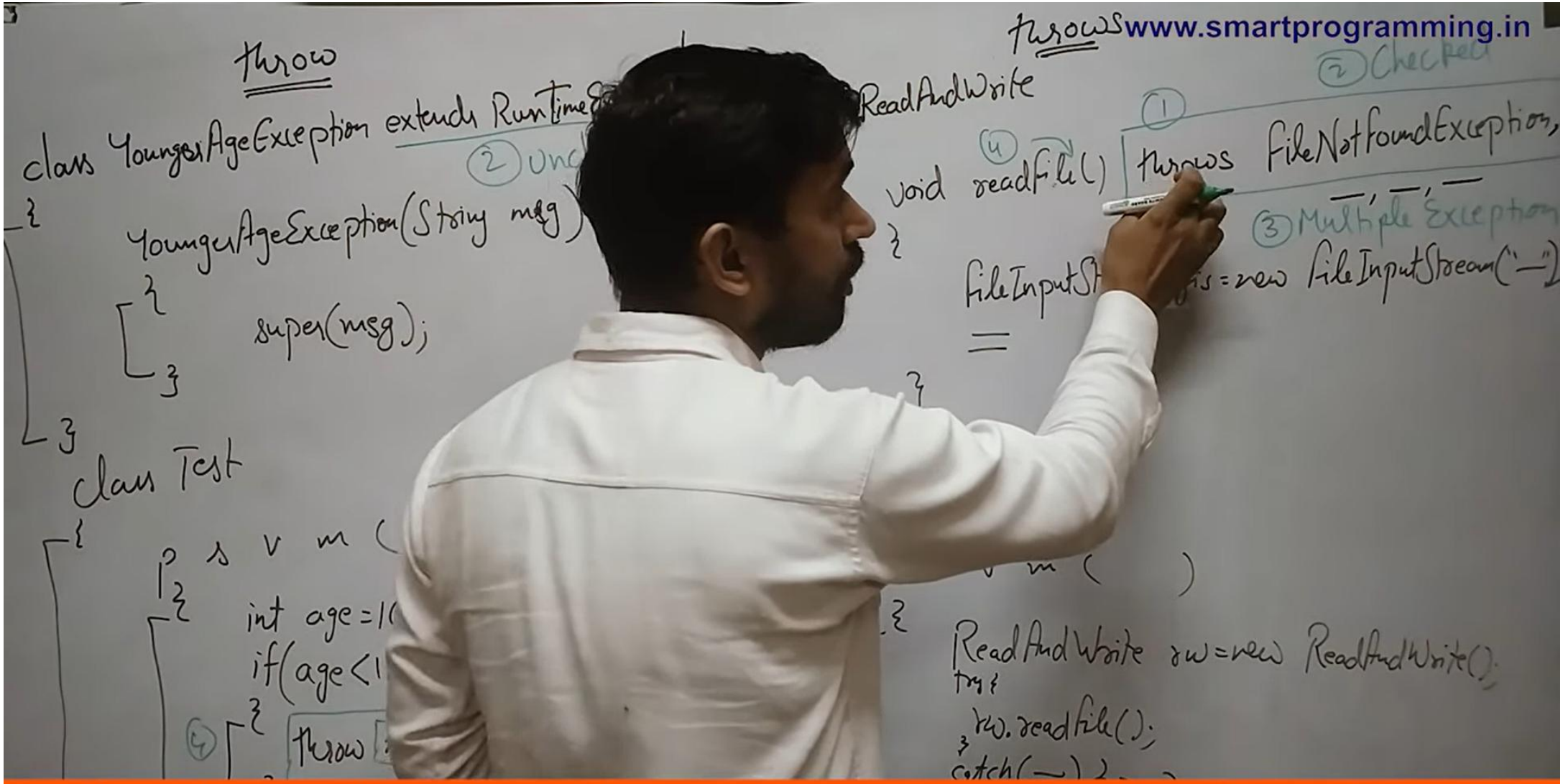
2. throw keyword is mainly used for runtime exceptions or unchecked exceptions but throws keyword is mainly used for compile time exceptions or checked exceptions



