Exercice Date

Ecrire un programme Java pour créer un objet Date en utilisant la classe Calendar

```
import java.util.*;
public class Exercise1 {
    public static void main(String[] args)
    {
     int year = 2016;
     int month = 0; // January
     int date = 1;

     Calendar cal = Calendar.getInstance();
     // Sets the given calendar field value and the time value
     // (millisecond offset from the Epoch) of this Calendar undefined.
     cal.clear();
     System.out.println();
     cal.set(Calendar.YEAR, year);
     cal.set(Calendar.MONTH, month);
     cal.set(Calendar.DATE, date);

     System.out.println(cal.getTime());
     System.out.println();
 }
}
```

Écrire un programme Java pour obtenir et afficher les informations (année, mois, jour, heure, minute) d'un calendrier par

défaut

```
import java.util.*;
public class Exercise2 {
 public static void main(String[] args)
    {
       // Create a default calendar
        Calendar cal = Calendar.getInstance();
       // Get and display information of current date from the calendar:
    System.out.println();
        System.out.println("Year: " + cal.get(Calendar.YEAR));
        System.out.println("Month: " + cal.get(Calendar.MONTH));
        System.out.println("Day: " + cal.get(Calendar.DATE));
        System.out.println("Hour: " + cal.get(Calendar.HOUR));
        System.out.println("Minute: " + cal.get(Calendar.MINUTE));
    System.out.println();
    }
}
```

Écrivez un programme Java pour obtenir la valeur maximale de l'année, du mois, de la semaine et de la date à partir de la date actuelle d'un calendrier par défaut.

```
import java.util.*;
public class Exercise3 {
public static void main(String[] args)
    {
     // Create a default calendar
        Calendar cal = Calendar.getInstance();
System.out.println();
System.out.println("\nCurrent Date and Time:" + cal.getTime());
int actualMaxYear = cal.getActualMaximum(Calendar.YEAR);
int actualMaxMonth = cal.getActualMaximum(Calendar.MONTH);
int actualMaxWeek = cal.getActualMaximum(Calendar.WEEK_OF_YEAR);
```

```
    int actualMaxDate = cal.getActualMaximum(Calendar.DATE);
    
    System.out.println("Actual Maximum Year: "+actualMaxYear);
    System.out.println("Actual Maximum Month: "+actualMaxMonth);
    System.out.println("Actual Maximum Week of Year: "+actualMaxWeek);
    System.out.println("Actual Maximum Date: "+actualMaxDate+"\n");
    System.out.println();
    
  }
}
```

Écrivez un programme Java pour obtenir la valeur minimale de l'année, du mois, de la semaine et de la date à partir de la date actuelle d'un calendrier par défaut.

```
import java.util.*;
public class Exercise4 {
 public static void main(String[] args)
    {
     // Create a default calendar
        Calendar cal = Calendar.getInstance();
    System.out.println();
    System.out.println("\nCurrent Date and Time:" + cal.getTime());
    int actualMaxYear = cal.getActualMinimum(Calendar.YEAR);
    int actualMaxMonth = cal.getActualMinimum(Calendar.MONTH);
    int actualMaxWeek = cal.getActualMinimum(Calendar.WEEK_OF_YEAR);
    int actualMaxDate = cal.getActualMinimum(Calendar.DATE);
    
    System.out.println("Actual Minimum Year: "+actualMaxYear);
    System.out.println("Actual Minimum Month: "+actualMaxMonth);
    System.out.println("Actual Minimum Week of Year: "+actualMaxWeek);
    System.out.println("Actual Minimum Date: "+actualMaxDate+"\n");
    System.out.println();
  }
```

```
}
```

Écrivez un programme Java pour obtenir l'heure actuelle à New York.

```
import java.util.*;
public class Exercise5 {
   public static void main(String[] args)
    {
     Calendar calNewYork = Calendar.getInstance();
     calNewYork.setTimeZone(TimeZone.getTimeZone("America/New_York"));
    System.out.println();
      System.out.println("Time in New York: " + calNewYork.get(Calendar.HOUR_OF_DAY) + ":"
        + calNewYork.get(Calendar.MINUTE)+":"+calNewYork.get(Calendar.SECOND));
   System.out.println();
    }
}
```

