Intro to programming in Python - Getting started

CSC 109 Lyon College @ Batesville High

August 17, 2023

Contents

1 Entry survey

Welcome message:

> Welcome to CSC 109 Introduction to Programming in Python!
> We'll meet for our first session on Monday, August 21 at 11 AM.
> I don't know exactly where yet but I'll let you know before class.

It would be great if you could fill in a short survey for me before class. It will help me prepare and it'll warm you up for the course. Here is the link: `https://forms.gle/vhEpfn4Nt4uNJxfH6Links` to an external site.. At the end of the survey is a short (100 secs) introduction to Python followed by a few quiz questions.

The class will be **highly interactive** - you'll be programming almost the whole time. I've been told that you've got Chromebooks: please bring them with you to class.

I'm looking forward to meeting you and to teaching this class - my first ever at a high school! If you need to know anything else before Monday, shoot me an email (birkenkrahe@lyon.edu) or send me a text (501-422-4725).

Cheers, Marcus Birkenkrahe

PS. I am writing this in Canvas, Lyon's learning management system. You should try to login and you can take a look at the syllabus already.

2 Getting started

Course and instructor overview information. The original version of this lecture is on GitHub. There are changes for technical details[1].

- ☐ Who am I?: my first programming languages: BASIC, FORTRAN, C++; my first computer: TI/99. Learnt Python (properly) only this year.
- ☐ Why Python, why not?
- ☐ Which other languages do you know or have you heard about?
- ☐ What are you expectations for this course?
- ☐ What will you learn in this course?
- ☐ What will you learn in this course? - Programming paradigms, Python basics, data types, functions, scientific computing, plotting, data frame manipulation, control flow.

[1] In the summer 2023 course when the material was created, we used Google Colaboratory, replit.com and IDLE, while in this course we only use the online DataCamp Workspace platform. Instead of GitHub, we'll use Google Drive.

- ☐ How will you be evaluated? - 25% each for weekly assignments, monthly sprint reviews, weekly tests and one final exam.
- ☐ Which tools are we going to use? - Canvas, GitHub (old files), DataCamp assignments, DataCamp workspace (with AI assistance).
- ☐ Textbooks? See python.org for examples. I used mainly "Automate the boring stuff with Python" for the first iteration of this course.
- ☐ Infinite skills exercise: come up with three programs you would create if you had infinite programming skills and if you could build anything you wanted using any computer and Python.
- ☐ First assignment: "What are programming paradigms?" (DataCamp)
- ☐ Next: using DataCamp Workspace in the classroom and first Python program.

3 Introducing the workspace

Our integrated development and interactive notebook environment: using DataCamp Workspace in the classroom. See also: DataCamp webinar 'Using Workspace in the Classroom' (video recording, Aug 30, 2022). For more Python platforms, see the GitHub practice file: Command line, IDLE, Google Colaboratory, replit.com.

- ☐ Why DataCamp workspace? Want interactive, shareable "literate programming" notebooks that support different languages (though unlike Emacs, these online notebooks do not support different languages in one notebook).
- ☐ The workspace contains a bunch of assets: Jupyter notebook with an IPython shell (but for Python, R, and SQL); a Linux terminal (container); a coding editor (for source code); a Markdown editor (for documentation); an AI coding assistant; a debugger; many pre-installed packages.
- ☐ Can you think of any reasons not to make it too convenient to develop, test and execute your programs? (Sounds crazy, right?) Is development and analysis speed the only goal?

4 Workspace demo - setup

☐ Demo workspace: bit.ly/ws-unicorn - analysis of unicorn company data (companies with a valuation > USD 1 bn)

☐ The demo workspace opens in your workspace as `YourName/Code-Along: Unicorn Companies`. There are three parts to it:

1. Left sidebar with files, databases, and environment information.
2. Top menu for the workspace editor.
3. Text, code and output cells or blocks in the center. Text cells can be edited, commented upon, AI-assisted, or deleted. Code cells can be run, commented upon, AI-assised, or deleted.

☐ The purpose of the notebook format is that you can build a data report as you go along, including any idea or input, any code (in Python), and any output generated by your code.

☐ Within data science (including AI, machine learning, data analysis) this interactive notebook format is the gold standard for data story-telling - developing and presenting data-driven computational insights to a human audience.

☐ Jupyter notebook (`.ipynb` files) are an open source standard so there is no lock-in: you can import and export notebooks to and from this platform, and if you lose DataCamp access, no big deal.

☐ You can download and use an offline version of "Jupyter Lab" to your PC or work in another online environment like Google Colaboratory.

