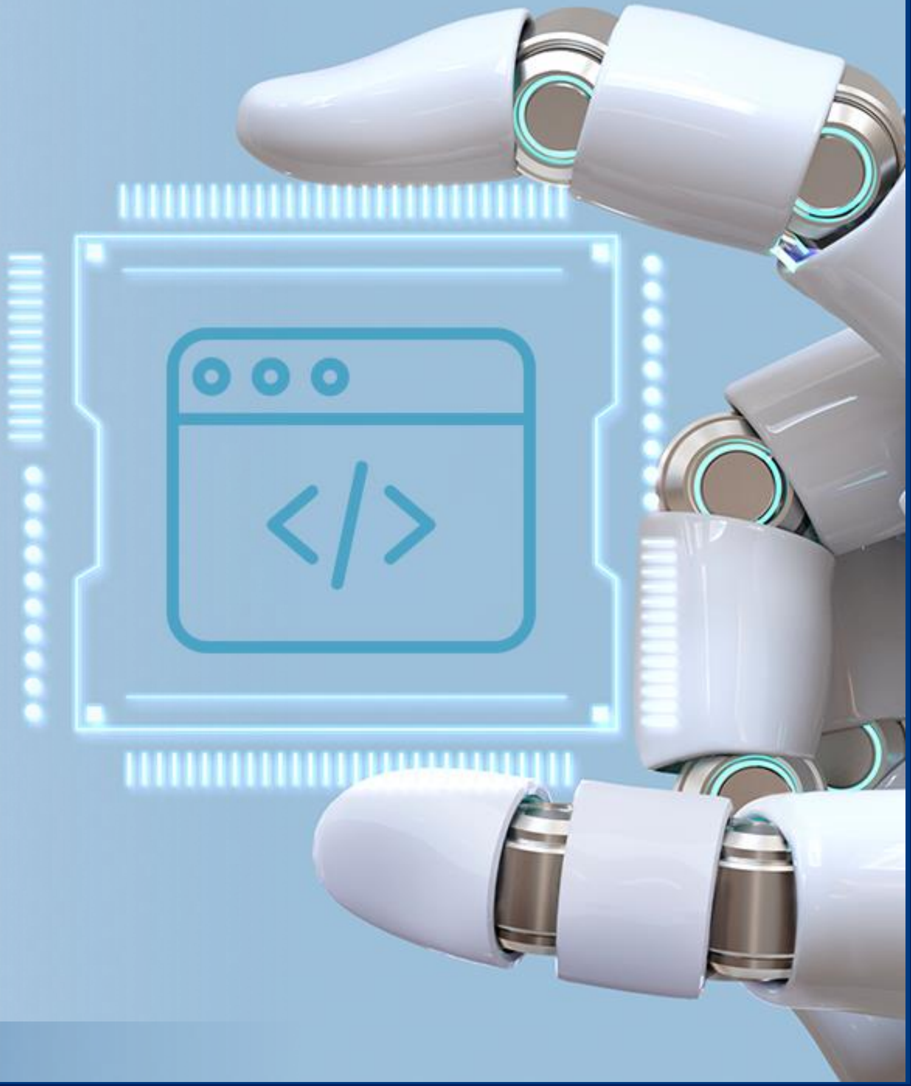
# Error Handling and Advance Java



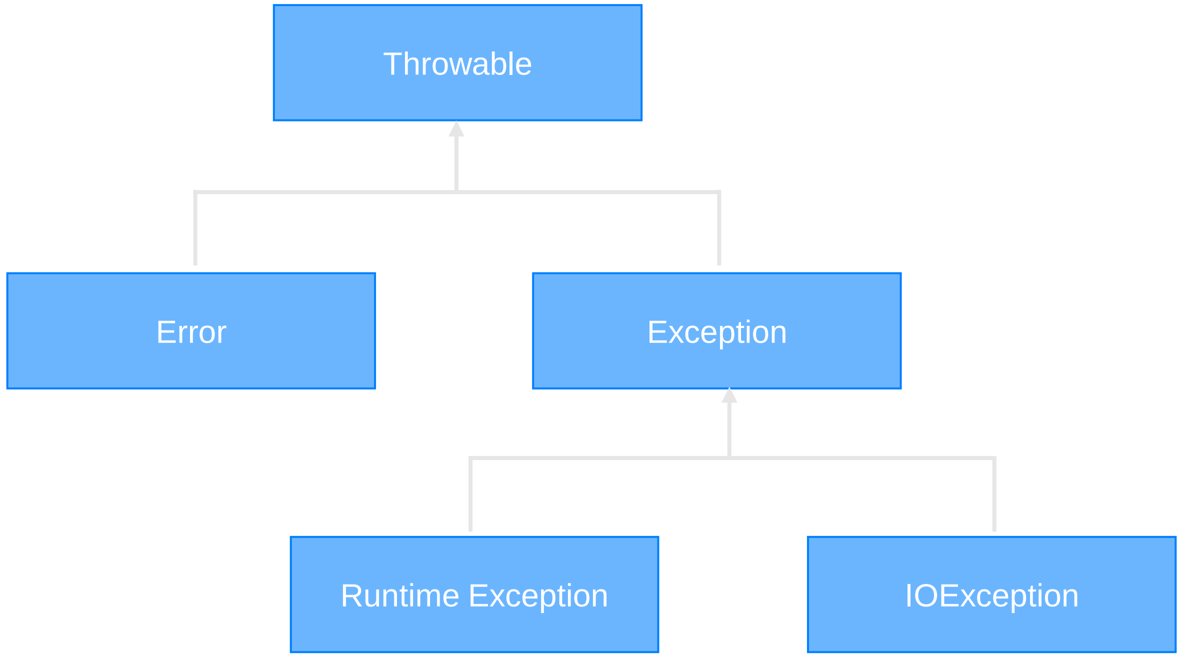
## Agenda

You will learn in this lesson:

