Introduction to Programming for Physicists

Dr. Charanjit Kaur

charanjit.kaur@manchester.ac.uk

University of Manchester

MANCHESTER
1824
The University of Manchester

PHYS20161 Pre-lecture 1
Semester 1, 2023-24

The material is adapted from previous year’s course

Learning Objectives

After covering week 1 material, you will get to know

- Basic introduction to programming
- Introduction to python and how to get started
  - Data types and variables
  - Simple python programs
  - Lists in python

Plan

Every week, you will get multiple short videos and a single pdf file for the slides.

- Part 1: Introduction to Programming [Video 1]
- Part 2: Introduction to Python and getting started [Video 2]
- Part 3: Data Types and Variables [Video 3]
- Part 4: First Python Program [Video 4, `first_program.py`]
- Part 5: More Example Programs [Video 5, `hello_user.py, number_cube.py`]
- Part 6: Lists [Video 6, `list_examples.py`]

Part 1
Introduction to Programming

Programming

It is telling the computer to do something and how to do; it’s about giving instructions to achieve something:

- Automation - make receptive and/or large tasks almost trivial.
- Calculation - work something out quickly or work something out that can’t be done analytically (numerical analysis).
- Simulation - model a process (a system) and learn about it by watching how it behaves and/or tweaking parameters.
- Optimisation - work out the ‘best’ way of doing something or the ‘best’ set of parameters to achieve a desired result.

Programming Applications

- Computer/phone operating system, Apps.
- Websites: Google, YouTube, University website etc.
- Artificial Intelligence Tools: face recognition, virtual chatbots, recommendations, etc.
- Social Media Platforms
- Games
- Data Analysis (academia, finance, healthcare, business, etc).

Computer Program

- A program is a set of instructions for a computer to perform a particular task.
- Often task involves performing operations using a given information (data input) to produce a result (output).

- *How does a computer understand a program?*

  Computer-readable (machine) form

- *How does it store data in the memory?*

  In binary form (bits and bytes)

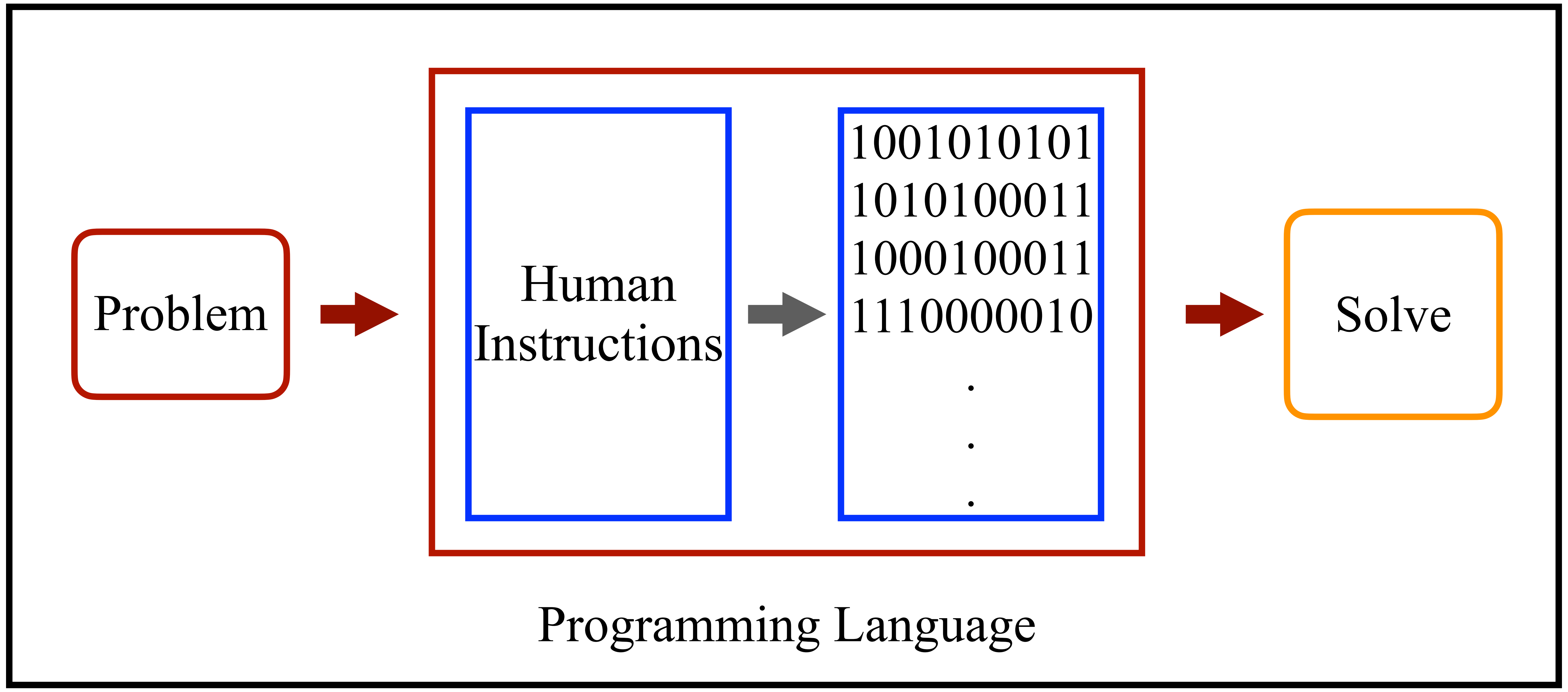
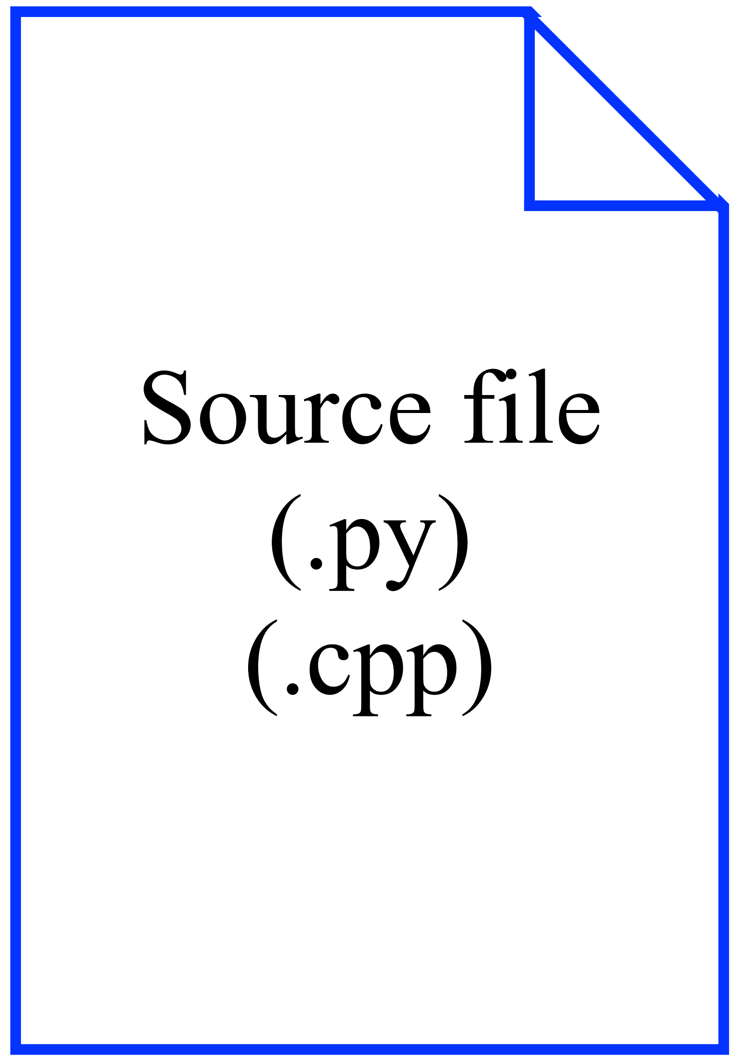


