LocalDate & LocalTime Classes

What is the LocalDate class

- A class that provides the ability to use years, months, and days to represent dates numerically

- The class is in the java.time package

- Oracle documentation:
  https://docs.oracle.com/javase/8/docs/api/java/time/LocalDate.html

What is the LocalTime class

- A class that provides the ability to use hours, minutes, and seconds to represent times

- The class is in the java.time package

- Oracle documentation:
  https://docs.oracle.com/javase/8/docs/api/java/time/LocalTime.html

What is the LocalDateTime class

- A class that provides the ability to use years, months, days, hours, minutes, and seconds to represent both dates and times as one object. It is a combination of LocalDate and LocalTime.

- The class is in the java.time package

- Oracle documentation:
  https://docs.oracle.com/javase/8/docs/api/java/time/LocalDateTime.html

Using the LocalDate class

1) Declare LocalDate reference
   Ex: LocalDate date;

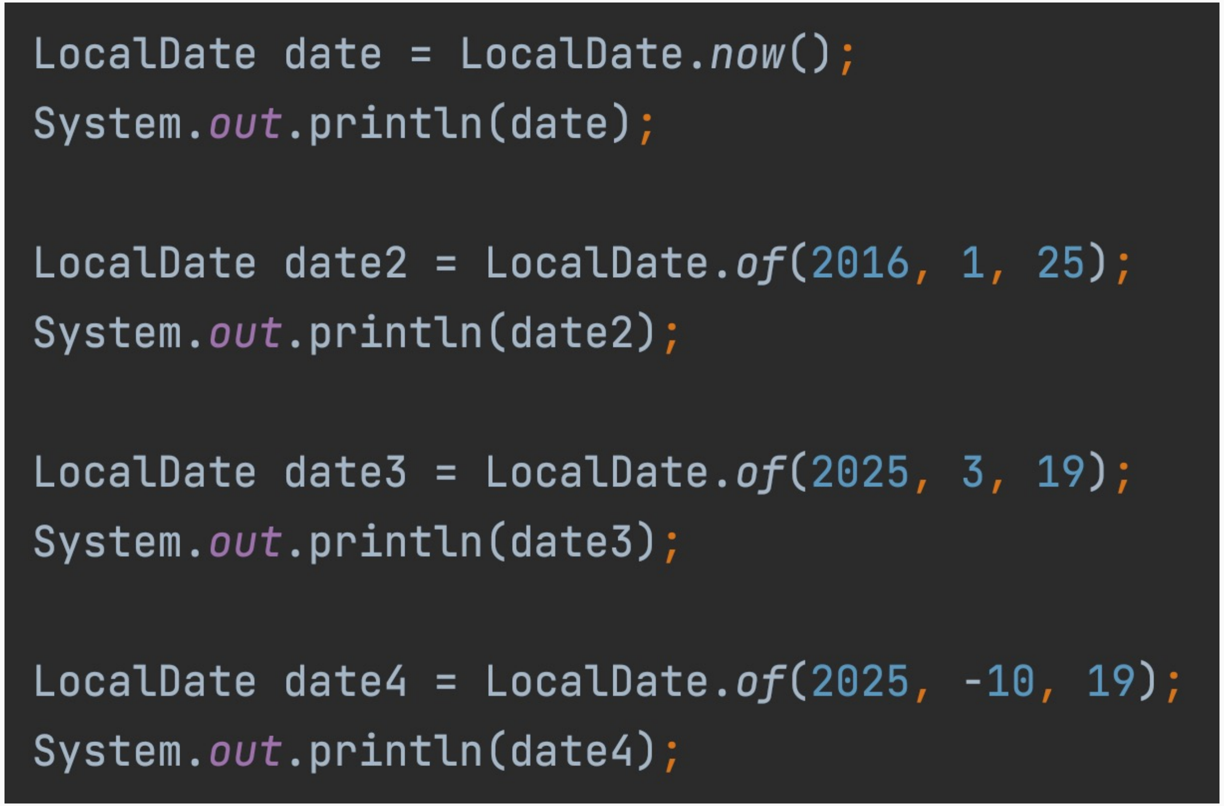
2) Use one of the following methods to get a LocalDate object:

   - LocalDate.now(): this method will get the current date and return an object with those values. If this date is printed by default, it will be in the year-month-day format

   - LocalDate.of(int, int, int): this method will accept 3 int values as the argument which will create and return a date object. The order of the arguments will correspond to the specific value being assigned. The first argument will be the year, the second argument will be the month and the last argument will be the day of the month