- Java Exception
- Types of Java Exception
- Exception Handling
- Try and Catch the block
- Threading
- Collection
- Lambda
- Annotation

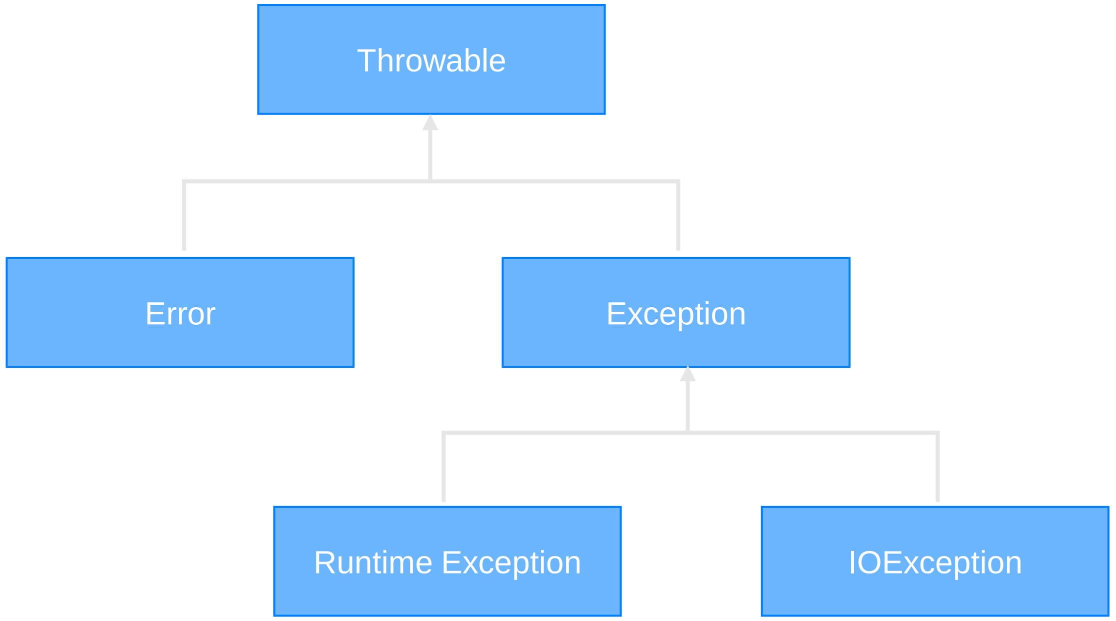
## Introduction

- Welcome everyone to the presentation on Error Handling and Advanced Java Concepts.
- Error handling, also known as exception handling, is a crucial aspect of software development that involves dealing with and managing errors, exceptions, and abnormal conditions that may occur during the execution of a program.
- It is essential for creating robust and reliable software systems.



## Java Exceptions

- An exception is an unexpected event that occurs during program execution.
- An exception is an event that disrupts the normal flow of the program

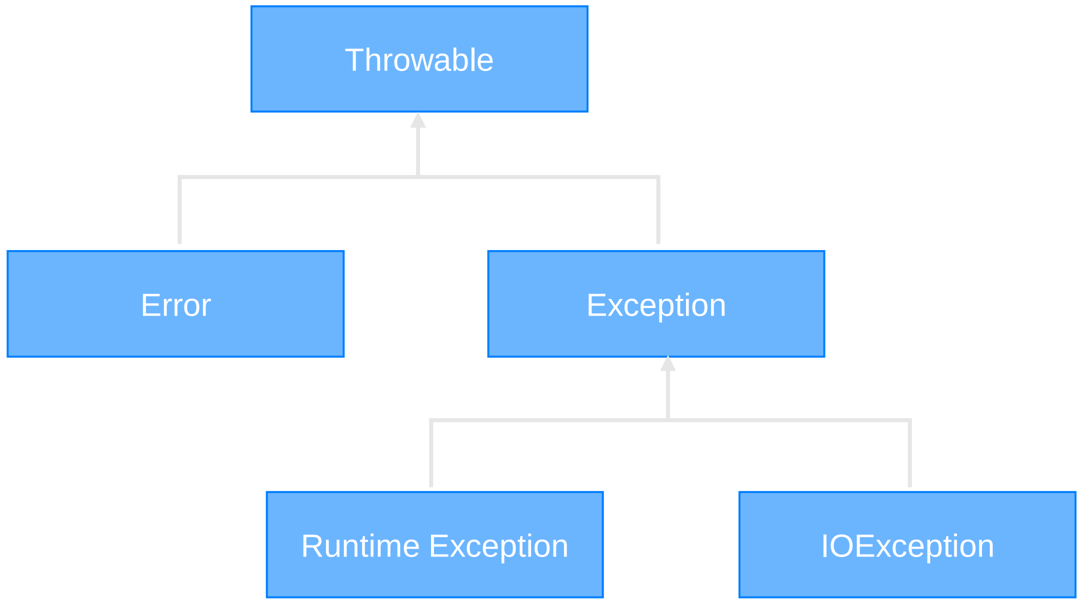


Java Exception Hierarchy

Source: [https://www.programiz.com/java-programming/exceptions](https://www.programiz.com/java-programming/exceptions)

## Types of Java Exceptions

- **Errors**
  - Errors represent irrecoverable conditions
  - Errors are usually beyond the control of the programmer, and we should not try to handle errors.
- **Exceptions**
  - Exceptions can be caught and handled by the program.

