DAY-55

-------

Can we have multiple try block in a program?

--> continuous try block is no permitted in java where as continous catch block is permitted.

NOTE: for a single try block we can have multiple catch block in java.

MULTIPLE CATCH BLOCK:

---------------------

EXAMPLE:

--------

import java.util.\*;

class Demo

{

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

try

{

System.out.println("enter the value of A");

int a = sc.nextInt();

System.out.println("enter the value of B");

int b = sc.nextInt();

System.out.println(a/b);

}

catch(ArithmeticException e)

{

System.out.println("learn maths..!");

}

catch(InputMismatchException e)

{

System.out.println("plz enter valid inputs ");

}

catch (Exception e)

{

System.out.println("invalid");

}

}

}

OUTPUT:

-------

enter the value of A

10

enter the value of B

0

learn maths..!

enter the value of A

10

enter the value of B

JBSJVBSJ

plz enter valid inputs

final, finally , finalize in exception handling:

-------------------------------------------------

1. final keyword:

-----------------

final is a keyword which can be used with classes , methods , variables

final class : class will not participate in inheritance.

final methods : that method will not permit to override.

final variables : constants, values associated with it can not be modified.

finally block:

--------------

finally block always associated with try and catch blocks to maintain the 'clean up code'.

EXAMPLE:

--------

import java.util.\*;

class Demo

{

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

try

{

System.out.println("enter the value of A");

int a = sc.nextInt();

System.out.println("enter the value of B");

int b = sc.nextInt();

System.out.println(a/b);

}

catch (Exception e)

{

System.out.println("invalid");

}

finally

{

System.out.println("finally executed..!");

}

}

}

OUTPUT:

--------

enter the value of A

10

enter the value of B

2

5

finally executed..!

enter the value of A

10

enter the value of B

0

invalid

finally executed..!

--> finally block is used to excute some set of statements which has to be executed if exception occur or it it does not occur .

finally block is used w.r.t the codes like closing the connection with database,file,network etc..

finalize() method :

--------------------

finalize method is invoked by 'garbage collector' just before destroying an object to perform cleap up activities.

Once finalize work is completed the G.C will destory the object.

NOTE: finally block is responisble to clean up the things w.r.t try block whereas finalize deal with exception object.

EXCEPTION OBJECT:

-----------------

The main purpose of the catch block is to catch the exception from try block.

The exception is created as a object inside the heap memory segment.

REFER DIA

To get the information present in the exception object:

--------------------------------------------------------

1. e.printStackTrace();

2. s.o.p(e);

3. s.o.p(e.getMessage());

NOTE: exception object consist of name,description, location of the exception.

EXAMPLE:

--------

import java.util.\*;

class Demo1

{

public static void main(String[] args)

{

try

{

Scanner sc = new Scanner(System.in);

System.out.println("enter the value of A");

int a = sc.nextInt();

System.out.println("enter the value of B");

int b = sc.nextInt();

System.out.println(a/b);

}

catch (Exception e)

{

e.printStackTrace();

System.out.println();

System.out.println(e);

System.out.println();

System.out.println(e.getMessage());

}

}

}

OUTPUT:

-------

enter the value of A

10

enter the value of B

0

java.lang.ArithmeticException: / by zero

at Demo1.main(Demo1.java:15)

java.lang.ArithmeticException: / by zero

/ by zero

enter the value of A

10

enter the value of B

SDSCFSVF

java.util.InputMismatchException

at java.util.Scanner.throwFor(Scanner.java:864)

at java.util.Scanner.next(Scanner.java:1485)

at java.util.Scanner.nextInt(Scanner.java:2117)

at java.util.Scanner.nextInt(Scanner.java:2076)

at Demo1.main(Demo1.java:13)

java.util.InputMismatchException

null

TYPES OF EXCEPTIONS:

---------------------

There are two types of exceptions in java:

1. PRE-DEFINED EXCEPTIONS

2. USER DIFINED EXCEPTIONS

REFER dia Exception hierarchy

1. pre - defined exceptions

-----------------------------

All the exceptions present in the exception hierarchy is considered as pre defiend exceptions

There are two types of pre - defined exceptions

1. Checked Exceptions

2. Unchecked Exceptions

What is the difference b/w checked exception and unchecked exceptions?

-->Chekced exceptions is an exception recognized at compile time,but, not occurred at compilation time.

Unchecked exceptions are not recognize at compile time,these exceptions are recognized at runtime by JVM.

NOTE: In exceptions hierarchy, Runtime exceptions and its sub-class and error and its sub-classes are the example for unchekced exceptions.

All the remaining exceptions are example for cheked exceptions.

There are two types of checked exceptions are:

1. pure checked exception

2. partially checked exceptions

what is the difference b/w pure and partially checked exceptions?

--> Pure checked exceptions is having only chekced exceptions as sub classes in it.

ex: IO exception

partially checked exception where atleast one sub class as unchecked exception present in it.

ex: Exception , throwable