Introduction to Computer Programming
-
Recitation 00

Spring 2020

Python

Why Python?

- **Friendly** and **readable high-level** programming language
- **Interpreted** programming language
- Well-known as a mainstream programming language

Already seen it?

- Web frameworks (Instagram, Dropbox, reddit, SourceForge, . . .)
- original Bittorrent client
- EVE online
- Embedded scripting in Maya and Blender

History

Python: snake or . . . ?

- named after Monty Python
- based on ABC (teaching/protoyping language)
- invented in 1991 by Guido van Rossum

Today

- 2 active branches of Python: **2.x** and **3.x**
- Python Enhancement Proposals (PEPs)
  - provide structure for evolution and dev of Python: add language features as well as conventions
- Python Software Foundation (PSF)
  - non-profit organization that provides organizational structure, governance and community building

Python 2.x or 3.x?

Which version will we use?

We are using **Python 3.x** (last version: Python 3.8.1, December 2019)

All of the in-class examples, exercises and homework solutions will target 3.x.
We are using it because:

- it is an excellent language for learning programming (readable, consistent, . . .)
- smooths over many of the perceived flaws and inconsistencies in Python 2.x
- is the future of Python

Writing Python programs

Python programs

Python programs are just **text files**!!!

- you might open a textfile in Notepad or TextEdit
- Python programs are usually text files with a **.py** extension (not .txt)
- Microsoft Word is not a text editor (.doc is not actually just text)

Then, which text editor?

2 possibilities:

- a text editor
  - Notepad, TextEdit are very simple text editors
  - some others (Emacs, . . .) offer features that aid in programming
- an IDE (Integrated Development Environment)
  - it adds features that help programmers

Writing Python programs

Important features for an IDE

- **Syntax highlighting** - to make your program more readable
- **Syntax checking** - helps preventing syntax error
- inline documentation/help
- code auto-completion

- an **integrated interactive shell** - to experiment statements
- an **integrated debugger** - to step through your program

IDLE

We will be using **IDLE**

- **IDLE** is an IDE (actually written in Pyhton!)
- means **I**ntegrated **D**eve**L**opment **E**nvironment
- but also refers to Eric Idle (member of Monthy Python)
- usually bounded with Python installation

Writing Python programs

Text Editor vs Interactive Python Shell

IDLE has 2 main features:

- a good **Text Editor**
- an **Interactive Python Shell**

Both allow you to write code! But have different purposes...

Interactive Python Shell (Interactive mode)

Interactive Python Shell is meant for quick experimentation

- code is entered **line-by-line**
- code is executed as **each line is entered!**
- immediate feedback
- you won't be able to save and re-run all the code you have entered

Writing Python programs

Text Editor (Script mode)

Text Editor is meant for writing programs

The usual workflow is:

- Start a new program or load an existing one
- Make changes to your program
- Save your changes
- Run your saved program

Installing Python

Existing installation?

Usually, Python is already installed on OSX or some Linux distributions (such as Ubuntu)
But it is probably Python 2.x

How to check?

- type `python -- version` in a terminal/commandline

terminal/commandline

- **Windows**: **cmd.exe** (Start → Run → type 'cmd.exe' and run)
- **OSX**: **terminal** (cmd+space to search spotlight → type 'terminal')
- **Ubuntu**: **terminal**

type `python -- version`

Installing Python

上海纽约大学
NYU SHANGHAI

If you don't have Python 3.x

- Windows/OSX:
  - go to http://python.org/download/
  - download the installer for your OS (**3.x** version, should be 3.8.1)
  - run it
- Ubuntu:
  - `sudo apt-get install python3`
  - `sudo apt-get install idle3`

Using IDLE - Demo

Starting IDLE

IDLE is the IDE we will use to read and write programs

When you first open IDLE, you should see a window titled **'Python Shell'**

- It is meant for entering and running programs **line-by-line**
- This **shell** can also be referred to as the **Python interpreter**, **console** or **interactive shell**

IDLE Text Editor

The **Text Editor** can be opened through:

- Create a new program: File → New File
- or modify an existing program: File → Open...

The new opened window is the **IDLE Text Editor**
You can **write a program** here, **save it** as a .py file and **run it**

Using IDLE - Demo

A first program

`'Hello World!'` is traditionally the first program you write when learning a new language.
It simply outputs `Hello World!`

- Create a new file
- type in `print("Hello World!")`
- Run the file (Run → Run Module)
  - you may be prompted to save your file
  - if you have not saved it yet, you will be prompted for a file name

Modifying a program

- If you closed your first program, reopen it
- Add a second line displaying another sentence
- Run it

Using IDLE - Demo

Making mistakes

- Add a third statement such as `print("Mistake?`
- The last parenthese is missing
- Run this program

- You should see a pop-up indicating an error
- The Text Editor detected a Syntax Error
- **Fix the problem** (How?)

Using IDLE - Demo

Making mistakes

The previous mistake in your program prevented it from running

- Replace the third statement by `print("Mistake?" + 2)`
- Run this program
- What happened?
  (look at the shell, it should indicate which line has a problem)
- Fix the problem

Using IDLE - Demo

Syntax Highlighting

In the **shell** or in the **Text Editor**, you should have noticed that some elements of you code have specific color

Different colors represents different syntactic elements

- strings (sequence of characters)
- built-in functions
- keywords (reserved words with a special meaning in Python)

Using IDLE - Demo

Help

You can access **help** directly from the **shell**

- Type `help()`
  You should see a help utility in the shell
- Type in this utility `print`
  It will display some basic help about *print* function
- In order to quit the utility, type `quit`

You can also access help directly for specific things by typing in the shell
`help(print)`

Additionally, you can check out the Python docs by going to Help → Python Docs

Exercises

Check `Recitation01.pdf`