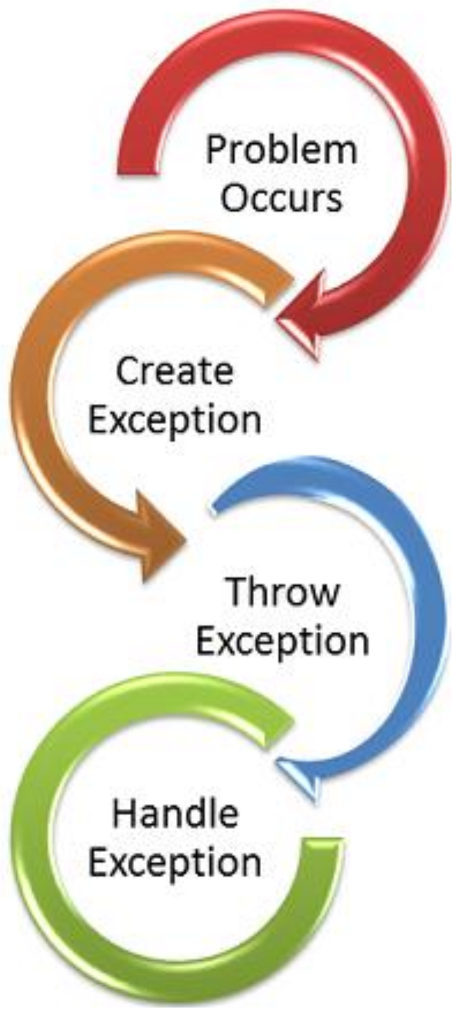


Java Exception Hierarchy

Source: https://www.programiz.com/java-programming/exceptions

## Exception Handling

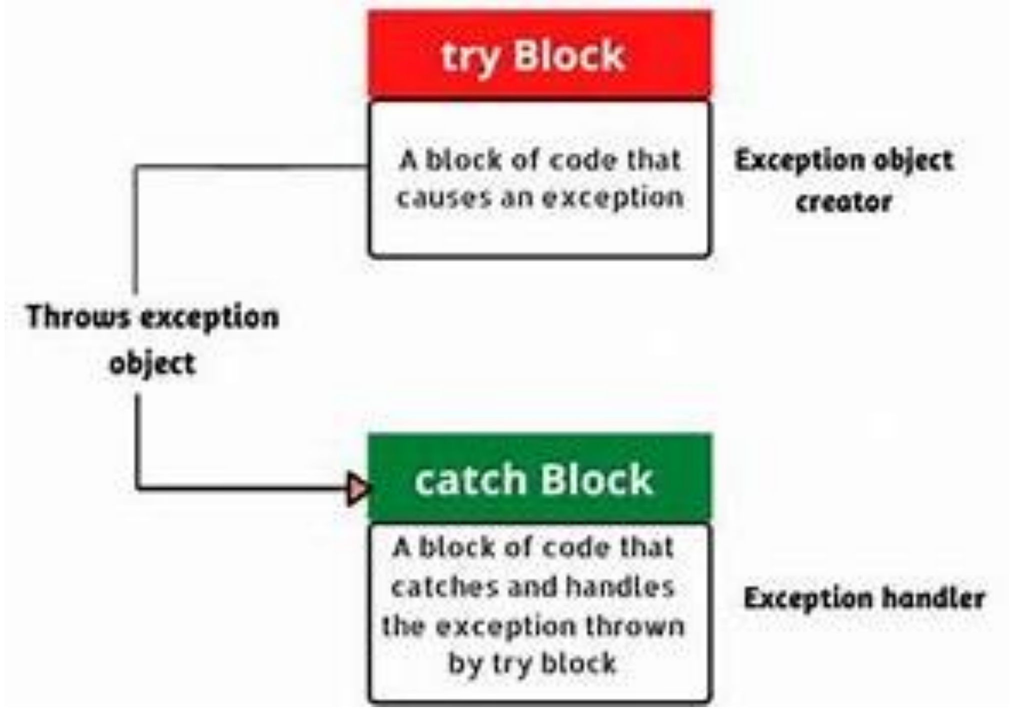
- Exception handling is a programming technique that allows for the detection, handling, and recovery of exceptional conditions or errors that may occur during the execution of a program.
- It involves using try-catch blocks to catch and handle exceptions, ensuring the program continues to run smoothly and preventing unexpected crashes.



