Group Project (2021-2022)

> **Rules:** *The projects will be done by groups of 4 (or 5) students. You will need to upload exactly 5 files. The python scripts for the projects will be named exclusively* `stockTrends.py` *and* `pokemonYearbook.py`*, respectively. The automatically generated pokemon « yearbook » must be in PDF format and will be named exclusively* `PYearbook.pdf`*. The log file should be a plain text file named exclusively* `PYlog.txt`*. A requirement file (* `requirements.txt` *) is also demanded. Requirement files can be generated automatically (your task is to find out how). A* `requirements.txt` *example is provided at the end of this document. Any change in file names will be penalised. Other files besides the scripts, the « Yearbook », the log file and the requirements file will not be considered. The files must be uploaded on the platform before the deadline. The deadline will be clearly marked on the upload section created for this project. Any delay will be sanctioned with 0/20 for the missing uploads.*

1. Stock Trends

General description:

Acquiring stock data for a period of time and visualising them represent important aspects for financial data analysis. Thus, this project aims to load stock prices data, for a specific company, during a specific period of time, and to generate a graph that would illustrate the trends.

Input:

- A distant text file containing the company name, the stock acronym, the beginning date, the end date and the time frequency https://adrianchifu.com/teachings/AMSE/MAG1/stocktrends/stock_parameters.txt

> **Hint**: *Check the file structure and display intermediary prints in order to figure out how to extract the necessary data.*

Command line arguments:

- the URL for the distant text file.

> *Check the command line interaction example below.*

Output:

- The graph, which will be generated directly on the screen (no file output).

> **Hint**: *Check out the* `matplotlib` *module.*

Check out the example below:

