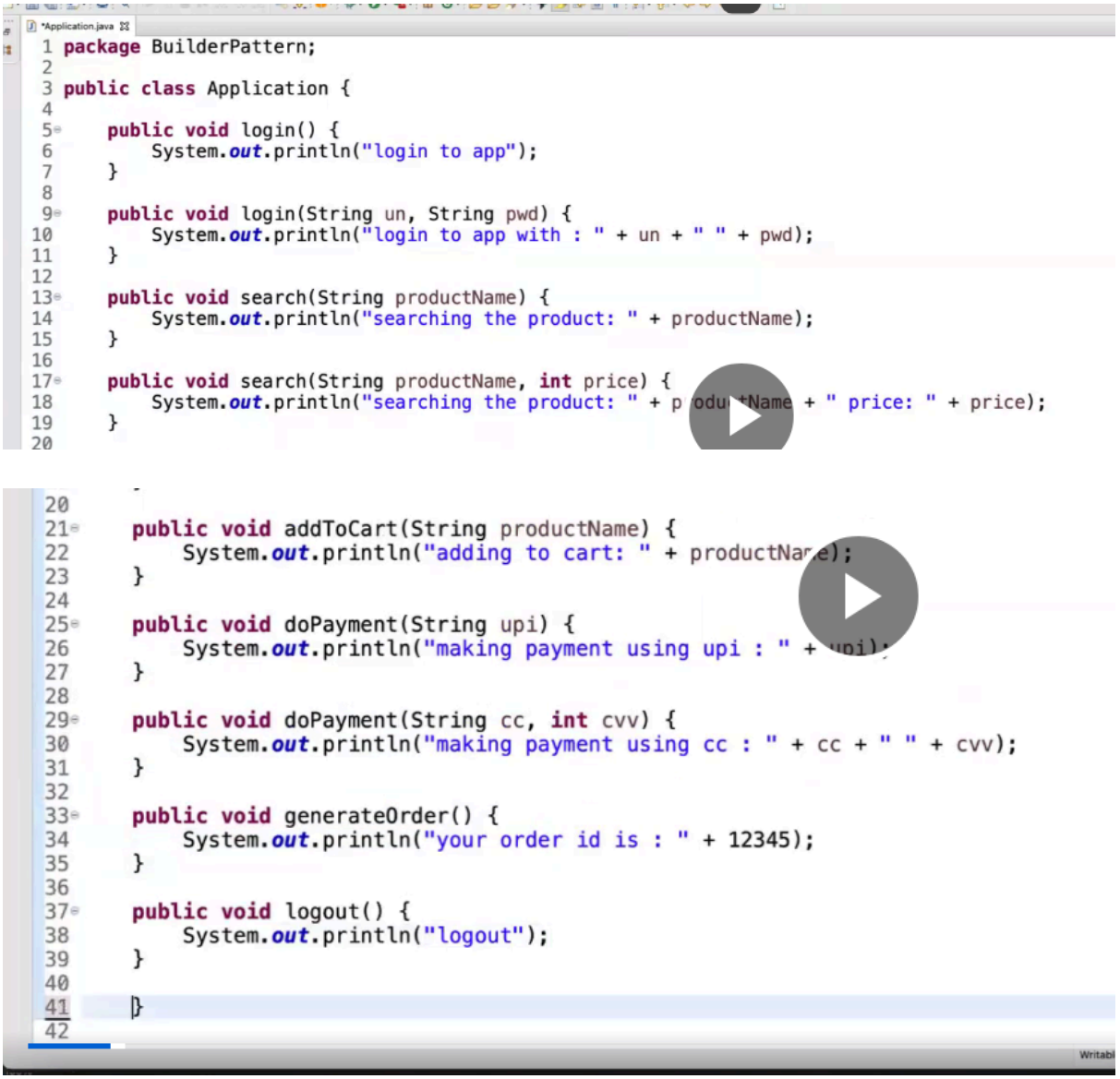
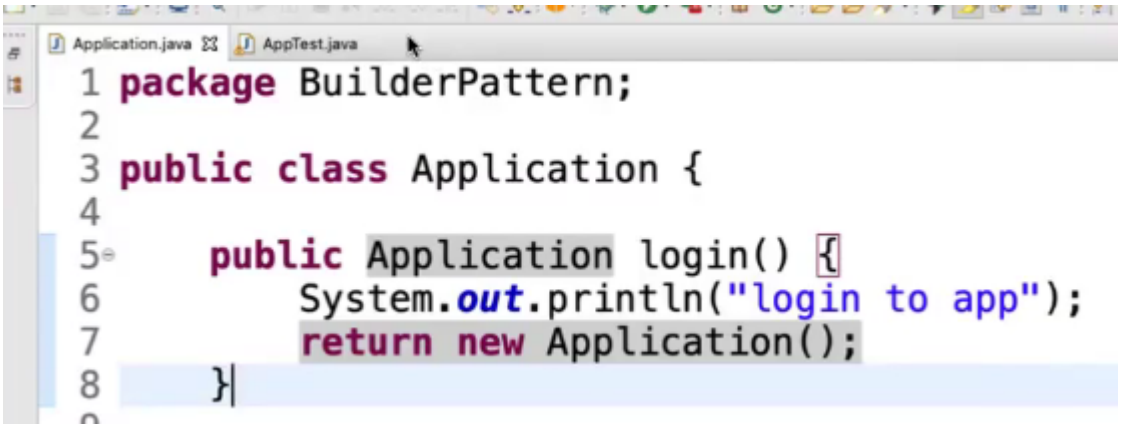
Builder pattern-



```java
package BuilderPattern;

public class Application {

    public void login() {
        System.out.println("login to app");
    }

    public void login(String un, String pwd) {
        System.out.println("login to app with : " + un + " " + pwd);
    }

    public void search(String productName) {
        System.out.println("searching the product: " + productName);
    }

    public void search(String productName, int price) {
        System.out.println("searching the product: " + productName + " price: " + price);
    }

    public void addToCart(String productName) {
        System.out.println("adding to cart: " + productName);
    }

    public void doPayment(String upi) {
        System.out.println("making payment using upi : " + upi);
    }

    public void doPayment(String cc, int cvv) {
        System.out.println("making payment using cc : " + cc + " " + cvv);
    }

    public void generateOrder() {
        System.out.println("your order id is : " + 12345);
    }

    public void logout() {
        System.out.println("logout");
    }

}
```

Login returning the object of application class-



```java
package BuilderPattern;

public class Application {

    public Application login() {
        System.out.println("login to app");
        return new Application();
    }
```

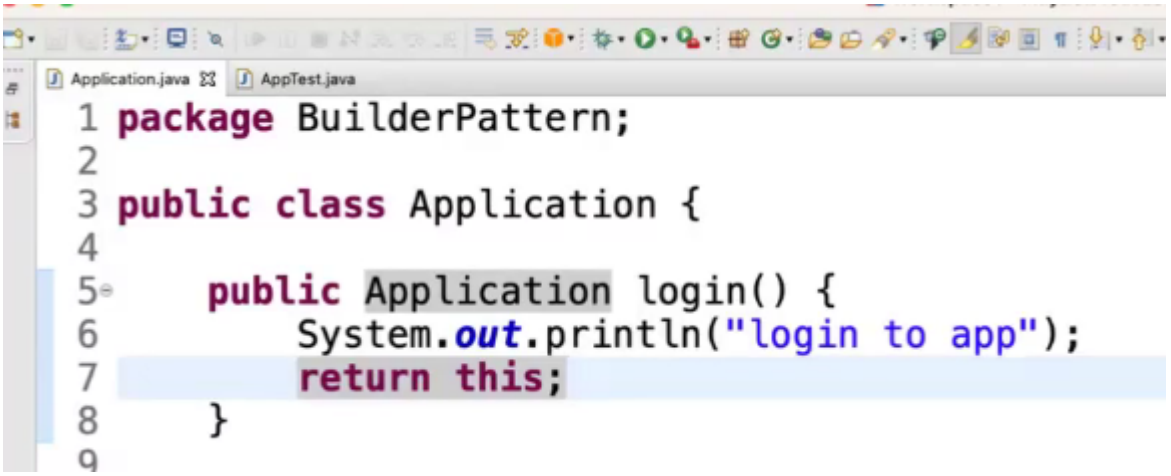
Call login-

Problems-

Two objects created – one in application class when we wrote “return new application” and other in the above class when we call the login. Memory wastage.

In application class, there is no reference to the object, so garbage collector can destroy it anytime.

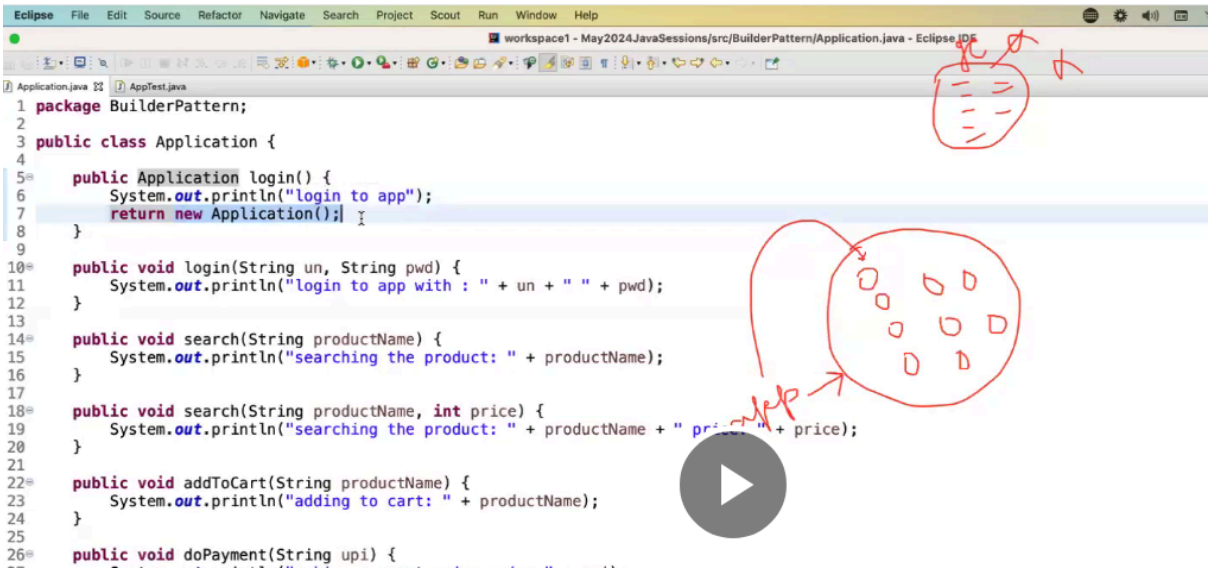
This will solve the problem-



Return “this;” means return the current class object if any existing in memory.

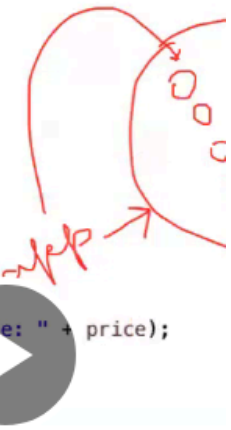
See memory wastage when we write

“return new Application;”-



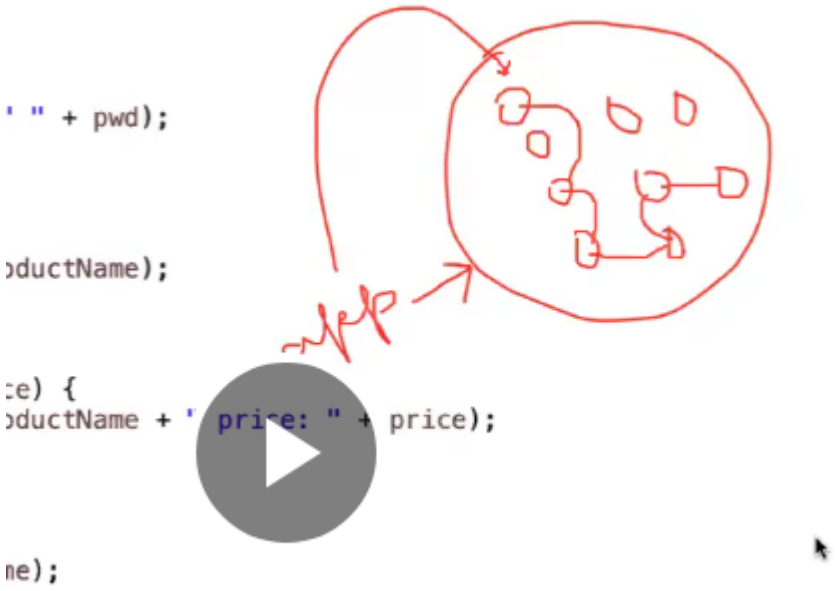
Every method returns this-

```
    public Application login() {
        System.out.println("login to app");
        return this;
    }

    public Application login(String un, String pwd) {
        System.out.println("login to app with : " + un + " " + pwd);
        return this;
    }

    public Application search(String productName) {
        System.out.println("searching the product: " + productName);
        return this;
    }

    public Application search(String productName, int price) {
        System.out.println("searching the product: " + productName + " price: " + price);
        return this;
    }

    public Application addToCart(String productName) {
        System.out.println("adding to cart: " + productName);
        return this;
    }

    public Application doPayment(String upi) {
        System.out.println("making payment using upi : " + upi);
        return this;
    }

    public Application doPayment(String cc, int cvv) {
        System.out.println("making payment using cc : " + cc + " " + cvv);
        return this;
    }

    public Application generateOrder() {
        System.out.println("your order id is : " + 12345);
        return this;
    }
```



```
    public void logout() {
        System.out.println("logout");
        return this;
    }

}
```

Builder pattern-



With one object we are playing around with the entire class.

See builder code in test-

Application.java AppTest.java

```
1  package BuilderPattern;
2
3  public class AppTest {
4
5      public static void main(String[] args) {
6
7
8          Application app = new Application();
9
10
11
12
13         //workflow chain scenario: Builder pattern
14
15         app.login("naveen@gmail.com", "naveen@123")
16             .search("macbook pro")
17                 .addToCart("macbook pro 16 inches")
18                     .doPayment("naveen@okicici")
19                         .generateOrder()
20                             .logout();
21
22
23
24
25
26     }
27
28 }
29
```

For builder pattern, every method has to return “this”.