Source: https://www.scientecheasy.com/2020/08/exception-handling-in-java.html/

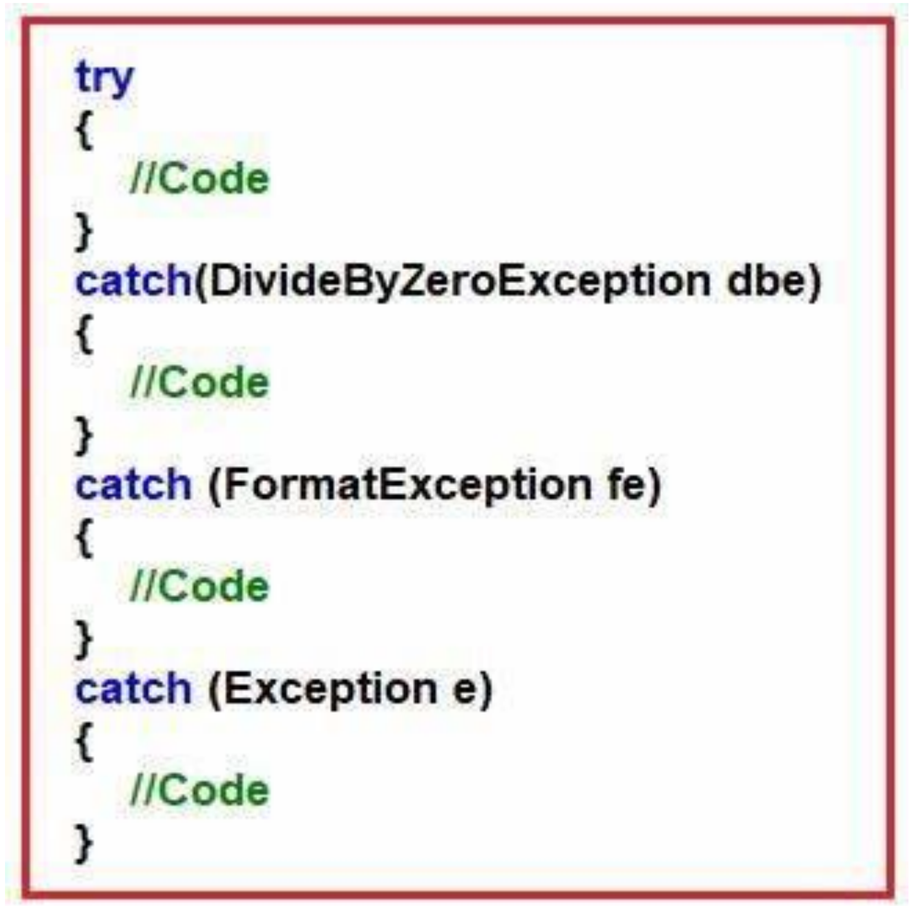
## Java try...catch

- The try...catch block in Java is used to handle exceptions and prevent the abnormal termination of the program.
- The try block includes the code that might generate an exception.
- The catch block includes the code that is executed when there occurs an exception inside the try block.
- In Java, we can use a try block without a catch block. However, we cannot use a catch block without a try block.


try Block
A block of code that causes an exception
Exception object creator
Throws exception object
catch Block
A block of code that catches and handles the exception thrown by try block
Exception handler

## Multiple Catch Blocks

- For each try block, there can be zero or more catch blocks.
- Multiple catch blocks allow us to handle each exception differently.
- The argument type of each catch block indicates the type of exception that can be handled by it.

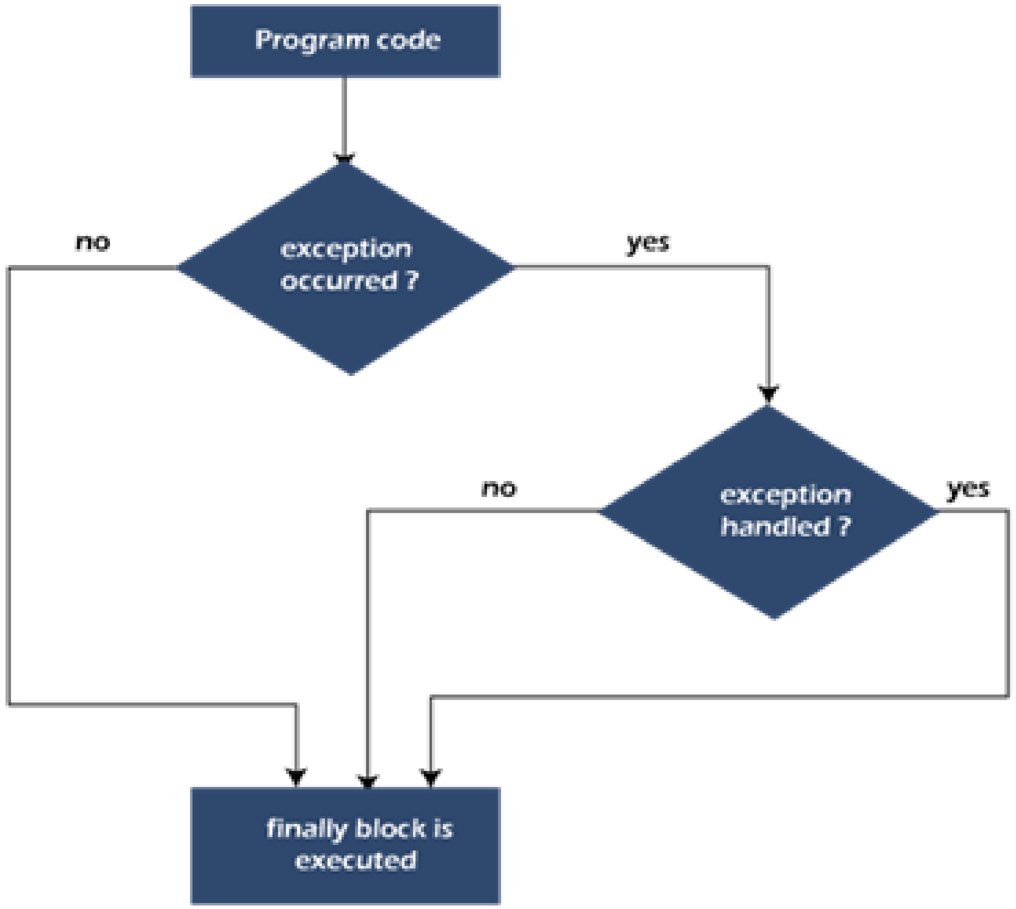


```
try
{
    //Code
}
catch(DivideByZeroException dbe)
{
    //Code
}
catch (FormatException fe)
{
    //Code
}
catch (Exception e)
{
    //Code
}
```

Source:

## Finally Block

- The finally block is always executed whether there is an exception inside the try block or not.
- The code inside the finally block is executed irrespective of the exception.
- It is a good practice to use finally block to include important cleanup code like closing a file or connection.



## Java Throw and Throws

### Java throws keyword

The throws keyword in the method declaration to declare the type of exceptions that might occur within it.

### Java throw keyword

The throw keyword is used to explicitly throw a single exception.