Figure adapted from wikiversity.org

Source Code

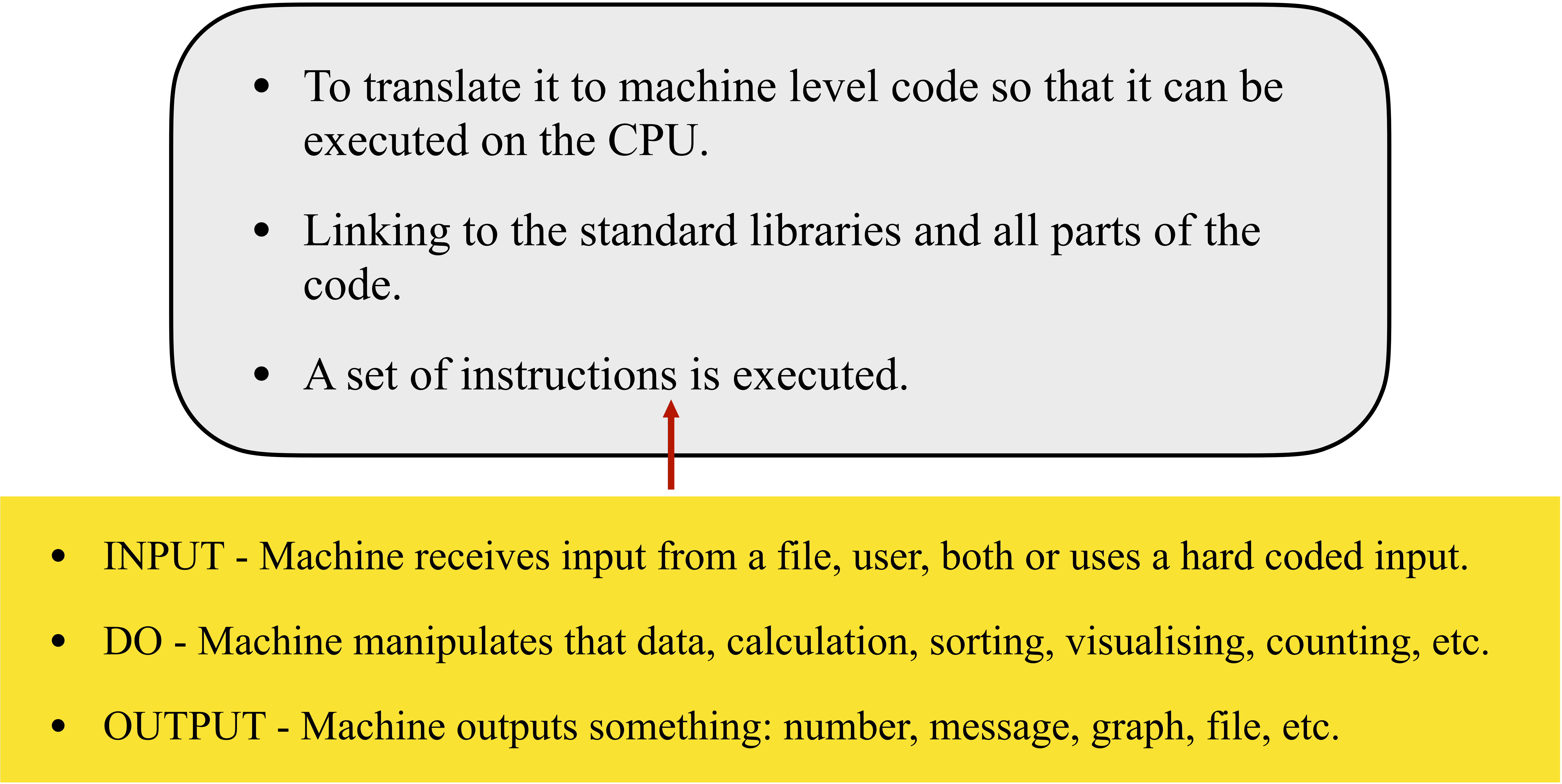
- Source code: code written by a programmer (using any editor) with proper file extensions such as .py for python (.cpp for C++).
- All programming languages have built-in standard libraries to perform several tasks.
- Often in coding, we use pre-existing code available in the language (standard libraries/modules).
- We need to know the syntax[1] of a chosen language.


Source file
(.py)
(.cpp)

We will refer to a source code just as code/program.

A syntax is a set of rules which should be followed to construct a chosen language code.

What does it mean to run a code?



