1. Problem 1: Time Zones and ZonedDateTime Write a program that displays the current

time in different time zones:

➢ GMT (Greenwich Mean Time)

➢ IST (Indian Standard Time)

➢ PST (Pacific Standard Time)

Solution:

import java.time.ZonedDateTime;

import java.time.ZoneId;

import java.time.format.DateTimeFormatter;

public class TimeZonesExample {

public static void main(String[] args) {

// Define a formatter for displaying date and time

DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss z");

// GMT (Greenwich Mean Time)

ZonedDateTime gmtTime = ZonedDateTime.now(ZoneId.of("GMT"));

System.out.println("GMT Time: " + gmtTime.format(formatter));

// IST (Indian Standard Time)

ZonedDateTime istTime = ZonedDateTime.now(ZoneId.of("Asia/Kolkata")); // IST is Asia/Kolkata

System.out.println("IST Time: " + istTime.format(formatter));

// PST (Pacific Standard Time)

ZonedDateTime pstTime = ZonedDateTime.now(ZoneId.of("America/Los\_Angeles"));

System.out.println("PST Time: " + pstTime.format(formatter));

}

}

2. Problem 2: Date Arithmetic Create a program that:

➢ Takes a date input and adds 7 days, 1 month, and 2 years to it.

➢ Then subtracts 3 weeks from the result.

Solution:

import java.time.LocalDate;

import java.time.format.DateTimeFormatter;

import java.util.Scanner;

public class DateArithmeticExample {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

// Input formatter

DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");

// Take date input from user

System.out.print("Enter a date (yyyy-MM-dd): ");

String inputDate = sc.nextLine();

// Parse the input

LocalDate date = LocalDate.parse(inputDate, formatter);

System.out.println("Original Date: " + date);

// Add 7 days, 1 month, and 2 years

LocalDate addedDate = date.plusDays(7).plusMonths(1).plusYears(2);

System.out.println("After adding 7 days, 1 month, 2 years: " + addedDate);

// Subtract 3 weeks

LocalDate finalDate = addedDate.minusWeeks(3);

System.out.println("After subtracting 3 weeks: " + finalDate);

}

}

Problem 3: Date Formatting Write a program that:

➢ Displays the current date in three different formats:

■ dd/MM/yyyy

■ yyyy-MM-dd

■ EEE, MMM dd, yyyy

Solution:

import java.time.LocalDate;

import java.time.format.DateTimeFormatter;

public class DateFormattingExample {

public static void main(String[] args) {

// Get current date

LocalDate currentDate = LocalDate.now();

System.out.println("Original Date: " + currentDate);

// Format 1: dd/MM/yyyy

DateTimeFormatter format1 = DateTimeFormatter.ofPattern("dd/MM/yyyy");

System.out.println("Format 1 (dd/MM/yyyy): " + currentDate.format(format1));

// Format 2: yyyy-MM-dd

DateTimeFormatter format2 = DateTimeFormatter.ofPattern("yyyy-MM-dd");

System.out.println("Format 2 (yyyy-MM-dd): " + currentDate.format(format2));

// Format 3: EEE, MMM dd, yyyy

DateTimeFormatter format3 = DateTimeFormatter.ofPattern("EEE, MMM dd, yyyy");

System.out.println("Format 3 (EEE, MMM dd, yyyy): " + currentDate.format(format3));

}

}

Problem 4: Date Comparison Write a program that:

➢ Takes two date inputs and compares them to check if the first date is before, after,

or the same as the second date.

Solution:

import java.time.LocalDate;

import java.time.format.DateTimeFormatter;

import java.util.Scanner;

public class DateComparisonExample {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");

// Take first date input

System.out.print("Enter first date (yyyy-MM-dd): ");

String input1 = sc.nextLine();

LocalDate date1 = LocalDate.parse(input1, formatter);

// Take second date input

System.out.print("Enter second date (yyyy-MM-dd): ");

String input2 = sc.nextLine();

LocalDate date2 = LocalDate.parse(input2, formatter);

// Compare the two dates

if (date1.isBefore(date2)) {

System.out.println("The first date is BEFORE the second date.");

} else if (date1.isAfter(date2)) {

System.out.println("The first date is AFTER the second date.");

} else if (date1.isEqual(date2)) {

System.out.println("Both dates are the SAME.");

}

}

}