Another workflow we created-

```
24
25         app.login("naveen@gmail.com", "naveen@123")
26             .search("tshirt", 1000)
27                 .addToCart("nike tshirt")
28                     .logout();
29
```

Third workflow-

```
                    .logout();

        //3.
        app.login("naveen@gmail.com", "naveen@123")
        .search("tshirt", 1000)
            .addToCart("nike tshirt")
                .doPayment("7878 8888 9999 7676", 111)
                    .generateOrder()
                            .logout();
```

Another workflow-

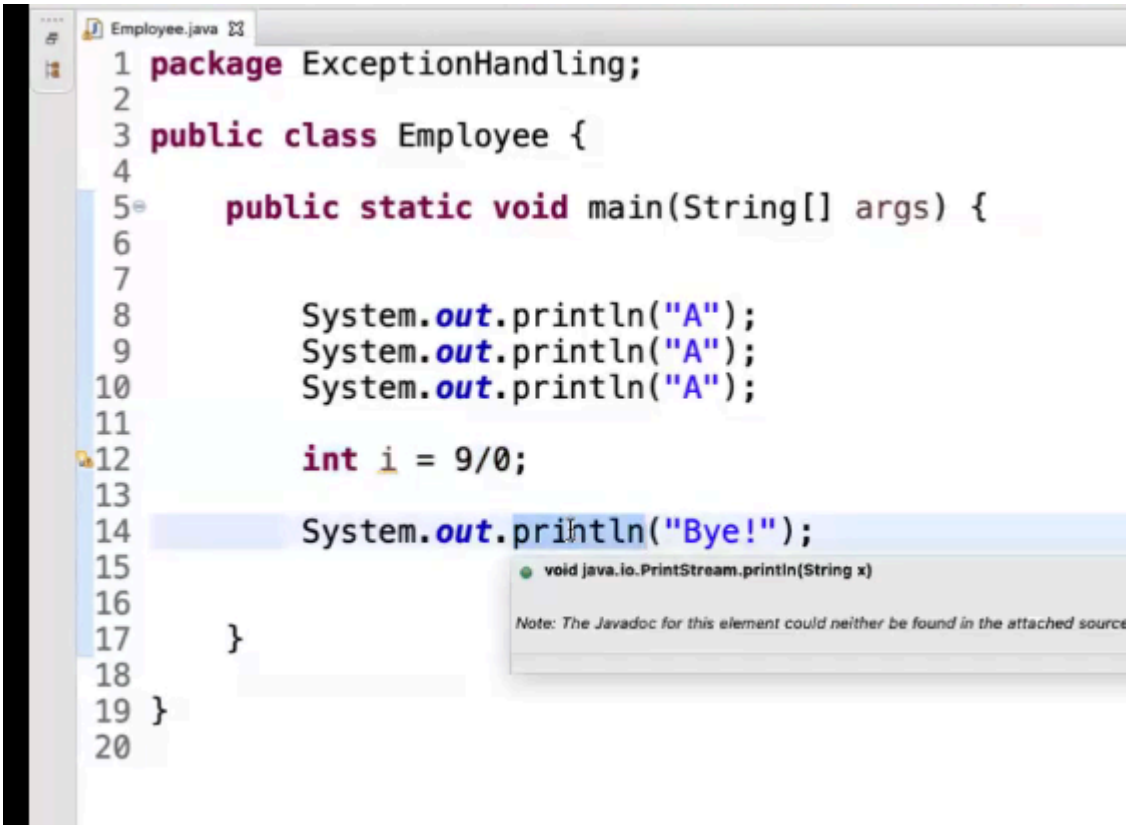
```
        //4.
        app.login("naveen@gmail.com", "naveen@123")
            .logout();
```

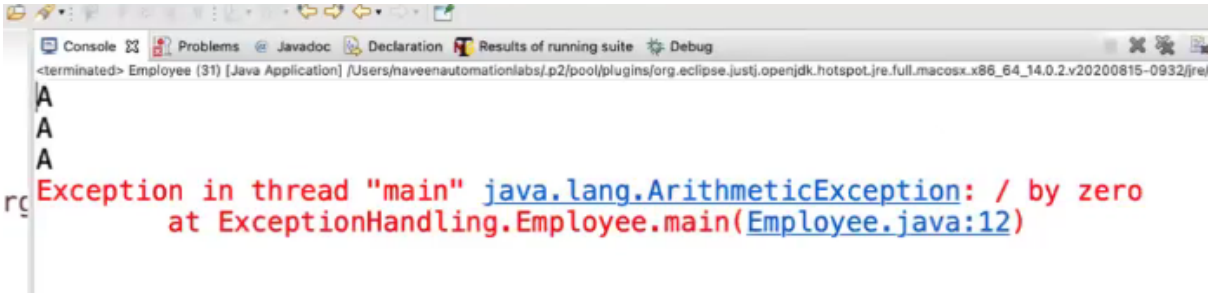
```
        //5.
        app.login("vibha@gmail.com", "vibha@123");


        //6.
        app.login()
            .search("keyboard")
                .addToCart("keyboard apple")
                    .doPayment("vibha@hdfc")
                        .generateOrder()
                            .logout();
```

Builder works for one class only at a time. Every class has to have own “return this” statement. (26.12)

```
//this:
        //1. to init the class variable within const.. or method
        //2. can be used with getter/setter: encapsulation
        //3. same class const... calling
        //return this from a function: builder pattern
```

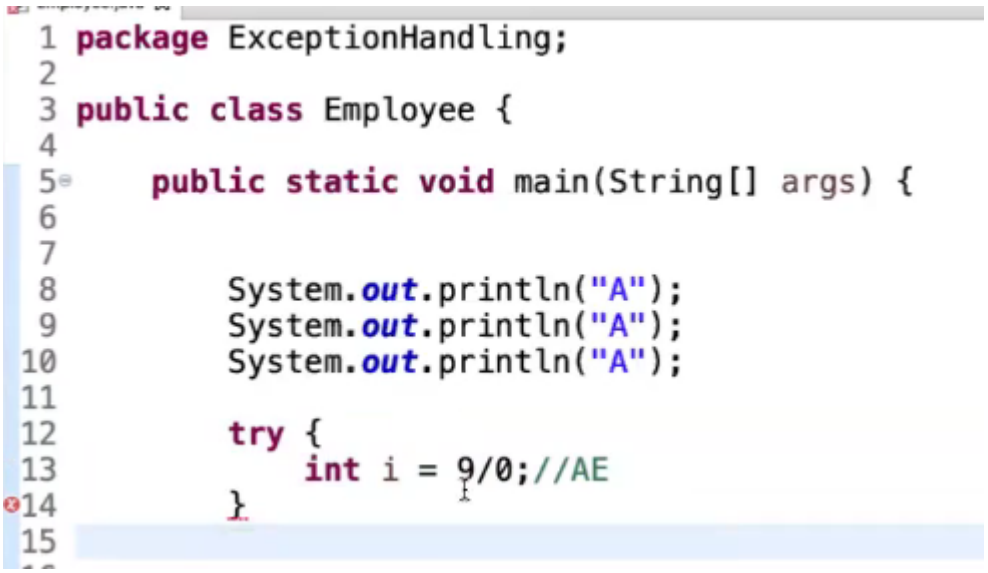
Exception-





Try cannot come alone-

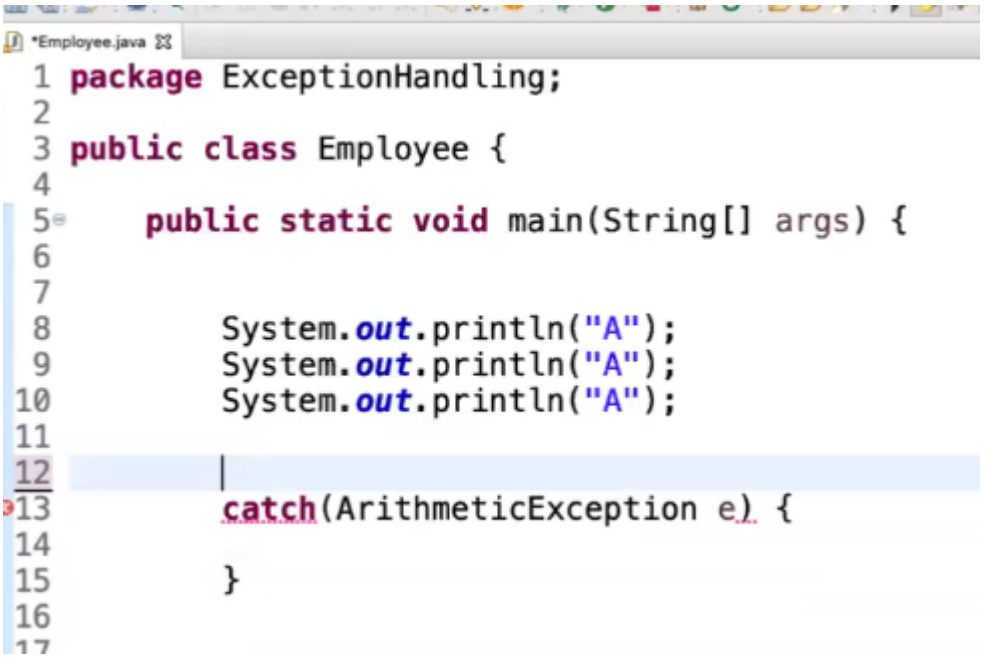
//Syntax error, insert "Finally" to complete BlockStatements



```java
package ExceptionHandling;

public class Employee {

    public static void main(String[] args) {


        System.out.println("A");
        System.out.println("A");
        System.out.println("A");

        try {
            int i = 9/0;//AE
        }

```

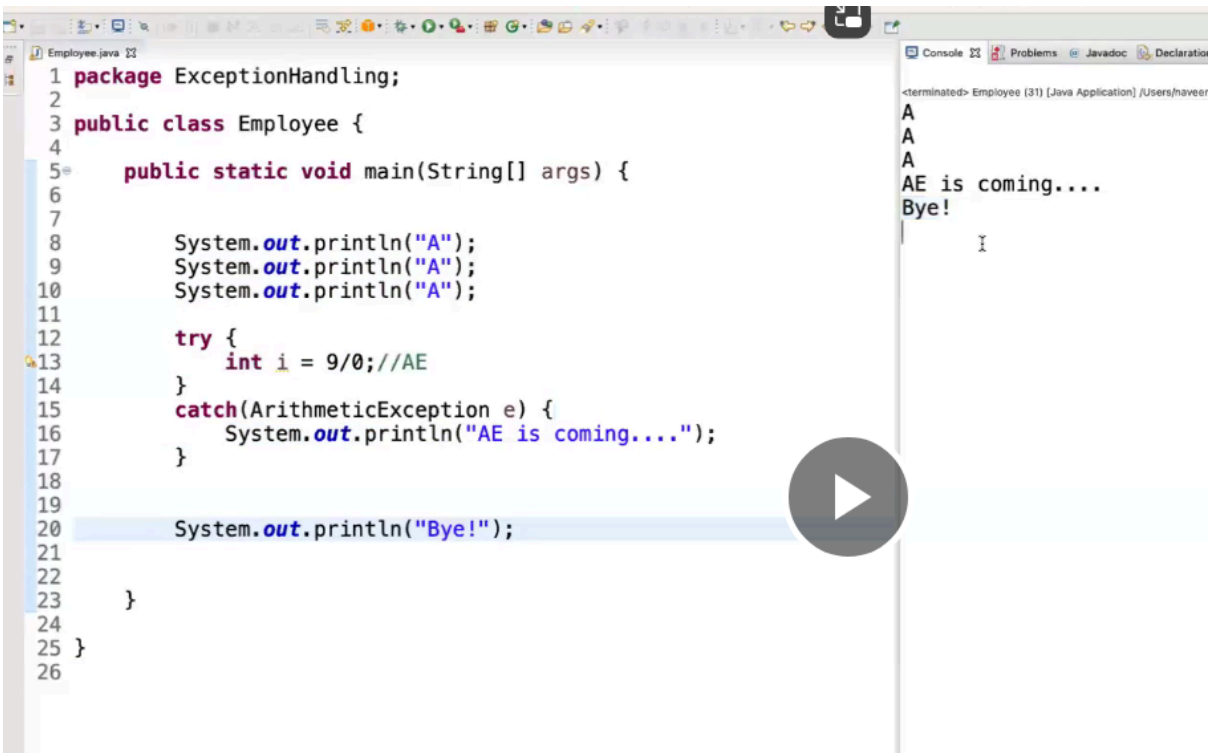
Catch cannot come alone-

//Syntax error on token "catch", invalid RecordHeaderName



```java
*Employee.java
package ExceptionHandling;

public class Employee {

    public static void main(String[] args) {


        System.out.println("A");
        System.out.println("A");
        System.out.println("A");

        catch(ArithmeticException e) {

        }

```

~~36~~



Print stack trace-

```
14          }
15          catch(ArithmeticException e) {
16              System.out.println("AE is coming....");
17              e.printStackTrace();
18          }
19
```

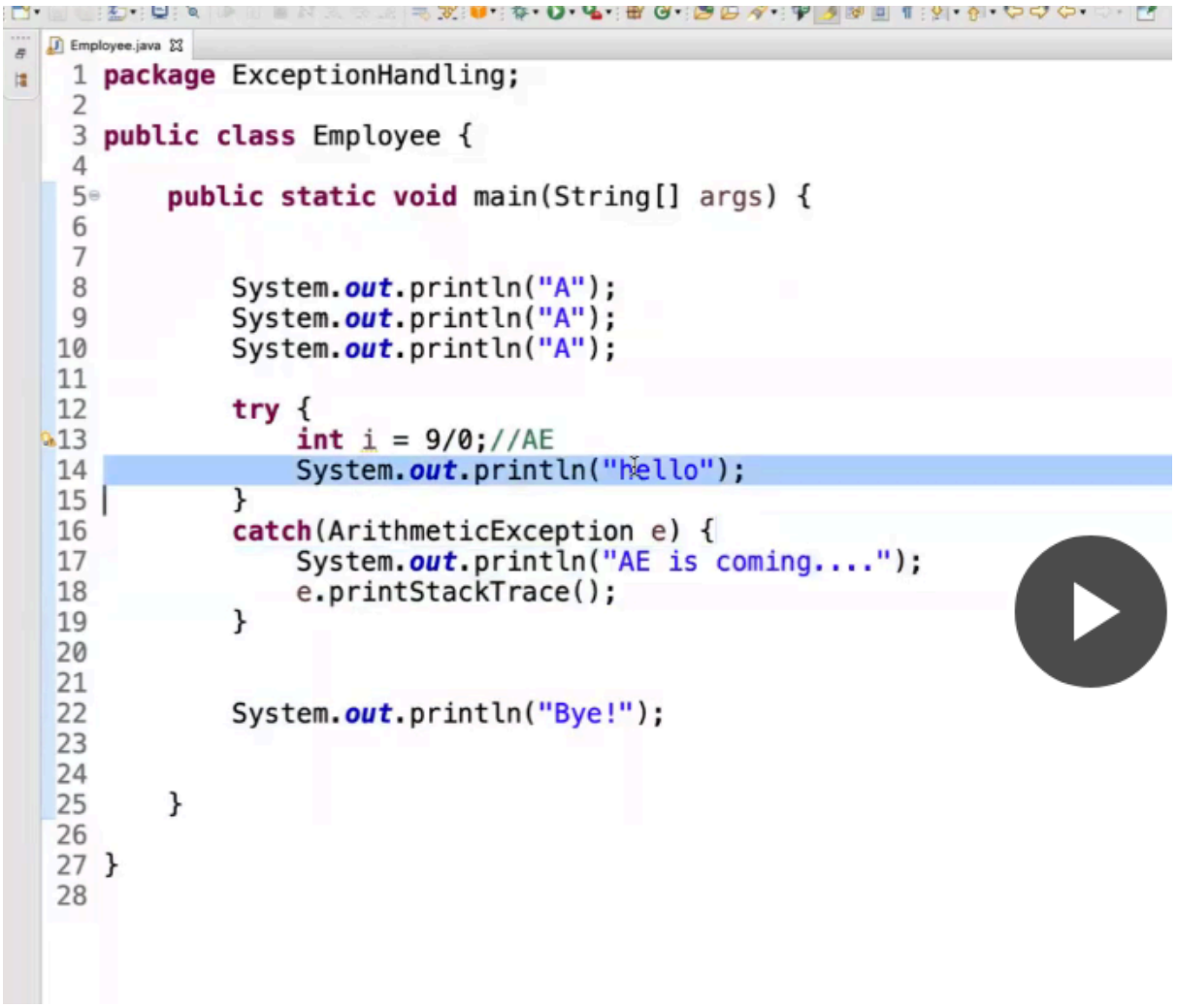
```
java.lang.ArithmeticException: / by zero
        at ExceptionHandling.Employee.main(Employee.java:13)
Bye!
```