Écrivez un programme Java pour obtenir la date et l'heure complètes actuelles

```
import java.util.*;
public class Exercise6 {
   public static void main(String[] args)
    {
   Calendar now = Calendar.getInstance();
    System.out.println();
    System.out.println("Current full date and time is : " + (now.get(Calendar.MONTH) + 1) + "-
"
        + now.get(Calendar.DATE) + "-" + now.get(Calendar.YEAR) + " "
        + now.get(Calendar.HOUR_OF_DAY) + ":" + now.get(Calendar.MINUTE) + ":"
        + now.get(Calendar.SECOND) + "." + now.get(Calendar.MILLISECOND));
System.out.println();
    }
}
```

Écrivez un programme Java pour obtenir le dernier jour du mois en cours.

```
import java.util.*;
public class Exercise7 {
   public static void main(String[] args)
    {
    //Gets a calendar using the default time zone and locale.
        Calendar calendar = Calendar.getInstance();
   System.out.println();
   System.out.println(calendar.getActualMaximum(Calendar.DAY_OF_MONTH));
   System.out.println();
   }
}
```

Écrivez un programme Java pour obtenir l'heure locale actuelle.

```
import java.time.*;
public class Exercise14 {
   public static void main(String[] args)
    {
        LocalTime time = LocalTime.now();
        System.out.println();
        // in hour, minutes, seconds, nano seconds
        System.out.println("Local time now : " + time);
        System.out.println();
    }
}
```

Écrivez un programme Java pour ajouter quelques heures à l'heure actuelle.

```
import java.time.*;
public class Exercise15 {
   public static void main(String[] args)
    {
      LocalTime time = LocalTime.now();
      // adding four hours
      LocalTime newTime = time.plusHours(4);
      System.out.println();
      System.out.println("Time after 2 hours : " + newTime);
      System.out.println();
    }
}
```

Écrivez un programme Java pour obtenir une date après 2 semaines.

```
import java.util.*;
public class Exercise16 {
   public static void main(String[] args)
    {
     //two weeks
     int noOfDays = 14;
     Calendar cal = Calendar.getInstance();
     Date cdate = cal.getTime();
     cal.add(Calendar.DAY_OF_YEAR, noOfDays);
     Date date = cal.getTime();
     System.out.println("\nCurrent Date: " + cdate+"\n");
     System.out.println("Day after two weeks: " + date+"\n");
    }
}
```

Écrivez un programme Java pour obtenir une date avant et après 1 an par rapport à la date actuelle.

```
import java.util.*;
public class Exercise17 {
   public static void main(String[] args)
    {
      Calendar cal = Calendar.getInstance();
      Date cdate = cal.getTime();
      // get next year
      cal.add(Calendar.YEAR, 1);
      Date nyear = cal.getTime();
      //get previous year
      cal.add(Calendar.YEAR, -2);
      Date pyear = cal.getTime();
      System.out.println("\nCurrent Date : " + cdate);
      System.out.println("\nDate before 1 year : " + pyear);
      System.out.println("\nDate after 1 year  : " + nyear+"\n");
    }
}
```

Écrivez un programme Java pour vérifier qu'une année est bissextile ou non.

```
public class Exercise18 {
   public static void main(String[] args)
    {
     //year to leap year or not
       int year = 2016;
       System.out.println();
       if((year % 400 == 0) || ((year % 4 == 0) && (year % 100 != 0)))
               System.out.println("Year " + year + " is a leap year");
       else
               System.out.println("Year " + year + " is not a leap year");
               System.out.println();
    }
}
```

Écrivez un programme Java pour obtenir l'année et les mois entre deux dates.

```
import java.time.*;
public class Exercise19 {
   public static void main(String[] args)
    {
     LocalDate today = LocalDate.now();
     LocalDate userday = LocalDate.of(2015, Month.MAY, 15);
     Period diff = Period.between(userday, today);
     System.out.println("\nDifference between "+ userday +" and "+ today +": "
     + diff.getYears() +" Year(s) and "+ diff.getMonths() +" Month()s\n");
    }
}
```

Écrivez un programme Java pour obtenir l'horodatage actuel.

```
import java.time.*;
public class Exercise20 {
   public static void main(String[] args)
    {
    Instant timestamp = Instant.now();
    System.out.println("\nCurrent Timestamp: " + timestamp+"\n");
    }
}
```