## Lab Exercise

**Hands On - 11:** Example of error handling in java

**Hands On - 12:** Example of exception handling in java

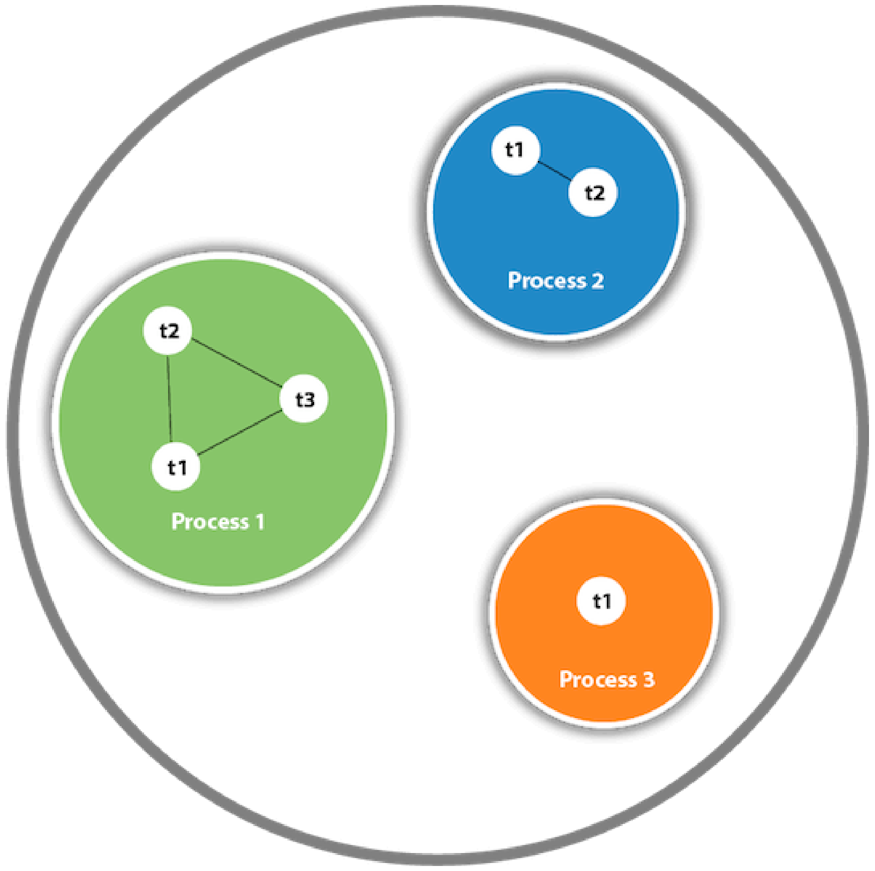
## Threading

- A **thread** is a lightweight subprocess, the smallest unit of processing. It is a separate path of execution.
- process of executing multiple threads simultaneously is called **Multithreading** in Java.

There are two ways to create a thread:

1) By extending the Thread class

2) By implementing a Runnable interface.



Source: https://www.javatpoint.com/multithreading-in-java

## Lab Exercise

**Hands On - 13:** Example of threading

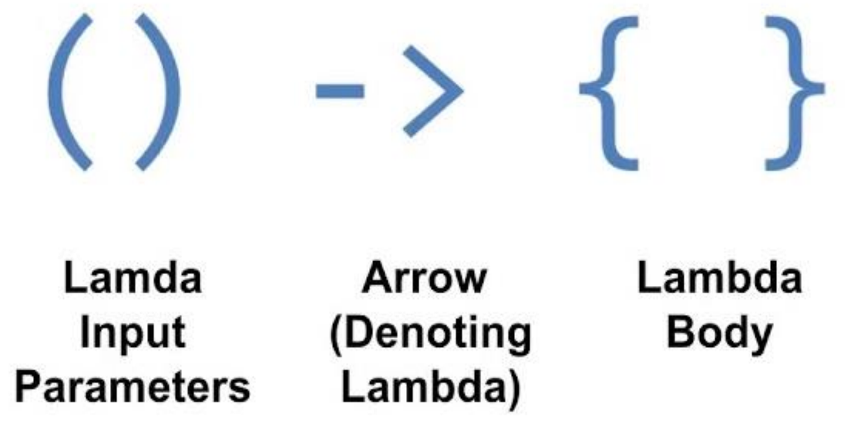
## Java Lambda Expressions

- The Lambda expression is used to provide the implementation of an interface that has a functional interface.
- It saves a lot of code. In the case of the lambda expression, we don't need to define the method again for providing the implementation.

**Why use Lambda Expression**

1. To provide the implementation of the Functional interface.

2. Less coding.



## Java Lambda Expressions

No Parameter Syntax

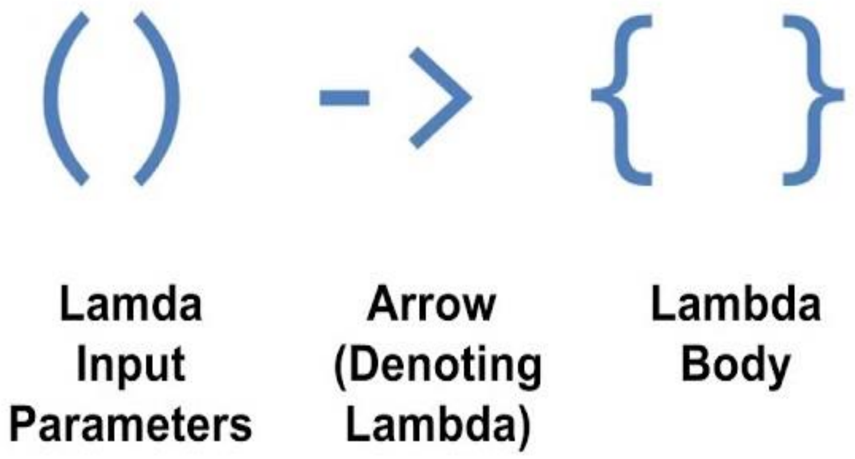
```
    () -> {
    //Body of no parameter lambda  }
```

One Parameter Syntax

```
    (p1) -> {
    //Body of single parameter lambda
}
```

Two Parameter Syntax

```
    (p1,p2) -> {
    //Body of multiple parameter lambda
}
```



## Lab Exercise

**Hands On - 14:** Example of Lambda

## Collections in Java

The Collection in Java is a framework that provides an architecture to store and manipulate a group of objects.