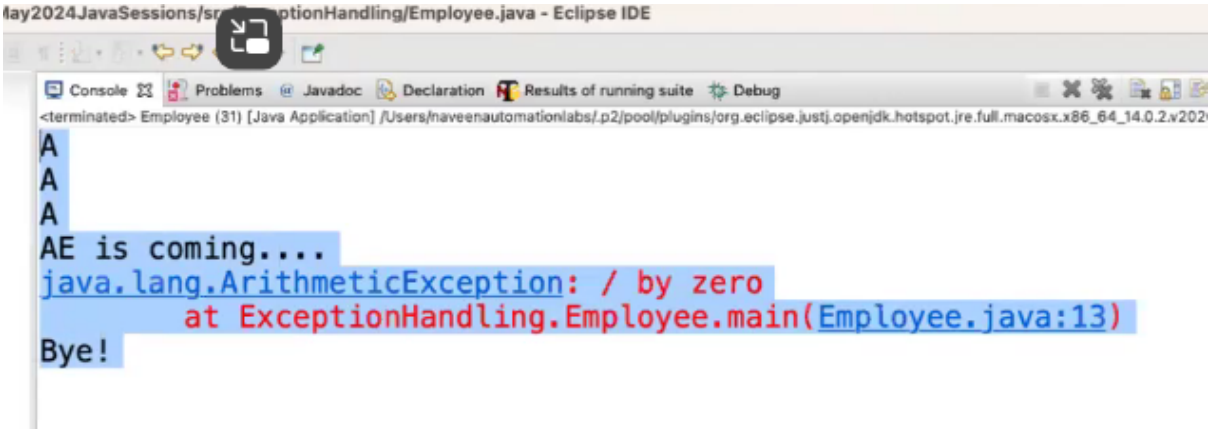
paste exception5-

exception5.java

```java
package com.day22;

public class exception5 {

    public static void main(String[] args) {

        System.out.println("start of the code");

        try {
            int i=9/0;
        }
        catch (ArithmeticException e) {
            System.out.println("arithmetic exception handled");
            //print stack trace.
            e.printStackTrace();//returns void
        }

        System.out.println("outside blocks");
    }

}

//start of the code
//arithmetic exception handled
//java.lang.ArithmeticException: / by zero
//  at com.day22.exception5.main(exception5.java:10)
//outside blocks

```

after exception no line runs in try-





No exception-

Employee.java

```
1  package ExceptionHandling;
2
3  public class Employee {
4
5      public static void main(String[] args) {
6
7
8          System.out.println("A");
9          System.out.println("A");
10         System.out.println("A");
11
12         try {
13             int i = 9/3;//AE
14             System.out.println("hello");
15
16         }
17         catch(ArithmeticException e) {
18             System.out.println("AE is coming....");
19             e.printStackTrace();
20         }
21
22
23         System.out.println("Bye!");
24
25
26     }
27
28 }
29
```

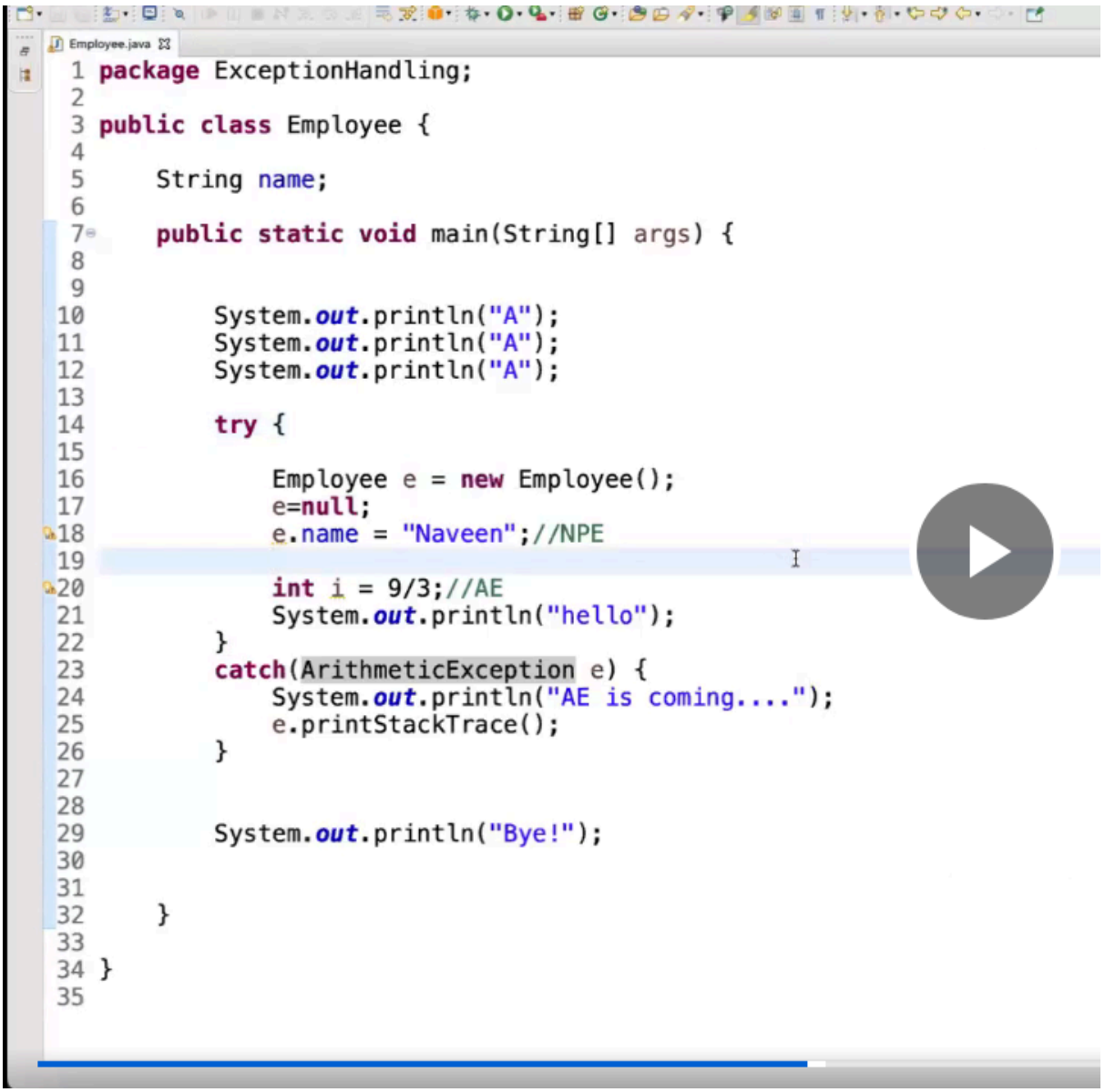
Console

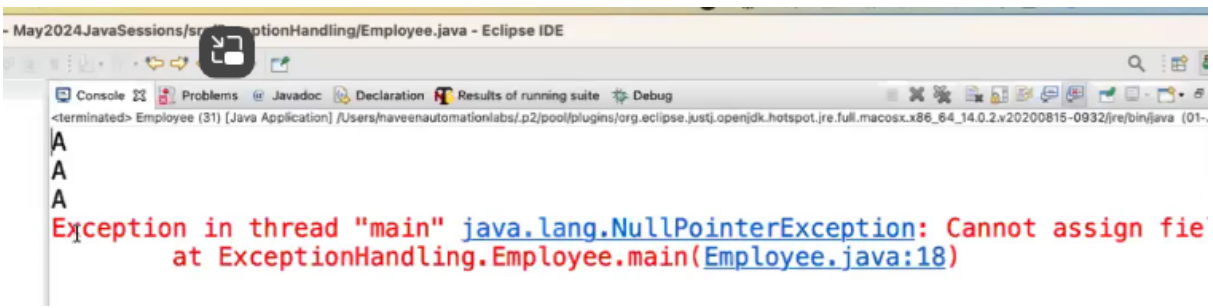
<terminated> Employee

```
A
A
A
hello
Bye!
```

Wont catch one(npe) error- 43.51→

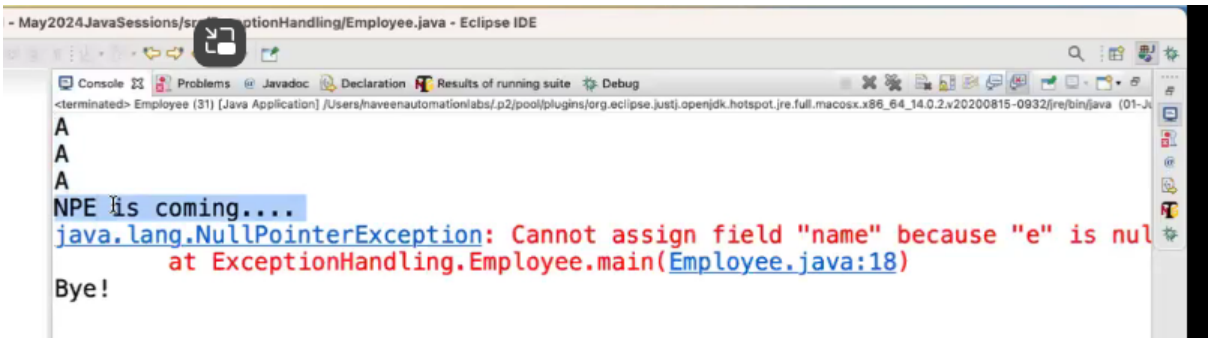
Only one exception captured(arithmetic)


Employee.java
1 package ExceptionHandling;
2
3 public class Employee {
4
5 String name;
6
7 public static void main(String[] args) {
8
9
10 System.out.println("A");
11 System.out.println("A");
12 System.out.println("A");
13
14 try {
15
16 Employee e = new Employee();
17 e=null;
18 e.name = "Naveen";//NPE
19
20 int i = 9/3;//AE
21 System.out.println("hello");
22 }
23 catch(ArithmeticException e) {
24 System.out.println("AE is coming....");
25 e.printStackTrace();
26 }
27
28
29 System.out.println("Bye!");
30
31
32 }
33
34 }
35


- May2024JavaSessions/sr
ptionHandling/Employee.java - Eclipse IDE
Console
Problems
Javadoc
Declaration
Results of running suite
Debug
<terminated> Employee (31) [Java Application] /Users/naveenautomationlabs/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86_64_14.0.2.v20200815-0932/jre/bin/java
A
A
A
Exception in thread "main" java.lang.NullPointerException: Cannot assign fie
at ExceptionHandling.Employee.main(Employee.java:18)

Add for npe-

```
21          System.out.println("hello");
22      }
23      catch(ArithmeticException e) {
24          System.out.println("AE is coming....");
25          e.printStackTrace();
26      }
27      catch(NullPointerException e) {
28          System.out.println("NPE is coming....");
29          e.printStackTrace();
30      }
31
```



paste exception10-

exception10.java

```java
package com.day22;

public class exception10 {

    String name;

    //both exceptions come.
    //after the first exception rest of the code in try ignored.

    public static void main(String[] args) {

        System.out.println("start of the code");

        try {

            exception10 e1=new exception10();
            e1=null;
            e1.name=null;
            e1.name="karan";
            System.out.println(e1.name);

            int i=9/0;
            System.out.println("after exception try block");
        }
        catch (ArithmeticException e) {
            System.out.println("arithmetic exception handled");
            e.printStackTrace();
        }
```

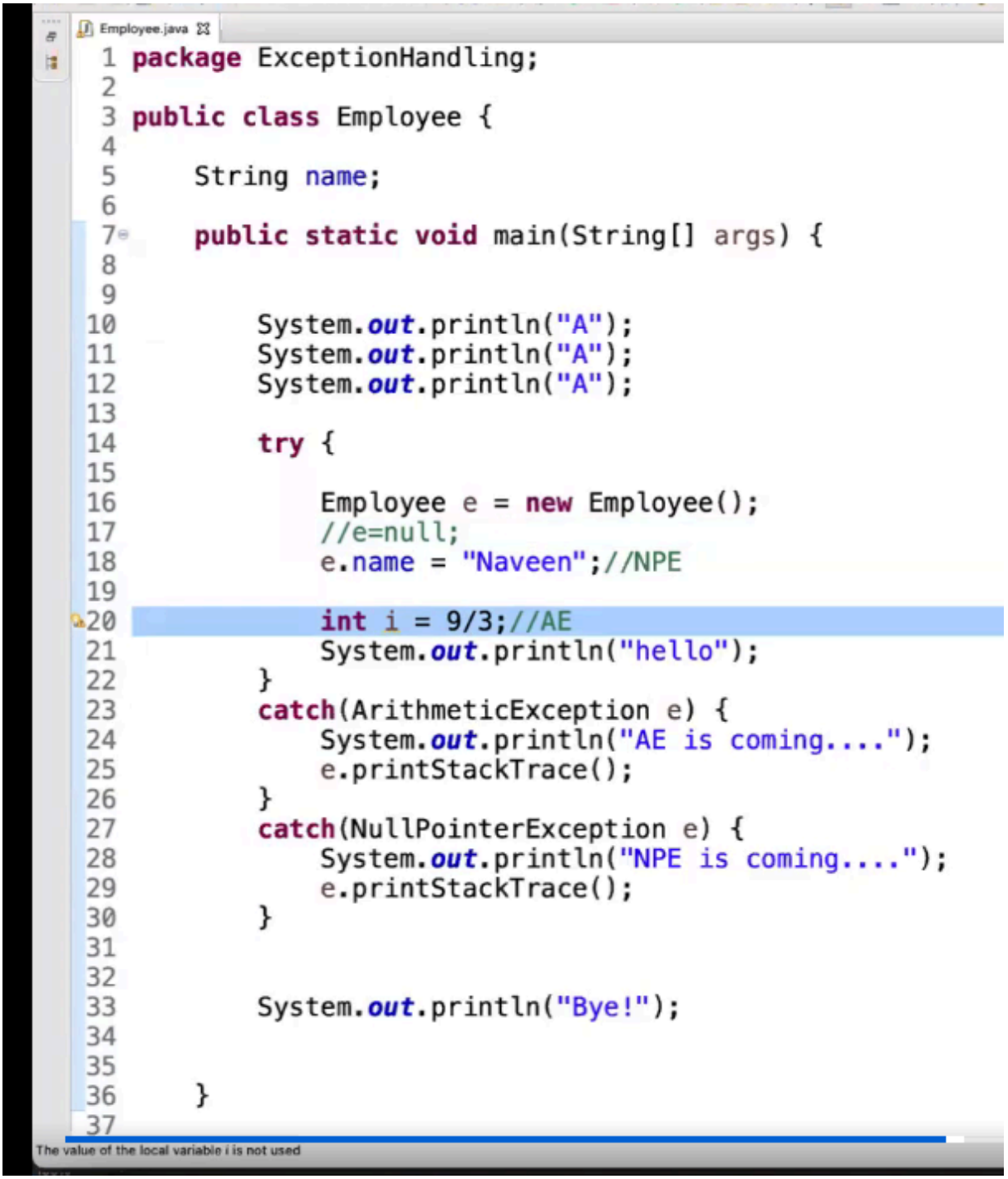
```java
            e.printStackTrace();
        }

        catch (NullPointerException e) {
            System.out.println("null pointer exception handled");
            e.printStackTrace();
        }

        System.out.println("outside blocks");
    }

}

//start of the code
//null pointer exception handled
//java.lang.NullPointerException: Cannot assign field "name" because "e1" is null
//  at com.day22.exception10.main(exception10.java:17)
//outside blocks
```

