Exception handling:
==========

Error
=======

    * Compile-time
    * Run-time -> Exception

There are 3 types of termination of programs:
==========
* Normal Termination
* Forceful Termination
* Abnormal Termination

Exception is an unwanted, unexpected event which disturbs the
normal flow of execution.

It is an unexpected and unwanted event triggered in the JVM, which
makes the JVM to terminate the program abnormally.

Default Exception handler:
==========
        When exception occurs in a program, JVM will create an object
    of respective type of exception and throws it back to the program.
    If user has not written any code to handle the thrown object,
    then JVM calls an assistant called DEFAULT EXCEPTION HANDLER
    to handle that object.

    Default Exception handler will first terminate the program
    abnormally and display the information about the exception
    to the user.

    If default exception handler handles the exception then
    it will terminate the program abnormally.
    If we want the normal termination then user should handle
    the exception.

    Handlers:
    ==========

    * try
    * catch
    * throw
    * throws
    * finally

    *   try block contains only those lines of code which might
    cause an exception. When the exception occurs in the try
    block JVM will create an object and throws it back to the
    program or user.

        User should have written appropriate catch block to catch
    the object thrown from the try block.
    If the exception occurs in the try block only then catch block
    will be executed.

        Once the control leaves from the try block, It will never
    come back to try block again. Deu to this all the lines of
    code which is written below the exception will not be executed.

        Just because we have written try block it doesn't mean
    exception will always occur.

        In one try block utmost one exception can occur.
    When the exception object is thrown from the try block that
    object will be caught in the respective catch block.

        If the user has not written the respective catch block the
    JVM will handle the exception.

        A try block can have multiple catch blocks.

Depending upon the type of exception occured , only one catch block will be executed but not all.

    If we want to handle more than one exception then we should go for multiple try catch blocks.

We can develop two types of catch blocks namely:
    * Generalized catch blocks
    * Specialized catch blocks

    Generalized catch blocks and Specialized catch blocks can be written together but order should be first Specialized and then Generalized else the Specific catch blocks will get into unreachable code.

throw keyword:
==========

1. It is a keyword used to throw both the checked and unchecked exception objects explicitly.

2. it throws only those objects which has the properties of Throwable class.

```
    throw new ArithmeticException()
    throw new NullPointerException()
    throw new SqlException()
    throw new IOException()
    throw new ClassCastException()

    Custom Exceptions:
    ==========
    throw new AgeInvalidException()
    throw new NotEligibleForMarriageException()
    throw new NotEligibleForJobException()

    Note: valid if and only if they have the properties
            of Throwable class.
```

3. Using throw we can throw only one exception object at a time.

4. throw must be used inside the method definition.

Example:

```
    try
    {
        sop("hello");
        sop(10/0);
        sop("good time");
        //not executed, Decision will be made during runtime.
    }
    catch(ArithmeticException ae)
    {
        sop(ae.getMessage());
    }
    ------------------------------------------------------------

    try
    {
        sop("hello");
        throw new ArithmeticException("Dont divide by zero");
        sop("good time");
        //not executed, Decision will be made during compiletime.
    }
    catch(ArithmeticException ae)
    {
        sop(ae.getMessage());
    }
```

Exception Propagation:

```
=========================

1. When exception occurs in any method and if that exception is not
handled by the user then that exception object will be
automatically propagated back to the caller.

This propagation continues till the JVM (caller)

2. Exception object will be automatically propagated if it is
an unchecked exception.
If the exception object is checked we should inform the caller
by using throws keyword.

This is because checked exception does not have the capability
of propagating to the caller by itself.


throws keyword:
================

1. It is a keyword used to inform the caller about the checked
exception.

2. throws keyword should be used in the method declaration.

3. Using throws keyword we can inform multiple checked
exception object to the caller.

public void m1() throws SqlException,InterruptedException
{
    // code
}

Difference B/W throw v/s throws
================================
throw:
* It is a keyword used to throw both checked and unchecked
exception objects.
* It must be used in the method definition.
* We can throw only one exception object at a time.

throws:
* It is a keyword used to inform the caller about the checked
exception.
* It must be used in the method declaration.
* We can inform more than one checked exception objects
to the caller.



Difference B/W Checked Exception v/s UnChecked Exception:
=========================================================

Checked Exception:
* known at compiletime but occurs at runtime.
* they dont have the capability of propagating the caller
    by itself.
* throws keyword is required.
* compiler forces to provide an alternative during compilation.


UnChecked Exception:
* known and occurs at runtime.
* they have the capability of propagating the caller by itself.
* throws keyword is required.
* compiler does not force.


finally block:
================

1. finally block usually contains closing related operations
like closing the open database connection, closing the open
```

file, closing the open session or terminal etc.

2. finally block will always be executed irrespective of the exception.

3. finally block will not be executed when there is forceful termination of program by the user.

4. try block can contain only finally block without catch block where JVM handles the exception using DEFAULT EXCEPTION HANDLER.