Collections in Java provide a framework for efficient storage, retrieval, and manipulation of groups of objects.

The Collection Framework in Java includes interfaces, implementations, and algorithms for working with collections.

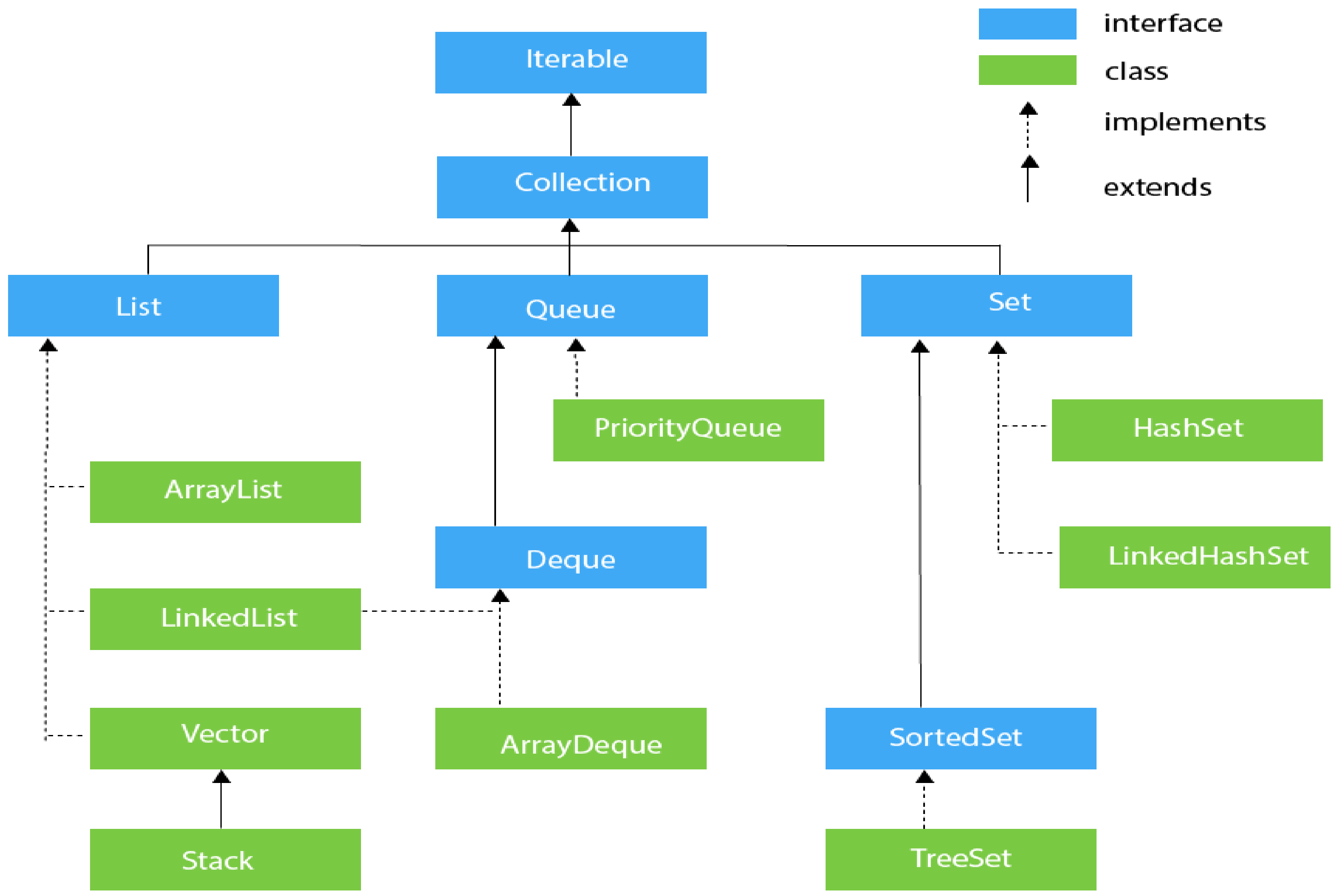
## Collection Framework

The Collection Framework in Java is a unified architecture that provides a set of interfaces, classes, and algorithms to work with collections of objects.

It offers a standardized way to store, manipulate, and process groups of objects efficiently.