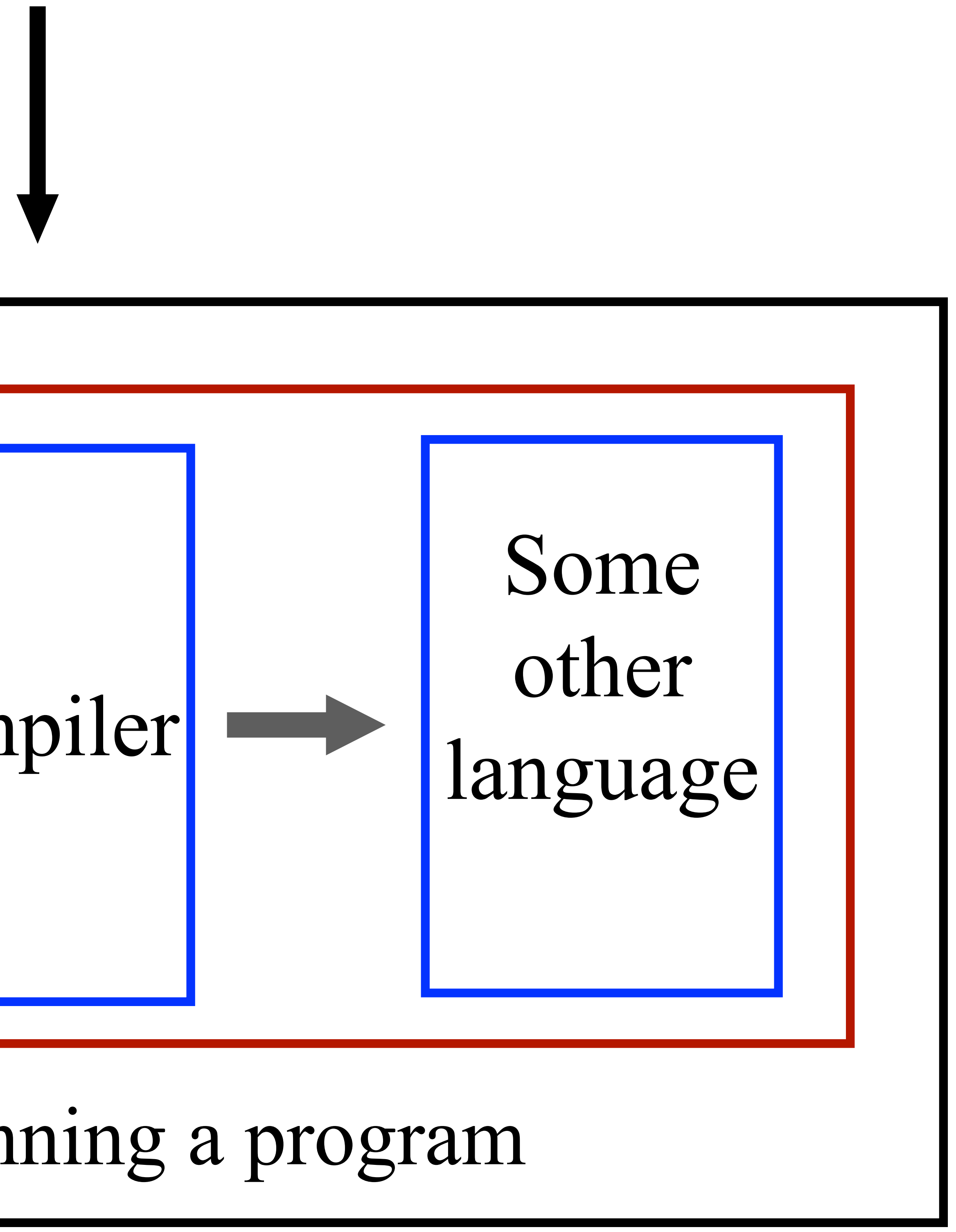
- To translate it to machine level code so that it can be executed on the CPU.
- Linking to the standard libraries and all parts of the code.
- A set of instructions is executed.

- INPUT - Machine receives input from a file, user, both or uses a hard coded input.
- DO - Machine manipulates that data, calculation, sorting, visualising, counting, etc.
- OUTPUT - Machine outputs something: number, message, graph, file, etc.

Everything happens in the order as written in the program. Also it only does what we ask it to do!

Compilation

It is a stage between writing the code and running it which converts human-readable code to either a low-level machine language or to some intermediate language.



Programming Languages

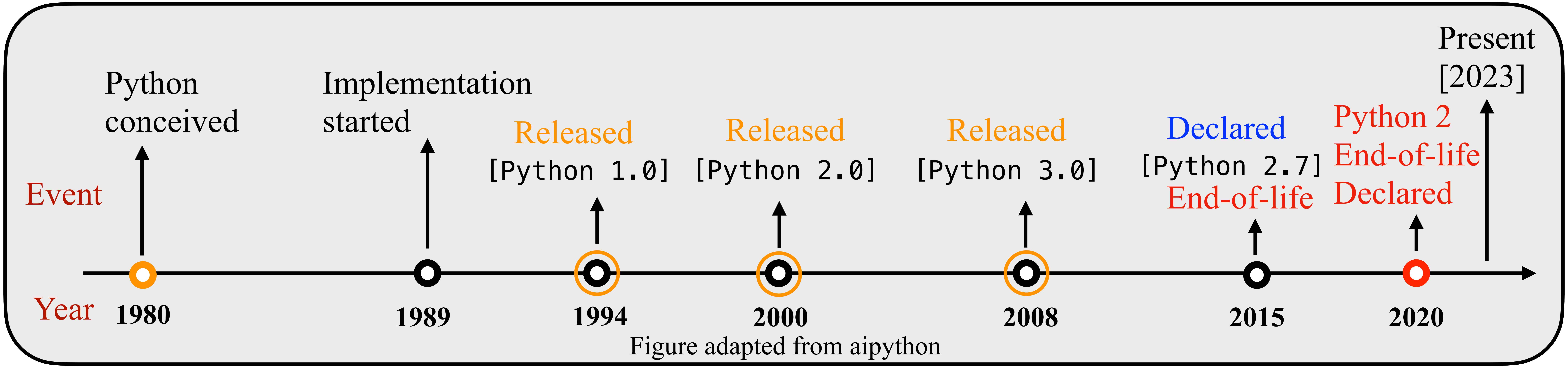
- There are hundreds of programming languages available. We distinguish between those those that need compiling and those that do not:
- Compiled:
  - C, C++, Fortran, Java etc.
- Interpreted:
  - MATLAB, Python, IDL, Perl, JavaScript.
- We think of all of these as different languages where they might have many common aspects, but work very differently.
- Once you know one language it is easier to learn more.

Part 2
Introduction to Python and getting started

Python

- Python is a general purpose interpreted programming language.
- Initially designed by Guido van Rossum in late 1980s.
- Named after BBC TV show “Monty Python's Flying Circus”.
- Developed by Python Software Foundation.

Source: wikipedia



Event

Year

Python conceived — 1980

Implementation started — 1989

Released [Python 1.0] — 1994

Released [Python 2.0] — 2000

Released [Python 3.0] — 2008

Declared [Python 2.7] End-of-life — 2015

Python 2 End-of-life Declared — 2020

Present [2023]

Figure adapted from aipython

We will use Python 3 (3.10+) in this course.

Key Features of Python

✓ Open source

✓ It is an interpreted language

✓ It is a high-level language (fundamental operations are automatically taken care of); syntax is very close to natural languages

✓ Dynamically typed language

✓ Functional and object oriented programming

✓ Being used for wide range of applications (high demand in industry)

✓ Large community support

Python Interpreter

It is another code which helps to run python codes.