Écrivez un programme Java pour obtenir l'heure actuelle dans tous les fuseaux horaires disponibles.

```
import java.time.*;
public class Exercise21 {
   public static void main(String[] args)
    {
    ZoneId.SHORT_IDS.keySet().
    stream().forEach(
    zoneKey ->System.out.println(" "+ ZoneId.of( ZoneId.SHORT_IDS.get( zoneKey ) ) +": "+
LocalDateTime.now(ZoneId.of(ZoneId.SHORT_IDS.get( zoneKey ) ) ) ) );
    }
}
```

Écrivez un programme Java pour obtenir les dates 10 jours avant et après aujourd'hui.

```
import java.time.*;
public class Exercise22 {
   public static void main(String[] args)
    {
     LocalDate today = LocalDate.now();
     System.out.println("\nCurrent Date: "+today);
     System.out.println("10 days before today will be "+today.plusDays(-10));
     System.out.println("10 days after today will be "+today.plusDays(10)+"\n");
   }
}
```

Écrivez un programme Java pour obtenir les mois restants dans l'année.

```
import java.time.*;
import java.time.temporal.TemporalAdjusters;
public class Exercise23 {
   public static void main(String[] args)
    {
```

```
      LocalDate today = LocalDate.now();
      LocalDate lastDayOfYear = today.with(TemporalAdjusters.lastDayOfYear());
      Period period = today.until(lastDayOfYear);
      System.out.println();
      System.out.println("Months remaining in the year: "+period.getMonths());
      System.out.println();
   }
}
```

Écrivez un programme Java pour afficher les dates dans les formats suivants.

LocalDate=2016-09-16
16::Sep::2016
LocalDateTime=2016-09-16T11:46:01.455
16::Sep::2016 11::46::01
Instant=2016-09-16T06:16:01.456Z
Default format after parsing = 2014-04-27T21:39:48

```
import java.time.Instant;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;

public class DateParseFormatExercise24 {

	public static void main(String[] args) {
		
		//Format examples
		LocalDate date = LocalDate.now();
		//default format
		System.out.println("\nDefault format of LocalDate="+date);
		//specific format
		System.out.println(date.format(DateTimeFormatter.ofPattern("d::MMM::uuuu")));

		LocalDateTime dateTime = LocalDateTime.now();
		//default format
		System.out.println("Default format of LocalDateTime="+dateTime);
```

```
    //specific format
    System.out.println(dateTime.format(DateTimeFormatter.ofPattern("d::MMM::uuuu HH::mm::ss")));
        Instant timestamp = Instant.now();
    //default format
    System.out.println("Default format of Instant="+timestamp);

    //Parse examples
    LocalDateTime dt = LocalDateTime.parse("27::Apr::2014 21::39::48",
            DateTimeFormatter.ofPattern("d::MMM::uuuu HH::mm::ss"));
    System.out.println("Default format after parsing = "+dt+"\n");
  }

}
```

Écrivez un programme Java pour afficher les informations de date et d'heure avant quelques heures et minutes à partir de la date et de l'heure actuelles

```
import java.time.*;

public class DateParseFormatExample28 {

  public static void main(String[] args) {

    // Before 7 houres and 30 minutes
     LocalDateTime dateTime = LocalDateTime.now().minusHours(5).minusMinutes(30);
      System.out.println("\nCurrent Date and Time:   " + LocalDateTime.now());
      System.out.println("Before 7 hours and 30 minutes: " + dateTime+"\n");
  }

}
```

Écrivez un programme Java pour convertir une chaîne en date.

```
import java.time.*;
import java.util.*;
import java.time.format.DateTimeFormatter;

public class MainEx29 {
  public static void main(String[] args) {
   String string = "May 1, 2016";
   DateTimeFormatter formatter = DateTimeFormatter.ofPattern("MMMM d, yyyy", Locale.ENGLISH);
   LocalDate date = LocalDate.parse(string, formatter);
   System.out.println();
   System.out.println(date);
   System.out.println();
  }
}
```

Ecrire un programme Java pour calculer la différence entre deux dates (année, mois, jours)

```
import java.time.*;
import java.util.*;

public class Exercise1 {
   public static void main(String[] args)
    {
        LocalDate pdate = LocalDate.of(2012, 01, 01);
        LocalDate now = LocalDate.now();

        Period diff = Period.between(pdate, now);

     System.out.printf("\nDifference is %d years, %d months and %d days old\n\n",
                    diff.getYears(), diff.getMonths(), diff.getDays());
```

```
  }
}
```