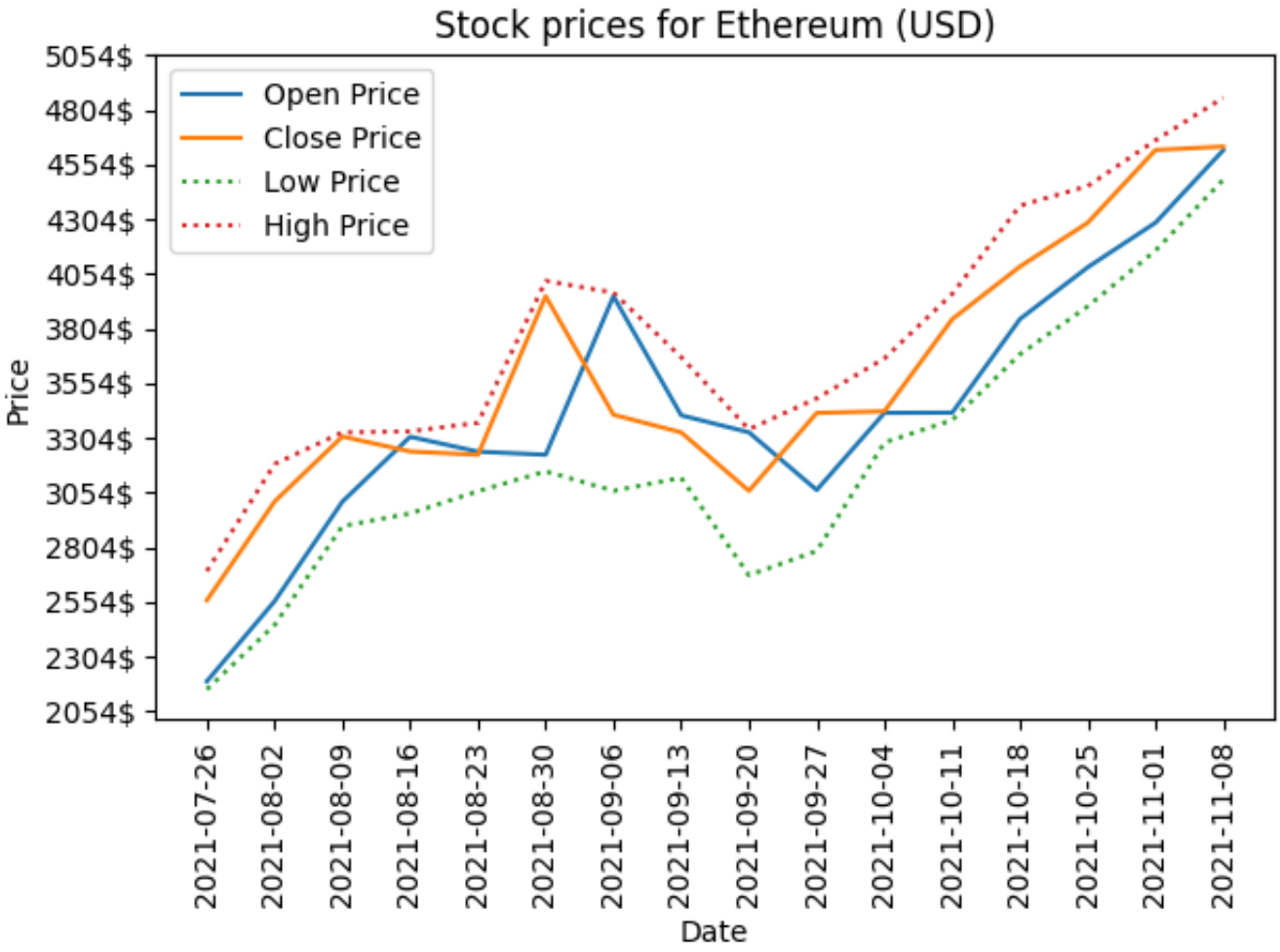

Figure 1. *Stock graph example*

Instructions:

- To obtain the data, you will need the **Yahoo Financials** python API.

> ***Hint:*** *Check out the GitHub page for the API https://github.com/JECSand/yahoofinancials.*

- You should read the input parameters (company name, stock acronym, begin date, end date and frequency) **exclusively** from the distant text file (no hardcoded input variables in your code).
- You should plot the open, closing, low and high prices, with the corresponding legends. The low and the high prices should be in dotted lines. Every plotted line should have a different colour.
- You should pay attention to the graph title, legend, x and y axis labels, and so on. In conclusion, you should try to obtain the graph from Figure 1.

> ***Hint:*** *Check out the following functions from* `matplotlib` : `ylabel(), xlabel(), xticks(), yticks(), legend(), tight_layout()` *and* `title()` .

- You should pay attention to the conversion of timestamps into dates.

> **Hint:** *Check out the* `time` *and* `datetime` *modules.*

- When obtaining data from the API, you should pay attention to the frequency specified in the distant input file (it may be weekly, daily, etc.).

Concepts to keep in mind:

- Installing python modules with `pip` (`pip install pandas`, for instance)
- Variables
- Input from file
- Reading from distant servers
- Loading data sets
- Generating graphs
- Command line arguments
- Data type conversion
- Displaying a graph
- Date conversion to string
- Functions
- Loops (while, for)
- Conditions

Modules to consider:

- `yahoofinancials`
- `sys`
- `matplotlib`
- `datetime`
- `time`
- `requests`
- `urllib.request`

Example of command line interaction:

```
python stockTrends.py https://adrianchifu.com/teachings/AMSE/MAG1/stocktrends/stock_parameters.txt
```

2. Pokemon Yearbook

General description:

Knowing how to deal with various file formats is a valued skill. The JSON format is widely used, especially by app APIs. Thus, practicing the manipulation of JSON files is crucial for data scientists.
Generating curated, proper and clear PDF reports is also important for data scientists. The reports may be, in most of the times, automatically generated.
Having all this in mind, this project aims at obtaining pokemon information from a distant JSON file and at automatically generating a pokemon « yearbook » in PDF format.

Input:

- A distant JSON file containing information about pokemons

https://adrianchifu.com/teachings/AMSE/MAG1/projpokedex/pokedex.json.

> **Hint**: *Check the file structure and the* `json` *module.*

- A distant ZIP file containing the necessary dependency files (the logo image, the image placeholder to replace missing pokemon images and two fonts that should be used for the pokemon names) https://adrianchifu.com/teachings/AMSE/MAG1/projpokedex/depend/data.zip.
- The path (including the filename) for the log file (the file where your program logs some information when running the code).
- The path (including the filename) for the PDF result file (the pokemon « yearbook »).