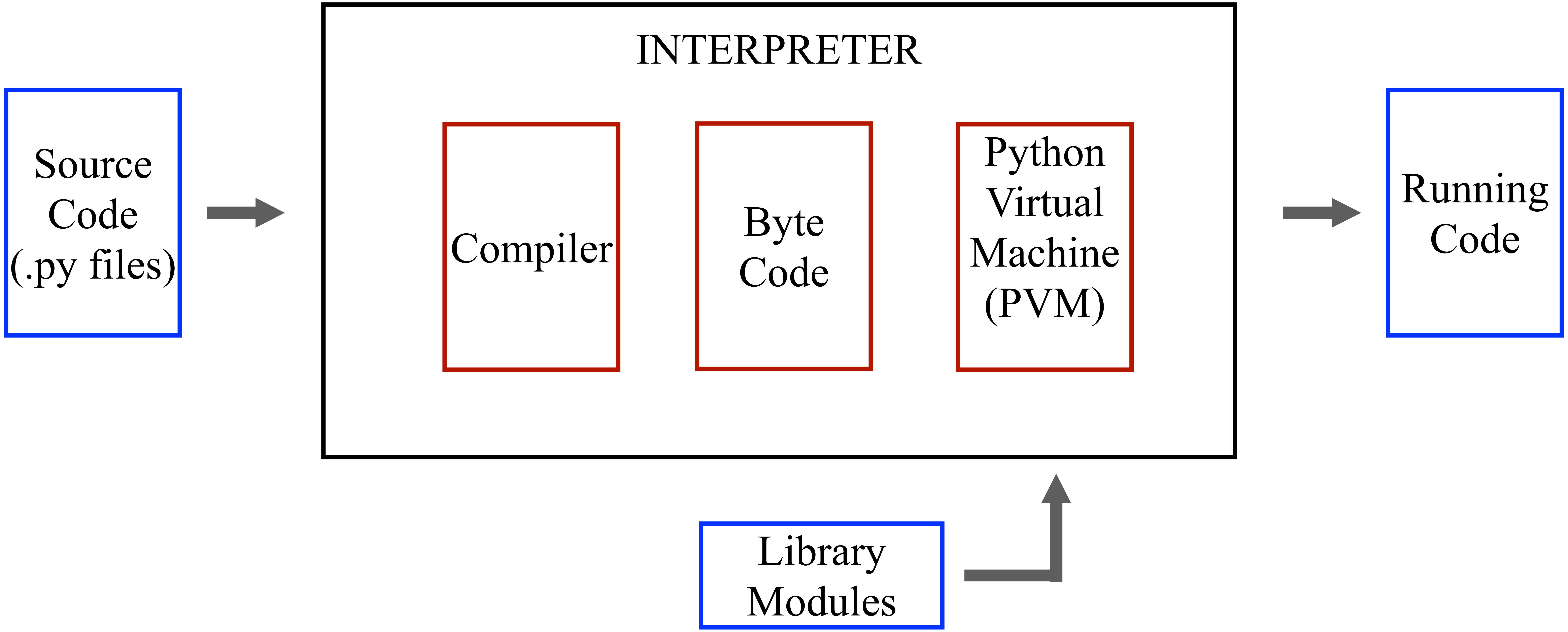


Figure adapted from www.geeksforgeeks.org

What we need to get started?

ANACONDA®

- Python Interpreter (Anaconda Python)
- Useful editor to write a code (Spyder)

Anaconda is a Python distribution which contains many packages needed for scientific computing.

Integrated Development Environments (IDEs)

- IDEs integrate several tools to run the program directly.
- There are many IDEs (suitable for different languages) where you can write code. The main purpose of them is to make it easier to work. They do things like autoformat, change text colour according to role, display variable values and let you debug.
    - Spyder, PyCharm, IDLE, Visual Studio Code, Visual Studio etc.

Anaconda and Spyder

- We will be using Spyder which is available (for free) from the Anaconda distribution which also includes the main libraries we will be using: NumPy, Matplotlib & SciPy.

- See installation instructions on BlackBoard. (https://docs.anaconda.com/free/anaconda/install/)

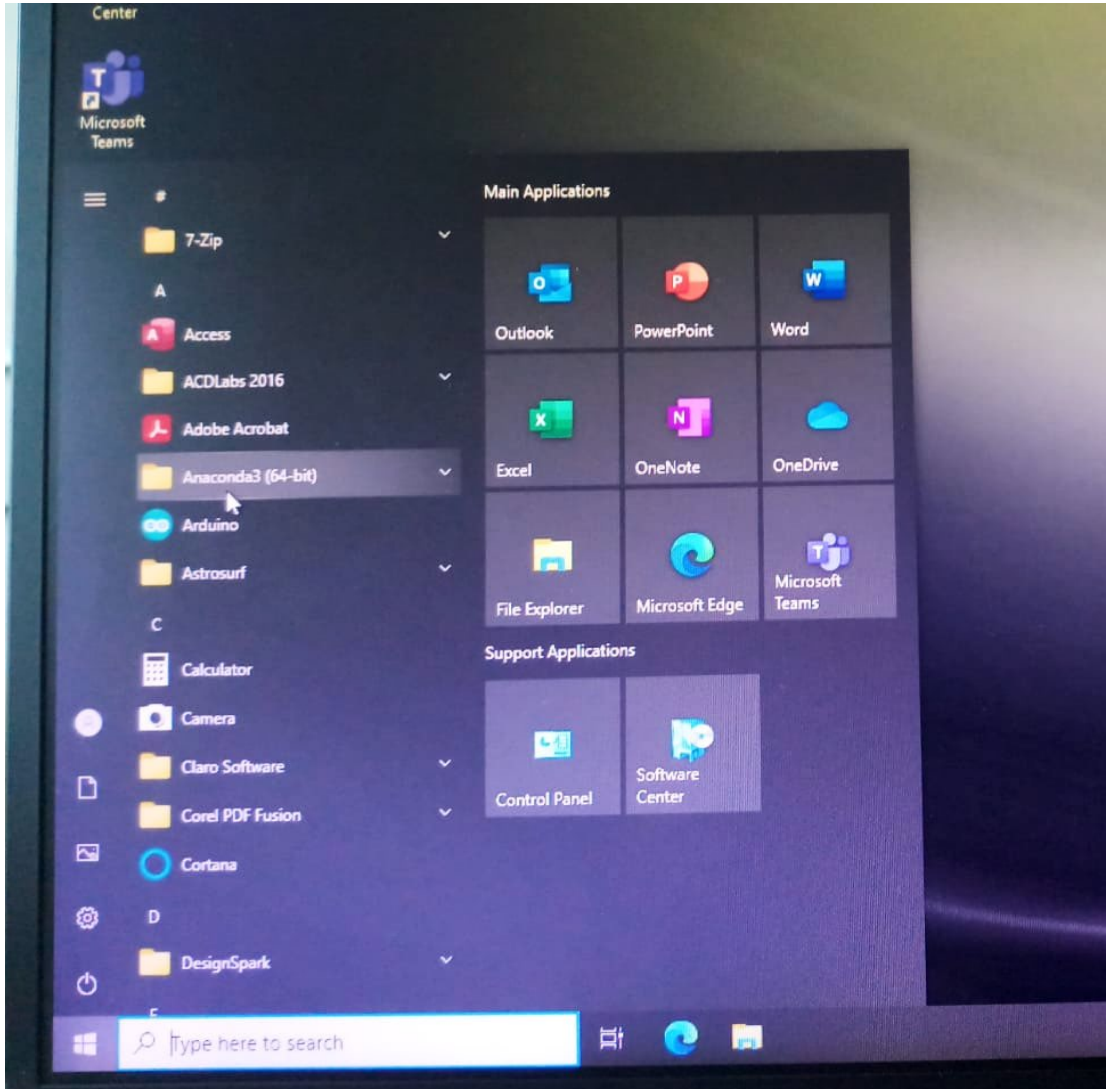
SPYDER: Scientific PYthon Development EnviRonment

