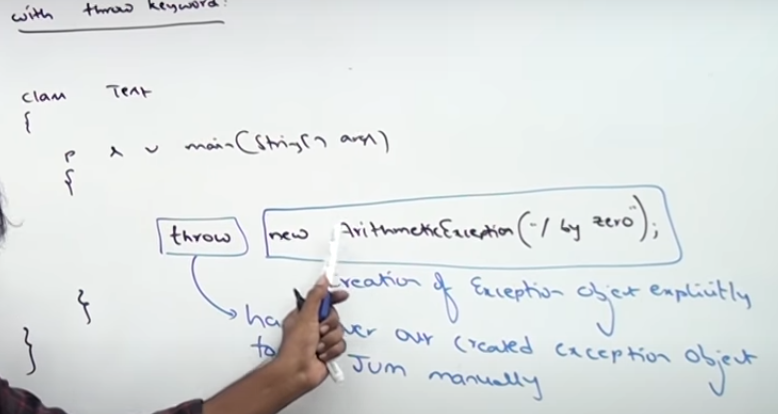
The usage of throw keyword:

To create the customized our own exception.

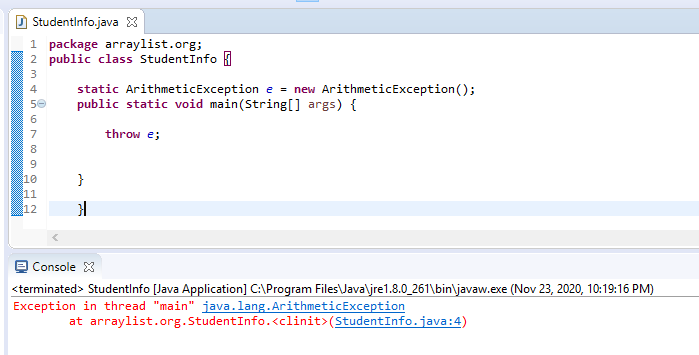
For example, I have to take Rs.10000 from my account but my available balance is Rs.9999 hence unable to take the amount and stating that insufficient fund.

**\*\*To handover our created exception object to the JVM Manually.**



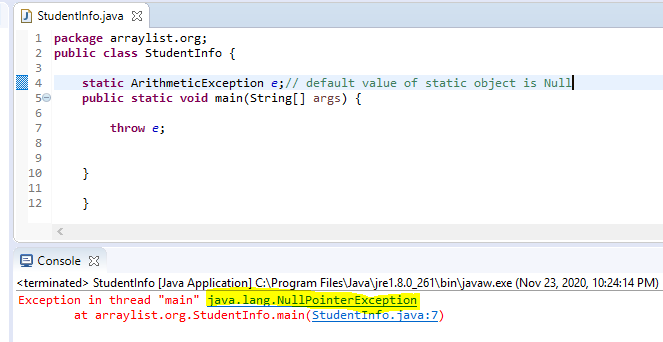
**Important cases related to throw keyword:**

**Case 1: We can create the exception object by using new keyword explicitly**

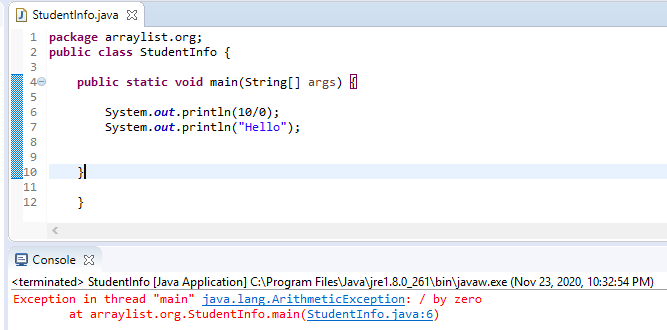


**Case 2: For default static value of exception object is Null. If we didn’t create the exception object by using new keyword mean it will show as Null pointer exception.**

