Chapter 5 Date and Calendar

Working with dates and times is essential in many real-world applications. In this lecture, you will learn how to use Python's modules `datetime` and `calendar` to work with dates and calendars effectively.

1. The datetime Module

Python has a module named `datetime` that provides functions and methods to work with dates and times.

Example 01: Create dates.

```
import datetime

# Create two date objects
date1 = datetime.date(2025, 10, 31) # must follow the order: year, month, day
date2 = datetime.date(day = 3, month = 12, year = 2024) # can be in any order

# Display the dates
print("date1:", date1)
print("date2:", date2)
```

Output

```
date1: 2025-10-31
date2: 2024-12-03
```

The `today()` method defined in the `date` class is used to return a date object containing the current date.

Example 02: Get the current date

```
import datetime

# Get the current date
date = datetime.date.today()

# Display the date
print(date)
```

Output

```
2025-07-20
```

You can use `year`, `month` and `day` attributes to extract the year, month and day as integers from a date.

Example 03: Get the year, month and day from a date object.

```
import datetime

# Get the current date
date = datetime.date.today()

# Display year, month and day of the date
print("year:", date.year)
print("month:", date.month)
print("day:", date.day)
```

Output

```
year: 2025
month: 7
day: 20
```

Example 04: Display date as day/month/year

```
import datetime

# Get the current date
date = datetime.date.today()

# Display the date as day/month/year
print("date:", str(date.day) + "/" + str(date.month) + "/" + str(date.year))
```

Output

```
date: 20/7/2025
```

Example 05: Add date.

```
import datetime

# Create a data object
date1 = datetime.date(2025, 1, 25)

# Add 20 days to date1
date2 = date1 + datetime.timedelta(20)

# Display the date
print(date2)
```

Output

```
2025-02-14
```

Example 06: Subtract date.

```
import datetime

# Create a date object
date1 = datetime.date.today()

# Subtract 60 days from date1
date2 = date1 - datetime.timedelta(60)

# Display the date
print(date2)
```

Output

```
2025-05-21
```

Example 07: Find the number of days between two dates.

```
import datetime

# Create two date objects
date1 = datetime.date(2018, 7, 12)
date2 = datetime.date(2017, 12, 23)

# Calcuate the number of days different between two dates
number_of_days = (date1 - date2).days

# Display the result
print("The number of days difference is", number_of_days, "days")
```

Output

```
The number of days difference is 201 days
```

The `replace()` method is used to create a new date object by replacing one or more of its components (like year, month, day, etc.) — without changing the original object.

Example 08: Using `replace()` method to change the dates.

```
import datetime

# Create two date objects
date1 = datetime.date(2018, 7, 12)
date2 = datetime.date(2017, 12, 23)

# Change the dates
date1 = date1.replace(year = 2025, day = 1)
date2 = date2.replace(month = 2)

# Display the result
print(date1)
print(date2)
```

Output

```
2025-07-01
2017-02-23
```

If you are trying to replace a date to another date which does not exist, it will raise an error:

```
date = datetime.date(2018, 7, 31)
date = date.replace(month = 2)      # error
```

ValueError: day is out of range for month

You can compare dates using comparison operators: `==`, `!=`, `<`, `<=`, `>`, and `>=`.

Example 09: Compare dates

```
import datetime

date1 = datetime.date(2018, 7, 12)
date2 = datetime.date(2017, 12, 23)

print(date1 == date2)
print(date1 != date2)
print(date1 > date2)
print(date1 < date2)
```

Output

```
False
True
True
False
```

Example 10: Get a date input from the user

```
import datetime

# Ask the user to enter a date as string
date = input("Enter a date (YYYY-MM-DD): ")

# Convert the string the user entered into date
date = datetime.datetime.strptime(date, "%Y-%m-%d").date()

# Display the result
print(date)
```

Output

```
Enter a date (YYYY-MM-DD): 2025-07-20
2025-07-20
```

2. The calendar Module

Python has a **calendar** module that allows us to output calendars, and provides additional useful functions related to the calendar.