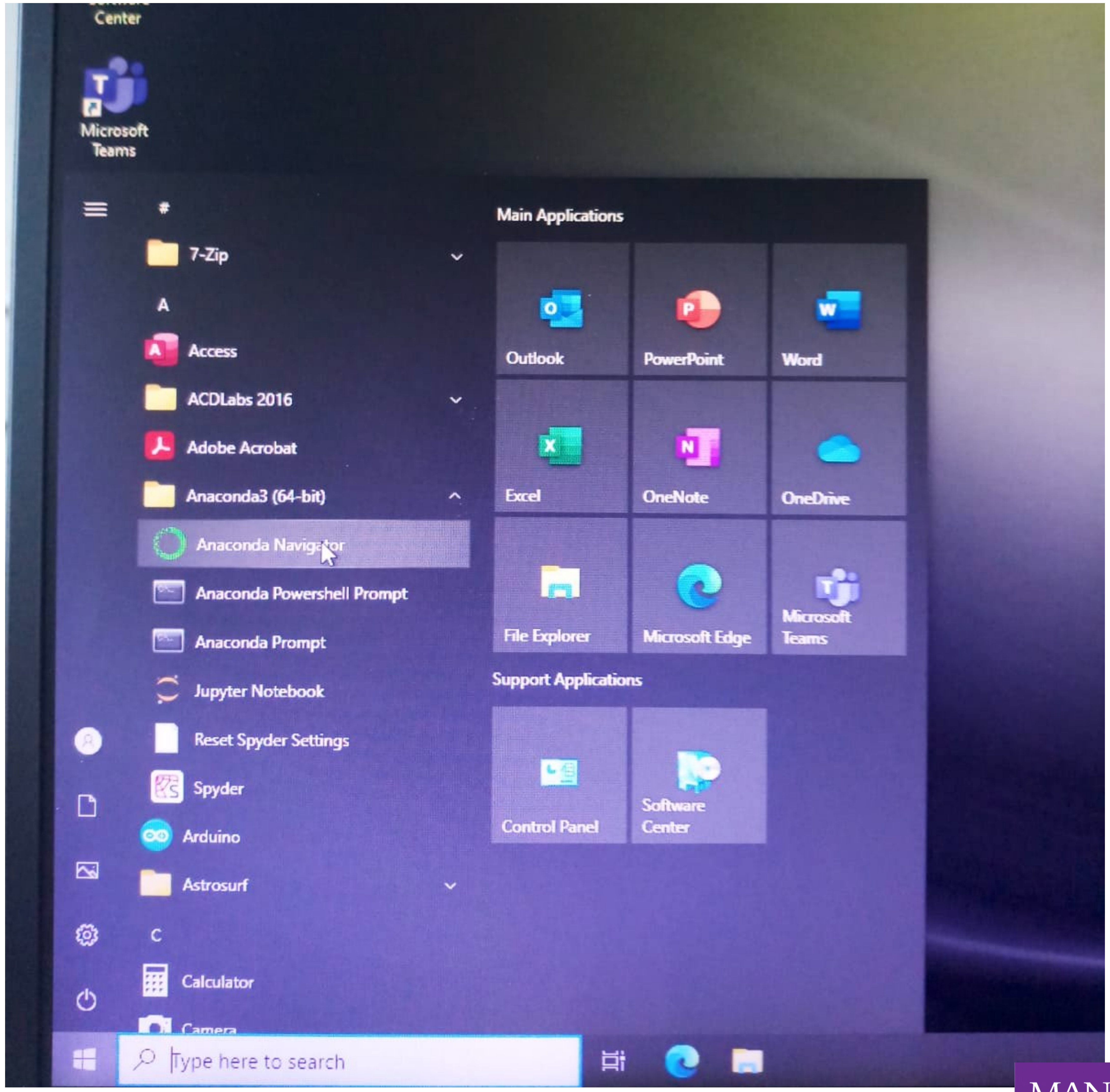
- It has an editor to write code, a console to evaluate it and view the results and other facilities needed for efficient code development.

Launching Spyder

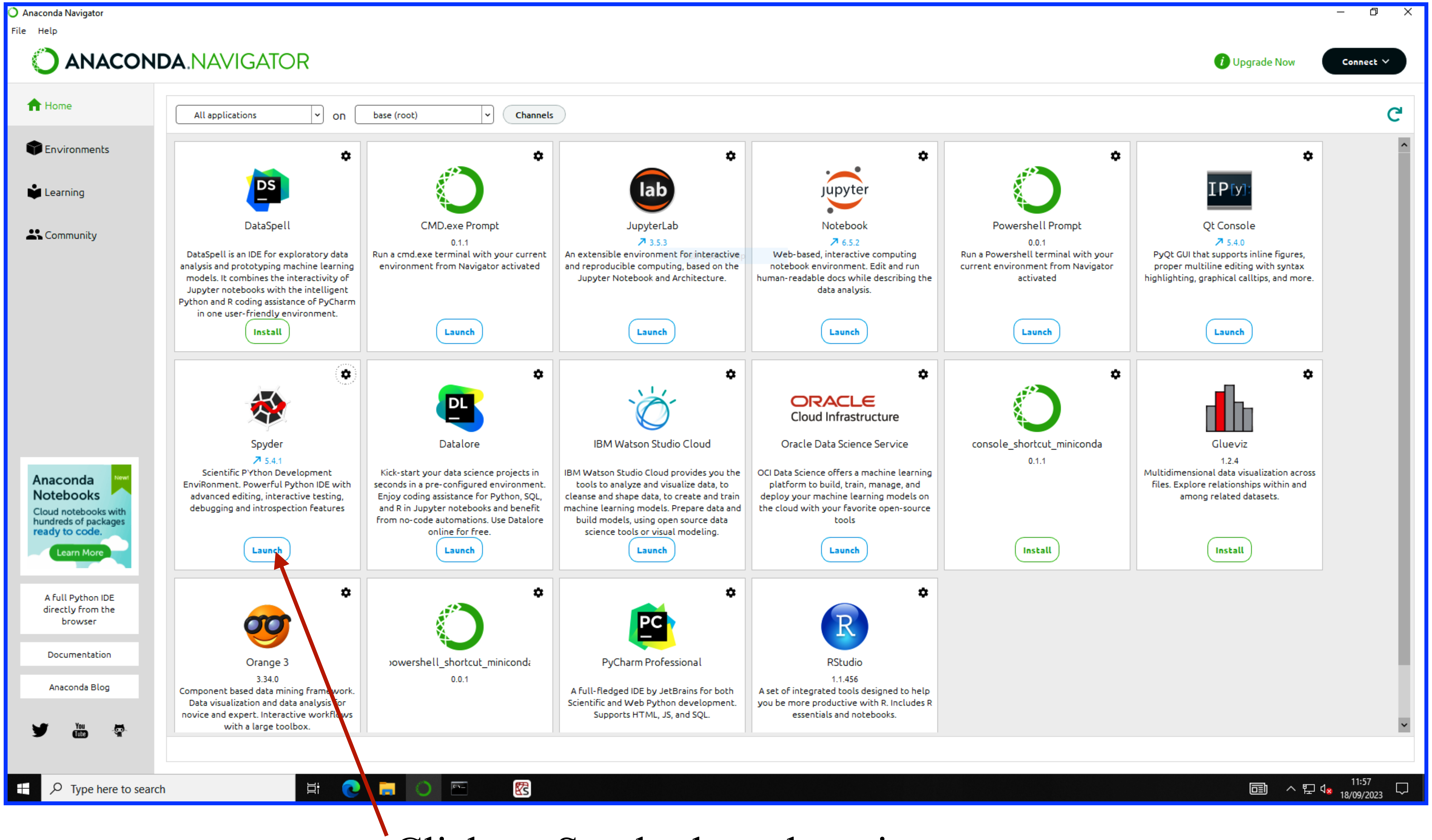
- Go to Anaconda Navigator
- Click on Spyder launch option
- If you have standalone version (on Mac) - just open Spyder

