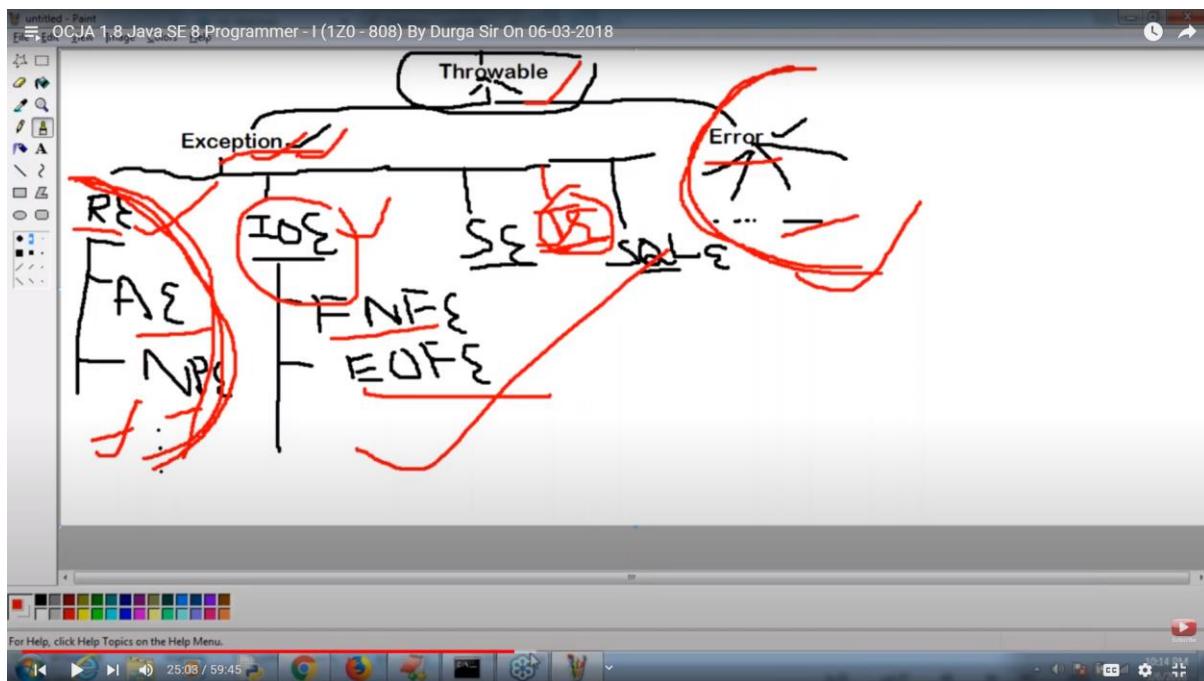


Exception Handling Root Class – Throwable

Throwable Child class are Exception and Error (Stack OverFlow Error)



Exception – Two Types Checked and Unchecked Exception – Needed for Overriding

RE and its child classes. Error and child class are Unchecked Exception

Rest are Checked Exception

Rule1: For Checked Exception

```
54 If child class method throws any checked exception then compulsory parent class  
55 method should throw the same checked exception or its parent.  
56 CE  
57
```

For UnChecked Exception – No Rule

Child class does not throw an any exception – So its valid



Child throws Exception which is checked then Parent class method should also throw the same exception or parent exception- Below is Invalid. Overriden method(Parent Method) does not throw Exception

```
34 P: public void m1()
35 C: public void m1() throws Exception
36 -----
```

Child Throws IOException parent throws parent of IOException->Exception value

```
37 P: public void m1() throws Exception
38 C: public void m1() throws IOException
39 -----
```

Child throws Exeption but parent throws child of Exception (Allowed Exception or its parent throwable only)- Fail

```
40 P: public void m1() throws IOException
41 C: public void m1() throws Exception
42 -----
```

FileNotFoundException and EOFException are child of IOException - Valid

```
43 P: public void m1() throws IOException
44 C: public void m1() throws FileNotFoundException,EOFException
45 -----
```

FileNotFoundException is a child of IOException , InterruptedException is not a child of IOException. Exception is the parent class

```
45 -----
46 P: public void m1() throws IOException
47 C: public void m1() throws FileNotFoundException,InterruptedException
48 -----
```

Valid. Child classes are unchecked Exceptions, child method throws checked then the rule applies .

For unchecked exception no rules.

```
48 -----
49 P: public void m1() throws IOException
50 C: public void m1() throws AE,NPE
51 -----
```

```
1 import java.io.*;
2 class P
3 {
4     public void m1() throws IOException
5     {
6     }
7 }
8 class C extends P
9 {
10    public void m1() throws ArithmeticException,FileNotFoundException
11    {
12    }
13 }
```

```
61 -----
62 overriding wrt static modifier:
63 -----
```

Static Methods are class level methods, Instance methods are Object Level methods. So, we can't override static and non-static methods and vice versa. Static method can be overridden with static method only and Instance method with instance method only

CE. How static method with non static

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

```
1 D:\durgaclasses>javac P.java
1 P.java:10: error: m1() in C cannot override m1() in P
1     public void m1()
1           ^
1   overridden method is static
1 error
```

Invalid

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

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

Below the code, a terminal window shows the output of the javac command:

```
D:\durgaclasses>javac P.java
P.java:10: error: m1() in C cannot override m1() in P
      public static void m1()
                           ^
overriding method is static
```

Static Method Overloading is not called overloading but Method Hiding

Method Hiding. Resolution is carried by ref type by compiler and compiler calls the ref var method

Consider non static methods below , this is overriding

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

```
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 m1()
11    {
12        System.out.println("Child");
13    }
14 }
```

```

15 class Test
16 {
17     public static void main(String[] args)
18     {
19         P p = new P();I
20         p.m1();
21
22         C c = new C();
23         c.m1();
24
25         P p1 = new C();
26         p1.m1();
27
28     }

```

For Help, press F1 In 26 col 17 30 00 PC ANSI

```

1 D:\durgaclasses>java Test
2 Parent
2 Child
2 Child
2

```

Static method Method Hiding carried out by compiler calls Ref Var Methods

```

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

```



```

15 class Test
16 {
17     public static void main(String[] args)
18     {
19         P p = new P();I
20         p.m1();
21
22         C c = new C();
23         c.m1();
24
25         P p1 = new C();
26         p1.m1();
27
28     }

```

For Help, press F1 In 26 col 17 30 00 PC ANSI

```

D:\durgaclasses>java Test
Parent
Child
Parent

```

Overriding concept is applicable only for methods and not variables, How to resolve

```
1 class P
2 {
3     int x=777;
4 }
5 class C extends P
6 {
7     int x=888;    I
8 }
9 class Test
10 {
11     public static void main(String[] args)
12     {
13         P p = new P();
14         System.out.println(p.x);
15
16         C c = new C();
17         System.out.println(c.x);
18
19         P p1 = new C();
20         System.out.println(p1.x);
21     }
22 }
```

Variable resolution is always taken care by compiler based on ref type. JVM does not play a role here.

Rule is same if its static variable non static or not- based only on ref variable

```
1 class P
2 {
3     int x=777;
4 }
5 class C extends P
6 {
7     int x=888;
8 }
9 class Test
10 {
11     public static void main(String[] args)
12     {
13         P p = new P();
14         System.out.println(p.x);
15
16         C c = new C();
17         System.out.println(c.x);
18
19         P p1 = new C();
20         System.out.println(p1.x);
21     }
22 }
```

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

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) By Durga Sir On 06-03-2018
1 class P
2 {
3     static int x=777;
4 }
5 class C extends P
6 {
7     static int x=888;
8 }
9 class Test
10 {
11     public static void main(String[] args)
12     {
13         P p = new P();
14         System.out.println(p.x);
15
16         C c = new C();
17         System.out.println(c.x);
18
19         P p1 = new C();
20         System.out.println(p1.x);
21
22     }
}
File: Untitled3.java 57:33 / 59:45
ln 7 col 13 24 69 PC ANSI
```

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

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir On 06-03-2018
1 class P
2 {
3     int x=777;
4 }
5 class C extends P
6 {
7     static int x=888;
8 }
9 class Test
10 {
11     public static void main(String[] args)
12     {
13         P p = new P();
14         System.out.println(p.x);
15
16         C c = new C();
17         System.out.println(c.x);
18
19         P p1 = new C();
20         System.out.println(p1.x);
21
22     }
}
For Help, press F1  * Untitled2.java  * P.java  * Test.java
57:49 / 59:45
```