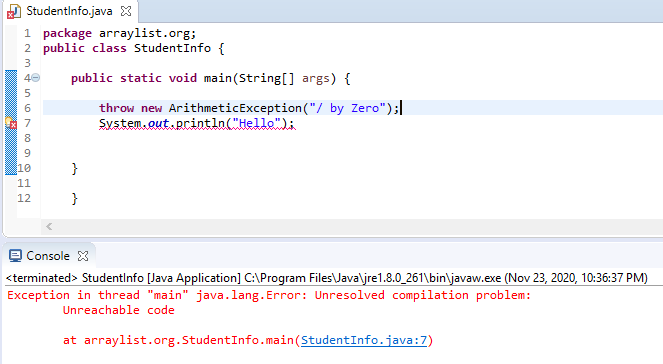
**Here, we didn’t assign any value to exception object.**



**Case 3: The below program will get run time error sine compiler will not check 10/0 and at run time only we will get run time error**

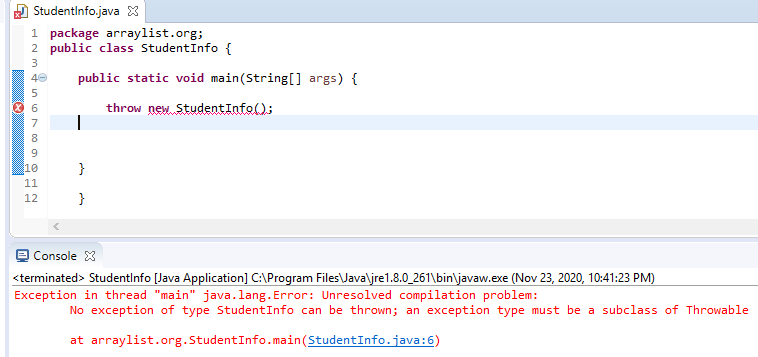


**Case 4: The below program will get the compile time error since we are declaring the exception explicitly by using the throw keyword and hello statement will not get chance to print.**



**Throw keyword only applicable for throwable exception not for classes.**

**Example:**



**Throws Keyword:**

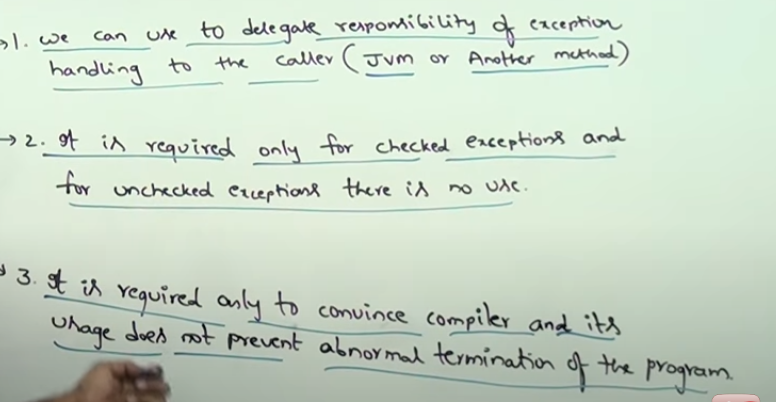
**In our program if there is a checked exception we should handle it otherwise will get compile time error. To avoid this issue we can handle in 2 ways.**