5 Workspace demo - code along

☐ The demo involves:

1. Importing CSV data as a data frame: `pd.read_csv`
2. Printing the unique data frame column names: `print`, `df.unique`
3. Cleaning the data frame: `df.replace`
4. Grouping the unicorns by category: `df.groupby`
5. Making a quick bar plot of the features: `plotly.express.bar`

5.1 Sidebar

- ☐ Open the `Files` menu in the sidebar: you see the notebook (open) and the CSV file.
- ☐ Click on the three dots next to the file name to see different options.
- ☐ The option `Query in new SQL cell` opens a new code cell with a SQL query command on all features (columns) of the CSV file. To execute this command, the CSV data are converted to a dataframe first.
- ☐ Create the SQL cell and run it, then press CTRL-Z twice to get back to the original notebook. You don't have to test the other option, `Load as DataFrame` because we're going to do this explicitly. But if you wanted to, this would create a Python cell with the commands to import the CSV data as a DataFrame.
- ☐ Click on the CSV file `unicorn_companies.csv` to open it.
- ☐ You see a headline with several features and 917 records of these features, one for each unicorn company.

5.2 Importing data

- ☐ Get back to your notebook. Next to the CSV file, select `Copy path to clipboard`. Click on `Files` to close the menu.
- ☐ In the first code cell, replace the three underscores in the `pd.read_csv` argument by the clipboard content.
- ☐ When you run this cell, either with the mouse or by entering CTRL-ENTER, the first 10 records of the DataFrame `df` and the headline with the features.
- ☐ Though the data look quite clean and appealing, a table view is not the best way to get an overview - there are many records.

5.3 Cleaning data

- ☐ For investment purposes, the `Category` column or feature is most interesting: this is the type of company. How many of these types are there?

- ☐ To print out all unique categories, we can use the `unique` function, which will return all unique entries in the `Category` column if we index the data frame accordingly:

  ```
  df["Category"].unique()
  ```

- ☐ To remove the extraneous information about data types in the printout (`array`), use a list comprehension:

  ```
  [print(i) for i in df["Category"].unique()];
  ```

- ☐ Here, we generate a new line with `print` for every unique record of the column. The semi-colon at the end stops a bunch of `None` values to be printed afterwards (an IPython artefact).

- ☐ You can see that there are duplicates because of typos (`Finttech`) and capitalization (`Artificial Intelligence`). Let's remove the ambiguities.

- ☐ We can use `df.replace` to replace one value by another value inside our dataframe. We do not need to repeat the command but we can append methods to one another:

  ```
  df_clean = df.replace(to_replace='Artificial intelligence',
                        value='Artificial Intelligence')\
              .replace(to_replace='Finttech',
                        value='Fintech')
  ```

5.4 Share editing rights

- ☐ Click on the sharing sign at the top and share **editing** access with your neighbor.

- ☐ Print the new dataframe `df_clean` in each other's notebooks by adding a new code block with the command `df_clean`.

- ☐ Once this is done, `Remove` access from your workspace for the other person.

5.5 Grouping data by column values

☐ We want to group records within the same industry to see how the unicorn companies are distributed across industries.

☐ We use three functions: `df.groupby()` on the `Category` column (ChatGPT summary), `size` to extract the number of records in each group, and `sort_values` to sort the result in descending order:

```
category_counts =\
    df_clean.groupby(by = 'Category', as_index=False)\
            .size()\
            .sort_values(by=['size'])
```

☐ `category_counts` is a pandas DataFrame with two columns, `Category`, and `size`.

5.6 Plotting data

☐ To plot the distribution, we use a bar chart that plots the frequency (counts) for each industry using `plotly.express`:

```
import plotly.express as px
px.bar(category_counts, x = 'Category', y = 'size')
```

☐ `plotly` is a plotting library, and `plotly.express` is a module to provide a range of plot types quickly (ChatGPT help and online doc).

5.7 Magic commands

☐ Create a new code block to check if IPython "magic commands" are supported, and enter `%whos`, which should generate a table of variables with their types and some environment information.

6 Workspace - Summary

- Workspace offers Jupyter notebooks in Python, R and SQL.
- WS Notebooks contain text, code, output ("literate programming").
- WS Notebooks have pre-installed libraries and sample data

- WS notebooks run an IPython shell
- WS notebooks can be downloaded/uploaded as `.ipynb` files
- WS notebooks can be shared with other [DataCamp] users
- WS notebooks can be published to [DataCamp] portfolios