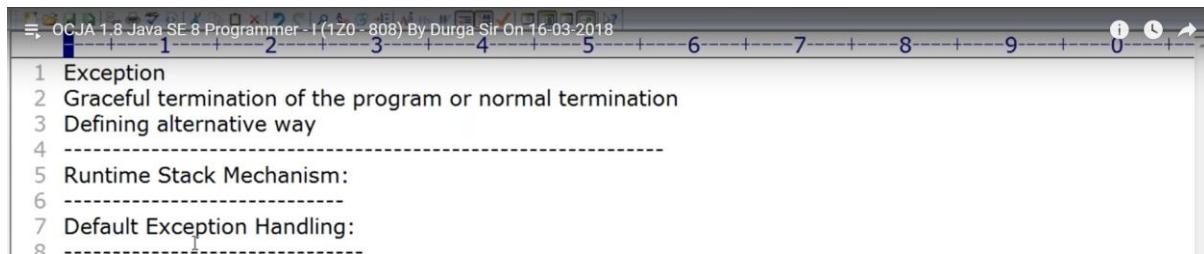
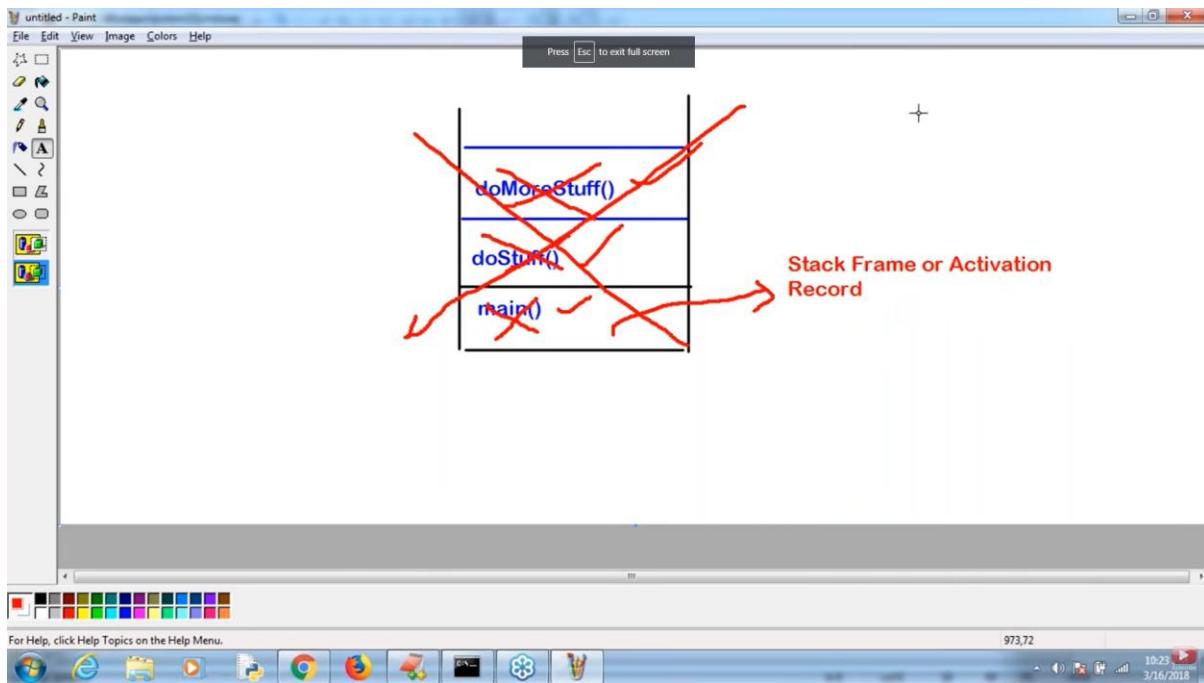
```
D:\durgaclasses>java Test
777
888
777
```

Exception

Every Thread has a stack

After main method is completed the, main thread is done and the stack will be destroyed

Every stack is called a Stack Frame or Activation Record



```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir On 16-03-2018
```

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         doStuff();
6     }
7     public static void doStuff()
8     {
9         doMoreStuff();
10    }
11    public static void doMoreStuff()
12    {
13        System.out.println(10/0);
14    }
15 }
```

If DoMoreStuff fails what happens is defaultExceptionMechanism. It creates an Exception object as shown below and JVM checks if there is a handling code

It checks if there is an handling code if not terminate the method.

Checks who is the caller- Do u have an handling code ? Do stuff has no – Terminate DoStuff

At last

JVM checks if MainMethod , do you have any handling code? No -> Terminate Main Method

Caller of JVM method – JVM handles the exception -> DefaultExceptionHandler. -> Terminates the program abnormally.

```
OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir On 16-03-2018
```

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         doStuff();
6     }
7     public static void doStuff()
8     {
9         doMoreStuff();
10    }
11    public static void doMoreStuff()
12    {
13        System.out.println(10/0);
14    }
15 }
```

```

OCJA 1.8 Java SE 8 Programmer - I (120 - 808) By Durga Sir On 16-03-2018
----- 1 ----- 2 ----- 3 ----- 4 ----- 5 ----- 6 ----- 7 ----- 8 ----- 9 ----- 0 -----
1 Exception
2 Graceful termination of the program or normal termination
3 Defining alternative way
4 -----
5 Runtime Stack Mechanism:
6 -----
7 Default Exception Handling:
8 -----
9
10
11Exception in thread main java.lang.AE: / by zero
12      at Test.doMoreStuff()
13      at Test.doStuff()
14      at Test.main()
15

```

This is called Stack Trace, taken care by DefaultExceptionHandler

```

Select C:\Windows\system32\cmd.exe
D:\durgaclasses>javac Test.java

D:\durgaclasses>java Test
Exception in thread "main" java.lang.ArithmetricException: / by zero
      at Test.doMoreStuff(Test.java:13)
      at Test.doStuff(Test.java:9)
      at Test.main(Test.java:5)

```

Here DomoreStuff executes successfully and doStuff line 10 will Throw Exception

At Test doMoreStuff will not be available in StackTrace

Diagram illustrating the execution stack:

- The timeline shows frames for doMoreStuff, doStuff, and main methods.
- Red annotations show the flow from doMoreStuff to doStuff, and from doStuff to main.
- Handwritten notes include "doMoreStuff()", "doStuff()", and "Hello".

X	Question	Asker	R...
1	Q: Sir if there is handling code in doStuff, there will not be an entry of the same	Shubham Gupta	
2	Q: 3	Shubham Gupta	
3	Q: ?	Deepakaj Yadav	
4	Q: ok sir	Deepakaj Yadav	
5	Q: AE	Shubham Gupta	

```

C:\Windows\system32\cmd.exe
D:\durgaclasses>javac Test.java

D:\durgaclasses>java Test
Hello
Exception in thread "main" java.lang.ArithmetricException: / by zero
      at Test.doStuff(Test.java:10)
      at Test.main(Test.java:5)

```

Error and child classes and Runtime Exception and child classes are Unchecked . Compiler is not going to throw an error.

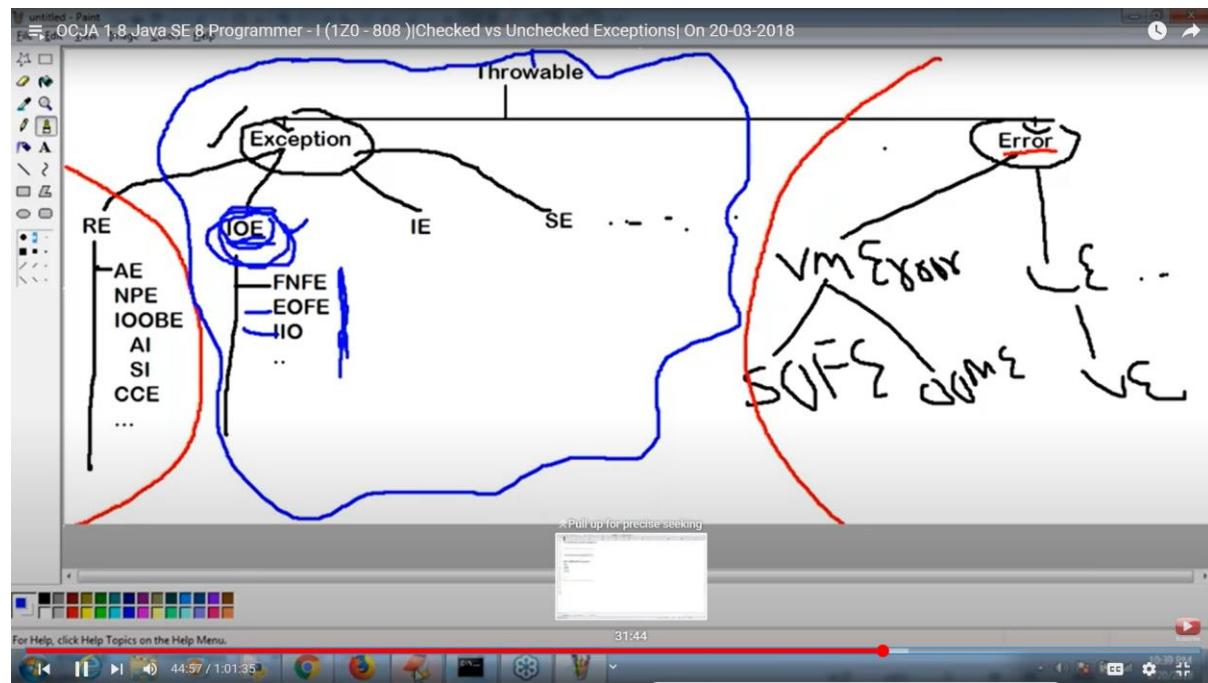
```
22 -----
23 RE and its child classes
24 Error and its child classes
25 unchecked
26
27 FNFE==>checked
28 IOE==>checked
29 Exception==>checked
30 AE==>unchecked
31 RE==>unchecked
32 Object==>NA
33
```

Fully Checked and Partially Checked. Parent is checked and all childs are checked. Eg IOE, IE(interrupted).

Exception class has RE and IOE. RE is unchecked and IOE is checked hence its partially checked

Throwable is partially Checked.Errors unchecked and RE is also unchecked.