1. **By using try catch**
2. **Throws keyword**

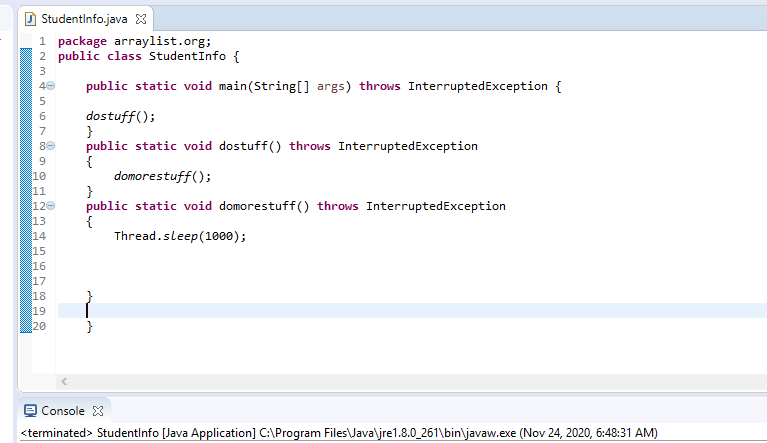
**Purpose of Throws keyword:**

**To delegate the responsibility of exception handling to the caller.**

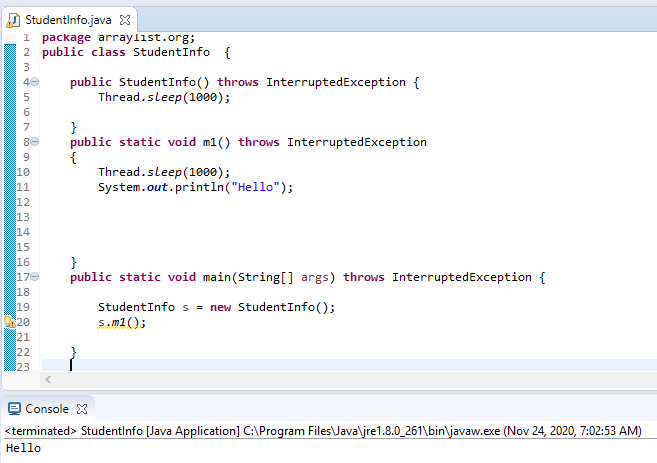
**Important points of throws keyword:**



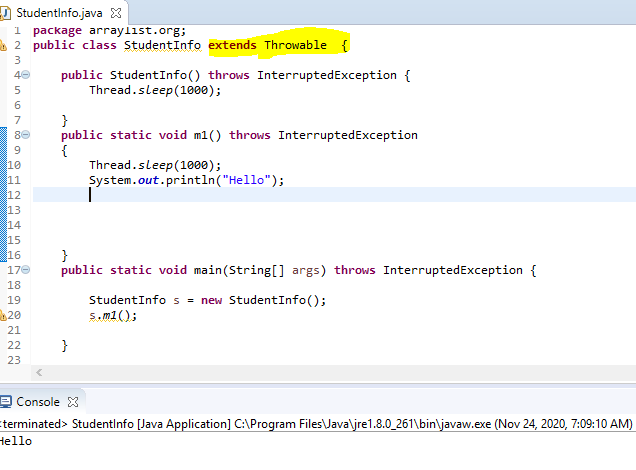
**Throws keyword with multiple methods example:**



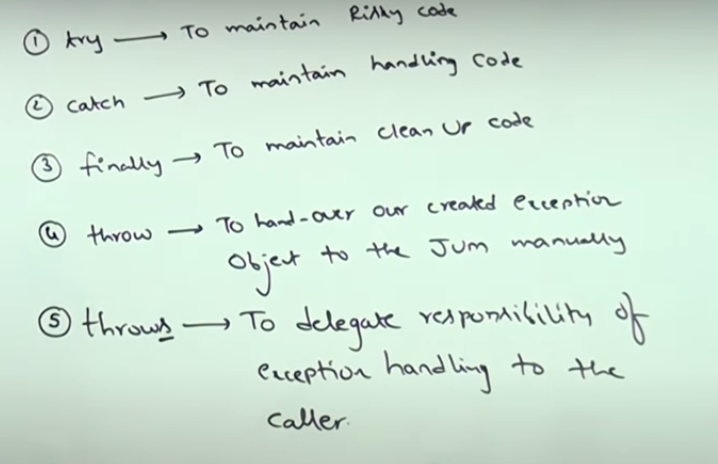
**Throws keyword is only applicable for constructor and methods not for classes.**