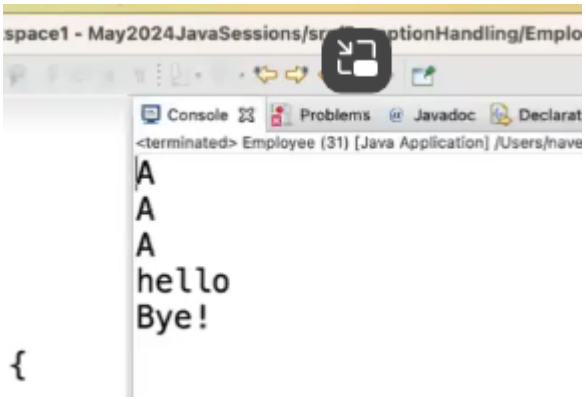
paste exception11-

exception11.java

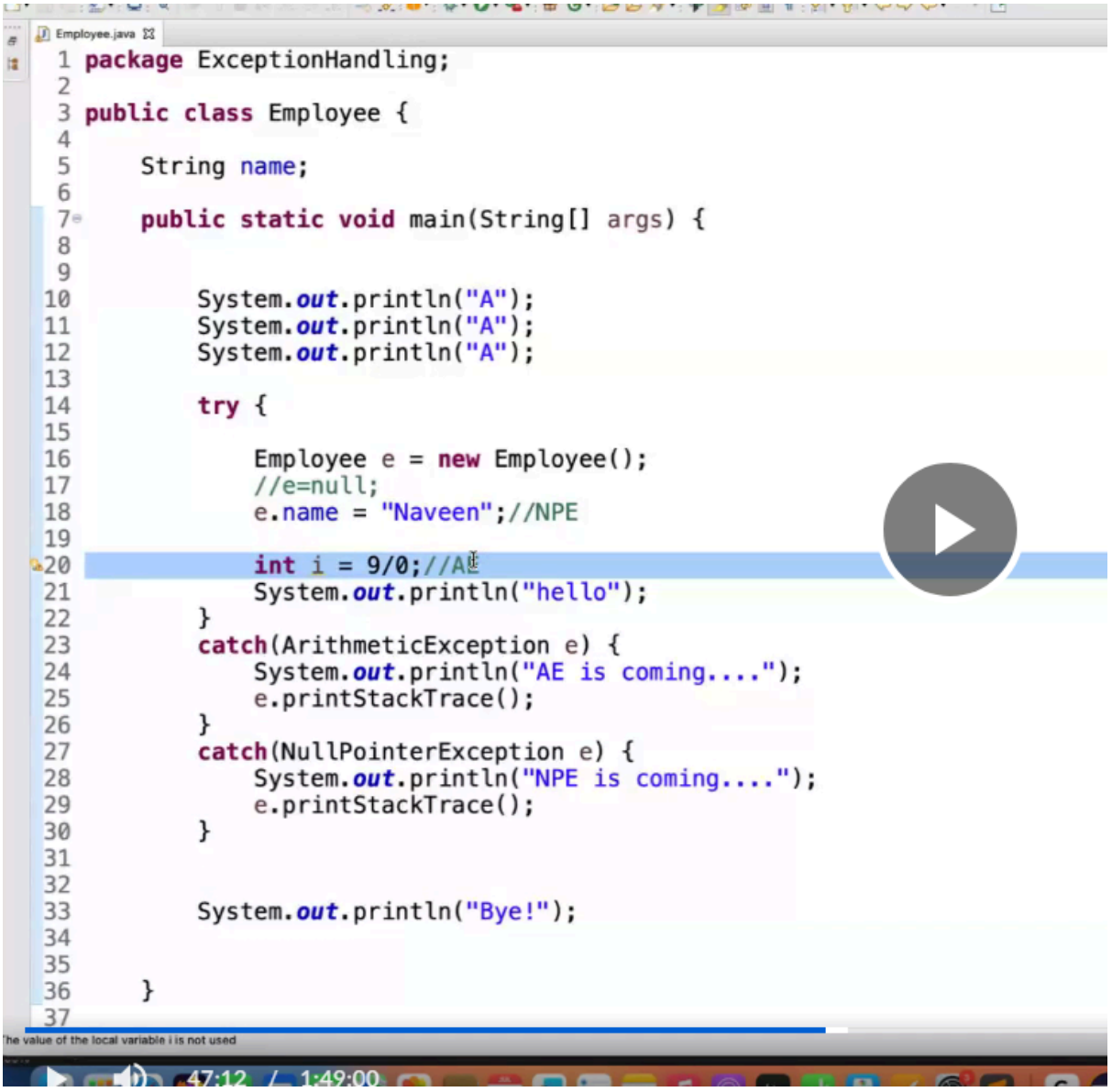
```java
package com.day22;

public class exception11 {

    String name;

    //both exceptions come.
    //have exceptions in different try blocks.

    public static void main(String[] args) {

        System.out.println("start of the code");

        try {

            exception11 e1=new exception11();
            e1=null;
            e1.name=null;
            e1.name="karan";
            System.out.println(e1.name);
        }
        catch (NullPointerException e) {
            System.out.println("null pointer exception handled");
            e.printStackTrace();
        }
        try {
            int i=9/0;
            System.out.println("after exception try block");
        }
        catch (ArithmeticException e) {
            System.out.println("arithmetic exception handled");
            e.printStackTrace();
        }



        System.out.println("outside blocks");
    }

}

//start of the code
//null pointer exception handled
//java.lang.NullPointerException: Cannot assign field "name" because "e1" is null
//  at com.day22.exception11.main(exception11.java:18)
//arithmetic exception handled
//java.lang.ArithmeticException: / by zero
//  at com.day22.exception11.main(exception11.java:27)
//outside blocks
```

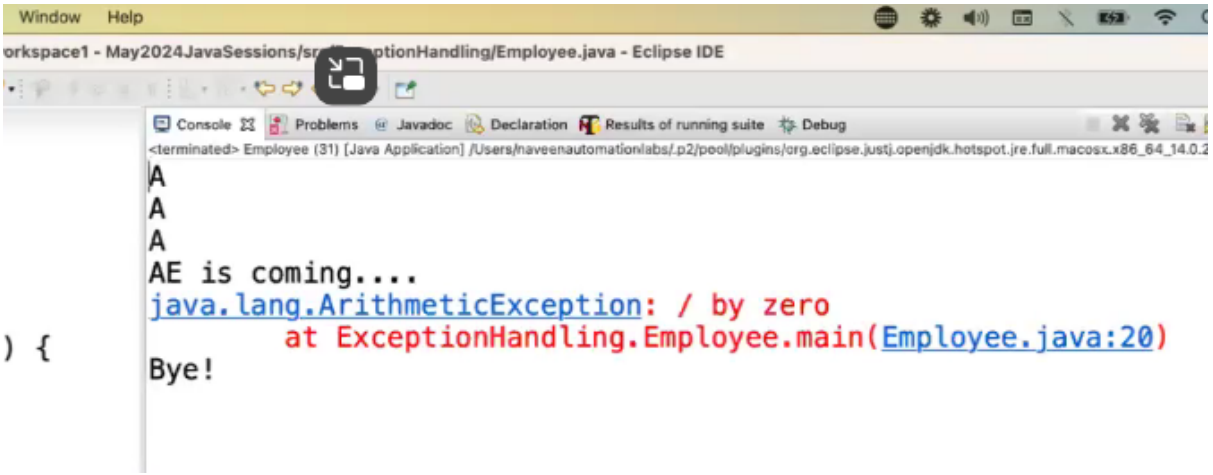
Normal code without exception-


Employee.java
1 package ExceptionHandling;
2
3 public class Employee {
4
5 String name;
6
7 public static void main(String[] args) {
8
9
10 System.out.println("A");
11 System.out.println("A");
12 System.out.println("A");
13
14 try {
15
16 Employee e = new Employee();
17 //e=null;
18 e.name = "Naveen";//NPE
19
20 int i = 9/3;//AE
21 System.out.println("hello");
22 }
23 catch(ArithmeticException e) {
24 System.out.println("AE is coming....");
25 e.printStackTrace();
26 }
27 catch(NullPointerException e) {
28 System.out.println("NPE is coming....");
29 e.printStackTrace();
30 }
31
32
33 System.out.println("Bye!");
34
35
36 }
37
The value of the local variable i is not used



catch arith exception not null -

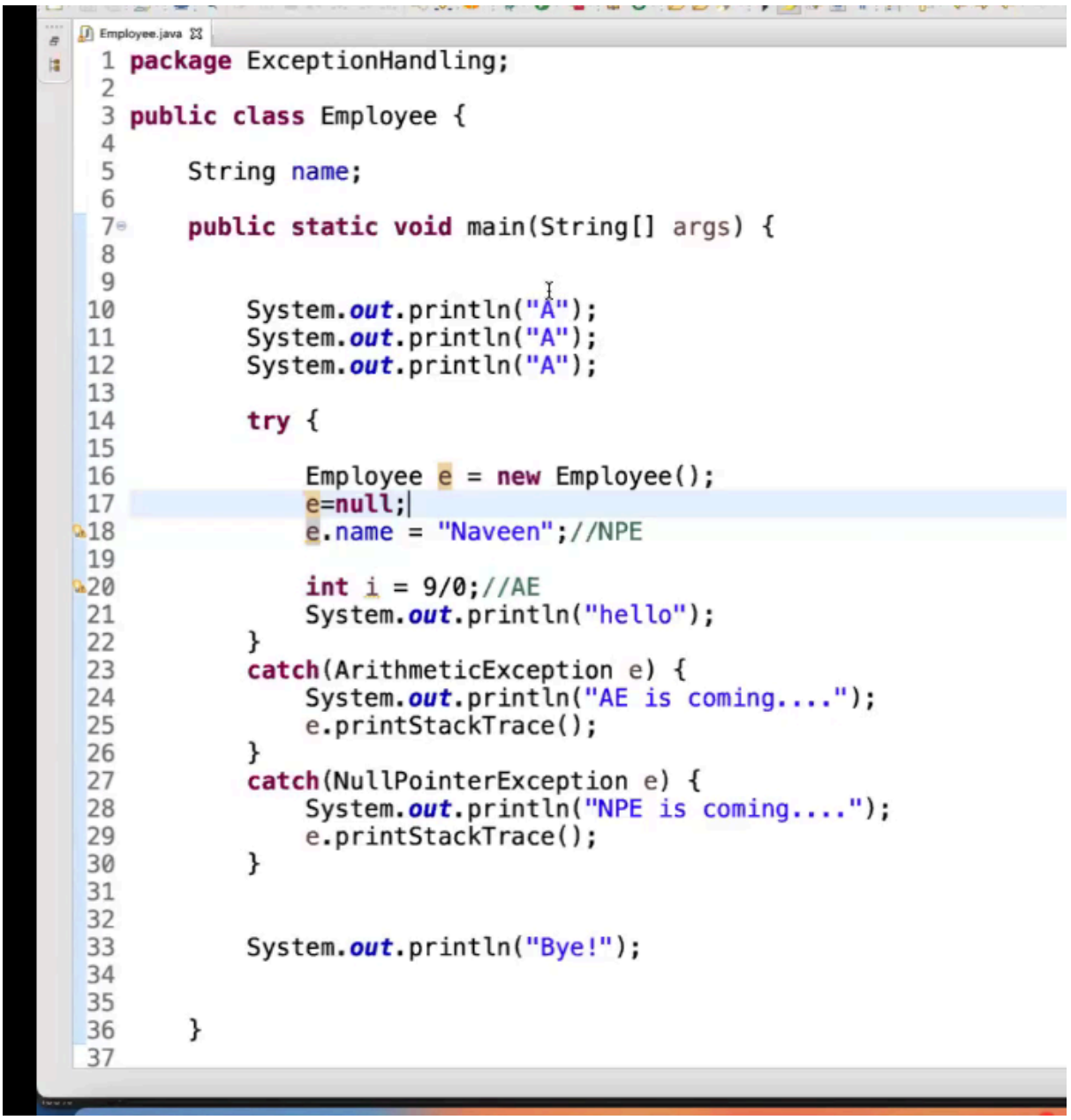


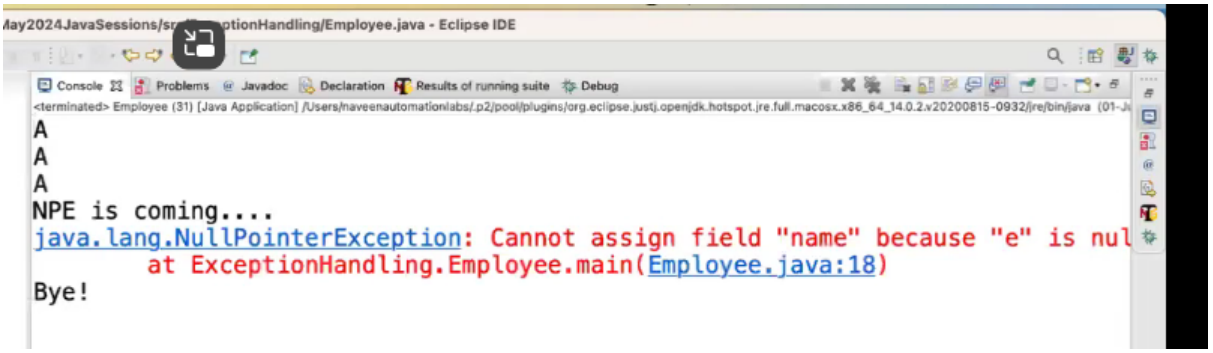


At a time only one exception can be thrown because execution happens line by line and the moment first exception comes all other lines not executed below it. All lines from catch and below catch will run.