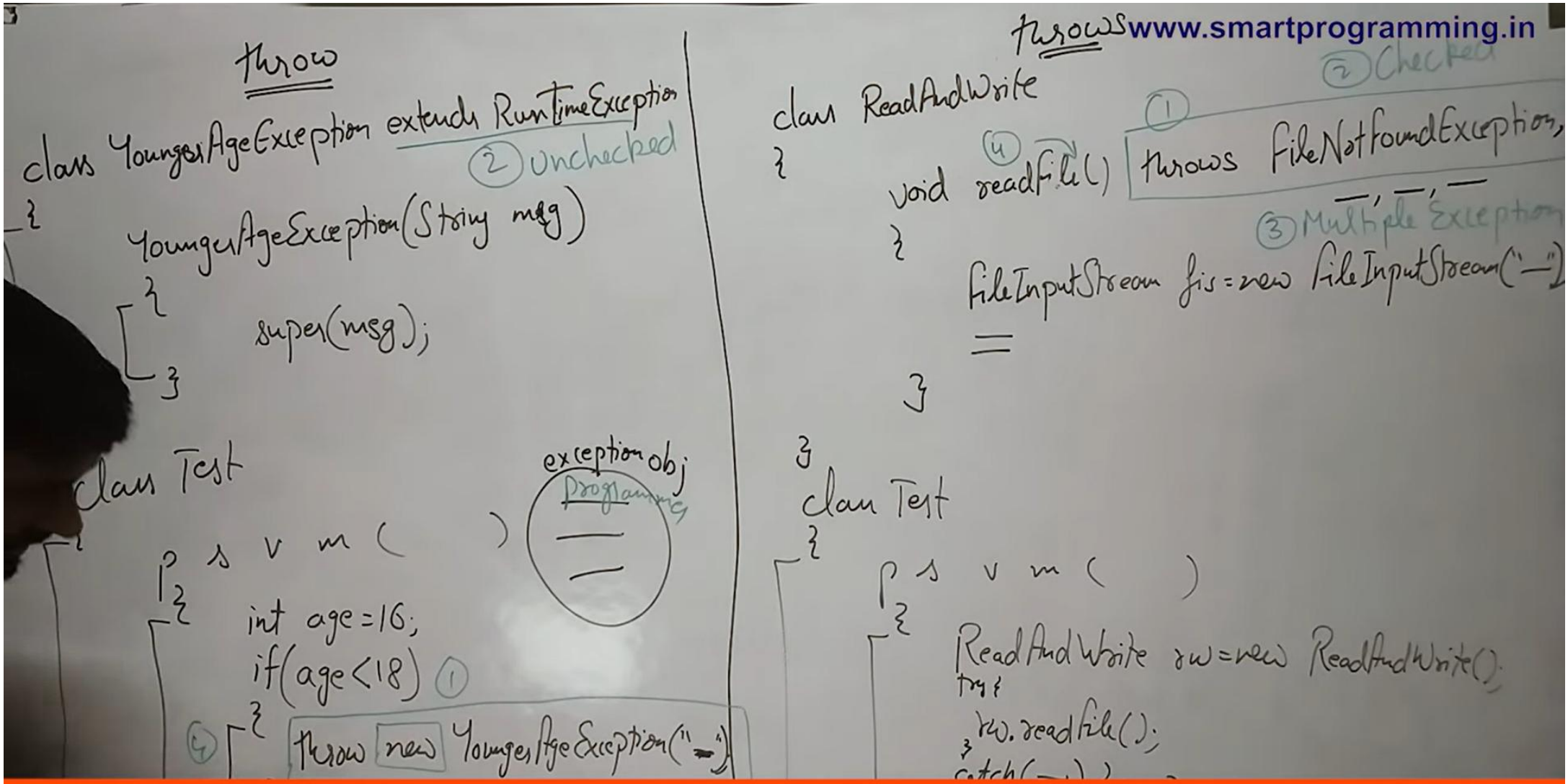
3. In case of throw keyword we can throw only single exception but in case of throws keyword we can declare multiple exceptions i.e. void readFile() throws FileNotFoundException, NullPointerException, etc.



4. throw keyword is used with the method but throws keyword is used with method signature.



5. throw keyword is followed by new instance i.e. object but throws keyword is followed by class



6. We cannot write any statement after throw keyword and thus it can be used to break the statement but there is not such rule for throws keyword

Difference between throw and throws keyword

throw keyword	throws keyword
1. throw keyword is used to create an exception object manually i.e. by programmer (otherwise by default method is responsible to create exception object)	1. throws keyword is used to declare the exceptions i.e. it indicate the caller method that given type of exception can occur so you have to handle it while calling.
2. throw keyword is mainly used for runtime exceptions or unchecked exceptions	2. throws keyword is mainly used for compile time exceptions or checked exceptions
3. In case of throw keyword we can throw only single exception	3. In case of throws keyword we can declare multiple exceptions i.e. void readFile() throws FileNotFoundException, NullPointerException, etc.
4. throw keyword is used within the method	4. throws keyword is used with method signature
5. throw keyword is followed by new instance	5. throws keyword is followed by class
6. We cannot write any statement after throw keyword and thus it can be used to break the statement	6. throws keyword does not have any such rule