> **Hint**: *You may use relative or absolute paths.*

Command line arguments:

- The URL for the distant JSON file.
- The URL for the distant ZIP file.
- The path for the log file.
- The path for the PDF result file.

> *Check the command line interaction example below.*

Output:

- The PDF file containing the pokemon « yearbook ».

> **Hint**: *Check out the* `fpdf` *module.*

- The log file containing information when running the code (see the Instructions section).

Check out the PDF file sample below:

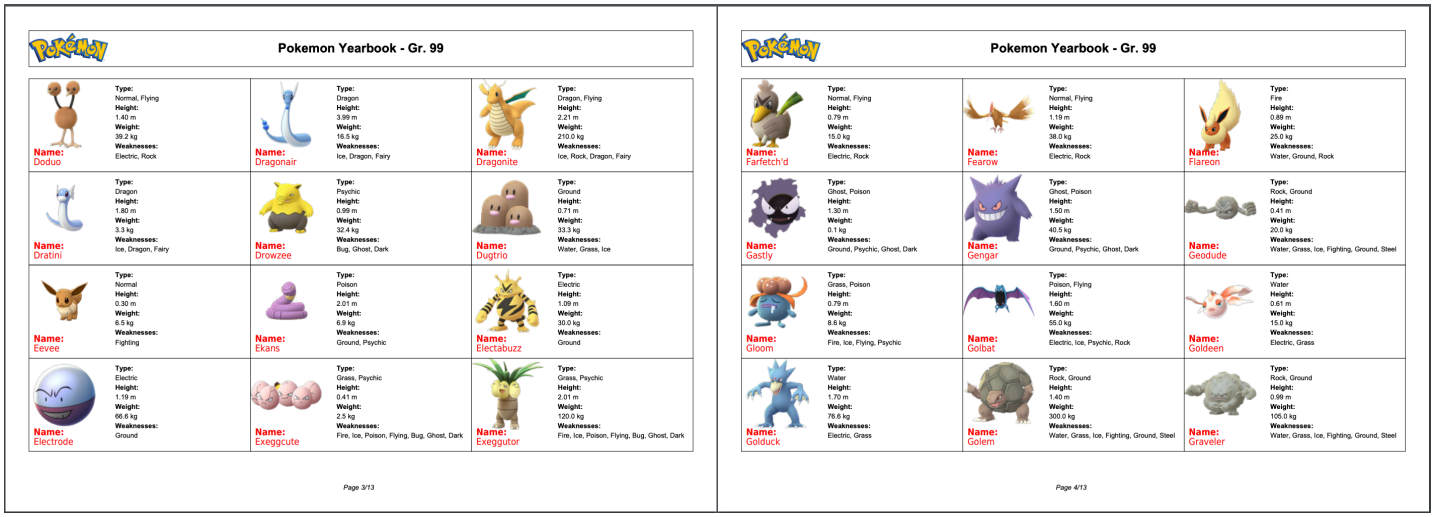

Figure 2. *Pokemon « yearbook » sample*

Instructions:

- PDF format instructions:
  - Header:
    - It should have a border
    - It should contain on the left hand side the Pokemon logo (`logo.png` image from the ZIP file)
    - It should contain the title « Pokemon Yearbook - Gr. » followed by your group number. The title font is Arial, with bold face, and size 15.
  - Footer:
    - It should contain the text « Page x/n », where x is the current page number and n the total number of pages. The font is Arial, with italic face, of size 8. The footer text should be centred.
  - The PDF pages should be in landscape mode, the unit should be millimetres (mm), and the format should be A4.
  - Pokemon images:
    - Should be square, of size of 30mm.
    - If there is a missing or unaccessible image link in the json file, the image must be replaced with the placeholder image (`placeholder.png` from the ZIP file).
  - The information text:
    - Every information label (such as name, type, height, weight, etc.) should be with bold face.
    - The pokemon names should be in DejaVu font (`DejaVuSansCondensed-Bold.ttf` and `DejaVuSansCondensed.ttf` from the ZIP file), written in red, and of size 11.
    - The other pieces of text have the font Arial, of size 8.
  - Every pokemon « card » should have a border.