Note: LocalDate objects are immutable

Examples of LocalDate



```java
LocalDate date = LocalDate.now();
System.out.println(date);

LocalDate date2 = LocalDate.of(2016, 1, 25);
System.out.println(date2);

LocalDate date3 = LocalDate.of(2025, 3, 19);
System.out.println(date3);

LocalDate date4 = LocalDate.of(2025, -10, 19);
System.out.println(date4);
```

Outputs:

If the current date is December 5th, 2021 the LocalDate would print as: 2021-12-05

2016-01-25

2025-03-19

Exception occurs in this usage because an invalid month number is given

Using the LocalTime class

1) Declare LocalTime reference
   Ex: LocalTime time;

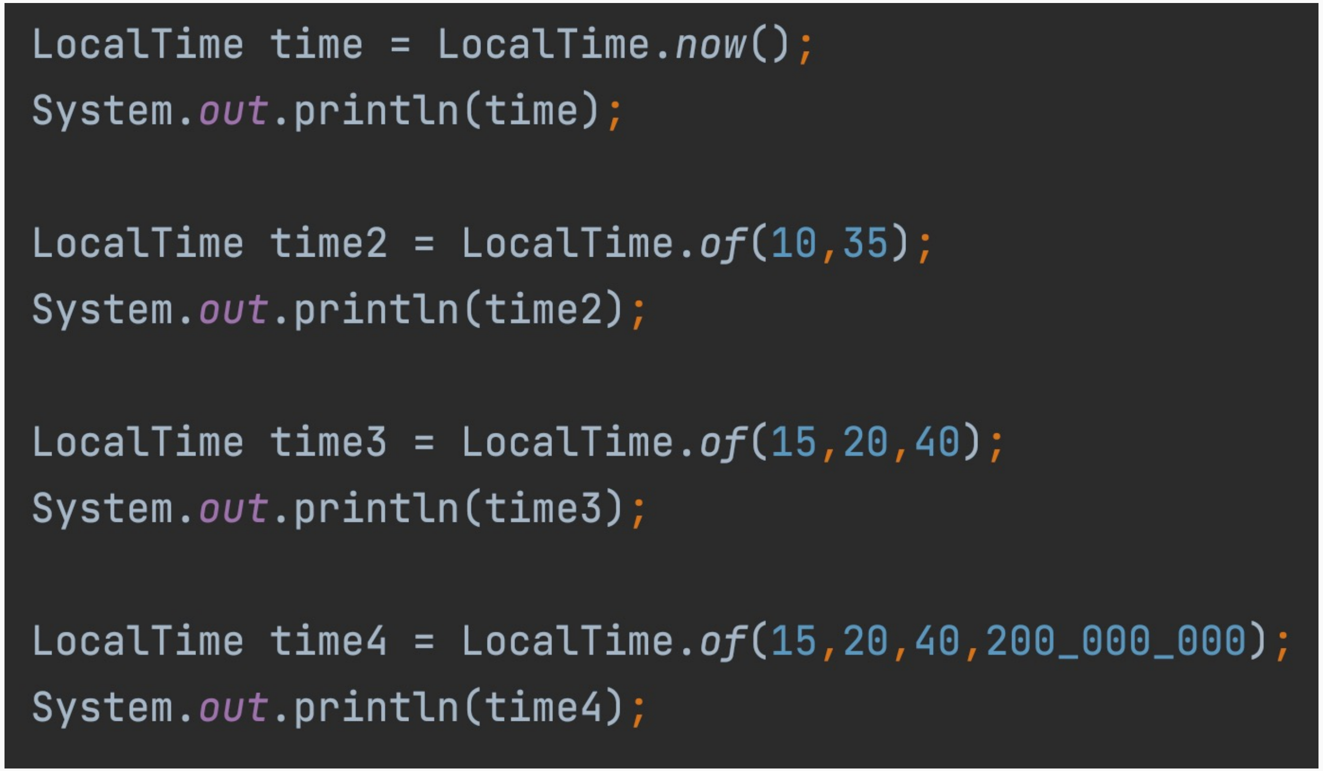
2) Use one of the following methods to get a LocalTime object:

   - LocalTime.now(): this method will get the current time and return an object with those values. If this time is printed by default, it will be in the hours: minutes: seconds. nanoseconds format where the hours are in 24-hour format