**We can give the throwable exception like Throwable,exception or RuntimeException in class level**



**Important points:**



**Various possible compile time errors in exception handling:**

* **Try without catch or finally**
* **Catch without try**
* **Finally, without try**
* **We should handle the checked exceptions like file not found exception, Interrupted exception through try catch finally / throws keyword. Otherwise, we will get compile time error stating that “Unreported compilation exception XXXX must be caught or declared to be thrown”**
* **If there is no chance to get an exception in try block but still we provided the exception handling code in catch block it will get compile time error and stating that “ exception XXXX is never thrown in body of corresponding try block. (But this rule is only applicable for fully checked exceptions)**
* **The order of exception handling code in catch block is must. It must be child to parent.**

**If parent to child mean will get an exception and stating that “ Exception XXX has been already caught”**

* **In between try catch block statement we can’t pass any statement. if pass it will get compilation error and stating that “ Unreachable code”**

**Difference between final, finally and finalize ():**

**Final:** It is one of the access modifiers like static, private. It is applicable for classes, methods and variables also.

If declared as class as final – Inheritance is not possible which mean we can’t extend the functionality.

If declared as method as final – We can’t override the functionality in child classes.

If declared variable as final – We can’t change the value. It will get constant.

**Finally:** It is a block associated with try and catch. It contains clean up code (resource deallocation code). It is always executed whether exception raised /not raised, handled / not handled.

**Finalize:** It is a method and associated with garbage collector.

**(If object is not having any reference variable mean eligible for garbage collector)**

**Just before destroying the object, Garbage collector calls the finalize method to perform the cleanup activities.**

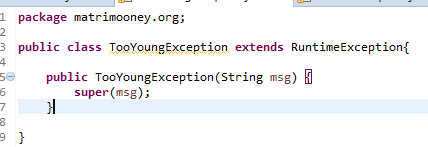
**Finally – it meant for clean up the activates associated with try block opened activities**

**Finalize – Object related cleanup activities**

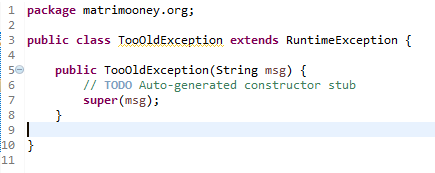
**User defined or customized exception:**

These exceptions are defined by the programmer explicitly to meet the programming requirement.

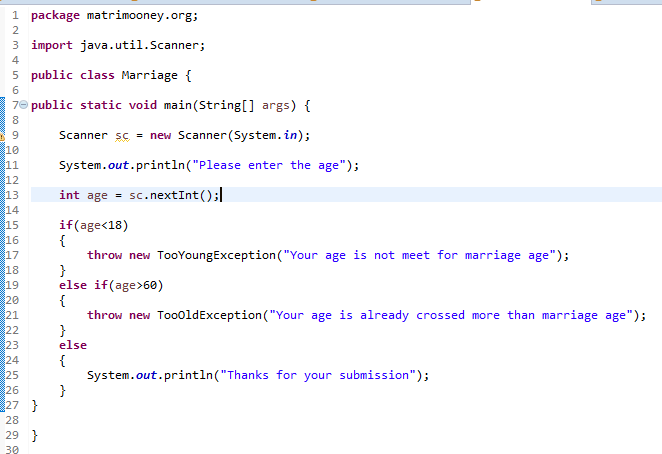
Class 1:



Class 2:



Class 3:



Why we use super () class?

We are creating our own created exceptions not using by try catch.

If not try catch block mean abnormal termination. If abnormal termination mean JVM will provide the default exception handling code.

The default exception handling code is taken care by default exception handler which is presented in Throwable class.

If we use super () mean that our created message will allocate from TooYoungException to Runtime Exception. Then, Runtime Exception to Exception,then Exception to Throwable class.

Finally,our created text message will print in console instead of printstacktrace().

The purpose of super() is to make our description available to throwable class.

**Why we are using RuntimeException instead of Exception or Throwable?**

**Exception and throwable is the checked exception but runtime Exception is the unchecked exception.**

**We are using our customized exceptions by using throw keyword instead of try catch / throws keyword.**

**If checked exception mean compiler will ask the question. If unchecked exception mean compiler will not ask any question and print the our exception message in console.**