Écrivez un programme Java pour calculer la différence entre deux dates (heures, minutes, milli, secondes et nano).

```
import java.time.*;
import java.util.*;

public class Exercise31 {
   public static void main(String[] args)
    {
     LocalDateTime dateTime = LocalDateTime.of(2016, 9, 16, 0, 0);
     LocalDateTime dateTime2 = LocalDateTime.now();
     int diffInNano = java.time.Duration.between(dateTime, dateTime2).getNano();
     long diffInSeconds = java.time.Duration.between(dateTime, dateTime2).getSeconds();
     long diffInMilli = java.time.Duration.between(dateTime, dateTime2).toMillis();
     long diffInMinutes = java.time.Duration.between(dateTime, dateTime2).toMinutes();
     long diffInHours = java.time.Duration.between(dateTime, dateTime2).toHours();
     System.out.printf("\nDifference is %d Hours, %d Minutes, %d Milli, %d Seconds and %d
Nano\n\n",
                   diffInHours, diffInMinutes, diffInMilli, diffInSeconds, diffInNano );

  }
}
```

Écrivez un programme Java pour extraire la date, l'heure de la chaîne de date.

```
import java.util.*;
import java.text.*;
```

```
public class Exercise35 {
  public static void main(String[] args)
   {
    try {
      String originalString = "2016-07-14 09:00:02";
      Date date = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss").parse(originalString);
      String newstr = new SimpleDateFormat("MM/dd/yyyy, H:mm:ss").format(date);
       System.out.println("\n"+newstr+"\n");
      }
    catch (ParseException e) {
    //Handle exception here
     e.printStackTrace();
     }
    }
}
```

Écrivez un programme Java pour convertir un horodatage Unix en date en Java.

```
import java.util.*;
import java.text.*;

public class Exercise36 {
  public static void main(String[] args)
   {
   //Unix seconds
   long unix_seconds = 1372339860;
   //convert seconds to milliseconds
   Date date = new Date(unix_seconds*1000L);
   // format of the date
   SimpleDateFormat jdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss z");
   jdf.setTimeZone(TimeZone.getTimeZone("GMT-4"));
   String java_date = jdf.format(date);
   System.out.println("\n"+java_date+"\n");
   }
}
```