Error is completely Unchecked Exception. Other than Throwable and Exception no other partially checked



Not recommended to write all code in one try block. Will stop all execution

```
1 try
2 {
3     stmt-1
4     stmt-2
5     stmt-1
6     stmt-1
7     stmt-1
8     stmt-100
9 }
10 catch (Exception e)
11 {
12     Handling code//train ticket//local file//mysql db
13 }
```

Exceptions are caused by program and recoverable

Errors are not caused by program and not recoverable

```
1 try
2 {
3     stmt-1
4     stmt-2
5     stmt-3
6 }
7 catch (X e)
8 {
9     stmt-4
10 }
11 stmt-5
12
13 case-1: If there is no exception:
14 -----
15 1,2,3,5 Normal termination
16
17 case-2: If exception at stmt-2 and the corresponding catch block matched?
18 -----
19 1,4,5,normal termination
20
21 case-3: If exception at stmt-2 and the corresponding catch block not matched?
22 -----
23 1,abnormally termination
24
```

```
1 Exception handling:
2 -----
3 introduction
4 runtime stack mechanism
5 default exception handling
6 exception hierarchy
7 customized exception handling by using try-catch
8 control-flow in try-catch:
9 -----
10 Methods to print exception information:
11 -----
12 e.printStackTrace()
13 e.toString()=>sop(e)
14 e.getMessage()
```

default exception handling
exception hierarchy
customized exception handling by using try-catch
control-flow in try-catch:

Methods to print exception information:

e.printStackTrace()
e.toString()=>sop(e)
e.getMessage()
Exception in thread "main" java.lang.ArithException: / by zero
at Test.main()

Questions

Question	Asker
complete info-printstacktrace	john acharya
s	Shubham Gupta
s	Pooja Chavan
ys	Deepankaj Yadav

Sir if we dont catch checked exception in , then compiler will complain?
ok sir

Send Privately Send to All

e.printStackTrace()

OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) by Durga Sir On 22-03-2018

```
D:\durga_classes>javac Test.java
D:\durga_classes>java Test
java.lang.ArithException: / by zero
at Test.main(Test.java:7)
```

e.toString()

sysout(e)-> will call e.toString()

```
D:\durga_classes>java Test
java.lang.ArithException: / by zero
```

e.getMessage()

```
D:\durga_classes>javac Test.java
D:\durga_classes>java Test
/ by zero
```

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0-808) by Durga Sir On 22-03-2018  
8 9  
22 try with multiple catch blocks  
23  
24 try  
25 {  
26     read data from the remote file  
27     write data to the oracle database  
28     send data over the network  
29  
30 }  
31 catch(FileNotFoundException e)  
32 {  
33     use some local file  
34 }  
35 catch(SQLException e)  
36 {  
37     use some other database like mysql  
38 }  
39 catch(NetworkException e)  
40 {  
41     use alternative network  
42 }  
43 catch(Exception e)  
44 {  
45     default exception handling  
46 }  
47
```

Don't use throws unchecked exception -> Throws keyword is used for checked exception. You can use Throws for unchecked but still makes no sense.

Order of catch block is important

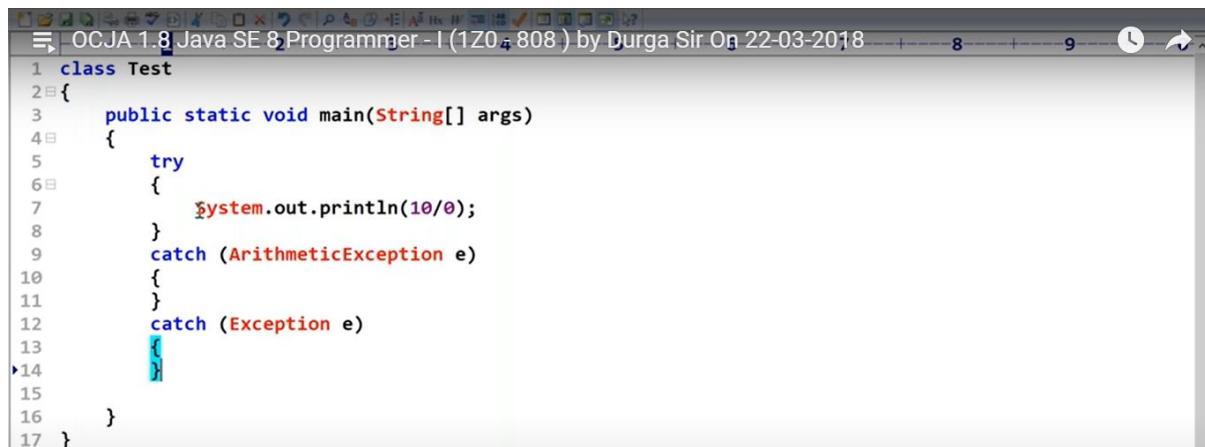
JVM considers top to bottom. Exception e -> will handle all exception and will not handle arithmetic exception. First child followed by Parent

```
1 class Test  
2 {  
3     public static void main(String[] args)  
4     {  
5         try  
6         {  
7             System.out.println(10/0);  
8         }  
9         catch (Exception e)  
10        {  
11        }  
12        catch (ArithmaticException e)  
13        {  
14        }  
15    }  
16 }  
17 }
```

Select Command Prompt

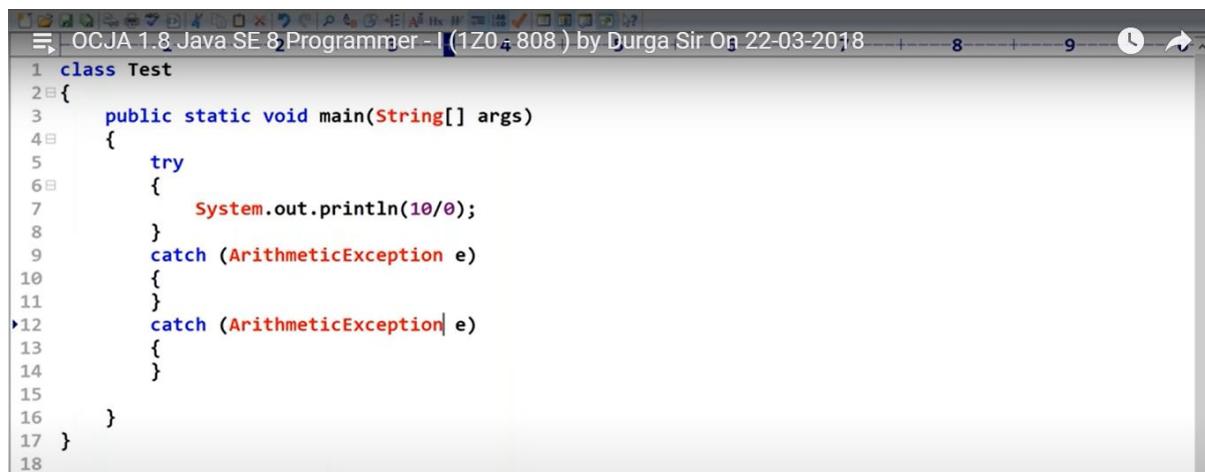
```
D:\durga_classes>javac Test.java  
Test.java:12: error: exception ArithmaticException has already been caught  
        catch (ArithmaticException e)  
                           ^  
1 error
```

Below is valid



```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) by Durga Sir On 22-03-2018
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         try
6         {
7             System.out.println(10/0);
8         }
9         catch (ArithmaticException e)
10        {
11        }
12        catch (Exception e)
13        {
14        }
15    }
16 }
17 }
```

InValid



```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808) by Durga Sir On 22-03-2018
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         try
6         {
7             System.out.println(10/0);
8         }
9         catch (ArithmaticException e)
10        {
11        }
12        catch (ArithmaticException| e)
13        {
14        }
15    }
16 }
17 }
18 }
```



```
D:\durga_classes>java Test
D:\durga_classes>javac Test.java
Test.java:12: error: exception ArithmaticException has already been caught
                                catch (ArithmaticException e)
                                         ^
1 error
```

```
OCJA 1.8 Java SE 8 Programmer-I (1Z0-808) by Durga Sir On 22-03-2018
```

```
34 }
35
36 finally:
37 -----
38
39 open database connection
40 read data from the database
41 close database connection
42
43
44
45
```

clean up activity
Resource deallocation

Finally will be executed during normal termination. Even if exception is handled or if exception is not handled and abnormal termination happens, finally block will be executed. Highly recommended

```
OCJA 1.8 Java SE 8 Programmer-I (1Z0-808) by Durga Sir On 22-03-2018
```

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         try
6         {
7             open database connection
8             read data from the database
9
10        }
11        catch (ArithmaticException e)
12        {
13        }
14    }
15    finally
16    {
17        close database connection
18    }
19
20
21 }
22 }
```

Questions

X	Question	Asker	R...
hahah		Shubham Gupta	
s		Pooja Chavan	
sir we may take separate try catch for cleanup code		Shubham Gupta	
s		Pooja Chavan	

but sir we have to take Exception catch in the end to handle these uncommon exception na

Send Privately Send to All

Only 1 finally for a try block is allowed

Exception Handled case

The screenshot shows an IDE interface with a code editor and a terminal window. The code editor displays a Java file named 'Test.java' with the following content:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         try
6         {
7             System.out.println("try");
8             System.out.println(10/0);
9         }
10        catch (ArithmaticException e)
11        {
12            System.out.println("catch");
13        }
14        finally
15        {
16            System.out.println("finally");
17        }
18    }
19 }
20 }
```

The terminal window below shows the command-line output of running the Java program:

```
8 D:\durga_classes>javac Test.java
9
10 D:\durga_classes>java Test
try
catch
finally
```

Exception UnHandled Case

The screenshot shows an IDE interface with a code editor. The code editor displays a Java file named 'Test.java' with the following content:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         try
6         {
7             System.out.println("try");
8             System.out.println(10/0);
9         }
10        catch (NullPointerException e)
11        {
12            System.out.println("catch");
13        }
14        finally
15        {
16            System.out.println("finally");
17        }
18    }
19 }
20 }
```

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         try
6         {
7             System.out.println("try");
8             System.out.println(10/0);
9         }
10        catch(NullPointerException e)
11        {
12            System.out.println("catch");
13        }
14        finally
15        {
16            System.out.println("finally");
17        }
18    }
19 }
20 }
```

Questions

X	Question	Asker
1	try finally abnormal flow	Kalyan TC
2	person will die like our code	Shubham Gupta
3	try finally	Shubham Gupta

but sir we have to take Exception catch in the end to handle these uncommon exception na

Finally Cant be written anywhere only after try catch .

When we use System.exit(0) bcos JVM will stop execution and the Finally will not be executed. That's the only situation where a Finally will not be executed

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         try
6         {
7             System.out.println("try");
8             System.exit(0);
9         }
10        catch(NullPointerException e)
11        {
12            System.out.println("catch");
13        }
14        finally
15        {
16            System.out.println("finally");
17        }
18    }
19 }
20 }
```

```
7
8 D:\durga_classes>javac Test.java
9
10 D:\durga_classes>java Test
11 try
```

When return is there as below finally will be executed before returning and , the return in finally will stop return and takes higher priority . Hence return 999 will execute and not Return 777

The screenshot shows a Java IDE interface. The code editor displays the following Java code:

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         System.out.println(m1());
6     }
7     public static int m1()
8     {
9         try
10        {
11            + return 777;
12        }
13        catch(NullPointerException e)
14        {
15            return 888;
16        }
17        finally
18        {
19            return 999;
20        }
21
22    }
23 }
```

The line `+ return 777;` is highlighted with a red box. To the right of the code editor is a "Questions" panel showing a list of user questions and answers.

Question	Asker
finally	Kalyan TC
finally	Uday Baba
ohhh is their any reason sir	Shubham Gupta
999	Pooja Chavan

A note at the bottom of the panel says: "but sir we have to take Exception catch in the end to handle these uncommon exception na".

Garbage Collector -> Finalize Method- Before destroying an object- To Perform clean up activities

The screenshot shows a Java IDE interface. The code editor displays the following Java code:

```
1 final,finally and finalize
2 control flow in try-catch-finally
3 control flow in nested try-catch-finally
4 various possible combinations of try-catch-finally
5
6
7 finalize()
```

Overlaid on the code editor are several hand-drawn red annotations:

- A large circle with a diagonal slash through it is positioned above the `finalize()` method.
- A stick figure is standing next to the circle.
- The word "finalize()" is written in large red letters with a checkmark.
- A brace is drawn around the word "cleanup" in a large oval.

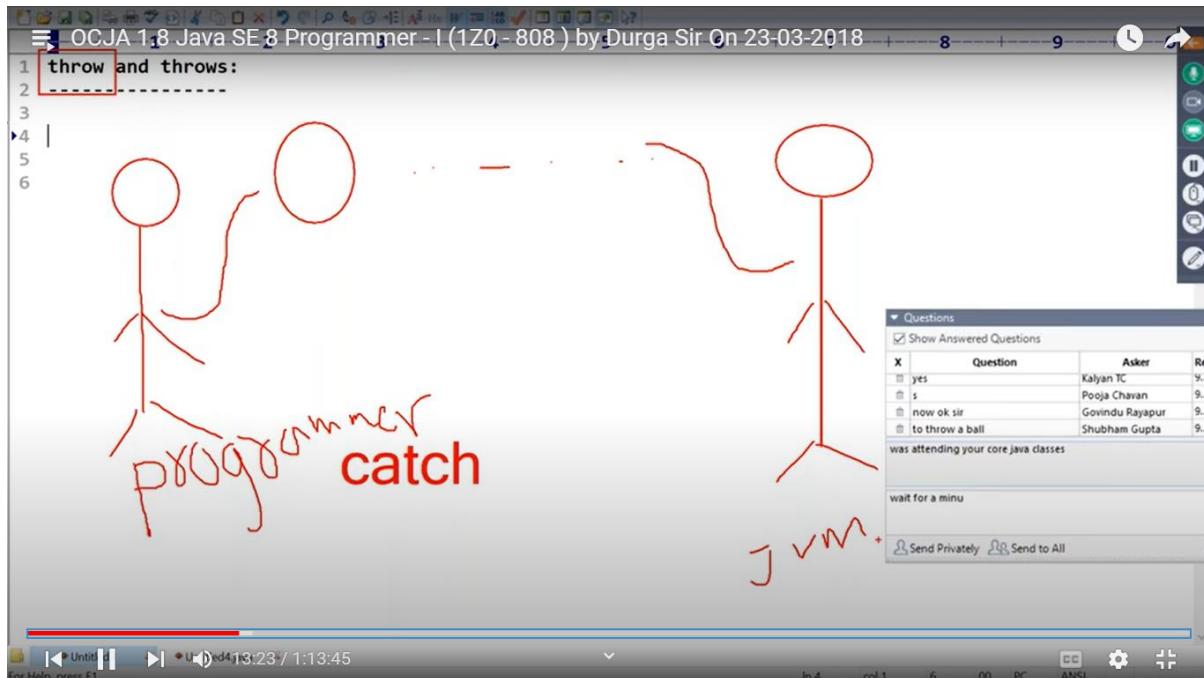
To the right of the code editor is a "Questions" panel showing a list of user questions and answers.

Question	Asker
code cleanup	Shubham Gupta
close resources	Pooja Chavan
finalize	Deepankaj Yadav
finalize	Deepankaj Yadav

A note at the bottom of the panel says: "was attending your core java classes".

Throw and Throws

Programmer throws Exception and JVM catches.



AE -> The method Main is responsible to create the Exception Object (with help of JVM only). JVM will see if the exception is being handled. And if not then it calls the default exception handler.

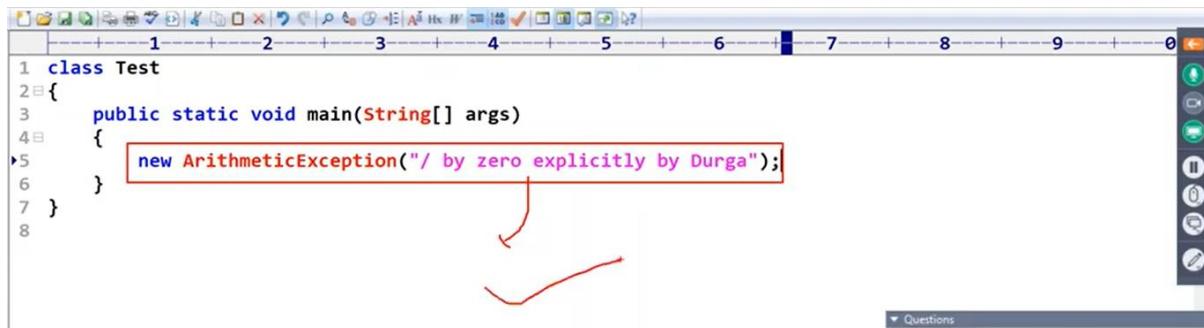
```
OCJA 1.8 Java SE 8 Programmer - I (1Z0-808) by Durga Sir On 23-03-2018
```

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         System.out.println(10/0);
6     }
7 }
```

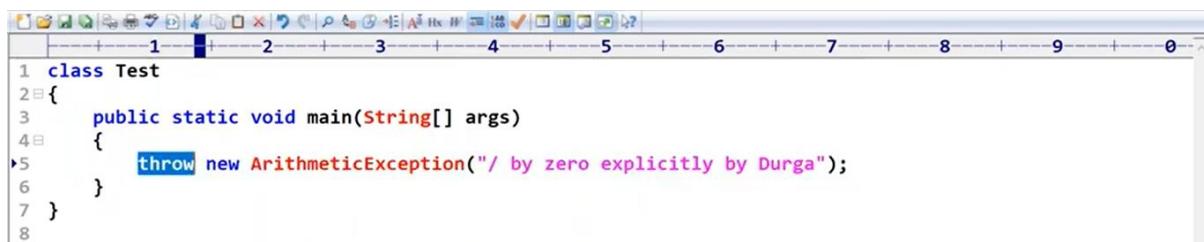
```
My Webin Command Prompt
D:\durga_classes>javac Test.java
D:\durga_classes>java Test
Exception in thread "main" java.lang.ArithmeicException: / by zero
at Test.main(Test.java:5)
```

The Main method created the exception Object and handed over by the method to JVM

Sometimes we can create an exception object explicitly and hand it over to JVM. Then we will use throw keyword, after creating the Exception Object hand it over to the JVM



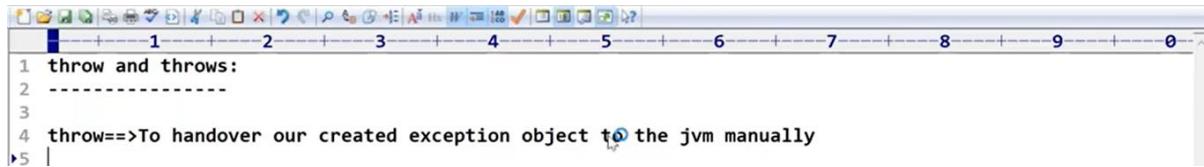
```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         new ArithmeticException("/ by zero explicitly by Durga");
6     }
7 }
8
```



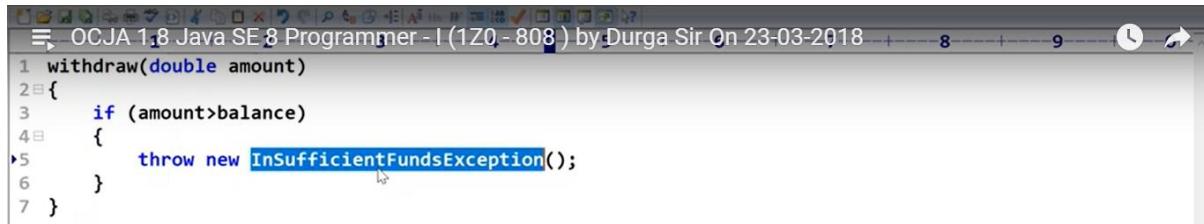
```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         throw new ArithmeticException("/ by zero explicitly by Durga");
6     }
7 }
8
```

See the Explicitly by Durga in the below

```
D:\durga_classes>javac Test.java
D:\durga_classes>java Test
Exception in thread "main" java.lang.ArithmetiException: / by zero explicitly by
Durga
at Test.main(Test.java:5)
```



```
1 throw and throws:
2 -----
3
4 throw==>To handover our created exception object to the jvm manually
5 |
```



```
OCJA 1.8 Java SE 8 Programmer - I (1Z0-808) by Durga Sir On 23-03-2018
1 withdraw(double amount)
2 {
3     if (amount>balance)
4     {
5         throw new InSufficientFundsException();
6     }
7 }
```

```

18 Matrinomial website:
19 -----
20
21 Age: 99
22
23 throw new TooYoungException("plz wait some more time definately u will get best match")
24
25

```

Throws

If there are any chance of Checked Exception being thrown then we need to handle them by try catch or throws Keyword. Eg below

There could be a chance of file not found exception but u r not handling it, CE

The screenshot shows a Java IDE interface with a code editor and a terminal window.