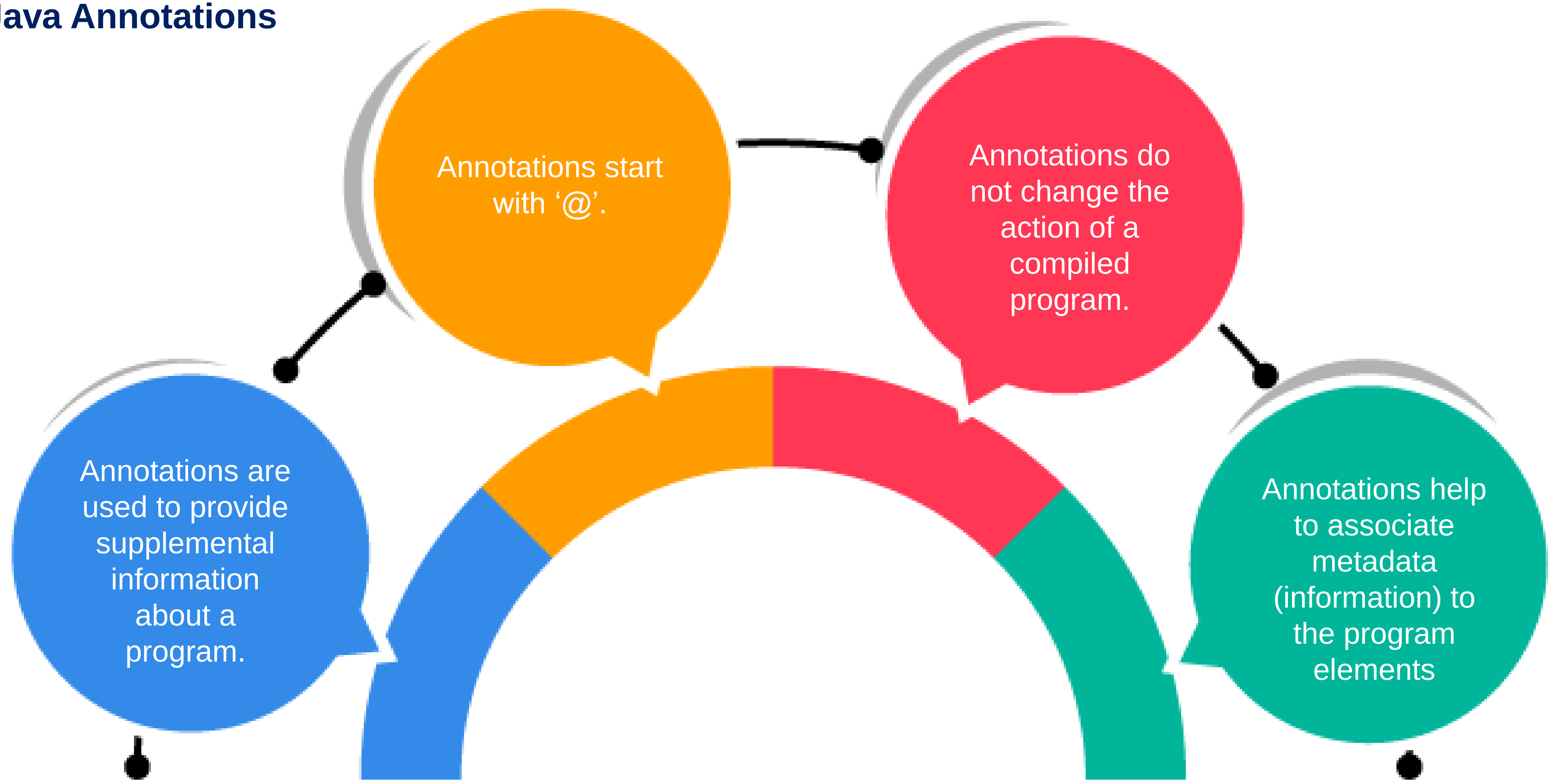
The Collection Framework is part of the java.util package and serves as a foundation for working with collections in Java.

## Hierarchy of Collection Framework



Source:

## Java Annotations



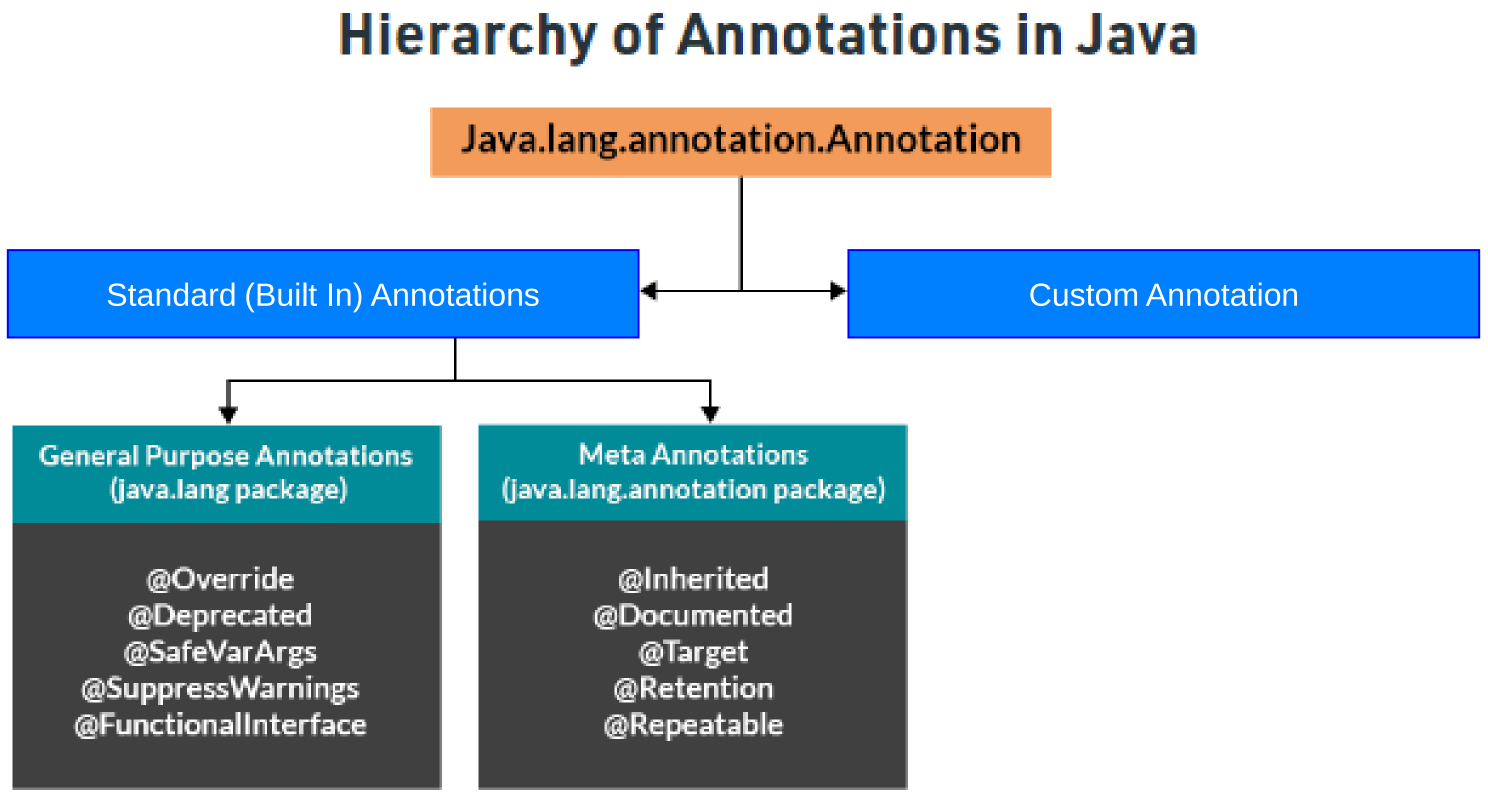
Annotations are used to provide supplemental information about a program.