Open Anaconda Navigator (Windows- Lab PC)





Anaconda Navigator



DataSpell

DataSpell is an IDE for exploratory data analysis and prototyping machine learning models. It combines the interactivity of Jupyter notebooks with the intelligent Python and R coding assistance of PyCharm in one user-friendly environment.

CMD.exe Prompt

0.1.1

Run a cmd.exe terminal with your current environment from Navigator activated

JupyterLab

↗ 3.5.3

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Notebook

↗ 6.5.2

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Powershell Prompt

0.0.1

Run a Powershell terminal with your current environment from Navigator activated

Qt Console

↗ 5.4.0

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Spyder

↗ 5.4.1

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Datalore

Kick-start your data science projects in seconds in a pre-configured environment. Enjoy coding assistance for Python, SQL, and R in Jupyter notebooks and benefit from no-code automations. Use Datalore online for free.

IBM Watson Studio Cloud

IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.

ORACLE Cloud Infrastructure

Oracle Data Science Service

OCI Data Science offers a machine learning platform to build, train, manage, and deploy your machine learning models on the cloud with your favorite open-source tools

console_shortcut_miniconda

0.1.1

Glueviz

1.2.4

Multidimensional data visualization across files. Explore relationships within and among related datasets.

Orange 3

3.34.0

Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

›owershell_shortcut_minicond‹

0.0.1

PyCharm Professional

A full-fledged IDE by JetBrains for both Scientific and Web Python development. Supports HTML, JS, and SQL.

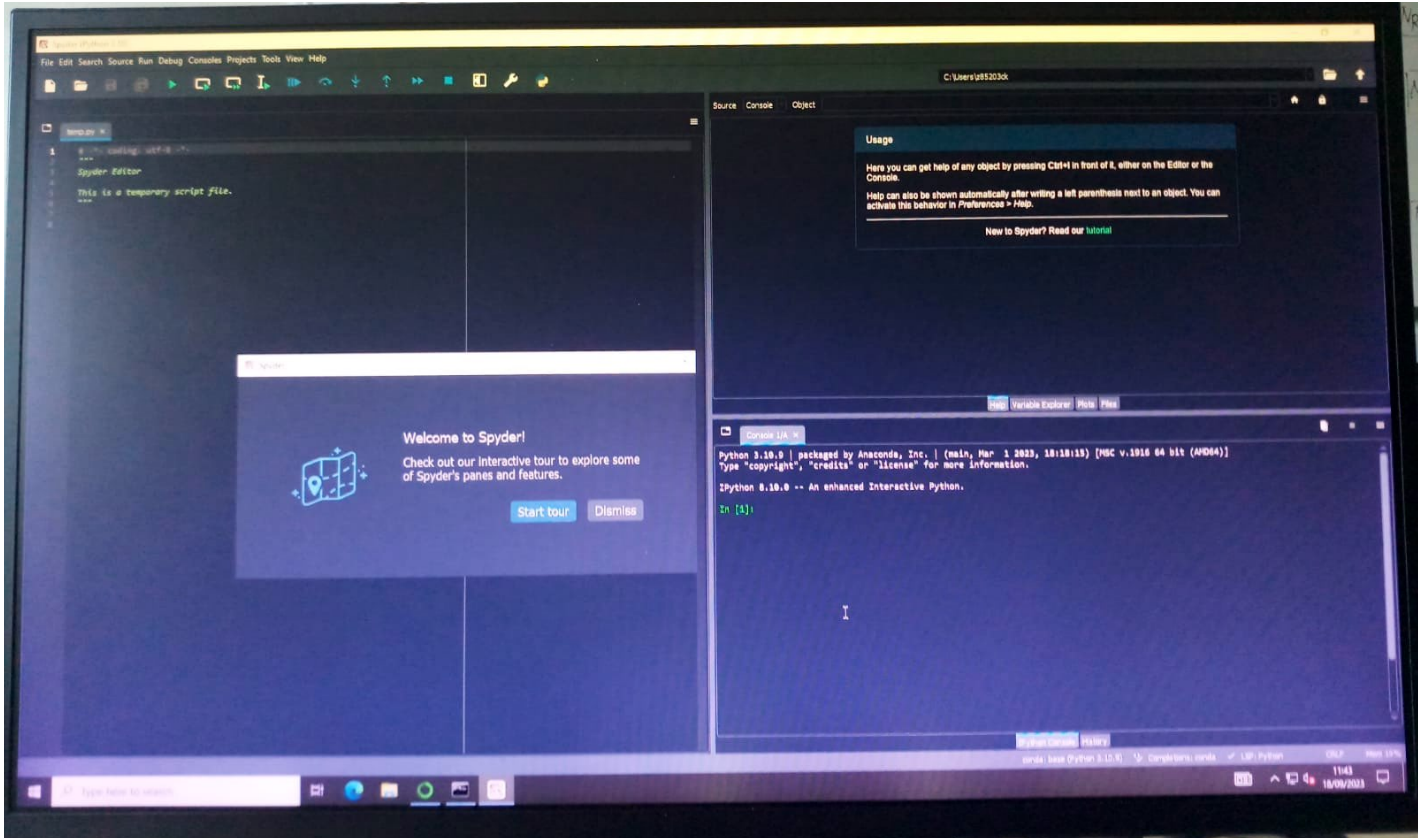
RStudio

1.1.456

A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

Click on Spyder launch option.

