Intro to programming 7

Henri Vandendriessche
henri.vandendriessche@ens.fr

2023-11-13

Terminal cheat sheet reminder

- Bash commands to navigate directories
  - Print Working Directory. Print the path of the current directory

```
pwd
```

  - List all files of the current directory

```
ls folder
```

  - Moving into folder1 and subfolder2 at once.

```
cd folder1/subfolder2
```

  - Moving out of a directory

```
cd ..
```

  - Going back and forth in the directory tree

```
cd ../../folder1/subfolder1
```

  - Going back to the root directory

```
cd ~
```

- "**Tab**" to use the auto-completion
- **Ctrl + C** to stop a program execution
- "**Upper arrow**" to see last commands
- Many more bash commands to use...

Previously on Intro to Programming (Python)

- Data types:
  - integer
  - float
  - string
  - boolean
- **If**, **For** and **While** loops:
  - syntax
  - indentation
- Data collections:
  - list
  - tuple
  - set
  - dictionary
- Python Standard library
  - Python modules
  - Python built-in functions
- Functions:
  - Parameters and arguments
  - Return values
  - Scope of variable
- Building abstraction with :
  - Recursive functions
  - High order functions
- Read and write files

Today

- csv
- Date format
- Exercise based on a real data set

CSV

- Comma-separated-value (CSV) file is a text file that uses comma to delimit data entries.

CSV

- Comma-separated-value (CSV) file is a text file that uses comma to delimit data entries.
- The .csv format is simple and basic, commonly used when data needs to be compatible with various programs and applications.

CSV

- Comma-separated-value (CSV) file is a text file that uses comma to delimit data entries.
- The .csv format is simple and basic, commonly used when data needs to be compatible with various programs and applications.
- As a matter of fact you can change the separator and chose “semicolon”, “Tab” or “space” or any other character depending on your need.

CSV

- Comma-separated-value (CSV) file is a text file that uses comma to delimit data entries.
- The .csv format is simple and basic, commonly used when data needs to be compatible with various programs and applications.
- As a matter of fact you can change the separator and chose “semicolon”, “Tab” or “space” or any other character depending on your need.
- CSV files can be opened by any text editor, spreadsheet software like Excel, and many other programs.

CSV

- Comma-separated-value (CSV) file is a text file that uses comma to delimit data entries.
- The .csv format is simple and basic, commonly used when data needs to be compatible with various programs and applications.
- As a matter of fact you can change the separator and chose “semicolon”, “Tab” or “space” or any other character depending on your need.
- CSV files can be opened by any text editor, spreadsheet software like Excel, and many other programs.
- For instance, databases often support CSV as an export format.

CSV

- Comma-separated-value (CSV) file is a text file that uses comma to delimit data entries.
- The .csv format is simple and basic, commonly used when data needs to be compatible with various programs and applications.
- As a matter of fact you can change the separator and chose “semicolon”, “Tab” or “space” or any other character depending on your need.
- CSV files can be opened by any text editor, spreadsheet software like Excel, and many other programs.
- For instance, databases often support CSV as an export format.
- example:

```
Name,Email,Phone,Address
John Doe,johndoe@example.com,123-456-7890,123 Fake Street
Michel Dupont,michel.dupont@test.com,098-765-4321,321 Ghost Avenue
```

Date format 1/3

- Python itself doesn't have a built-in date format, but it provides a module for handling dates and times called datetime.

Date format 1/3

- Python itself doesn't have a built-in date format, but it provides a module for handling dates and times called datetime.
- The datetime module documentation can be found at: https://docs.python.org/3/library/datetime.html

Date format 1/3

- Python itself doesn't have a built-in date format, but it provides a module for handling dates and times called datetime.
- The datetime module documentation can be found at: https://docs.python.org/3/library/datetime.html
- This module offers classes for manipulating dates and times.