Example 11: Display a calendar of a given year and month.

```
import calendar

# Display a calendar
print(calendar.month(2025, 8))
```

Output

```
    August 2025
Mo Tu We Th Fr Sa Su
                1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31
```

Example 12: Display calendar of a given year

```
import calendar

# Display the calendar of year 2025
print(calendar.calendar(2025))
```

Output

```
                                  2025

      January                   February                  March
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
       1  2  3  4  5                      1  2                      1  2
 6  7  8  9 10 11 12       3  4  5  6  7  8  9       3  4  5  6  7  8  9
13 14 15 16 17 18 19      10 11 12 13 14 15 16      10 11 12 13 14 15 16
20 21 22 23 24 25 26      17 18 19 20 21 22 23      17 18 19 20 21 22 23
27 28 29 30 31            24 25 26 27 28            24 25 26 27 28 29 30
                                                    31

      April                     May                       June
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
    1  2  3  4  5  6                1  2  3  4                         1
 7  8  9 10 11 12 13       5  6  7  8  9 10 11       2  3  4  5  6  7  8
14 15 16 17 18 19 20      12 13 14 15 16 17 18       9 10 11 12 13 14 15
21 22 23 24 25 26 27      19 20 21 22 23 24 25      16 17 18 19 20 21 22
28 29 30                  26 27 28 29 30 31         23 24 25 26 27 28 29
                                                    30

      July                      August                    September
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
    1  2  3  4  5  6                   1  2  3       1  2  3  4  5  6  7
 7  8  9 10 11 12 13       4  5  6  7  8  9 10       8  9 10 11 12 13 14
14 15 16 17 18 19 20      11 12 13 14 15 16 17      15 16 17 18 19 20 21
21 22 23 24 25 26 27      18 19 20 21 22 23 24      22 23 24 25 26 27 28
28 29 30 31               25 26 27 28 29 30 31      29 30

      October                   November                  December
Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su      Mo Tu We Th Fr Sa Su
       1  2  3  4  5                      1  2       1  2  3  4  5  6  7
 6  7  8  9 10 11 12       3  4  5  6  7  8  9       8  9 10 11 12 13 14
13 14 15 16 17 18 19      10 11 12 13 14 15 16      15 16 17 18 19 20 21
20 21 22 23 24 25 26      17 18 19 20 21 22 23      22 23 24 25 26 27 28
27 28 29 30 31            24 25 26 27 28 29 30      29 30 31
```

The `calendar` module has two lists: `day_name` and `month_name` as below:

```
day_name = ['Monday', 'Tuesday', ..., 'Sunday']
month_name = ['', 'January', 'February', ..., 'December']
```

To get the name of the day or month, you need to know its index, (0-6 ~ Monday-Sunday) and (1-12 ~ January-December).

Example 13: Get a day name of a given day's index and month's index.

```
import calendar

# Get the day name
name_of_day = calendar.day_name[2]

# Get the month name
name_of_month = calendar.month_name[8]

# Display the result
print(name_of_day)
print(name_of_month)
```

Output

```
Wednesday
August
```

To get the day index of a given date, you can use a method called `weekday()`.

Example 14: Get a day's index of a given date.

```
import datetime

date1 = datetime.date(2025, 8, 19) # create a date object

# Get the day index of the date
day_index = date1.weekday()  # returns (0-6 ~ Monday-Sunday)

# Display the result
print(day_index)
```

Output

```
1
```

Example 15: Get a day's index of a given date.

```
import calendar

# Get the day index of a date
day_index = calendar.weekday(2025, 8, 19) # returns (0-6 ~ Monday-Sunday)

# Display result
print(day_index)
```

Output

```
1
```

The `calendar.monthrange(year, month)` function returns a tuple of two values: `(first_weekday, total_days)`

Example 16: Get the first day and the total days in a given month of a year.

```
import calendar

# Get the first day's index and the total days of a given month of a year
first_day, total_days = calendar.monthrange(2025, 2)

# Display the results
print(first_day)
print(total_days)
```

Output

```
5
28
```

Exercises

Write the following program:

1. Calculate how many days remain until the next new year (January 1, 2026)
2. Ask the user to enter a month and year, and count how many Saturdays are in that month.
   Here is a sample run:

```
Enter year: 2025
Enter month (1-12): 8

The total number of Saturdays in August 2025 is 5
```

3. Ask the user for a month and year, and display all dates that are Mondays. Here is a sample run:

```
Enter year: 2025
Enter month (1-12):  8

Mondays in August 2025:
2025-08-04
2025-08-11
2025-08-18
2025-08-25
```

4. Generate five random dates in a year input by the user, and display the day of the week for each. Here is a sample run:

```
Enter a year: 2025

Random dates in 2025:
2025-02-06, Thursday
2025-07-08, Tuesday
2025-08-25, Monday
2025-01-06, Monday
2025-01-19, Sunday
```

5. Ask the user for their birthdate and calculate: How many days until their next birthday, and What day of the week their birthday falls on this year. Here is a sample run:

```
Enter your birthday (YYYY-MM-DD): 2000-8-1


Your next birthday is on a Friday.
Days until your next birthday is 12
```

6. Ask the user to enter two dates, and count the number of workdays (Mon–Fri) between the two dates. Here is a sample run:

```
Enter start date (YYYY-MM-DD): 2025-6-10
Enter end date (YYYY-MM-DD): 2025-7-5

Total workdays: 19
```