**Exception Handling – Part-05 – Finally block**

* **Final:**

final is a modifier applicable for classes, methods and variables.

If a class declared as final then we can’t extend that class. That is, we can’t create child class for that class. That is inheritance is not possible for final classes.

If a method is final then we can’t override that method in the child class.

If a variable declared as final then we can’t perform reassignment for that variable.

* **finally:**

finally, is a block always associated with try/catch to maintain cleanup code.

try{

Risky code;

} catch(Exception e){

Handling code;

} finally{

Cleanup code;

}

The specialty of finally block is it will be executed always irrespective of exception is raised or not raised or whether handled or not handled.

* **finalize:**

finalize is a method always invoked by garbage collector just before destroying an object to perform cleanup activities. Once finalize() completes immediately garbage collector destroys that object.

Note:

finally block is responsible to perform cleanup activities related to try block that is whatever resources we opened as part of try block will be closed inside finally block.

Whereas finalize() is responsible to perform cleanup activities related to object. That is, whatever resources associated with object will be deallocated before destroying an object by using finalize method.

* **Various possible combinations of try/catch/finally:**

In try catch finally order is important.

Whenever we are writing try compulsory we should write either catch or finally, otherwise we will get compile time error. That is, try without catch or finally is invalid.

Whenever we are writing catch block compulsory try block must be required that is catch without try is invalid.

Whenever we are writing finally block compulsory we should write try block that is finally without try is invalid.

Inside try catch finally blocks we declare try catch and finally blocks that is nesting of try catch finally is allowed.

For try catch and finally blocks curly braces are mandatory.

Example:

Case\_01:

try{

} catch(X e){

}

Valid

Case\_02:

try{

} catch(X e){

} catch(Y e){

}

Valid

Case\_03:

try{

} catch(X e){

} catch(X e){

}

Invalid

CE: Exception X has already been caught

Case\_04:

try{

} catch(X e){

} finally{

}

Valid

Case\_05:

try{

} finally{

}

Valid

Case\_06:

try{

} catch(X e){

}

try{

} catch(Y e){

}

Valid

Case\_07:

try{

} catch(X e){

}

try{

} finally{

}

Valid

Case\_08:

try{

}

Invalid

CE: try without catch or finally

Case\_09:

catch(X e){

}

Invalid

CE: catch without try

Case\_10:

finally{

}

Invalid

CE: finally without try

Case\_11:

try{

} finally{

}

catch(X e){

}

Invalid

CE: catch without try

Case\_12:

try{

}

System.out.println(“Hello”);

catch(X e){

}

Invalid

CE1: try without catch or finally

CE2: catch without try

Case\_13:

try{

} catch(X e){

}

System.out.println(“Hello”);

catch(Y e){

}

Invalid

CE: catch without try

Case\_14:

try{

} catch(X e){

}

System.out.println(“Hello”);

finally{

}

Invalid

CE: finally without try

Case\_15:

try{

try{

} catch(X e){

}

} catch(X e){

}

Valid

Case\_16:

try{

try{

}

} catch(X e){

}

CE: try without catch or finally

Case\_17:

try{

try{

} finally{

}

} catch(X e){

}

Valid

Case\_18:

try{

} catch(X e){

try{

} finally{

}

}

Casey\_19:

try{

} catch(X e){

finally{

}

}

CE: finally without try.

Casey\_20:

try{

} catch(X e){

} finally {

try{

} catch(X e){

}

}

Valid

Casey\_21:

try{

} catch(X e){

} finally{

try{

} catch(X e){

}

}

Valid

Case\_22:

try{

} catch(X e){

} finally {

}

finally{

}

CE: finally without try

Casey\_23:

try

System.out.println(“try”);

catch(X e){

System.out.println(“try”);

} finally{

}

Invalid – Curly braces are important for try/catch/finally

Casey\_24:

try{

}

catch

System.out.println(“catch”);

finally{

}

Invalid – Curly braces are important for try/catch/finally

Casey\_25:

try{

}

catch(X e){

}

finally

System.out.println(“finally”);

Invalid – Curly braces are important for try/catch/finally