Code Editor:

```

1 import java.io.*;
2 class Test
3 {
4     public static void main(String[] args)
5     {
6         PrintWriter pw= new PrintWriter("abc.txt");
7         pw.println("Hello");
8     }
9 }
10

```

Terminal Window (Command Prompt):

```

D:\durga_classes>{
'{ is not recognized as an internal or external command,
operable program or batch file.

D:\durga_classes>javac Test.java
Test.java:6: error: unreported exception FileNotFoundException; must be caught or
declared to be thrown
        PrintWriter pw= new PrintWriter("abc.txt");
                           ^
1 error

```

Eg2

The screenshot shows a Java IDE interface with a code editor.

Code Editor:

```

1 import java.io.*;
2 class Test
3 {
4     public static void main(String[] args)
5     {
6         Thread.sleep(3000); I
7     }
8 }

```

Code Editor (Continued):

```

32 throws:
33 -----
34 InterruptedException=>checked
35

```

CE is say the exception is not reported

```
D:\durga_classes>javac Test.java
Test.java:5: error: unreported exception InterruptedException; must be caught or declared to be thrown
        Thread.sleep(3000);
                           ^
1 error
D:\durga_classes>_
```

```
28
29
30
31
32 throws:
33 -----
34 InterruptedException==>checked
35
36
37 small work to you...
38 -----
39 1. you can do on your own==>try-catch
40 2. You can delegate that work to the next person==>throws|
41
42
43
^ ^
```

Line no 5 throws the exception, which is in main method. Main Method can delegate its called JVM to handle the exception by throwing the exception

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0-808) by Durga Sir On 23-03-2018
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         Thread.sleep(3000);
6     }
7 }
```

Question	Asker
0	Hooja Chavan
5	Shubham Gupta
line 5	Kalyan TC
5	Deependra Yadav
main	Govindu Rayapur

was attending your core java classes

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0-808) by Durga Sir On 23-03-2018
1 class Test
2 {
3     public static void main(String[] args) throws InterruptedException
4     {
5         Thread.sleep(3000);
6     }
7 }
```

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0-808) by Durga Sir On 23-03-2018 8 9

34 InterruptedException====>checked
35
36
37 small work to you...
38 -----
39 1. you can do on your own==>try-catch
40 2. You can delegate that work to the next person==>throws
41
42
43
44 throws ==>To delegate responsibility of exception handling to the caller(may be another method or
> jvm)
```

Three levels of method calls

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0-808) by Durga Sir On 23-03-2018 8 9

1 class Test
2 {
3     public static void main(String[] args)
4     {
5         doStuff();
6     }
7     public static void doStuff()
8     {
9         doMoreStuff();
10    }
11    public static void doMoreStuff()
12    {
13        Thread.sleep(3000);
14    }
15 }
```

Line13 triggers the exception. CE asks doMoreStuff for Handling

```
D:\durga_classes>javac Test.java
D:\durga_classes>javac Test.java
Test.java:13: error: unreported exception InterruptedException; must be caught or
declared to be thrown
        Thread.sleep(3000);
                           ^
1 error
```

Either use try catch or delegate the responsibility to caller (do Stuff). So u can use throws to ask CE compiler to check with doStuff

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0-808) by Durga Sir On 23-03-2018 8 9

1 class Test
2 {
3     public static void main(String[] args)
4     {
5         doStuff();
6     }
7     public static void doStuff()
8     {
9         doMoreStuff();
10    }
11    public static void doMoreStuff() throws InterruptedException
12    {
13        Thread.sleep(3000);
14    }
15 }
```

Check the error its in line 9 . Now do stuff can either use Try catch or delegate the responsibility to the caller (Main Method) _

Test.java:9: error: unreported exception InterruptedException; must be caught or declared to be thrown
 Thread.sleep(3000);
 ^
1 error

D:\durga_classes>javac Test.java
Test.java:9: error: unreported exception InterruptedException; must be caught or declared to be thrown
 doMoreStuff();
 ^
1 error

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         doStuff();
6     }
7     public static void doStuff() throws InterruptedException
8     {
9         doMoreStuff();
10    }
11    public static void doMoreStuff() throws InterruptedException
12    {
13        Thread.sleep(3000);
14    }
15 }
```

The exception will be thrown at line 5

D:\durga_classes>javac Test.java
Test.java:5: error: unreported exception InterruptedException; must be caught or declared to be thrown
 doStuff();
 ^
1 error

Now Main Method has delegated the exception handling to JVM. CE will now ignore it

```
1 class Test
2 {
3     public static void main(String[] args) throws InterruptedException
4     {
5         doStuff();
6     }
7     public static void doStuff() throws InterruptedException
8     {
9         doMoreStuff();
10    }
11    public static void doMoreStuff() throws InterruptedException
12    {
13        Thread.sleep(3000);
14    }
15 }
```

D:\durga_classes>javac Test.java
D:\durga_classes>

Delegating

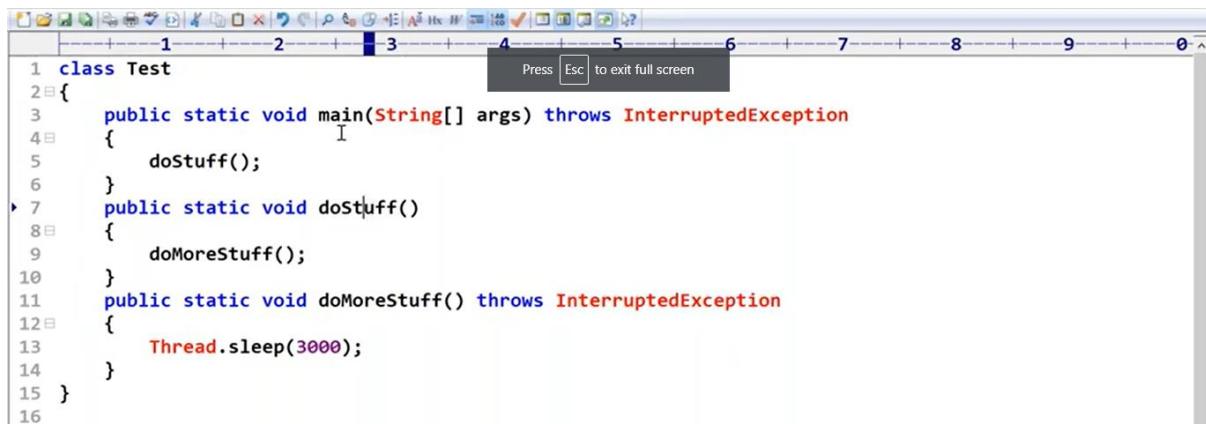
```
OCJA 1.8 Java SE 8 Programmer - I (1Z0_808) by Durga Sir On 23-03-2018
1 class Test
2 {
3     public static void main(String[] args) throws InterruptedException
4     {
5         doStuff();
6     }
7     public static void doStuff() throws InterruptedException
8     {
9         doMoreStuff();
10    }
11    public static void doMoreStuff() throws InterruptedException
12    {
13        Thread.sleep(3000);
14    }
15 }
```

InValid- Compiler will check doMoreStuff to handle it , if we don't delegate it by throws in Line 11 how would it check with doStuff and then to Main Method . CE knows that Main is throwing this but u have not delegated it to Main Method. First Delegate

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0_808) by Durga Sir On 23-03-2018
1 class Test
2 {
3     public static void main(String[] args) throws InterruptedException
4     {
5         doStuff();
6     }
7     public static void doStuff()
8     {
9         doMoreStuff();
10    }
11    public static void doMoreStuff()
12    {
13        Thread.sleep(3000);
14    }
15 }
```

```
D:\durga_classes>javac Test.java
D:\durga_classes>javac Test.java
Test.java:13: error: unreported exception InterruptedException; must be caught or
declared to be thrown
        Thread.sleep(3000);
                           ^
1 error
D:\durga_classes>throws InterruptedException
```

The ball comes to DoStuff



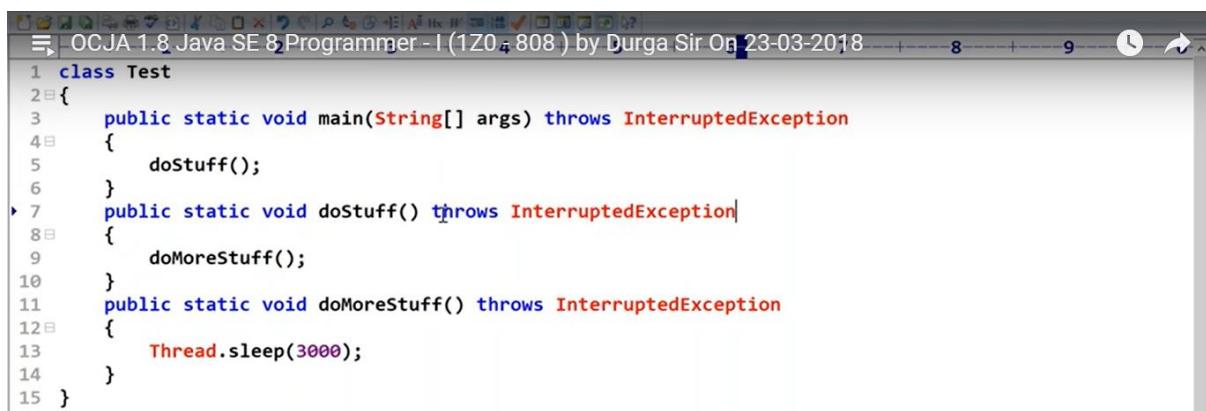
```
1 class Test
2 {
3     public static void main(String[] args) throws InterruptedException
4     {
5         doStuff();
6     }
7     public static void doStuff()
8     {
9         doMoreStuff();
10    }
11    public static void doMoreStuff() throws InterruptedException
12    {
13        Thread.sleep(3000);
14    }
15 }
```

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

```
D:\durga_classes>javac Test.java
Test.java:13: error: unreported exception InterruptedException; must be caught or
declared to be thrown
        Thread.sleep(3000);           ^
1 error
```

