APIs and AnnotationAssignment Questions

Assignment Questions:

1. Program to display current date and time in Java?

Ans → import java.util.Date;

public class CurrentDateTime {

public static void main(String[] args) {

// create a new Date object

Date currentDate = new Date();

// display the current date and time

System.out.println("Current date and time: " + currentDate);

}

}

1. Write a program to convert a date to a String in the format “MM/dd/yyyy”.

Ans → import java.text.SimpleDateFormat;

import java.util.Date;

public class DateToStringExample {

public static void main(String[] args) {

// create a new Date object

Date currentDate = new Date();

// create a SimpleDateFormat object with the desired format

SimpleDateFormat dateFormat = new SimpleDateFormat("MM/dd/yyyy");

// use the format method of the SimpleDateFormat object to format the date

String dateString = dateFormat.format(currentDate);

// display the formatted date string

System.out.println("Formatted date string: " + dateString);

}

}

1. What is the difference between collections and streams? Explain with an Example.

Ans → A collection is an object that holds a group of elements, typically of the same type, and provides operations to add, remove, and access elements in the group. Collections can be traversed using iterators or for-each loops, and they can be sorted, filtered, and manipulated using a variety of methods provided by the collection classes.

A stream, on the other hand, is a sequence of elements that supports various operations, such as filtering, mapping, and reducing, that can be pipelined to produce the desired result. Streams can be thought of as lazy collections that don't store the elements themselves but instead generate them on demand based on the operations that are performed on the stream.

1. What are enums in Java? Explain with an example.

Ans → In Java, an enum is a special type of class that represents a fixed set of constants. Enums can be used to define a group of related constants that are often used together in a program.

public enum DayOfWeek {

MONDAY,

TUESDAY,

WEDNESDAY,

THURSDAY,

FRIDAY,

SATURDAY,

SUNDAY

}

1. What are inbuilt annotations in Java?

Ans → Java provides several built-in annotations that can be used to add metadata and additional information to our code. Here are some of the most commonly used built-in annotations in Java:

1. @Override: Indicates that a method overrides a method from a superclass or interface.
2. @Deprecated: Indicates that a class, method, or field is deprecated and should no longer be used. This annotation is often used to provide information about why the deprecated element was removed and what alternative should be used instead.
3. @SuppressWarnings: Suppresses compiler warnings for a specific code element or for an entire method or class. This annotation is useful for ignoring warnings that are safe to ignore or for code that is generated by a third-party tool and cannot be modified.
4. @SafeVarargs: Indicates that a method or constructor does not perform any potentially unsafe operations on its variable arguments (varargs) parameter. This annotation is used to suppress unchecked warnings that might otherwise be generated when using varargs.
5. @FunctionalInterface: Indicates that an interface is intended to be a functional interface, which is an interface that has a single abstract method. This annotation is useful for preventing accidental addition of additional methods to a functional interface, which would break existing code that relies on the interface's single method.
6. @SuppressWarnings: This annotation can be used to suppress warnings that are generated by the compiler.
7. @Nullable and @NonNull: These annotations are used to indicate whether a parameter, variable or return value can be null or not.