Spyder - First View


File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\z85203ck
temp.py
Source Console Object
Help Variable Explorer Plots Files
Console 1/A
Start tour
Dismiss

```
# -*- coding: utf-8 -*-
"""
Spyder Editor

This is a temporary script file.
"""
```

Usage

Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our tutorial

Welcome to Spyder!

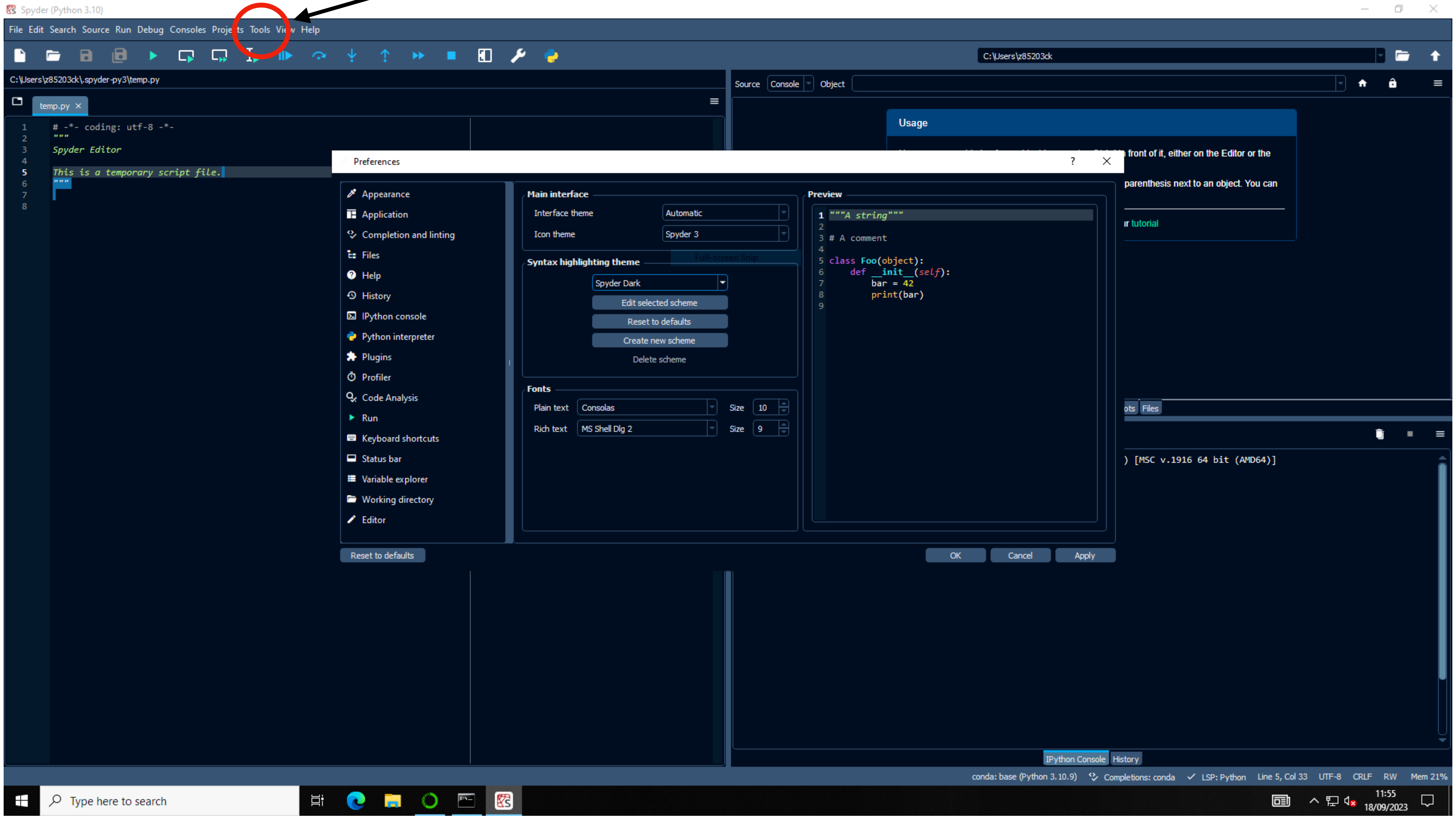
Check out our interactive tour to explore some of Spyder's panes and features.

```
Python 3.10.9 | packaged by Anaconda, Inc. | (main, Mar  1 2023, 18:18:15) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.10.0 -- An enhanced Interactive Python.

In [1]:
```

MANCHESTER 1824 The University of Manchester

Spyder: Updating Preferences

Tools → Preferences → Appearance



```
# -*- coding: utf-8 -*-
"""
Spyder Editor

This is a temporary script file.
"""
```

Main interface

Interface theme: Automatic

Icon theme: Spyder 3

Syntax highlighting theme: Spyder Dark

Fonts

Plain text: Consolas Size: 10

Rich text: MS Shell Dlg 2 Size: 9

Preview

```
"""A string"""

# A comment

class Foo(object):
    def __init__(self):
        bar = 42
        print(bar)
```

```
) [MSC v.1916 64 bit (AMD64)]
```