> - ***Important!*** *If you manage to read the ZIP file and use the files inside without actually extracting the archive (using temp files, and so on), you will get bonus points.*
> - ***Important!*** *You may have one pokemon card by line (not displayed as a grid), without penalties. However, if you manage to obtain a grid of 4 x 3 pokemon cards (as displayed in Figure 2), you will get bonus points.*

- The pokemons must be sorted by their name.
- The log file should contain the total number of pokemons, the names of pokemon that is treated at any current time, what images links have not worked and a message when everything is done (see the log file sample below).

> ***Hint:*** *Check out the following functions from the* `fpdf` *module:* `add_page(), ln(), set_font(), cell(), text(), set_text_color(), add_font(), get_y(), get_x(), alias_nb_pages(), image()`.
> ***Hint:*** *Check out the following functions from the* `zipfile` *and* `tempfile` *modules:* `read(), NamedTemporaryFile(), write()`.

- You should try to obtain the same result as displayed in the Figure 2 (or getting very close).

Concepts to keep in mind:

- Installing python modules with `pip` (`pip install pandas`, for instance)
- Variables
- Input from files
- Writing to files

- Reading from distant servers
- Loading data sets
- Dealing with JSON files
- Dealing with ZIP archives
- Dealing with external fonts
- Generating PDF files with python
- Functions
- Formatted printing
- Loops (while, for)
- Conditions

Modules to consider:

- `json`
- `sys`
- `fpdf`
- `zipfile`
- `io`
- `requests`
- `urllib.request`
- `tempfile`
- `shutil`

Example of command line interaction:

```
python pokemonYearbook.py https://adrianchifu.com/teachings/AMSE/MAG1/projpokedex/pokedex.json https://adrianchifu.com/teachings/AMSE/MAG1/projpokedex/depend/data.zip PYlog.txt PYearbook.pdf
```

Log file sample:

```
We have 151pokemons.
Treating the pokemon: Abra
Treating the pokemon: Aerodactyl
Treating the pokemon: Alakazam
Treating the pokemon: Arbok
Treating the pokemon: Arcanine
Treating the pokemon: Articuno
Treating the pokemon: Beedrill
Treating the pokemon: Bellsprout
Treating the pokemon: Blastoise
Treating the pokemon: Bulbasaur
Treating the pokemon: Butterfree
Treating the pokemon: Caterpie
Treating the pokemon: Chansey
Treating the pokemon: Charizard
Treating the pokemon: Charmander
Treating the pokemon: Charmeleon

.....

Treating the pokemon: Shellder
Treating the pokemon: Slowbro
!!! The image file http://www.serebii.net/pokemongo/pokemon/-080.png does not exist. It was replaced with the placeholder... !!!Treating the pokemon: Slowpoke
Treating the pokemon: Snorlax
Treating the pokemon: Spearow
```

```
Treating the pokemon: Squirtle
Treating the pokemon: Starmie
Treating the pokemon: Staryu
Treating the pokemon: Tangela
Treating the pokemon: Tauros
Treating the pokemon: Tentacool
Treating the pokemon: Tentacruel
Treating the pokemon: Vaporeon
Treating the pokemon: Venomoth
Treating the pokemon: Venonat
Treating the pokemon: Venusaur
Treating the pokemon: Victreebel
Treating the pokemon: Vileplume
Treating the pokemon: Voltorb
Treating the pokemon: Vulpix
Treating the pokemon: Wartortle
Treating the pokemon: Weedle
Treating the pokemon: Weepinbell
Treating the pokemon: Weezing
Treating the pokemon: Wigglytuff
Treating the pokemon: Zapdos
Treating the pokemon: Zubat
All done.
```

The requirements.txt file example:

```
Django==1.4.1
distribute==0.6.27
dj-database-url==0.2.1
psycopg2==2.4.5
wsgiref==0.1.2
```