Date format 1/3

- Python itself doesn't have a built-in date format, but it provides a module for handling dates and times called datetime.
- The datetime module documentation can be found at: https://docs.python.org/3/library/datetime.html
- This module offers classes for manipulating dates and times.
- The datetime module includes various objects, such as:

Date format 1/3

- Python itself doesn't have a built-in date format, but it provides a module for handling dates and times called datetime.
- The datetime module documentation can be found at: https://docs.python.org/3/library/datetime.html
- This module offers classes for manipulating dates and times.
- The datetime module includes various objects, such as:
  - **timedelta** object represents a duration, the difference between two dates or times.

Date format 1/3

- Python itself doesn't have a built-in date format, but it provides a module for handling dates and times called datetime.
- The datetime module documentation can be found at: https://docs.python.org/3/library/datetime.html
- This module offers classes for manipulating dates and times.
- The datetime module includes various objects, such as:
  - **timedelta** object represents a duration, the difference between two dates or times.
  - **date** object represents a date (year, month and day) in a calendar

Date format 1/3

- Python itself doesn't have a built-in date format, but it provides a module for handling dates and times called datetime.
- The datetime module documentation can be found at: https://docs.python.org/3/library/datetime.html
- This module offers classes for manipulating dates and times.
- The datetime module includes various objects, such as:
  - **timedelta** object represents a duration, the difference between two dates or times.
  - **date** object represents a date (year, month and day) in a calendar
  - **datetime** object is a single object containing all the information from a date object and a time object

Date format 1/3

- Python itself doesn't have a built-in date format, but it provides a module for handling dates and times called datetime.
- The datetime module documentation can be found at: https://docs.python.org/3/library/datetime.html
- This module offers classes for manipulating dates and times.
- The datetime module includes various objects, such as:
  - **timedelta** object represents a duration, the difference between two dates or times.
  - **date** object represents a date (year, month and day) in a calendar
  - **datetime** object is a single object containing all the information from a date object and a time object
  - **time** object represents a (local) time of day, independent of any particular day

Date format 1/3

- Python itself doesn't have a built-in date format, but it provides a module for handling dates and times called datetime.
- The datetime module documentation can be found at: https://docs.python.org/3/library/datetime.html
- This module offers classes for manipulating dates and times.
- The datetime module includes various objects, such as:
  - **timedelta** object represents a duration, the difference between two dates or times.
  - **date** object represents a date (year, month and day) in a calendar
  - **datetime** object is a single object containing all the information from a date object and a time object
  - **time** object represents a (local) time of day, independent of any particular day
- Example 1

```
from datetime import *
date1 = date(2022,11,8)
date2 = date(2022,12,24)
difference = date2 - date1
print(difference , "Before Christmas")
```

```
## 46 days, 0:00:00 Before Christmas
```

```
print(type(date1))
```

```
## <class 'datetime.date'>
```

```
print(type(difference))
```

Date format 2/3

- Example 2

```
from datetime import *

time = datetime.now()
print(time)
```

```
## 2023-11-14 00:01:57.184908
```

```
time2 = datetime.now()
print(time2)
```

```
## 2023-11-14 00:01:57.189595
```

```
if time > time2:
  print("time is posterior to time2")
elif time2 > time:
  print("time2 is posterior to time")
```

```
## time2 is posterior to time
```

Date format 3/3

- Example 3

```
from datetime import *

time = datetime(2022,11,8,14,30,10)
print(time)

time2 = date(2022,11,8)
print(time2)


if time > time2:
  print("time is posterior to time2")
elif time2 > time:
  print("time2 is posterior to time")
elif time == time2:
    print("They are the same date")
```

TypeError: can't compare datetime.datetime to datetime.date

Exercise

- We are gonna work today on the open data regarding location of cameras / video protection in Paris (2019)
- From the government website https://www.data.gouv.fr open and save the data in a csv file from the following link:

https://www.data.gouv.fr/fr/datasets/r/e6add12b-fa2e-437c-9f4c-b8195616d39a

- Save the file in .csv
- Exercise 1: Return the number of cameras in Paris
- Exercise 2: Print all information about cameras in the 5th arrondissement
- Exercise 3: Number of cameras in each arrondissement
- Exercise 4: Print arrondissement with the highest number of cameras
- Exercise 5: Date of first camera installed in Paris
- Exercise 6: Number of all the camera installed the first date