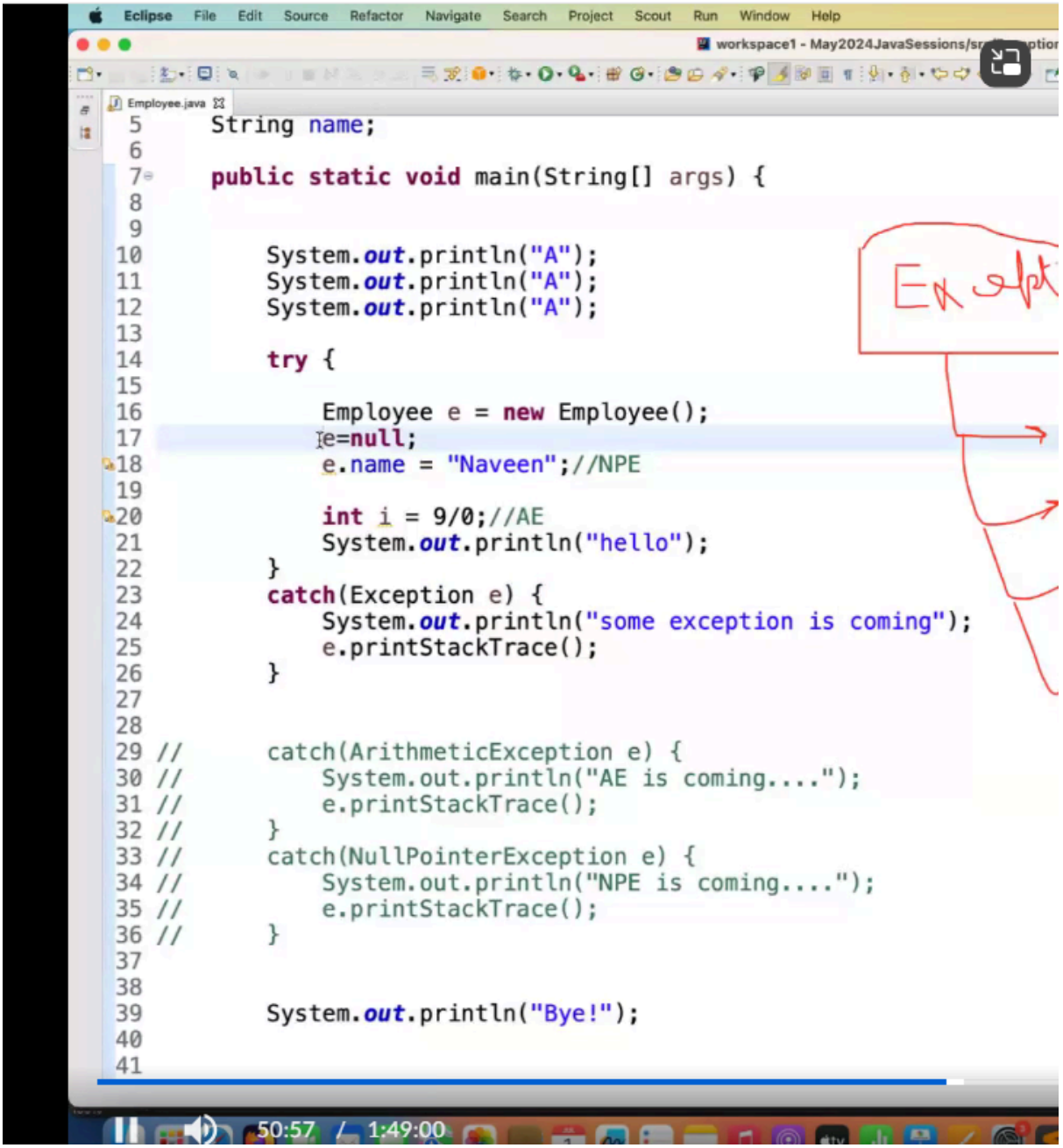
two exceptions at once-

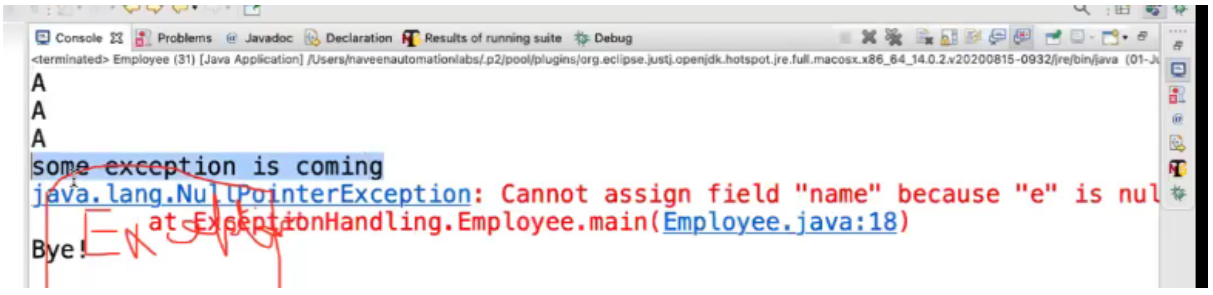
the first one will be caught and program goes ahead.





Parent exception instead of multiple catch blocks-





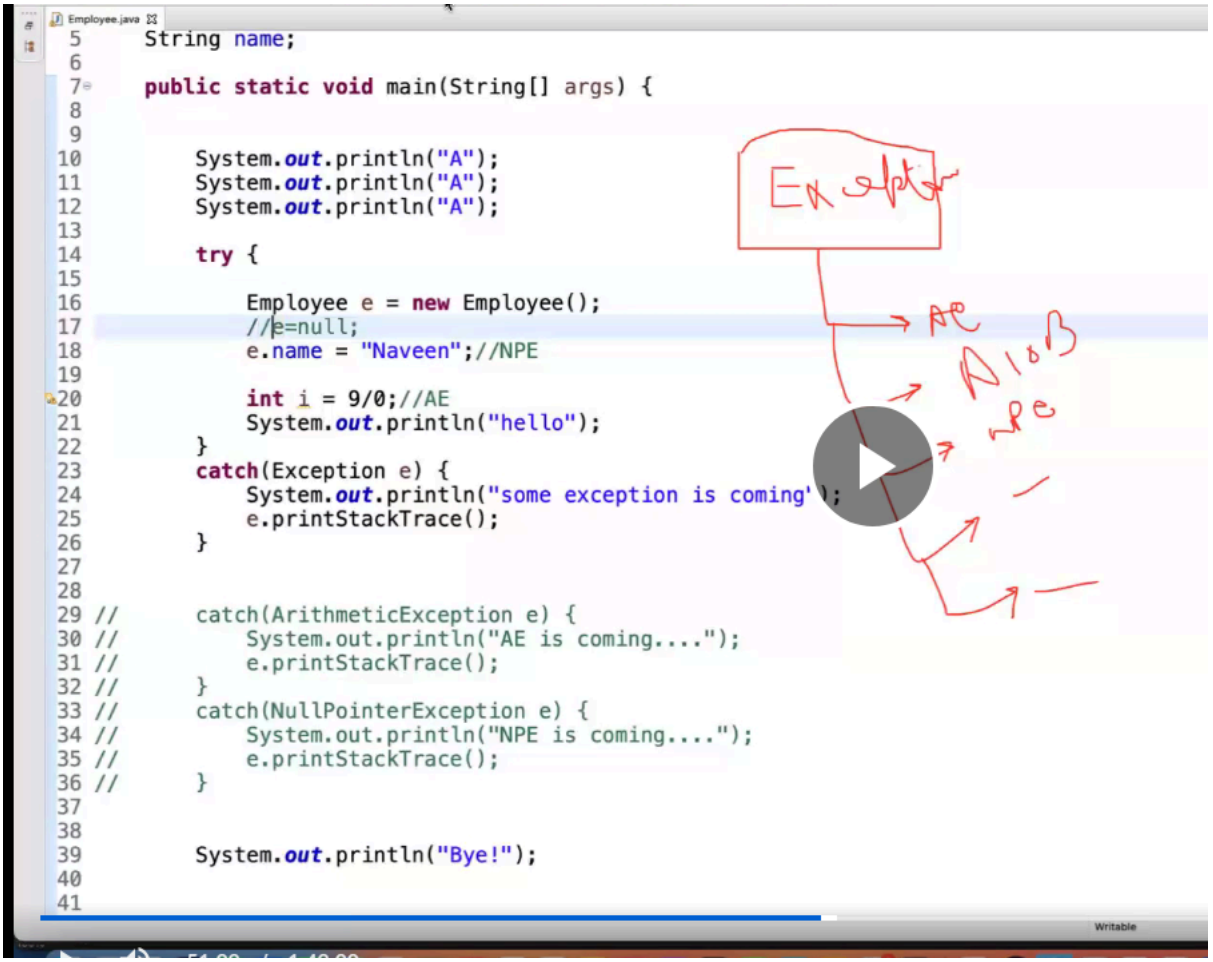
paste exception14-

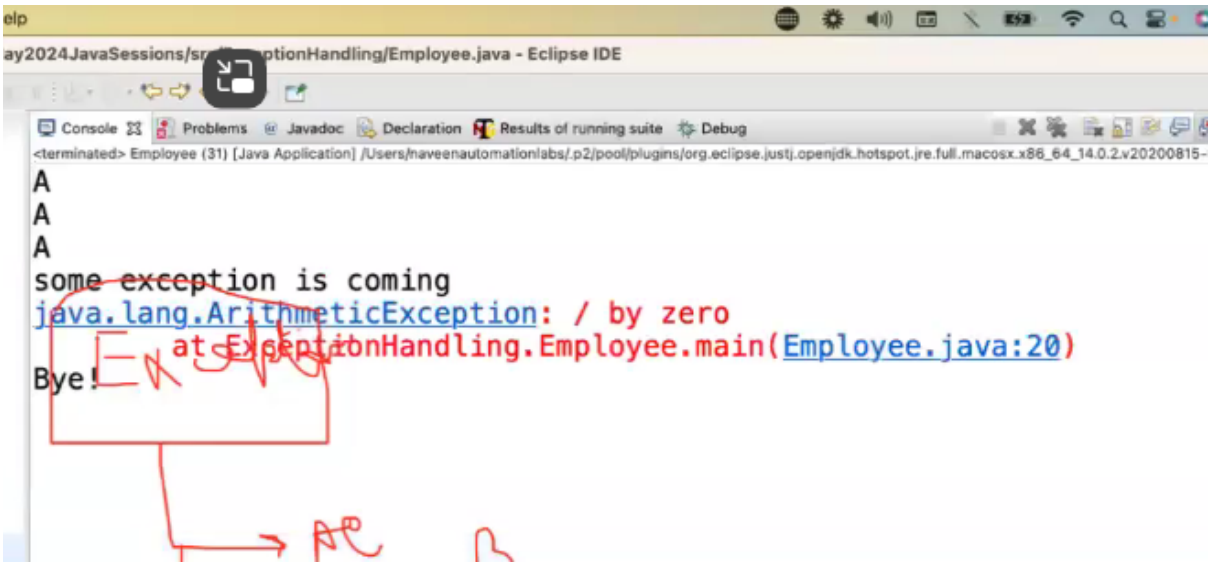
exception14.java

```java
package com.day22;

public class exception14 {

    String name;

    //write master class exceptions for any exceptions.
    //one catch will be enough.

    public static void main(String[] args) {

        System.out.println("start of the code");

        try {

            exception14 e1=new exception14();
            e1=null;
            e1.name=null;
            System.out.println(e1.name);
        }
        catch (Exception e) {
            System.out.println("null pointer exception handled");
            e.printStackTrace();
        }
        try {
            int i=9/0;
            System.out.println("after exception try block");
        }
        catch (Exception e) {
            System.out.println("arithmetic exception handled");
            e.printStackTrace();
        }



        System.out.println("outside blocks");
    }

}
```

```java
//start of the code
//null pointer exception handled
//java.lang.NullPointerException: Cannot assign field "name" because "e1" is null
//  at com.day22.exception14.main(exception14.java:18)
//arithmetic exception handled
//java.lang.ArithmeticException: / by zero
//  at com.day22.exception14.main(exception14.java:26)
//outside blocks
```

Master class is “exception” for all exceptions-





paste exception16-

exception16.java

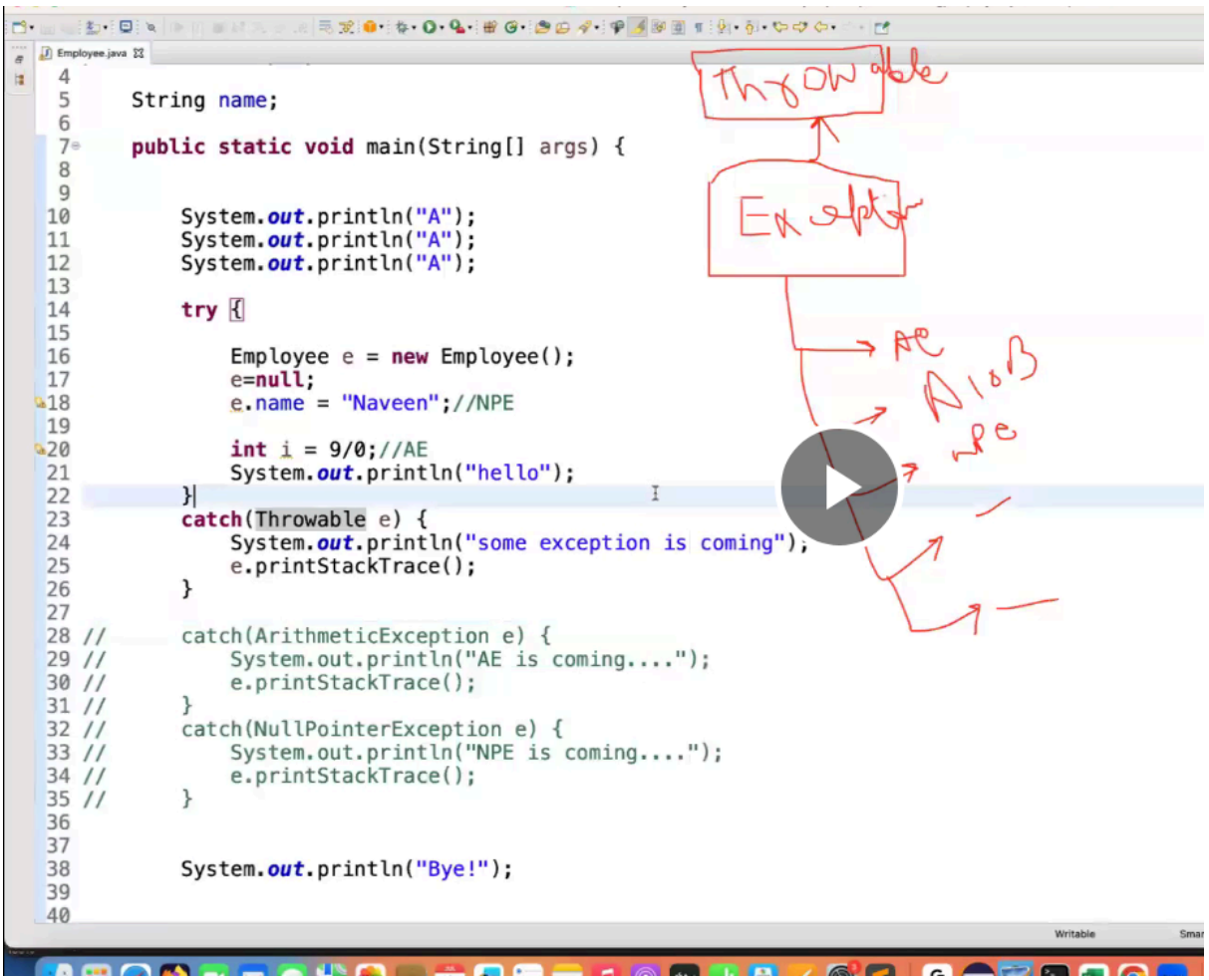
```java
package com.day22;

public class exception16 {

    String name;

    //catch arithmetic using exception.

    public static void main(String[] args) {

        System.out.println("start of the code");

        try {

            exception16 e1=new exception16();
            e1=null;
            e1.name="karan";
            System.out.println(e1.name);
            int i=9/0;
            System.out.println("after exception try block");
        }
        catch (Exception e) {
            System.out.println("Catch block with exception");
            e.printStackTrace();
        }

        System.out.println("outside blocks");
    }

}
```