Écrivez un programme Java pour calculer la différence entre deux dates en jours.

```
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.Period;
import java.time.ZoneId;
import java.time.ZonedDateTime;
import java.time.temporal.ChronoUnit;
import java.util.Calendar;
import java.util.concurrent.TimeUnit;

public class Main {

    public static void main(String[] args) {

        System.out.println("\nBefore JDK 8:");

        Calendar cal1 = Calendar.getInstance();
        cal1.set(2019, 0, 1);
        Calendar cal2 = Calendar.getInstance();
        cal2.set(2020, 2, 1);

        System.out.println("\nDate/Calendar case: " + cal1.getTime() + " <-> " +
cal2.getTime());

        long inMs = Math.abs(cal1.getTimeInMillis() - cal2.getTimeInMillis());
        long inDays = Math.abs(TimeUnit.DAYS.convert(inMs, TimeUnit.MILLISECONDS));

        System.out.println("Difference in milliseconds is: " + inMs);
        System.out.println("Difference in days is: " + inDays);

        System.out.println("\nStarting with JDK 8:");

        LocalDate ld1 = LocalDate.of(2019, 1, 1);
        LocalDate ld2 = LocalDate.of(2020, 3, 1);

        System.out.println("\nLocalDate case: " + ld1 + " <-> " + ld2);
```

```
        long between_In_Days = Math.abs(ChronoUnit.DAYS.between(ld1, ld2));
        long between_In_Months = Math.abs(ChronoUnit.MONTHS.between(ld1, ld2));
        long between_In_Years = Math.abs(ChronoUnit.YEARS.between(ld1, ld2));
        long until_In_Days = Math.abs(ld1.until(ld2, ChronoUnit.DAYS));
        long until_In_Months = Math.abs(ld1.until(ld2, ChronoUnit.MONTHS));
        long until_In_Years = Math.abs(ld1.until(ld2, ChronoUnit.YEARS));
        Period period = ld1.until(ld2);
        System.out.println("Difference as Period: "
                + period.getYears() + "y" + period.getMonths() + "m" + period.getDays() +
"d");
        System.out.println("Difference in days is via between(): " + between_In_Days);
        System.out.println("Difference in months is via between(): " + between_In_Months);
        System.out.println("Difference in years is via between(): " + between_In_Years);
        System.out.println("Difference in days is via until(): " + until_In_Days);
        System.out.println("Difference in months is via until(): " + until_In_Months);
        System.out.println("Difference in years is via until(): " + until_In_Years);

        LocalDateTime ldt1 = LocalDateTime.of(2019, 1, 1, 22, 15, 15);
        LocalDateTime ldt2 = LocalDateTime.of(2020, 1, 1, 23, 15, 15);

        System.out.println("\nLocalDateTime case: " + ldt1 + " <-> " + ldt2);

        long betweenInMinutesWithoutZone = Math.abs(ChronoUnit.MINUTES.between(ldt1, ldt2));
        long untilInMinutesWithoutZone = Math.abs(ldt1.until(ldt2, ChronoUnit.HOURS));
        System.out.println("Difference in minutes without zone: " +
betweenInMinutesWithoutZone);
        System.out.println("Difference in hours without zone: " + untilInMinutesWithoutZone);

        System.out.println("\nZonedDateTime case:");

        ZonedDateTime zdt1 = ldt1.atZone(ZoneId.of("Europe/Bucharest"));
        ZonedDateTime zdt2 =
zdt1.withZoneSameInstant(ZoneId.of("Australia/Perth")).plusHours(1);
        ZonedDateTime zdt3 = ldt2.atZone(ZoneId.of("Australia/Perth"));

        long betweenInMinutesWithZone12 = Math.abs(ChronoUnit.MINUTES.between(zdt1, zdt2));
        long untilInHoursWithZone12 = Math.abs(zdt1.until(zdt2, ChronoUnit.HOURS));
        long betweenInMinutesWithZone13 = Math.abs(ChronoUnit.MINUTES.between(zdt1, zdt3));
        long untilInHoursWithZone13 = Math.abs(zdt1.until(zdt3, ChronoUnit.HOURS));
```

```
        System.out.println("Europe/Bucharest: " + zdt1 + " <-> Australia/Perth: " + zdt2);
        System.out.println("Difference in minutes with zone (same instant): " +
betweenInMinutesWithZone12);
        System.out.println("Difference in hours with zone (same instant): " +
untilInHoursWithZone12);

        System.out.println("\nEurope/Bucharest: " + zdt1 + " <-> Australia/Perth: " + zdt3);
        System.out.println("Difference in minutes with zone (not same instant): " +
betweenInMinutesWithZone13);
        System.out.println("Difference in hours with zone: " + untilInHoursWithZone13);
    }
}
```

Écrivez un programme Java pour imprimer aaaa-MM-jj, HH:mm:ss, aaaa-MM-jj HH:mm:ss, E MMM aaaa HH:mm:ss.SSSZ et HH:mm:ss,Z, modèle de format pour la date et l'heure

```
import java.text.SimpleDateFormat;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.LocalTime;
import java.time.OffsetDateTime;
import java.time.ZonedDateTime;
import java.time.OffsetTime;
import java.time.format.DateTimeFormatter;
import java.util.Date;
public class Main {
public static void main(String[] args) {
               String result;
       //yyyy-MM-dd
       LocalDate localDate = LocalDate.now();
```

```
        DateTimeFormatter formatterLocalDate = DateTimeFormatter.ofPattern("yyyy-MM-dd");
        result = formatterLocalDate.format(localDate);
        System.out.println("\nyyyy-MM-dd: " + result);
        // HH:mm:ss
        LocalTime localTime = LocalTime.now();
        DateTimeFormatter formatterLocalTime = DateTimeFormatter.ofPattern("HH:mm:ss");
        result = formatterLocalTime.format(localTime);
                        System.out.println("\nHH:mm:ss: " + result);
        // yyyy-MM-dd HH:mm:ss
        LocalDateTime localDateTime = LocalDateTime.now();
        DateTimeFormatter formatterLocalDateTime =
                DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");
        result = formatterLocalDateTime.format(localDateTime);
                        System.out.println("\nyyyy-MM-dd HH:mm:ss: " + result);
        // E MMM yyyy HH:mm:ss.SSSZ
        ZonedDateTime zonedDateTime = ZonedDateTime.now();
        DateTimeFormatter formatterZonedDateTime =
                DateTimeFormatter.ofPattern("E MMM yyyy HH:mm:ss.SSSZ");
        result = formatterZonedDateTime.format(zonedDateTime);
                        System.out.println("\nE MMM yyyy HH:mm:ss.SSSZ: " + result);
        // HH:mm:ss,Z
        OffsetTime offsetTime = OffsetTime.now();
        DateTimeFormatter formatterOffsetTime =
                DateTimeFormatter.ofPattern("HH:mm:ss,Z");
        result = formatterOffsetTime.format(offsetTime);
                        System.out.println("\nHH:mm:ss,Z: " + result);
      }
}
```