Annotations start with ‘@’.

Annotations do not change the action of a compiled program.

Annotations help to associate metadata (information) to the program elements

## Java Annotations


Hierarchy of Annotations in Java
Java.lang.annotation.Annotation
Standard (Built In) Annotations
Custom Annotation
General Purpose Annotations (java.lang package)
@Override
@Deprecated
@SafeVarArgs
@SuppressWarnings
@FunctionalInterface
Meta Annotations (java.lang.annotation package)
@Inherited
@Documented
@Target
@Retention
@Repeatable

Source: https://www.geeksforgeeks.org/annotations-in-java/

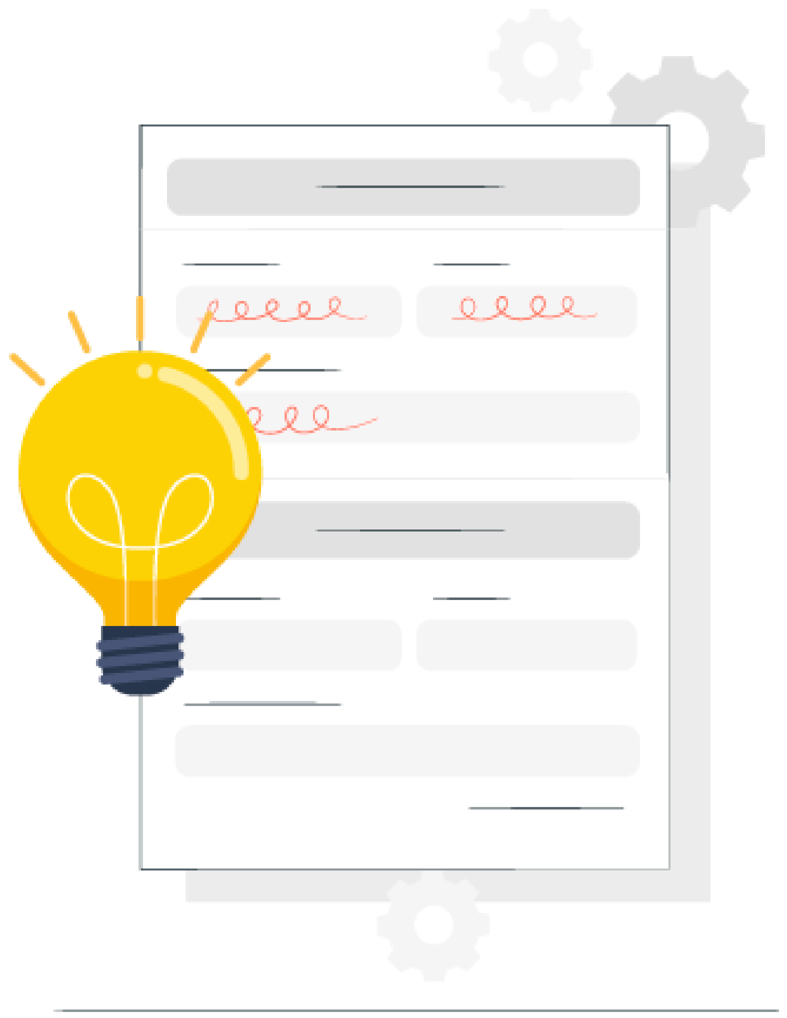
## Lab Exercise

**Hands On - 15:** Example of Annotations

## Conclusion

Well done! You have completed this course and now you have understand about:

- Java Exception
- Types of Java Exception
- Exception Handling
- Try and Catch the block
- Threading
- Collection
- Lambda
- Annotation



## Quiz

**1. Predicting the amount of rainfall in a region based on various cues is a ______ problem.**

a) Try

b) finally

c) thrown

d) catch

**Answer: c**
thrown

## Quiz

**2. What will be the output of the Java program?**

a) Hello

b) World

c) HelloWorld

d) Hello World

```
class exception_handling
  {
      public static void main(String args[])
      {
          try
          {
              System.out.print("Hello" + " " + 1 / 0);
          }
          catch(ArithmeticException e)
          {
              System.out.print("World");

          }
      }
  }
```

**Answer: b**
World

## Quiz

**3. Which version of Java introduced annotation?**

a) Java 5

b) Java 6

c) Java 7

d) Java 8

**Answer: a**
Java 5

## Quiz

**4. Which of these methods deletes all the elements from invoking collection?**

a) clear()

b) reset()

c) delete()

d) refresh()

**Answer: a**
clear()

## Quiz

**5. What is the use of try & catch?**

a) It allows us to manually handle the exception

b) It allows to fix errors

c) It prevents automatic terminating of the program in cases when an exception occurs

d) All of the mentioned

**Answer: d**
All of the mentioned

## References

- https://www.geeksforgeeks.org/errors-v-s-exceptions-in-java/
- https://www.javatpoint.com/exception-vs-error-in-java
- https://www.javatpoint.com/multithreading-in-java
- https://www.baeldung.com/java-errors-vs-exceptions
- https://www.youtube.com/watch?v=y-NlcLcxiKY&list=PLlhM4lkb2sEjaU-JAASDG4Tdwpf-JFARN
- https://www.youtube.com/watch?v=1xuDEPftKV0&t=191s

Thank You!