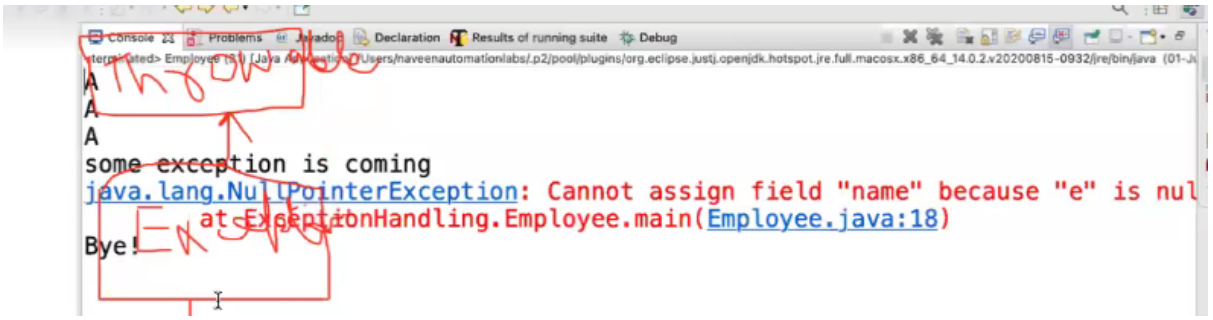
```
}

//start of the code
//Catch block with exception
//java.lang.NullPointerException: Cannot assign field "name" because "e1" is null
//  at com.day22.exception16.main(exception16.java:17)
//outside blocks
```

Best practice is multiple catch blocks.

Throwable is super class of exception.

Handling using throwable-



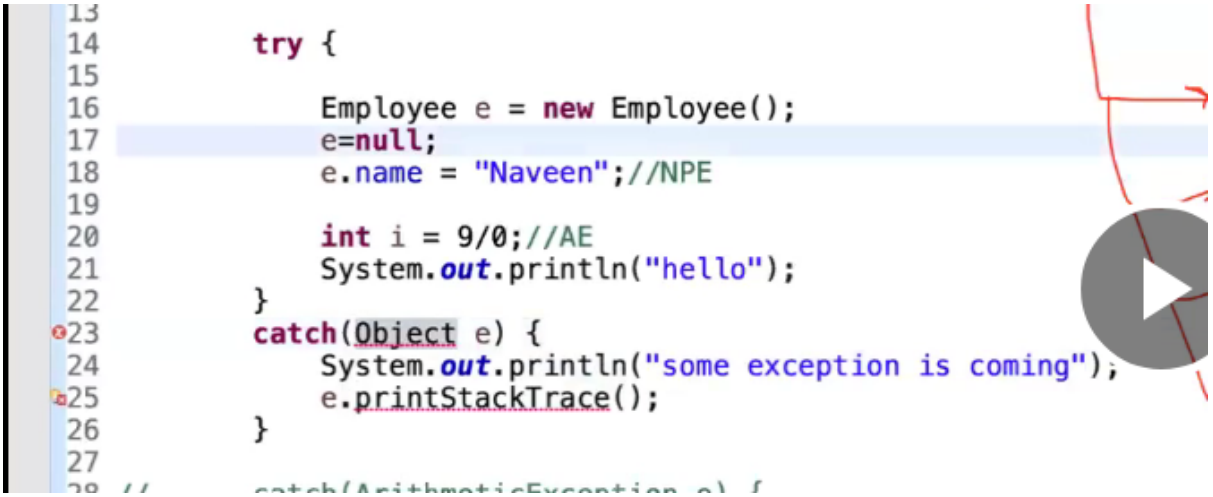


Object is super class of all class including throwable.

It will throw error when we use object for catching exceptions-

//No exception of type Object can be thrown;

//an exception type must be a subclass of Throwable



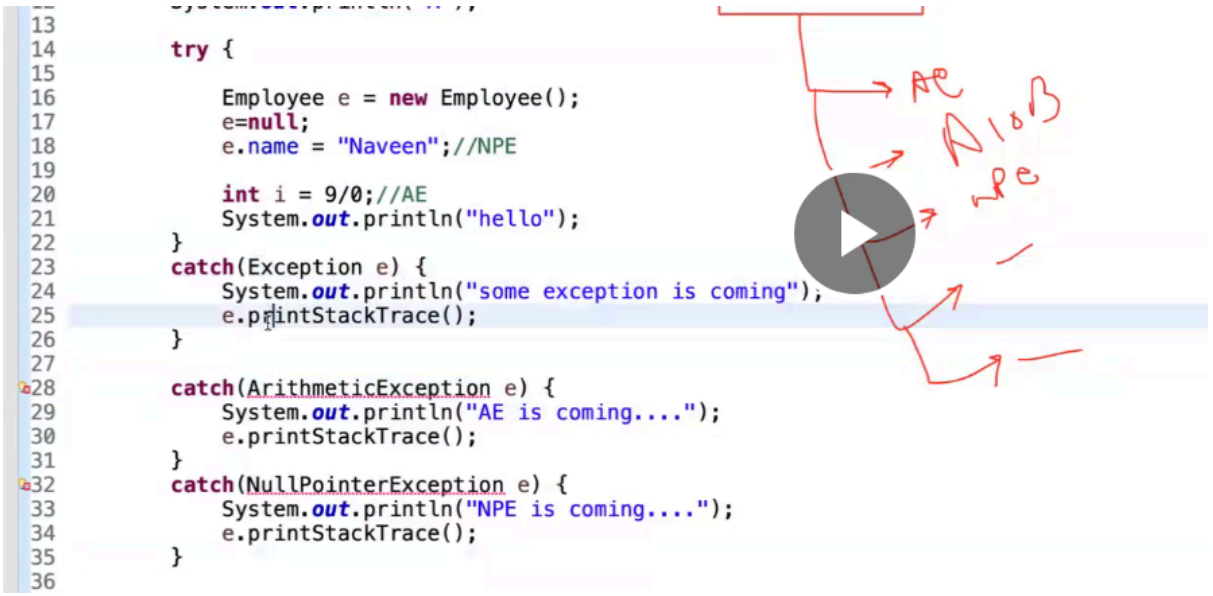
```
13
14          try {
15
16              Employee e = new Employee();
17              e=null;
18              e.name = "Naveen";//NPE
19
20              int i = 9/0;//AE
21              System.out.println("hello");
22          }
23          catch(Object e) {
24              System.out.println("some exception is coming");
25              e.printStackTrace();
26          }
27
```

Only throwable or lower than that is allowed.

Cant add the exception above and child exceptions below-

Top one already handling it.

//Unreachable catch block for ArithmeticException.
//It is already handled by the catch block for Exception

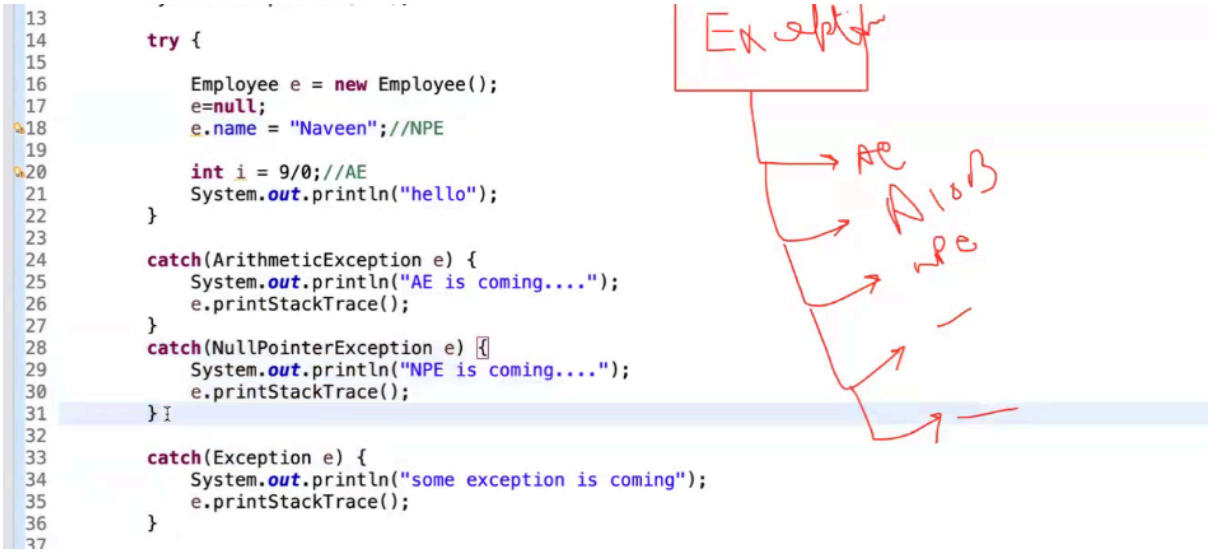


```
13
14      try {
15
16          Employee e = new Employee();
17          e=null;
18          e.name = "Naveen";//NPE
19
20          int i = 9/0;//AE
21          System.out.println("hello");
22      }
23      catch(Exception e) {
24          System.out.println("some exception is coming");
25          e.printStackTrace();
26      }
27
28      catch(ArithmeticException e) {
29          System.out.println("AE is coming....");
30          e.printStackTrace();
31      }
32      catch(NullPointerException e) {
33          System.out.println("NPE is coming....");
34          e.printStackTrace();
35      }
36
```

This one is ok-

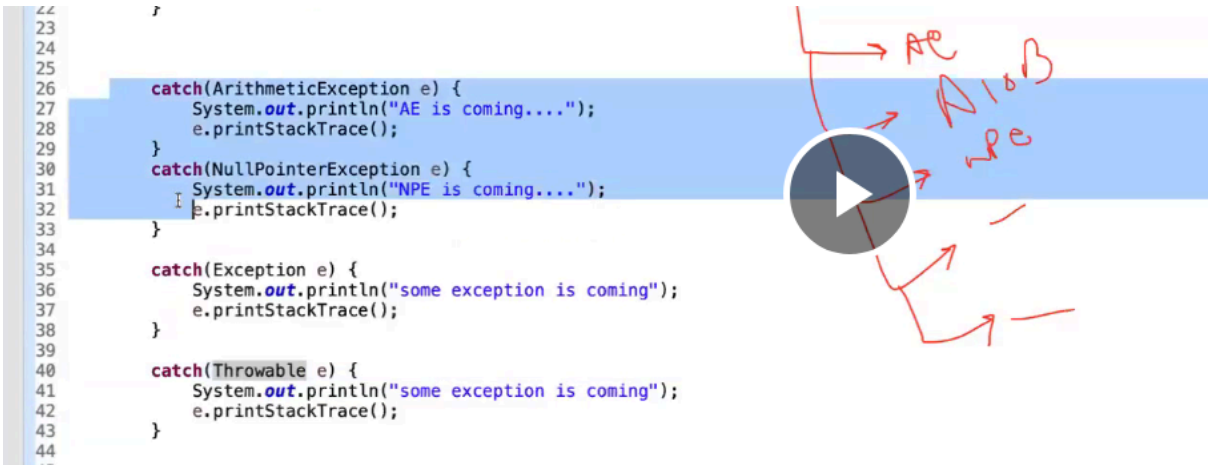
Line by line code runs.

Hence lower one can be at top.



```
13
14      try {
15
16          Employee e = new Employee();
17          e=null;
18          e.name = "Naveen";//NPE
19
20          int i = 9/0;//AE
21          System.out.println("hello");
22      }
23
24      catch(ArithmeticException e) {
25          System.out.println("AE is coming....");
26          e.printStackTrace();
27      }
28      catch(NullPointerException e) {
29          System.out.println("NPE is coming....");
30          e.printStackTrace();
31      }
32
33      catch(Exception e) {
34          System.out.println("some exception is coming");
35          e.printStackTrace();
36      }
37
```