Écrivez un programme Java pour afficher la date et l'heure locales combinées dans un seul objet.

```
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.LocalTime;
```

```
import java.time.format.DateTimeFormatter;

public class Main {
    public static void main(String[] args) {
        LocalDate local_Dt = LocalDate.now();
        String localDateAsString = local_Dt
                .format(DateTimeFormatter.ofPattern("yyyy-MMM-dd"));
        System.out.println("Local Date: " + localDateAsString);
        LocalTime local_Time = LocalTime.now();
        String localTimeAsString = local_Time
                .format(DateTimeFormatter.ofPattern("hh:mm:ss a"));
        System.out.println("Local Time: " + localTimeAsString);
        LocalDateTime localDateTime = LocalDateTime.of(local_Dt, local_Time);
        String localDateTimeAsString = localDateTime
                .format(DateTimeFormatter.ofPattern("yyyy-MMM-dd hh:mm:ss a"));
        System.out.println("\nCombine local Date Time: " + localDateTimeAsString);
    }
}
```

Écrivez un programme Java pour définir une période de temps en utilisant des valeurs basées sur la date (Period) et une durée en utilisant des valeurs basées sur le temps (Duration).

```
import java.time.LocalDate;
import java.time.Period;

public class Main {

    public static void main(String[] args) {

        Period fromDays = Period.ofDays(120);
        System.out.println("Period from days: " + fromDays);
```

```
        Period periodFromUnits = Period.of(2000, 11, 24);
        System.out.println("Period from units: " + periodFromUnits);

        LocalDate localDate = LocalDate.now();
        Period periodFromLocalDate = Period.of(localDate.getYear(),
                localDate.getMonthValue(), localDate.getDayOfMonth());
        System.out.println("Period from LocalDate: " + periodFromLocalDate);

        Period periodFromString = Period.parse("P2019Y2M25D");
        System.out.println("Period from String: " + periodFromString);

        LocalDate startLocalDate = LocalDate.of(2018, 3, 12);
        LocalDate endLocalDate = LocalDate.of(2019, 7, 20);
        Period periodBetween = Period.between(startLocalDate, endLocalDate);
        System.out.println("\nBetween " + startLocalDate + " and "
                + endLocalDate + " there are " + periodBetween.getYears() + " year(s)");
        System.out.println("Between " + startLocalDate + " and "
                + endLocalDate + " there are " + periodBetween.getMonths() + " month(s)");
        System.out.println("Between " + startLocalDate + " and "
                + endLocalDate + " there are " + periodBetween.getDays() + " days(s)");

        System.out.println("Expressed as y:m:d: " + periodToYMD(periodBetween));

        System.out.println(startLocalDate + " is after "
                + endLocalDate + " ? " + periodBetween.isNegative());

        Period periodBetweenPlus1Year = periodBetween.plusYears(1L);
        System.out.println("\n" + periodBetween + " has " + periodBetween.getYears() + "
year,"
                + " after adding one year it has " + periodBetweenPlus1Year.getYears());

        Period p1 = Period.ofDays(5);
        Period p2 = Period.ofDays(20);
        Period p1p2 = p1.plus(p2);
        System.out.println(p1 + "+" + p2 + "=" + p1p2);
    }

    private static String periodToYMD(Period period) {

        if (period == null) {
            // or throw IllegalArgumentException
```

```
            return "";
        }

        StringBuilder sb = new StringBuilder();
        sb.append(period.getYears())
                .append("y:")
                .append(period.getMonths())
                .append("m:")
                .append(period.getDays())
                .append("d");

        return sb.toString();
    }
}
```

Revision #2
Created 19 June 2022 10:55:28 by ggpilou2
Updated 19 June 2022 11:24:51 by ggpilou2