Monday, May 2, 2022 9:36 AM

Revision ----------
implements , default keywords , void, static property , static method

---
Syntax error , JVM error, Exception

To Handle Exception ---------- we use try { problematic code }
catch block will handle the exception.

(API)Library for exceptions and errors = All runtime exceptions and runtime errors are CLASSES in JAVA

class Throwable java.lang package is the root of all exceptions and errors in java

Two subclasses ------- Error , Exception !!

We have huge hierarchies of Exceptions under the Exception class!!!

Exceptions are of two types ---

a) Checked Exceptions ---- All the subclasses of **Exception** class except the **RuntimeException** Class sub hierarchy
b) Unchecked Exceptions ---- All the subclasses of RuntimeException

---

3 blocks in exception handling ----

1. try --- one try can have 0 or more catch blocks , 0 or 1 finally block , either catch or finally or both can be present
2. catch ---- will catch one or more exceptions and handle it, if the control catch block then exception can be handled
3. finally --- one try can have max one finally block [finally block WILL always execute -- the code that MUST run in any situation must be written in finally block]

	line after problematic code in try	Catch code	Finally code	Line after try catch block
If exception occurs and it is caught	NO	yes	yes	Yes
If exception does not occur	yes	no	yes	Yes
If exception occurs but it is not caught	NO	No matching	yes	NO -program crashes

---

One try can have many catch blocks
Or
One catch can catch many exceptions

When we have multiple catches --- the super class catch must be in the end and sub class catches must be above, other wise code does not compile ---Unreachable syntax error

---

throw , throws keywords

throw = it is written inside a method
throw object of any exception class

throw will explicitly raise / generate an exception

---

throws = it is written in the signature of the method
throws name of the exception class or classes

throws declares that the given method **MAY** throw an exception !! This is useful for the caller of the method!!!

Cascading the exception ----------
Worker---throws--->supervisor--**throws**--->manager----throws------>Director (main)-->throws (crashes)

worker ----throws ---->supervisor ---**(is handling it)**

Checked ------ Compiler treats them as SERIOUS exceptions
So compiler is INSISTING on safety measures
Compiler FORCES the caller to either RETHROW or to CATCH the exception

Unchecked ----------Compiler thinks that these are minor problems
So compiler DOES not insist/force the caller to CATCH or RETHROW

Write Custom Exceptions ----------User Defined Exceptions!!!

Writing our own Exception --------
Under18Exception

Whenever the age of the employee is under 18 , Under18Exception is thrown

Create a package study.errors.custom create a class Under18Exception
CHOICE --- should we make it a checked exception or unchecked exception ???

To make it a checked exception extend it from Exception
To make it a unchecked exception extend it from RuntimeException

Add a constructor without params and pass a message to the super class constructor

HW --------- Do the Under18Exception as discussed in class
throw it in Employee1 class and catch it in User class

Write one more custom exception Above70Exception
Throw it in Employee1 setDOB method and catch it in User class

Make Under18Exception as CHECKED Exception and Above70Exception as unchecked exception

---

Multithreading in Java ---------!!!!!!!
Thread = path of execution within a process
Why Multithreading ? Within a process many sub tasks can be done without waiting for each other,
Simultaneously in RR , Ex -- Zoom process --- chat, video , sharing threads

Every thread has a lifecycle ----- created ---(TCB, new stack in stack area) , ready , running , waiting, end of path(terminate) .

API ---- java.lang. Thread class that represents a thread

HW --- type the SingleThreadExample and MultiThreadExample , run them observe result

Part 2 --- instead of calling start method in MultithreadExample
Call run() of both threads and observe output!!!!!!!!!!

HW -----

Using Random API create a game

class Game
{
Property = hiddenNumber
= chances = 5
Constructor ()----- generate and set a random number in hiddenNumber

Getter , setter

int isMatching(pass the number given by user)
Return 0 if the number is matching the hidden number
Return -1 if the hidden number is less than number
Return 1 if the hidden number is greater than number

}

class PlayGame
Main
Add a play again loop
Create a game object for every game ----
Call the getchances , set chances

Call isMatching

GUESS the number

First create a random number in the beginning of the game between 0 and 20

For ex ------- random number is 7

ask the user to guess the number

I guess = 12

System should say wrong guess --- number is lesser

Guess again

3

System -------- wrong guess --- number is greater

Guess again 7

Bingo ----you win

Play AGAIN ?

Max 5 chances !!