```
D:\durga_classes>javac Test.java
Test.java:9: error: unreported exception InterruptedException; must be caught or d
eclared to be thrown
        doMoreStuff();            ^
1 error
```

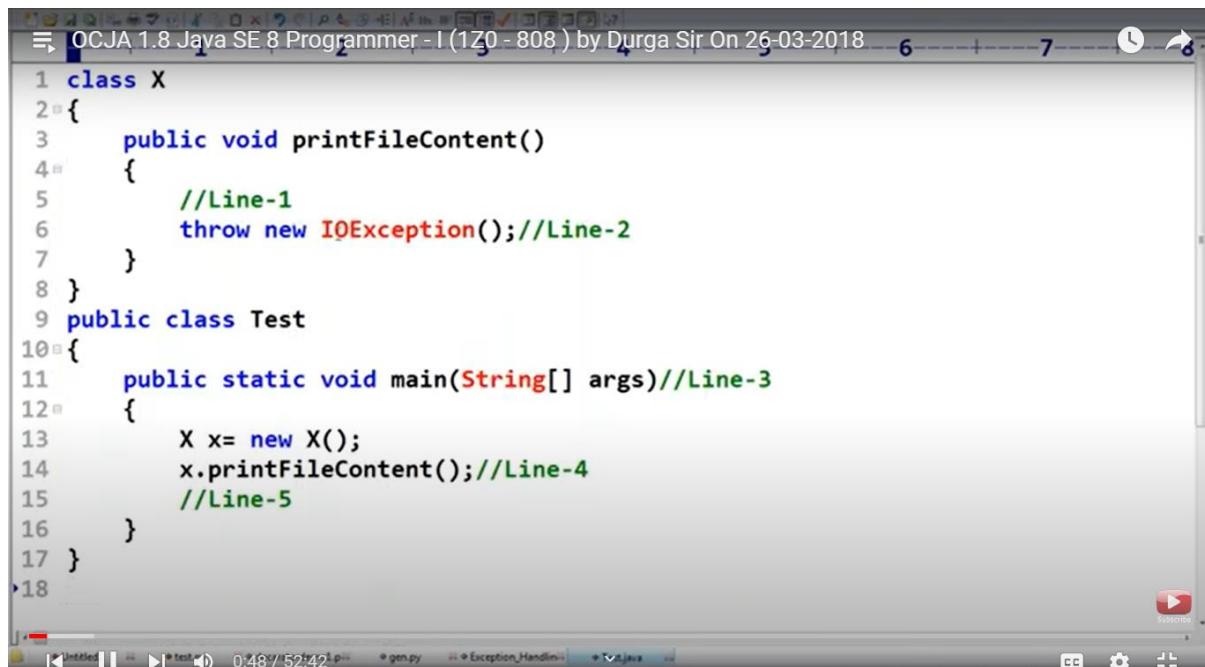
It compiles now



```
1 class Test
2 {
3     public static void main(String[] args) throws InterruptedException
4     {
5         doStuff();
6     }
7     public static void doStuff() throws InterruptedException
8     {
9         doMoreStuff();
10    }
11    public static void doMoreStuff() throws InterruptedException
12    {
13        Thread.sleep(3000);
14    }
15 }
```

Practice

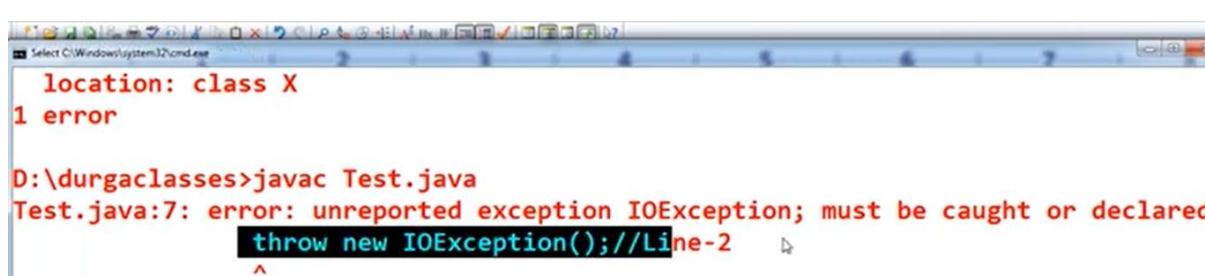
Prob1: If a program is throwing any exception it should be handled or caught by throws



A screenshot of a Java IDE showing a code editor with the following Java code:

```
1 class X
2 {
3     public void printFileContent()
4     {
5         //Line-1
6         throw new IOException(); //Line-2
7     }
8 }
9 public class Test
10{
11     public static void main(String[] args) //Line-3
12     {
13         X x= new X();
14         x.printFileContent(); //Line-4
15         //Line-5
16     }
17 }
```

The code editor has line numbers on the left and a status bar at the bottom.



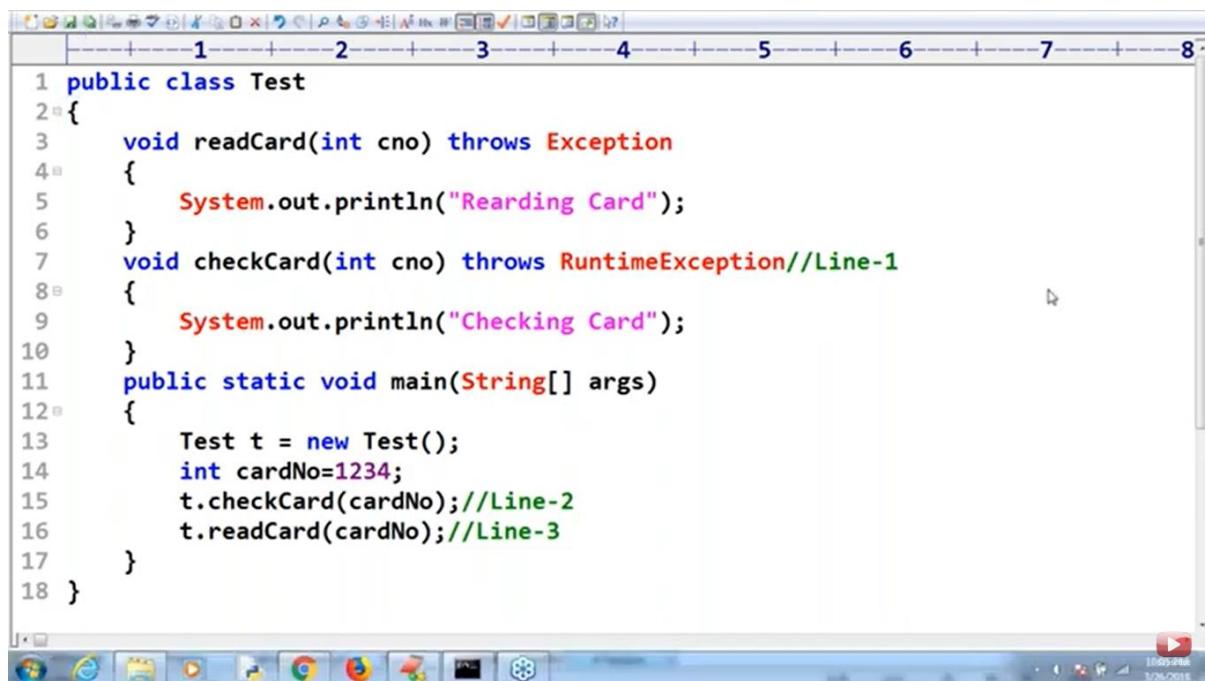
A screenshot of a terminal window showing the output of a javac compilation:

```
location: class X
1 error

D:\durgaclasses>javac Test.java
Test.java:7: error: unreported exception IOException; must be caught or declared
        throw new IOException(); //Line-2
                           ^
1 error
```

Using throws will not handle the exception, if the exception occurs it will terminate the program.
Using throws will not prevent abnormal termination of the program, just stops CE error

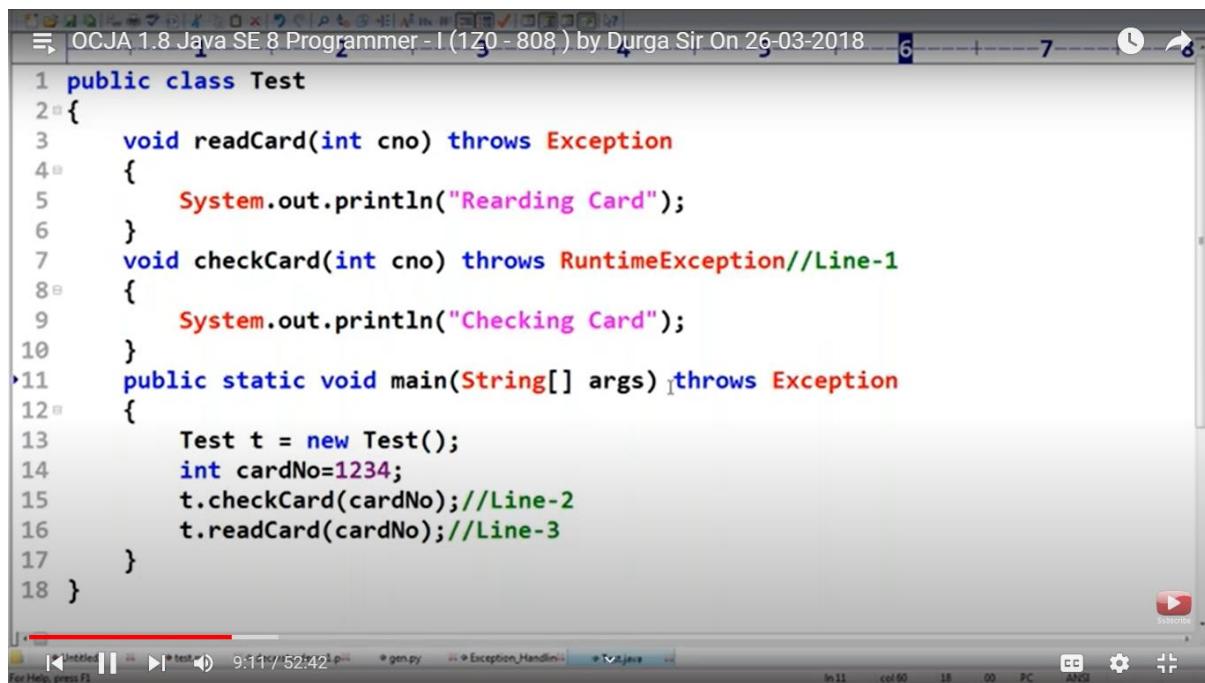
Throws Keyword can be used for Checked Exception for being more meaningful.



