package Level1\_Practice**;**import java.time.LocalDate**;**import java.time.ZonedDateTime**;**import java.time.ZoneId**;**import java.time.format.DateTimeFormatter**;**import java.util.Scanner**;**public class timeZoneDisplay {  
 public static void main(String[] args) {

**//problem-1**  
 // Define the formatter for displaying date and time  
 DateTimeFormatter formatter = DateTimeFormatter.*ofPattern*("dd-MM-yyyy HH:mm:ss z")**;** // Get current time in GMT(London (UK))  
 ZonedDateTime gmtTime = ZonedDateTime.*now*(ZoneId.*of*("GMT"))**;** System.*out*.println("Current Time in GMT: " + gmtTime.format(formatter))**;** // Get current time in IST (Asia/Kolkata)  
 ZonedDateTime istTime = ZonedDateTime.*now*(ZoneId.*of*("Asia/Kolkata"))**;** System.*out*.println("Current Time in IST: " + istTime.format(formatter))**;** // Get current time in PST (America/Los\_Angeles)  
 ZonedDateTime pstTime = ZonedDateTime.*now*(ZoneId.*of*("America/Los\_Angeles"))**;** System.*out*.println("Current Time in PST: " + pstTime.format(formatter))**;  
  
 //problem 2**  
 // Step 1: Get date input from user  
 Scanner scanner=new Scanner(System.*in*)**;** System.*out*.print("Enter a date (dd-MM-yyyy): ")**;** String input = scanner.nextLine()**;** // Step 2: Parse input to LocalDate  
 DateTimeFormatter formatter1 = DateTimeFormatter.*ofPattern*("dd-MM-yyyy")**;** LocalDate inputDate = LocalDate.*parse*(input**,** formatter1)**;** // Step 3: Add 7 days, 1 month, and 2 years  
 LocalDate addedDate = inputDate.plusDays(**7**).plusMonths(**1**).plusYears(**2**)**;** // Step 4: Subtract 3 weeks  
 LocalDate finalDate = addedDate.minusWeeks(**3**)**;** // Step 5: Display results  
 System.*out*.println("Original Date: " + inputDate.format(formatter1))**;** System.*out*.println("After Adding 7 Days, 1 Month, 2 Years: " + addedDate.format(formatter1))**;** System.*out*.println("After Subtracting 3 Weeks: " + finalDate.format(formatter1))**;  
  
  
 //problem-3**  
 // Get the current date  
 LocalDate currentDate = LocalDate.*now*()**;** // Define three different formatters  
 DateTimeFormatter format1 = DateTimeFormatter.*ofPattern*("dd/MM/yyyy")**;** DateTimeFormatter format2 = DateTimeFormatter.*ofPattern*("yyyy-MM-dd")**;** DateTimeFormatter format3 = DateTimeFormatter.*ofPattern*("EEE, MMM dd, yyyy")**;** // Display the date in different formats  
 System.*out*.println("Format 1 (dd/MM/yyyy): " + currentDate.format(format1))**;** System.*out*.println("Format 2 (yyyy-MM-dd): " + currentDate.format(format2))**;** System.*out*.println("Format 3 (EEE, MMM dd, yyyy): " + currentDate.format(format3))**;  
  
   
 //problem 4**  
 DateTimeFormatter formatter3 = DateTimeFormatter.*ofPattern*("dd-MM-yyyy")**;** // Step 1: Get first date input  
 System.*out*.print("Enter the first date (dd-MM-yyyy): ")**;** String firstInput = scanner.nextLine()**;** LocalDate firstDate = LocalDate.*parse*(firstInput**,** formatter3)**;** // Step 2: Get second date input  
 System.*out*.print("Enter the second date (dd-MM-yyyy): ")**;** String secondInput = scanner.nextLine()**;** LocalDate secondDate = LocalDate.*parse*(secondInput**,** formatter3)**;** // Step 3: Compare the dates  
 if (firstDate.isBefore(secondDate)) {  
 System.*out*.println("First date is before the second date.")**;** } else if (firstDate.isAfter(secondDate)) {  
 System.*out*.println("First date is after the second date.")**;** } else {  
 System.*out*.println("Both dates are the same.")**;** }  
  
  
 }  
  
}

**Problem Statements for Practice**

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)

Hint: Use ZonedDateTime and ZoneId to work with different time zones.

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.

Hint: Use LocalDate.plusDays(), plusMonths(), plusYears(), and

minusWeeks() methods.

3. 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

Hint: Use DateTimeFormatter with custom patterns for date formatting.

4. 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.

Hint: Use isBefore(), isAfter(), and isEqual() methods from the LocalDate

class.