This is also ok-




```
22
23
24
25
26      catch(ArithmeticException e) {
27          System.out.println("AE is coming....");
28          e.printStackTrace();
29      }
30      catch(NullPointerException e) {
31          System.out.println("NPE is coming....");
32          e.printStackTrace();
33      }
34
35      catch(Exception e) {
36          System.out.println("some exception is coming");
37          e.printStackTrace();
38      }
39
40      catch(Throwable e) {
41          System.out.println("some exception is coming");
42          e.printStackTrace();
43      }
44
```

paste exception22-

exception22.java

```
1  package com.day22;
2
3  public class exception22 {
4
5      String name;
6
7      //write higher exception at top and lower at bottom.
8      //not allowed.
9
10     public static void main(String[] args) {
11
12         System.out.println("start of the code");
13
14         try {
15
16             exception22 e1=new exception22();
17             e1=null;
18             e1.name=null;
19             System.out.println(e1.name);
20             int i=9/0;
21             System.out.println("after exception try block");
22         }
23
24         catch (Throwable e) {
25             System.out.println("Catch block with exception");
26             e.printStackTrace();
27         }
```

```
26              e.printStackTrace();
27          }
28
29          //Unreachable catch block for ArithmeticException.
30          //It is already handled by the catch block for Throwable
31          catch (ArithmeticException e) {
32              System.out.println("Catch block with arithmetic exception");
33              e.printStackTrace();
34          }
35
36          //Unreachable catch block for NullPointerException.
37          //It is already handled by the catch block for Throwable
38          catch (NullPointerException e) {
39              System.out.println("Catch block with  NullPointerException exception");
40              e.printStackTrace();
41          }
42
43
44
45          System.out.println("outside blocks");
46      }
```

```
45          System.out.println("outside blocks");
46      }
47
48  }
49
50  //start of the code
51  //Catch block with  NullPointerException exception
52  //java.lang.NullPointerException: Cannot assign field "name" because "e1" is null
53  //  at com.day22.exception21.main(exception21.java:18)
54  //outside blocks
55
```

paste exception23-

exception23.java

```java
package com.day22;

public class exception23 {

    String name;

    //write higher exception at top and lower at bottom.
    //not allowed.

    public static void main(String[] args) {

        System.out.println("start of the code");

        try {

            exception23 e1=new exception23();
            e1=null;
            e1.name=null;
            System.out.println(e1.name);
            int i=9/0;
            System.out.println("after exception try block");
        }

        catch (Throwable e) {
            System.out.println("Catch block with exception");
            e.printStackTrace();
        }
```

```java
            e.printStackTrace();
        }

        //error when lower exception is present.
        //Unreachable catch block for Exception.
        //It is already handled by the catch block for Throwable
        catch (Exception e) {
            System.out.println("Catch block with exception");
            e.printStackTrace();
        }

        //Unreachable catch block for ArithmeticException.
        //It is already handled by the catch block for Throwable
        catch (ArithmeticException e) {
            System.out.println("Catch block with arithmetic exception");
            e.printStackTrace();
        }

        //Unreachable catch block for NullPointerException.
        //It is already handled by the catch block for Throwable
        catch (NullPointerException e) {
            System.out.println("Catch block with  NullPointerException exception");
            e.printStackTrace();
        }

    }
```

```
        }


        System.out.println("outside blocks");
    }

}

//start of the code
//Catch block with  NullPointerException exception
//java.lang.NullPointerException: Cannot assign field "name" because "e1" is null
//  at com.day22.exception21.main(exception21.java:18)
//outside blocks
```

paste exception24-

exception24.java

```
package com.day22;

public class exception24 {

    String name;

    //write exception at top.
    //then write throwable.
    //its fine.

    public static void main(String[] args) {

        System.out.println("start of the code");

        try {

            exception24 e1=new exception24();
            e1=null;
            e1.name=null;
            System.out.println(e1.name);
            int i=9/0;
            System.out.println("after exception try block");
        }

        catch (Exception e) {
            System.out.println("Catch block with exception");
            e.printStackTrace();
        }
```

```
27              e.printStackTrace();
28          }
29
30          catch (Throwable e) {
31              System.out.println("Catch block with exception");
32              e.printStackTrace();
33          }
34
35          //Unreachable catch block for ArithmeticException.
36          //It is already handled by the catch block for Throwable
37          catch (ArithmeticException e) {
38              System.out.println("Catch block with arithmetic exception");
39              e.printStackTrace();
40          }
41
42          //Unreachable catch block for NullPointerException.
43          //It is already handled by the catch block for Throwable
44          catch (NullPointerException e) {
45              System.out.println("Catch block with  NullPointerException exception");
46              e.printStackTrace();
47          }
```

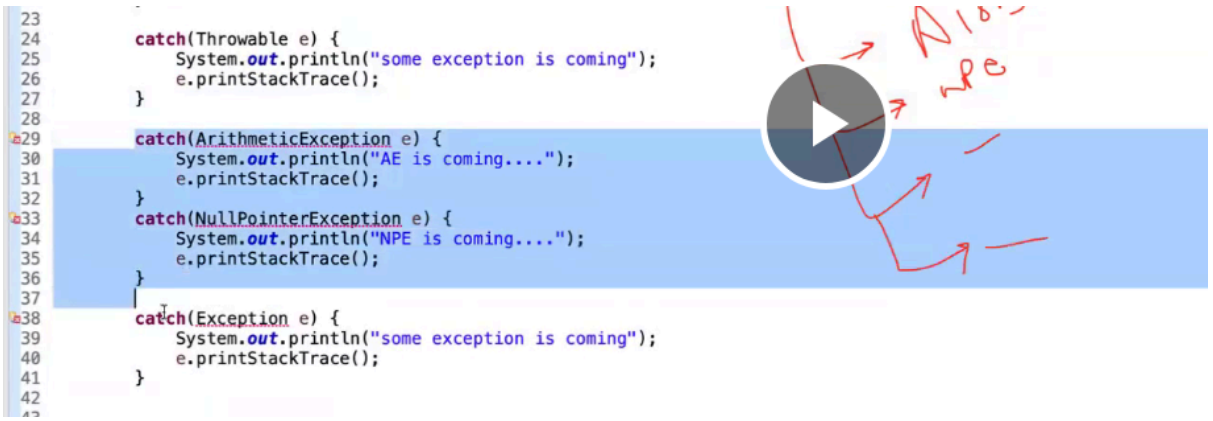
```
46              e.printStackTrace();
47          }
48
49
50
51          System.out.println("outside blocks");
52      }
53
54  }
55
56  //start of the code
57  //Catch block with  NullPointerException exception
58  //java.lang.NullPointerException: Cannot assign field "name" because "e1" is null
59  //  at com.day22.exception21.main(exception21.java:18)
60  //outside blocks
61
```

paste exception25-

exception25.java

```java
package com.day22;

public class exception25 {

    String name;

    //write exception at top.
    //then write throwable.
    //its fine.

    public static void main(String[] args) {

        System.out.println("start of the code");

        try {

            exception25 e1=new exception25();
            e1=null;
            e1.name=null;
            System.out.println(e1.name);
            int i=9/0;
            System.out.println("after exception try block");
        }

        catch (Exception e) {
            System.out.println("Catch block with exception");
            e.printStackTrace();
        }
            e.printStackTrace();
        }

        catch (Throwable e) {
            System.out.println("Catch block with exception");
            e.printStackTrace();
        }

        System.out.println("outside blocks");
    }

}

//start of the code
//Catch block with exception
//java.lang.NullPointerException: Cannot assign field "name" because "e1" is null
//  at com.day22.exception25.main(exception25.java:19)
//outside blocks
```

This is not allowed-

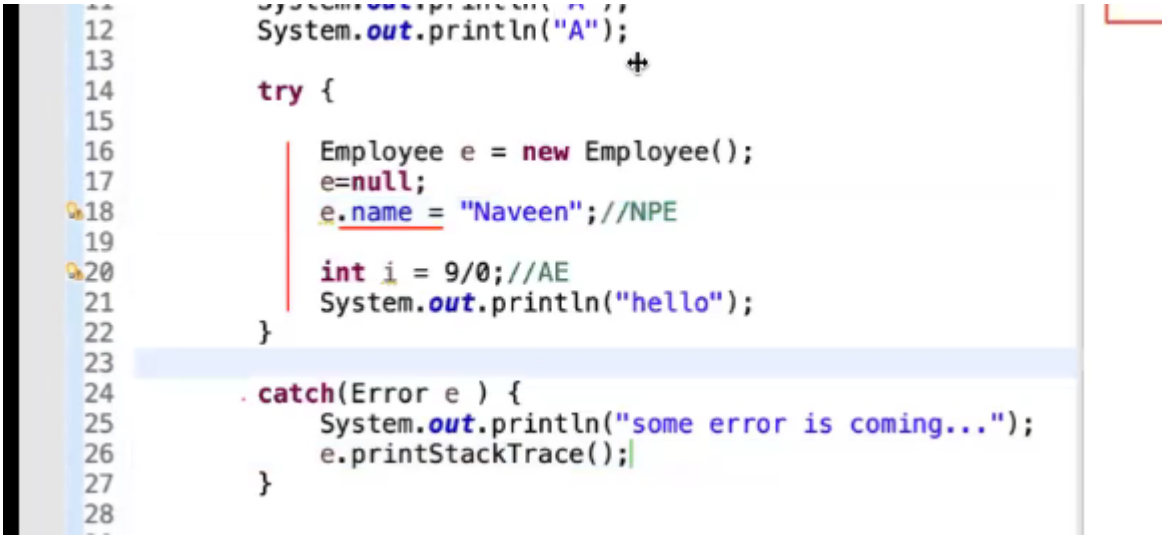


```
23
24        catch(Throwable e) {
25            System.out.println("some exception is coming");
26            e.printStackTrace();
27        }
28
29        catch(ArithmeticException e) {
30            System.out.println("AE is coming....");
31            e.printStackTrace();
32        }
33        catch(NullPointerException e) {
34            System.out.println("NPE is coming....");
35            e.printStackTrace();
36        }
37
38        catch(Exception e) {
39            System.out.println("some exception is coming");
40            e.printStackTrace();
41        }
42
```

Error-

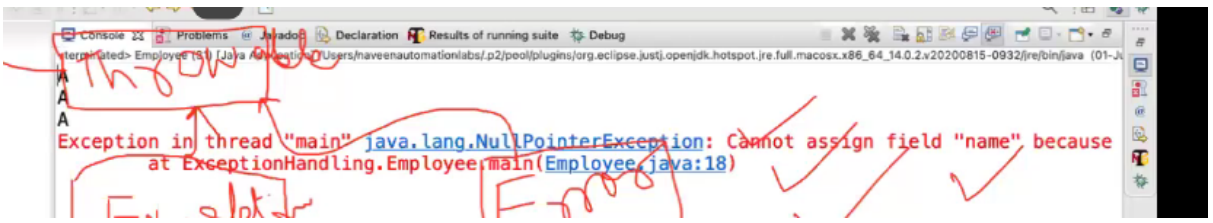
Child of throwable.

Try handling exception with error-



```
12          System.out.println("A");
13
14          try {
15
16              Employee e = new Employee();
17              e=null;
18              e.name = "Naveen";//NPE
19
20              int i = 9/0;//AE
21              System.out.println("hello");
22          }
23
24          catch(Error e ) {
25              System.out.println("some error is coming...");
26              e.printStackTrace();
27          }
28
```

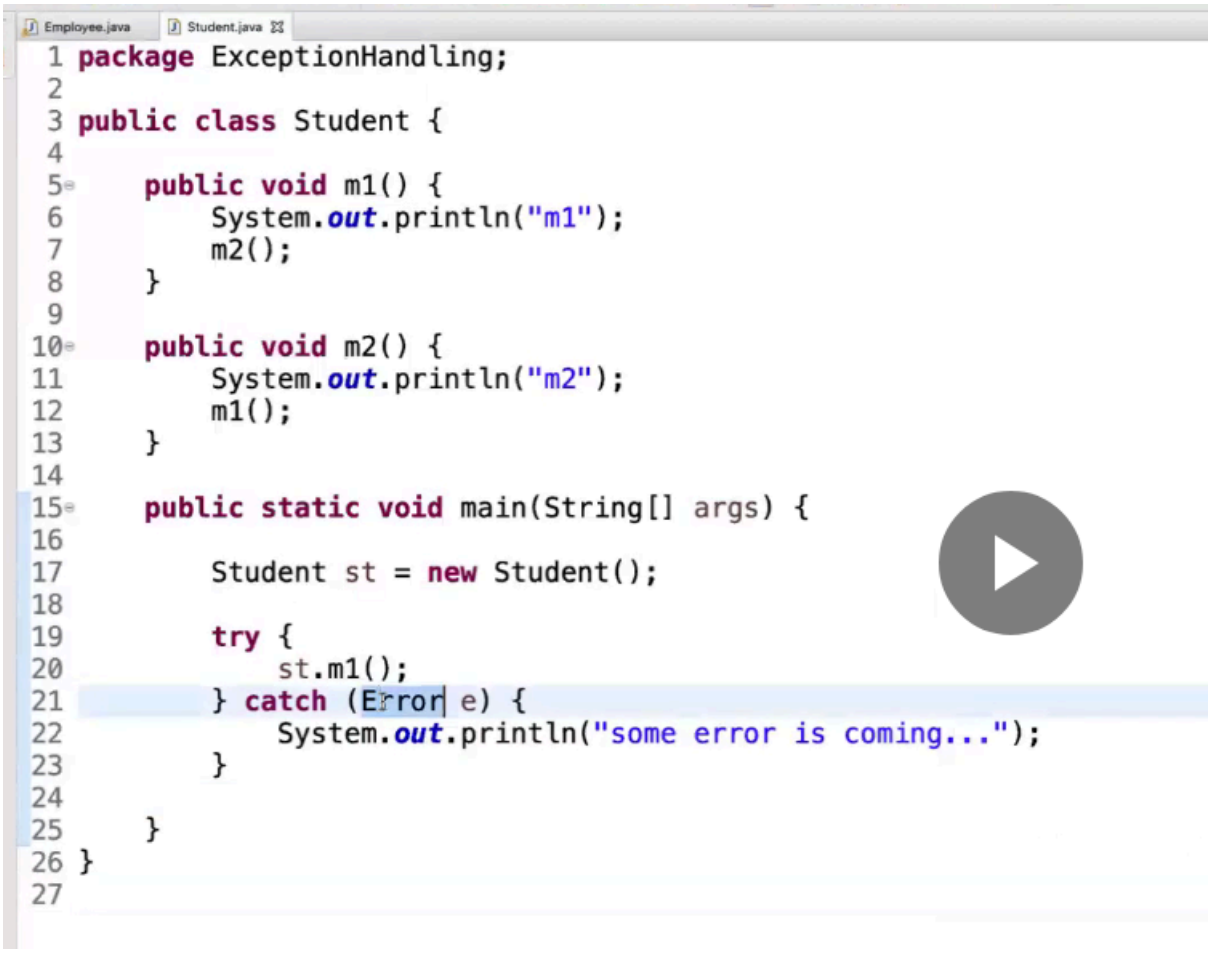
Cannot be handled.

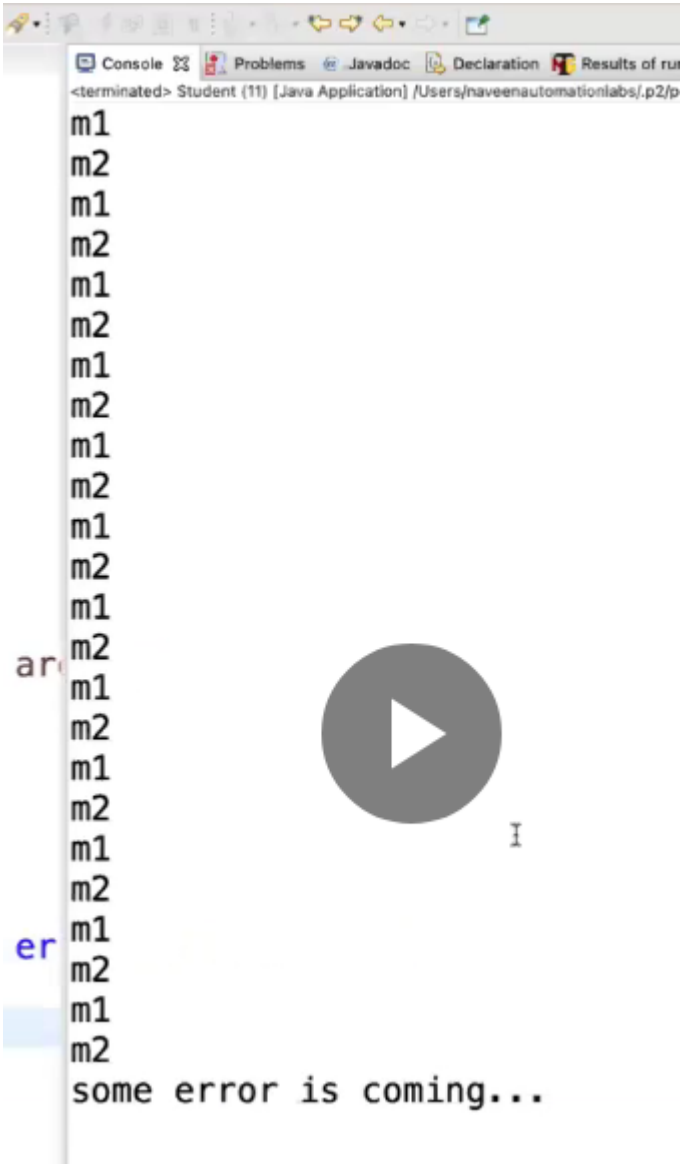


```
Console  Problems  Javadoc  Declaration  Results of running suite  Debug
<terminated> Employee (8) [Java Application] /Users/naveenautomationlabs/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86_64_14.0.2.v20200815-0932/jre/bin/java (01-J
A
A
A
Exception in thread "main" java.lang.NullPointerException: Cannot assign field "name" because
        at ExceptionHandling.Employee.main(Employee.java:18)
```

Errors can be handled with try catch but difficult to catch them.