   - LocalTime.of(): this method is overloaded to work with different time inputs.
     - of(hours, minutes): returns an object with only hours and minutes
     - of(hours, minutes, seconds): returns an object with hours, minutes, and seconds
     - of(hours, minutes, seconds, nanoseconds): returns an object with hours, minutes, seconds, and nanoseconds

Note: LocalTime objects are immutable and follow 24-hour format

Examples of LocalTime



```
LocalTime time = LocalTime.now();
System.out.println(time);

LocalTime time2 = LocalTime.of(10,35);
System.out.println(time2);

LocalTime time3 = LocalTime.of(15,20,40);
System.out.println(time3);

LocalTime time4 = LocalTime.of(15,20,40,200_000_000);
System.out.println(time4);
```

Outputs:

If the current time is 4:06 pm
it would give a time of:
16:06:18.493

10:35

15:20:40

15:20:40.200

CYDEO

Using the LocalDateTime class

1) Declare LocalDateTime reference

   Ex: LocalDateTime time;

2) Use one of the following methods to get a LocalDateTime object:

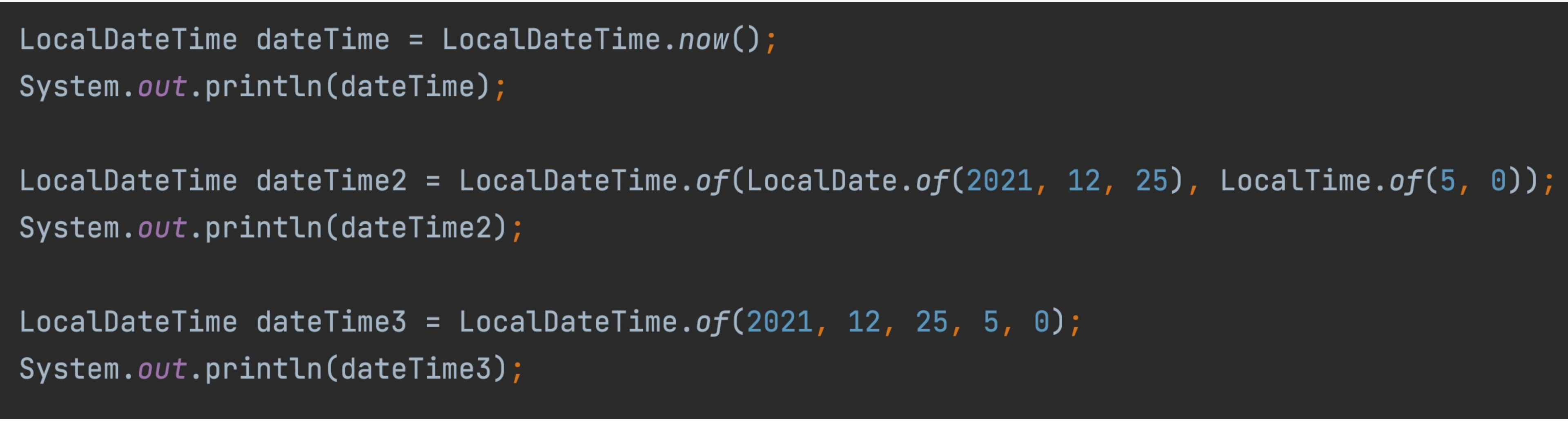
   - LocalDateTime.now(): this method will get the current date and time and return an object with those values. If this object is printed by default, it will be in the year-month-day‘T’ hours:minutes:seconds.nanoseconds format where the hours are in 24-hour format and a character ‘T’ is used to separate the date and time sections

   - LocalTime.of(): this method is overloaded to work with different inputs.
     - of(LocalDate, LocalTime)
     - of(years, months, days, hours, minutes)
     - of(years, months, days, hours, minutes, seconds)

     and other parameter combinations

Note: LocalDateTime objects are immutable and follow 24-hour format

Examples of LocalDateTime



```java
LocalDateTime dateTime = LocalDateTime.now();
System.out.println(dateTime);

LocalDateTime dateTime2 = LocalDateTime.of(LocalDate.of(2021, 12, 25), LocalTime.of(5, 0));
System.out.println(dateTime2);

LocalDateTime dateTime3 = LocalDateTime.of(2021, 12, 25, 5, 0);
System.out.println(dateTime3);
```

Outputs:

When the date is December 5th, 2021 and time is 4:52 pm

```
2021-12-05T16:52:56.381

2021-12-25T05:00
2021-12-25T05:00
```

What is the DateTimeFormatter class

- A class that provides the ability to format LocalDate, LocalTime, and LocalDateTime objects

- The class is in the java.time.format package

- Oracle documentation:
  https://docs.oracle.com/javase/8/docs/api/java/time/format/DateTimeFormatter.html

Using the DateTimeFormatter class

1) Declare a DateTimeFormatter object with a pattern

```
Ex: DateTimeFormatter formatter = DateTimeFormatter.ofPattern(”pattern here");
```

→ Patterns can be combined with in a String to create the pattern of the objects (See next page)

2) Use the format method from the LocalDate, LocalTime, or LocalDateTime objects and give the DateTimeFormatter object as the argument of the format method

```
LocalDate date = LocalDate.now();
System.out.println(date.format(formatter));

LocalTime time = LocalTime.now();
System.out.println(time.format(formatter));

LocalDateTime dt = LocalDateTime.now();
System.out.println(dt.format(formatter));
```

DateTimeFormatter patterns - Dates

Years:

yy: 2 digit year number
yyyy: 4 digit year number

Months:

M: month number (from 1-12) – doesn’t include 0 for number of 0-9
MM: month number (from 1-12) – includes 0 for number of 0-9
MMM: month abbreviation (Ex: Jan, Dec, etc)
MMMM: full month name (Ex: January, December, etc)

Days:

d: single digit day number
dd: double digit day number, includes 0 if it is a digit from 0-9

DateTimeFormatter patterns - Times

Hours:

h: hour format for am or pm in 1-12 – doesn’t include 0 for number of 0-9
hh: hour format for am or pm in 1-12 – includes 0 for number of 0-9
HH: hour format for 24-hour format – doesn’t include 0 for number of 0-9
HH: hour format for 24-hour format – includes 0 for number of 0-9

Minutes:

m: minute format – doesn’t include 0 for number of 0-9
mm: minute format – includes 0 for number of 0-9

Seconds:

s: second format – doesn’t include 0 for number of 0-9
ss: second format – includes 0 for number of 0-9

AM or PM

a: adding this to the format will put AM or PM based on the time

Examples of Formatted objects

```java
System.out.println(LocalDate.of(2021,3,5).format(DateTimeFormatter.ofPattern("MM/dd/yy")));
System.out.println(LocalDate.of(2021,3,5).format(DateTimeFormatter.ofPattern("MM/dd/yyyy")));
System.out.println(LocalDate.of(2021,3,5).format(DateTimeFormatter.ofPattern("MMM/dd/yy")));
System.out.println(LocalDate.of(2021,3,5).format(DateTimeFormatter.ofPattern("MMMM/dd/yyyy")));

System.out.println(LocalTime.of(14, 35).format(DateTimeFormatter.ofPattern("hh:mm")));
System.out.println(LocalTime.of(14, 35).format(DateTimeFormatter.ofPattern("HH:mm")));
System.out.println(LocalTime.of(14, 35).format(DateTimeFormatter.ofPattern("hh:mm a")));

System.out.println(LocalDateTime.of(2021,12,25,6,30).format(DateTimeFormatter.ofPattern("MMM dd,
                                                                                  yyyy | hh:mm a")));
```

Outputs:

```
03/05/21
03/05/2021
Mar/05/21
March/05/2021
```

```
02:35
14:35
02:35 PM
Dec 25, 2021 | 06:30 AM
```

CYDEO

Other methods

- The LocalDate, LocalTime, and LocalDateTime objects have many methods to use, here is some other useful ones:

  getDayOfWeek(), getDayOfYear(), getMonth, getMinute(), isBefore(), isAfter(), minusDays, plusDays(), plusYears(), minusYears(), parse(), etc

Remember the objects are immutable, so any changes applied do not change the original object

Practices

- Create a program that will ask the user to enter their birthday. Ask them what month they were born (number), which day, and which year and print their birthday together in a nice format

- Create a program that will check if the campus is open. Take the current time and compare it with campus open times: open from 7 am – 6 pm

- Create an ArrayList of LocalDates and store all federal holidays of 2022

- [Advanced] Create a Map that stores everyone's birthdays. Key will be the LocalDate and the values will be an ArrayList of Person object (You will need to create a Person class or use an existing one). Every person that has the same birthday will be on the same ArrayList value corresponding to the key of the LocalDate