```
1 public class Test
2 {
3     void readCard(int cno) throws Exception
4     {
5         System.out.println("Reading Card");
6     }
7     void checkCard(int cno) throws RuntimeException//Line-1
8     {
9         System.out.println("Checking Card");
10    }
11    public static void main(String[] args)
12    {
13        Test t = new Test();
14        int cardNo=1234;
15        t.checkCard(cardNo);//Line-2
16        t.readCard(cardNo);//Line-3
17    }
18 }
```

RunTimeException is unchecked Exception – So we don't need handle it so main need not throw the RunTimeException neither do checkCard. Having it is ok does not provide any meaning

ReadCard – throws Exception, its compulsory as its Partially Checked Exception. So, it has to be handled compulsory and thrown by Main Method, either use try catch or throws



```
1 public class Test
2 {
3     void readCard(int cno) throws Exception
4     {
5         System.out.println("Reading Card");
6     }
7     void checkCard(int cno) throws RuntimeException//Line-1
8     {
9         System.out.println("Checking Card");
10    }
11    public static void main(String[] args) throws Exception
12    {
13        Test t = new Test();
14        int cardNo=1234;
15        t.checkCard(cardNo);//Line-2
16        t.readCard(cardNo);//Line-3
17    }
18 }
```

A screenshot of a Java IDE interface. The code editor displays the following Java code:

```
1 public class MyException extends RuntimeException
2 {
3 }
4 public class Test
5 {
6     public static void main(String[] args)
7     {
8         try
9         {
10            m1();
11        } catch (MyException e)
12        {
13            System.out.print("A");
14        }
15    }
16    public static void m1()
17    {
18        try
19        {
20            throw Math.random() > 0.5 ? new Exception():new MyException();
21        } catch (RuntimeException e)
22        {
23            System.out.println("B");
24        }
25    }
26 }
27 }
28 }
```

The line `throw Math.random() > 0.5 ? new Exception():new MyException();` is highlighted in blue, indicating it is currently selected or being edited.

Line 21 throws Exception or MyException. Both are possible, We are handling RunTimeException but not Exception and MyException

A screenshot of a Java IDE interface. The code editor displays the following Java code:

```
10     m1();
11   } catch (MyException e)
12   {
13     System.out.print("A");
14   }
15 }
16 public static void m1()
17 {
18     try
19     {
20         throw Math.random() > 0.5 ? new Exception():new MyException();
21     } catch (Exception e)
22     {
23         System.out.println("B");
24     }
25 }
26 }
27 }
28 }
```

The line `throw Math.random() > 0.5 ? new Exception():new MyException();` is highlighted in blue, indicating it is currently selected or being edited.

```
D:\durgaclasses>javac Test.java  
D:\durgaclasses>javac Test.java  
Test.java:1: error: class MyException is public, should be declared in a file na  
public class MyException extends RuntimeException  
      ^  
1 error
```

A screenshot of a Java IDE interface. The code editor window shows a Java file named 'Test.java' with the following content:

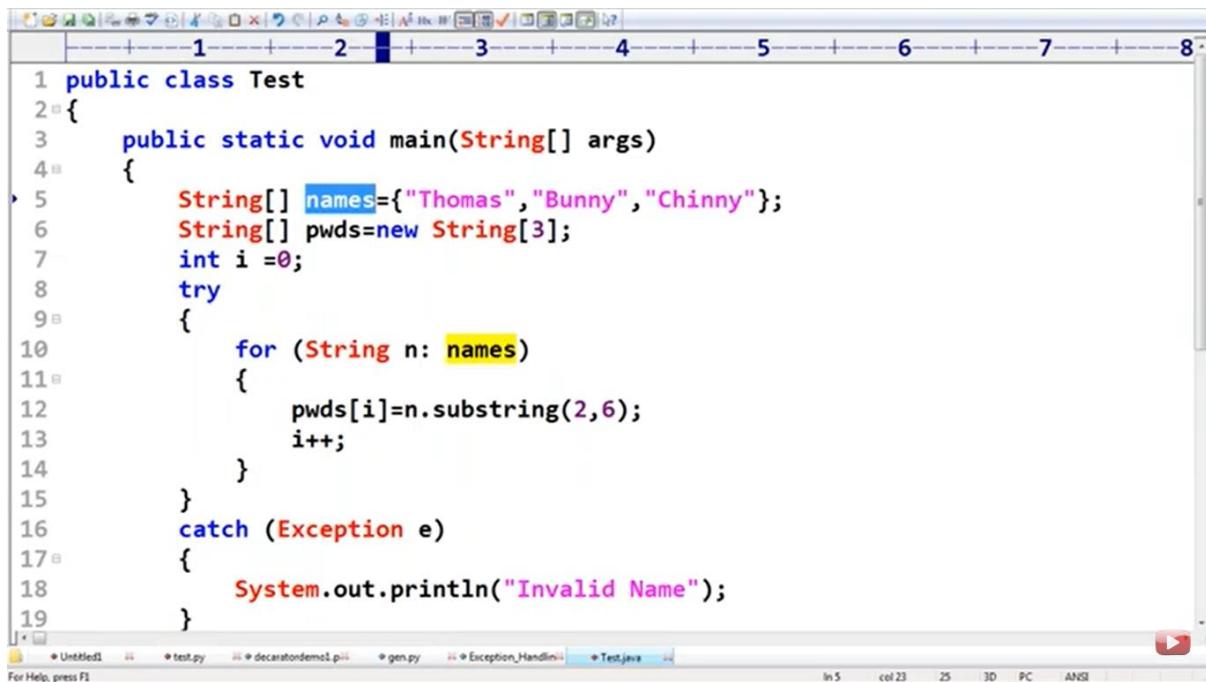
```
1 public class Test
2 {
3     public static void main(String[] args)
4     {
5         String[] s= new String[2];
6         int i=0;
7         for(String s1: s)
8         {
9             s[i].concat("element"+i);
10            i++;
11        }
12        for(i=0; i<s.length;i++)
13        {
14            System.out.println(s[i]);
15        }
16    }
17 }
```

The code editor has a toolbar at the top with various icons. Below the code editor is a tab bar containing several tabs: 'Untitled1', 'test.py', 'decoratordemo1.py', 'gen.py', 'Exception_Handling', and 'Test.java'. The 'Test.java' tab is currently selected. At the bottom of the interface, there is a status bar displaying 'in 9 col 17 18 2E PC ANSI'.

We have just declared, the array and not initialized. On null s[i] we cant call any method. Null Pointer Exception will be thrown. At run time not at compiler Time,

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

D:\durgaclasses>java Test
Exception in thread "main" java.lang.NullPointerException
at Test.main(Test.java:9)
```



```
1 public class Test
2 {
3     public static void main(String[] args)
4     {
5         String[] names={"Thomas", "Bunny", "Chinny"};
6         String[] pwds=new String[3];
7         int i =0;
8         try
9         {
10            for (String n: names)
11            {
12                pwds[i]=n.substring(2,6);
13                i++;
14            }
15        } catch (Exception e)
16        {
17            System.out.println("Invalid Name");
18        }
19    }
}
```

SubString(Begin to n-1)

Thomas->(2,6)= 2 to 5 = omas,

Bunny-> 2 to 5 – no six index

This is the exception that will be thrown

The below error is from another program to show u what would be the error

```
D:\durgaclasses>java Test1
Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String ind
at java.lang.String.substring(Unknown Source)
at Test1.main(Test1.java:5)
```

But the exception is handled . So pwds will be Omas, NULL , NULL

So the out put will be “invalid Name” -> which is in catch block

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

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

```
Invalid Name
```

```
omas
```

```
null
```

```
null
```

```
1 import java.util.*;
2 public class Test
3 {
4     public static void main(String[] args)
5     {
6         ArrayList l = new ArrayList();
7         String[] s;
8         try
9         {
10            while(true)
11            {
12                l.add("MyString");
13            }
14        }
15        catch (RuntimeException e)
16        {
17            System.out.println("Caught a RuntimeException");
18        }
19        catch (Exception e)
20        {
21            System.out.println("Caught an Exception");
22        }
23        System.out.println("Ready to use");
24    }
25 }
```

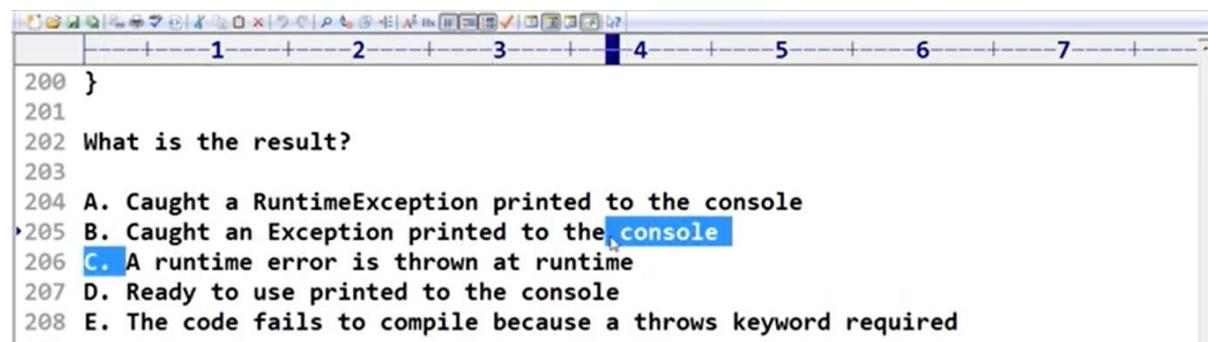
Its not an Exception not an Error and cant be handled. Program terminates abnormally

```
1 import java.util.*;
2 public class Test
3 {
4     public static void main(String[] args)
5     {
6         ArrayList l = new ArrayList();
7         try
8         {
9             while(true)
10            {
11                l.add("MyString");
12            }
13        }
14        catch (RuntimeException e)
15        {
16            System.out.println("Caught a RuntimeException");
17        }
18        catch (Exception e)
19        {
```

Asker
1. Out of memory error
2. Out of memory error
3. Out of memory error
4. Caught a runtimeexception
5. Out of memory error
6. 1-cth
7. 2nd
8. Out of memory error
9. 1-catch
10. RE
Sir can you please explain again why unchecked exp not necessary to handle?