Handled stack exception-





paste exception27-

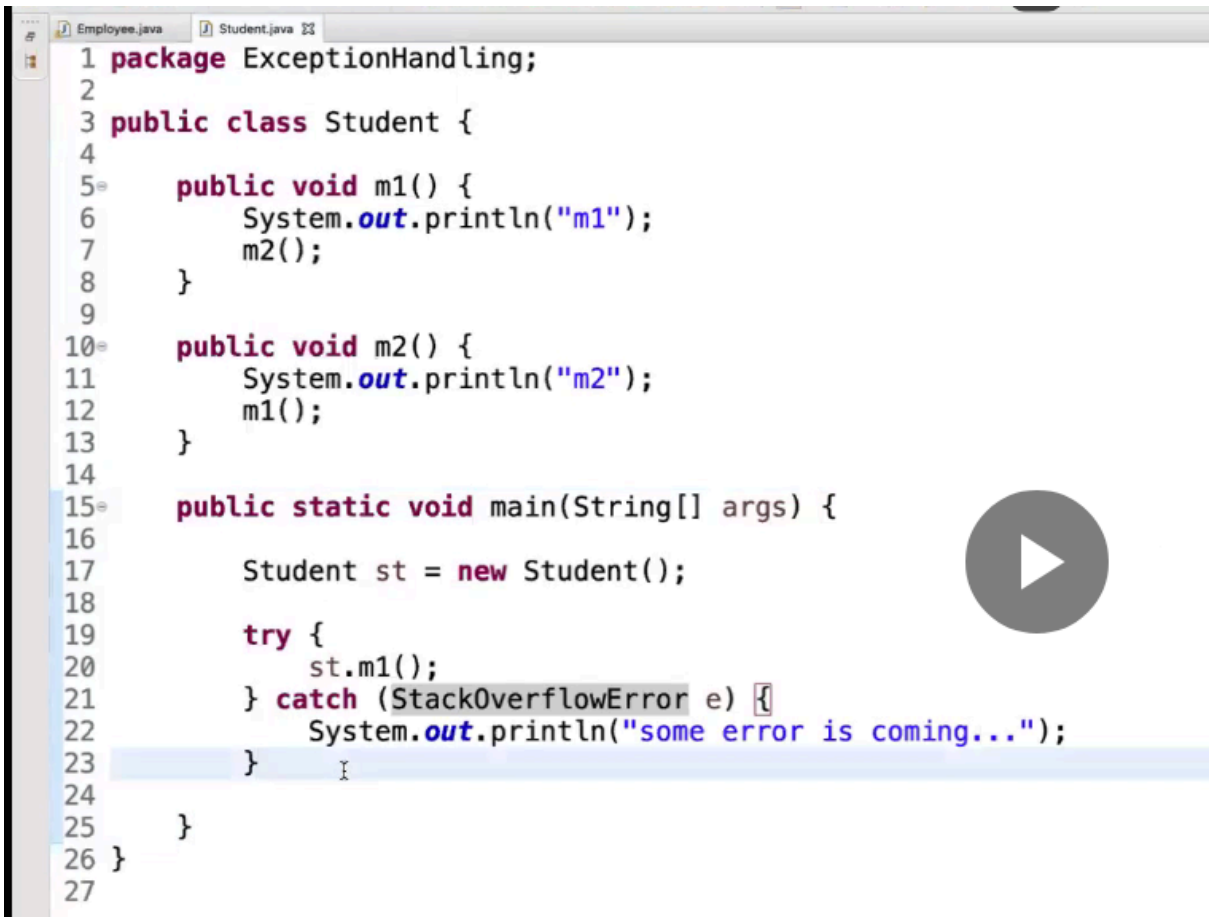
exception27.java

```java
package com.day22;

public class exception27 {

    //unreachable block in method m2.

    String name;

    public void m1() {
        System.out.println("m1");
        int i1=m2(name);
        System.out.println(i1);
    }

    public int m2(String name1) {
        System.out.println("m2 method " + name1);
        return 10;
        m1();//Unreachable code
    }

    public static void main(String[] args) {

        System.out.println("start of the code");
```

```java
        System.out.println("start of the code");

        try {

            exception27 e1=new exception27();
            e1=null;
            e1.name=null;
            System.out.println(e1.name);
            int i=9/0;
            System.out.println("after exception try block");
        }

        catch (Error e) {
            System.out.println("Catch block with exception");
            e.printStackTrace();
        }

        System.out.println("outside blocks");
    }

}
```

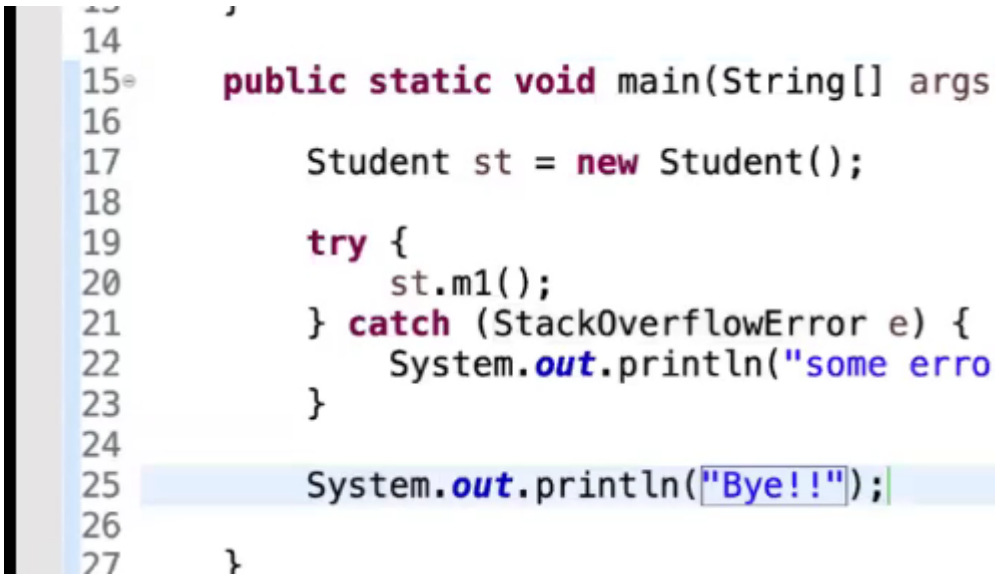
//tried catching error but not easy to debug.

Write exact error name-



```java
package ExceptionHandling;

public class Student {

    public void m1() {
        System.out.println("m1");
        m2();
    }

    public void m2() {
        System.out.println("m2");
        m1();
    }

    public static void main(String[] args) {

        Student st = new Student();

        try {
            st.m1();
        } catch (StackOverflowError e) {
            System.out.println("some error is coming...");
        }

    }
}
```

Print bye-

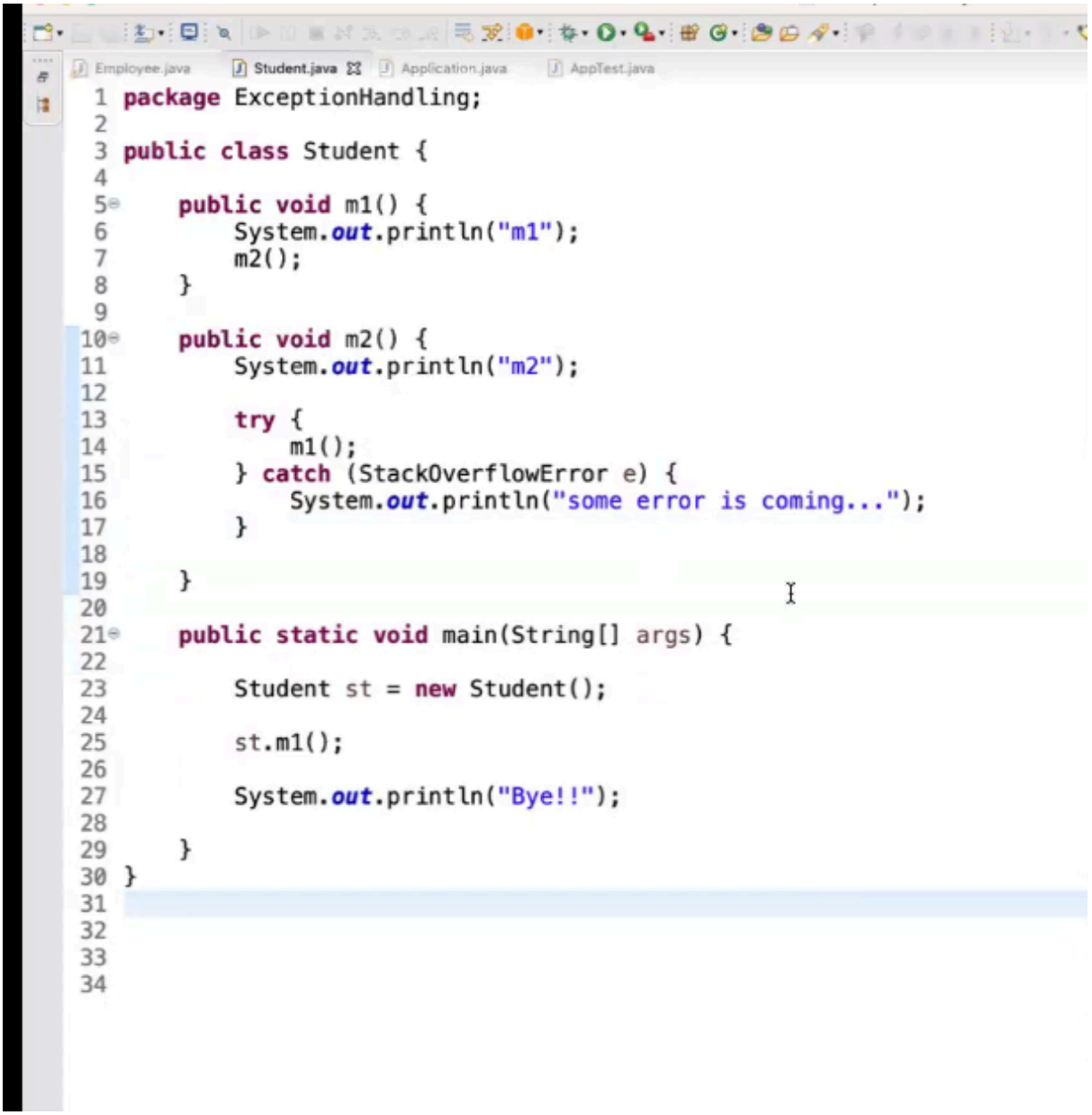


```java
    public static void main(String[] args

        Student st = new Student();

        try {
            st.m1();
        } catch (StackOverflowError e) {
            System.out.println("some erro
        }

        System.out.println("Bye!!");

    }
```

Stack will have multiple execution blocks-

One block full then stack error and then second block full will have another stack error and so on.

Can handle exceptions and errors anywhere where you like-


Employee.java
Student.java
Application.java
AppTest.java
1 package ExceptionHandling;
2
3 public class Student {
4
5 public void m1() {
6 System.out.println("m1");
7 m2();
8 }
9
10 public void m2() {
11 System.out.println("m2");
12
13 try {
14 m1();
15 } catch (StackOverflowError e) {
16 System.out.println("some error is coming...");
17 }
18
19 }
20
21 public static void main(String[] args) {
22
23 Student st = new Student();
24
25 st.m1();
26
27 System.out.println("Bye!!");
28
29 }
30 }
31
32
33
34

It keeps throwing error until stack is overflowing.

paste exception30-

exception30.java

```java
package com.day22;

public class exception30 {

    /**
     * Can handle exceptions and errors anywhere where you like
     * in the code.
     *it keeps going in circles and finally nothing after catch block runs
     *in this case.
     */

    public void m1() {
        System.out.println("m1");
        m2();
    }

    public void m2() {

        System.out.println("m2");
        try {
            m1();
        }
        catch (StackOverflowError e) {
            System.out.println("stack overflow inside method");
            e.printStackTrace();
        }
        //it throws this error for me.
        //just written for reference.
//      catch (NoClassDefFoundError e) {
//          System.out.println("no class def found overflow inside method");
//          e.printStackTrace();
//      }

        System.out.println("m2 method");
    }
```

```
34          System.out.println("m2 method");
35      }
36
37      public static void main(String[] args) {
38
39          System.out.println("start of the code");
40
41          exception30 e1=new exception30();
42
43          e1.m1();
44
45          System.out.println("outside blocks");
46      }
47
48  }
49
```

```
48 }
49
50
51 //start of the code
52 //m1
53 //m2
54 //m1
55 //m2
56 //m1
57 //m2
58 //m1
59 //m2
60 //m1
61 //m2
62 //m1
63 //m2
64 //m1
65 //m2
66 //m1
67 //m2
68 //stack overflow inside method
69 //stack overflow inside method
70 //java.lang.StackOverflowError
```

```
//stack overflow inside method
//java.lang.StackOverflowError
//java.lang.StackOverflowError
//stack overflow inside method
//java.lang.StackOverflowError
//stack overflow inside method
//java.lang.StackOverflowError
//stack overflow inside method
//java.lang.StackOverflowError
//stack overflow inside method
//java.lang.StackOverflowError
//stack overflow inside method
//java.lang.StackOverflowError
//stack overflow inside method
//java.lang.StackOverflowError
//stack overflow inside method
//java.lang.StackOverflowError
//stack overflow inside method
//java.lang.StackOverflowError
```

```
//stack overflow inside method
//java.lang.StackOverflowError
//Exception in thread "main" java.lang.NoClassDefFoundError:
//Could not initialize class java.lang.StackTraceElement$HashedModules
//
//Exception: java.lang.NoClassDefFoundError thrown from
//the UncaughtExceptionHandler in thread "main"
```