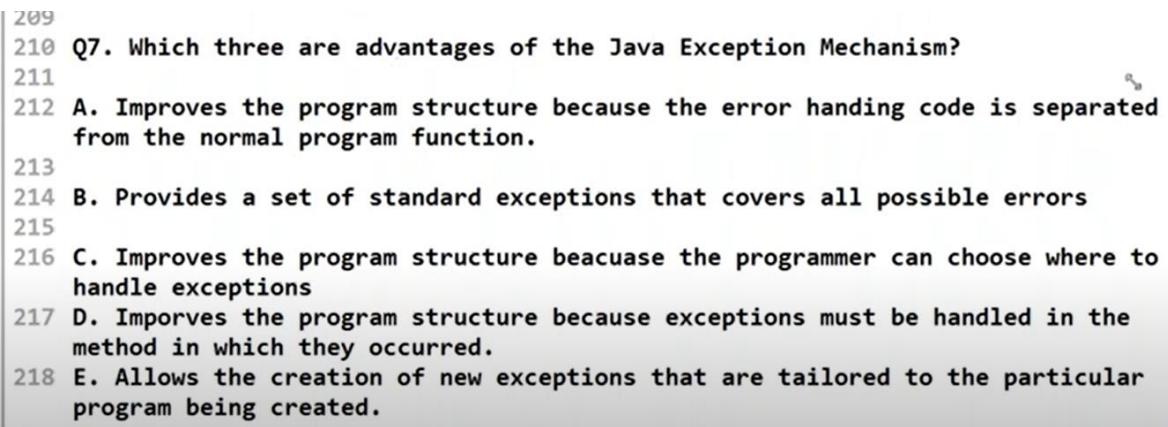
```
D:\durgaclasses>java Test
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
    at java.util.Arrays.copyOf(Unknown Source)
    at java.util.Arrays.copyOf(Unknown Source)
    at java.util.ArrayList.grow(Unknown Source)
    at java.util.ArrayList.ensureExplicitCapacity(Unknown Source)
    at java.util.ArrayList.ensureCapacityInternal(Unknown Source)
    at java.util.ArrayList.add(Unknown Source)
    at Test.main(Test.java:11)
```

C – RunTime Error



200 }
201
202 What is the result?
203
204 A. Caught a RuntimeException printed to the console
205 B. Caught an Exception printed to the console
206 C. A runtime error is thrown at runtime
207 D. Ready to use printed to the console
208 E. The code fails to compile because a throws keyword required
~~~

- A- True
- B- False- Does not Cover all possible Errors
- C- True
- D- False – must be handled not must. Can be delegated
- E- True



209  
210 Q7. Which three are advantages of the Java Exception Mechanism?  
211  
212 A. Improves the program structure because the error handing code is separated from the normal program function.  
213  
214 B. Provides a set of standard exceptions that covers all possible errors  
215  
216 C. Improves the program structure because the programmer can choose where to handle exceptions  
217 D. Improves the program structure because exceptions must be handled in the method in which they occurred.  
218 E. Allows the creation of new exceptions that are tailored to the particular program being created.  
219

```
221 Q8. Which 3 statements are true about exception handling?  
222  
223 A. Only unchecked exceptions can be rethrown  
224 B. All Subclasses of the RuntimeException are recoverable  
225 C. The parameter in catch block is of throwable type  
226 D. All subclasses of RuntimeException must be caught or declared to be thrown  
227 E. All Subclasses of the Exception except RuntimeException class are checked  
exceptions  
228 F. All subclasses of the Error class are checked exceptions and are recoverable  
229
```

Rethrown you can rethrow another exception from another exception. Throw can be used to rethrow

- A- False , Any Exception can be rethrown
- B- Exception and child classes are recoverable, Errors and its child class are not recoverable. So Runtime exception and its child classes are recoverable

Unchecked Exception does not throw compile time error so using throws means nothing to the compiler. Because Compiler is not going to check if there is throws keyword there are not. Its being triggered at run time. Errors cannot be even handled at RunTime and will fail

So all Subclasses of RunTime Exception are recoverable by using try catch

c- TRUE – The argument in catch block is of throwable type which is the root

D-False. RE are unchecked and is not mandatory to be declared to be thrown or to be handled by try catch

E= True , RE and Sub classes are unchecked and all subclasses of Exception other than RE are checked

F- False. Error are unchecked and are not recoverable

B,C,E

C- Error class is extendable can create child class of Error

D- Error is a throwable

OCJA 1.8 Java SE 8 Programmer - I (1Z0-808 ) by Durga Sir On 26-03-2018

```
228 F. All subclasses of the Error class are checked exceptions and are recoverable
229
230
231 Q9. Which two statements are true?
232
233 A. Error class is unextendable
234 B. Error class is extendable
235 C. Error is a RuntimeException
236 D. Error is an Exception
237 E. Error is a Throwable
238
239
```

OCJA 1.8 Java SE 8 Programmer - I (1Z0-808 ) by Durga Sir On 26-03-2018

```
1 class Test extends Throwable
2 {
3     public static void main(String[] args)
4     {
5         I |
6     }
7 }
```

OCJA 1.8 Java SE 8 Programmer - I (1Z0-808 ) by Durga Sir On 26-03-2018

```
1 class Test extends Throwable
2 {
3     public static void main(String[] args)
4     {
5         I |
6     }
7 }
```

```
at Test.main(Test.java:11)
D:\durgaclasses>java Test
D:\durgaclasses>javac Test.java
D:\durgaclasses>
```

We can extend Error RE and Throwanle

OCJA 1.8 Java SE 8 Programmer - I (1Z0-808 ) by Durga Sir On 26-03-2018

```
1 class Test extends Error
2 {
3     public static void main(String[] args)
4     {
5
6     }
7 }
```

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808 ) by Durga Sir On 26-03-2018
D:\durgaclasses>java Test
D:\durgaclasses>javac Test.java
D:\durgaclasses>java Test
D:\durgaclasses>
```

Why should we use RE.. The below is extending RE, its unchecked , if its unchecked even if its throwing compiler never gonna object even if its thrown or not, but at run time it will throw the exception.

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808 ) by Durga Sir On 26-03-2018
1 class TooYoungException extends RuntimeException
2 {
3 }
4 class Test
5 {
6     public static void main(String[] args)
7     {
8         int age=Integer.parseInt(args[0]);
9         if(age>60)
10        {
11            throw new TooYoungException();
12        }
13    }
14 }
15
16
```

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808 ) by Durga Sir On 26-03-2018
1 c
2 { D:\durgaclasses>java Test 99
3 } Exception in thread "main" TooYoungException
4 c
5 { at Test.main(Test.java:11)
```

Here the class is Exception ,which is checked and Compiler will throw and error that this is not handled or uses throws

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

```
1 class TooYoungException extends Exception
2 {
3 }
4 class Test
5 {
6     public static void main(String[] args)
7     {
8         int age=Integer.parseInt(args[0]);
9         if(age>60)
10        {
11            throw new TooYoungException();
12        }
13    }
14 }
15 }
16 }
```

The code defines a class `TooYoungException` that extends the built-in `Exception`. It also contains a `Test` class with a `main` method that checks if the input age is greater than 60 and throws a `TooYoungException` if it is.

When I am creating user defined exception We always prefer RE because we are creating the exception to fail.. if u want it to fail why do you want to handle it... if you can handle it handle it in the code only you don't have to throw exception

The below is valid only ..but does not has any meaning , Any throwable can be used with throws keyword

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

```
1 class Test
2 {
3     public static void main(String[] args) throws ArithmeticException
4     {
5
6     }
7 }
8 }
```

The code defines a `Test` class with a `main` method that declares a `throws` clause specifying `ArithmeticException`.

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

```
1 class Test
2 {
3     public static void main(String[] args) throws Error
4     {
5
6     }
7 }
8 }
```

The code defines a `Test` class with a `main` method that declares a `throws` clause specifying `Error`.

```
1 class Test
2 {
3     public static void main(String[] args) throws Throwable|
4     {
5
6     }
7 }
```

```
1 class Test
2 {
3     public static void main(String[] args) throws String|
4     {
5
6     }
7 }
```

```
OCJA 1.8 Java SE 8 Programmer - I (1Z0 - 808 ) by Durga Sir On 26-03-2018
1 class Test
2 {
3     public static void main(String[] args) throws String|
4     {
5
6     }
7 }
8 1 error

D:\durgaclasses>
```

Throw keyword can be used for any throwable type

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         throw new OutOfMemoryError();|
6     }
7 }
```

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         throw new Throwable();
6     }
7 }
```

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         throw new Thread();
6     }
7 }
8
```

```
D:\durgaclasses>javac Test.java
Test.java:5: error: incompatible types: Thread cannot be converted to Throw
        throw new Thread();
                           ^
1 error
```

```
1 class Test
2 {
3     public static void main(String[] args)
4     {
5         try
6         {
7             System.out.println("Hello");
8         }
9         catch (String e)
10        {
11        }
12    }
13 }
```

```
D:\durgaclasses>javac Test.java
Test.java:9: error: incompatible types: String cannot be converted to Throw
        catch (String e)
                           ^
1 error
```