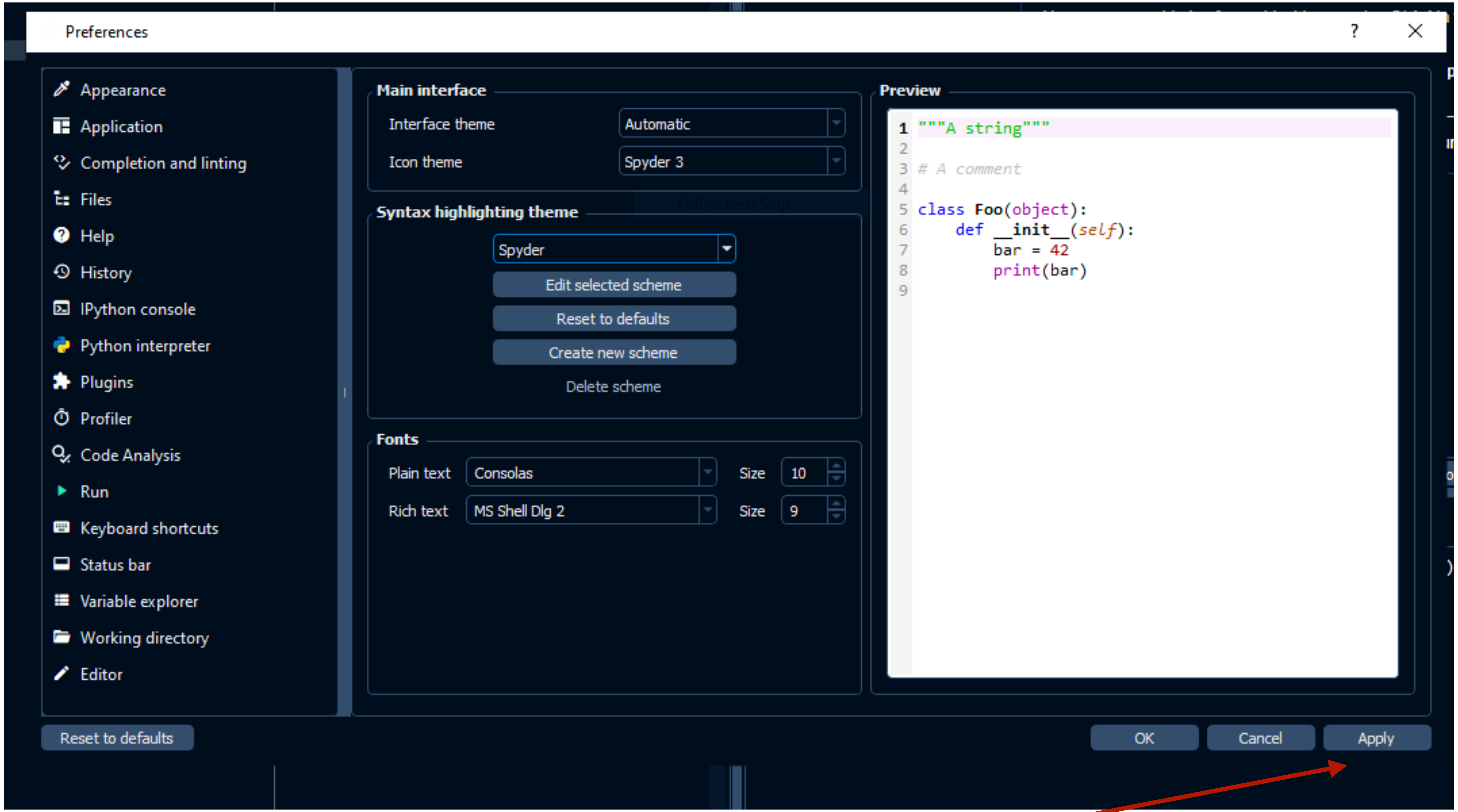
conda: base (Python 3.10.9) Completions: conda LSP: Python Line 5, Col 33 UTF-8 CRLF RW Mem 21%

11:55 18/09/2023

Default Syntax highlighting theme is Spyder Dark. You may also want to change font size.

Spyder: Updating Preferences

Change Syntax highlighting theme to Spyder



Main interface

Interface theme: Automatic

Icon theme: Spyder 3

Syntax highlighting theme

Spyder

Fonts

Plain text: Consolas — Size: 10

Rich text: MS Shell Dlg 2 — Size: 9

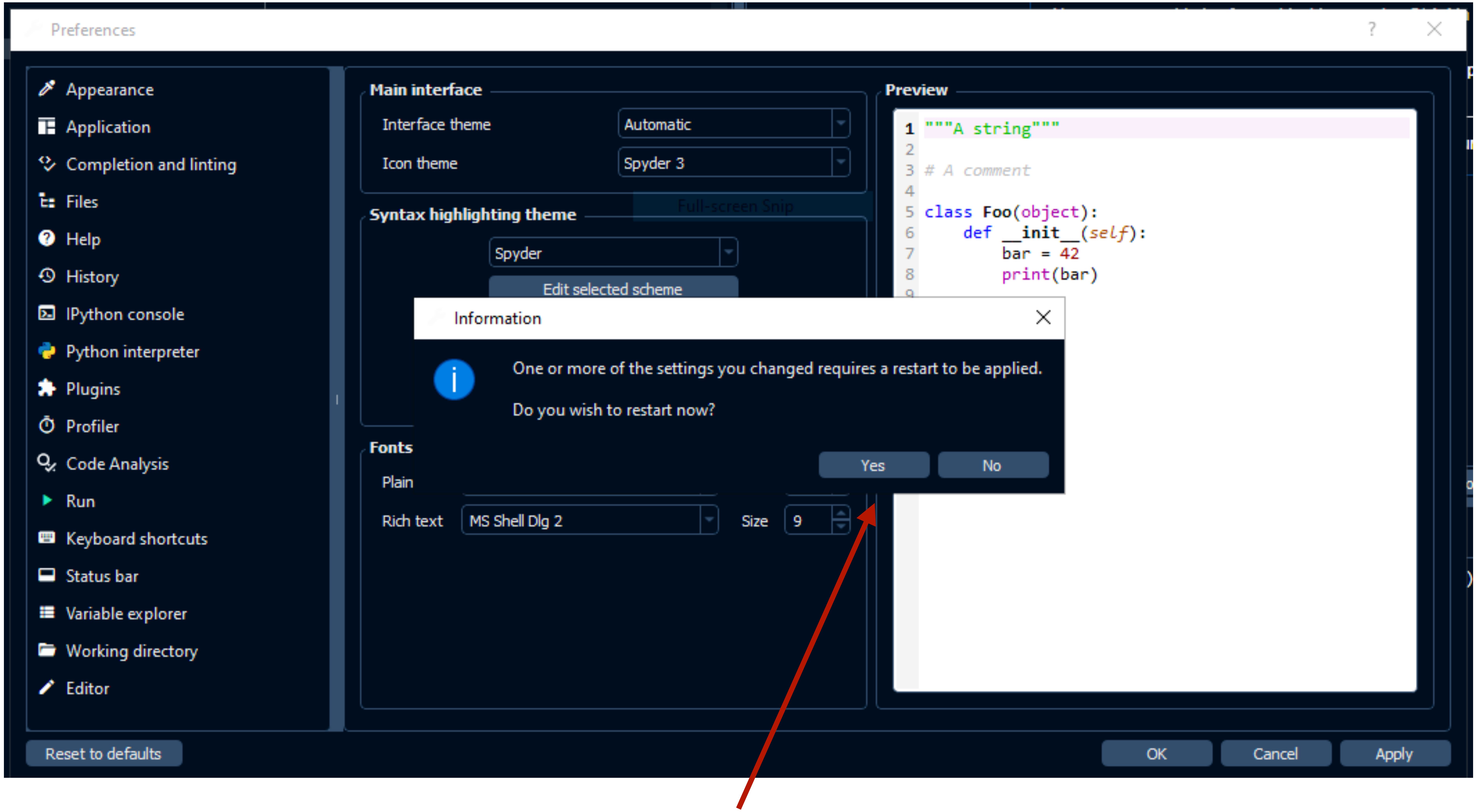
Preview

```
"""A string"""

# A comment

class Foo(object):
    def __init__(self):
        bar = 42
        print(bar)

```

Apply changes.

MANCHESTER
1824
The University of Manchester

Spyder: Updating Preferences



Main interface

Interface theme: Automatic

Icon theme: Spyder 3

Syntax highlighting theme

Spyder

Fonts

Plain

Rich text: MS Shell Dlg 2 Size: 9

Preview

```
"""A string"""

# A comment

class Foo(object):
    def __init__(self):
        bar = 42
        print(bar)
```

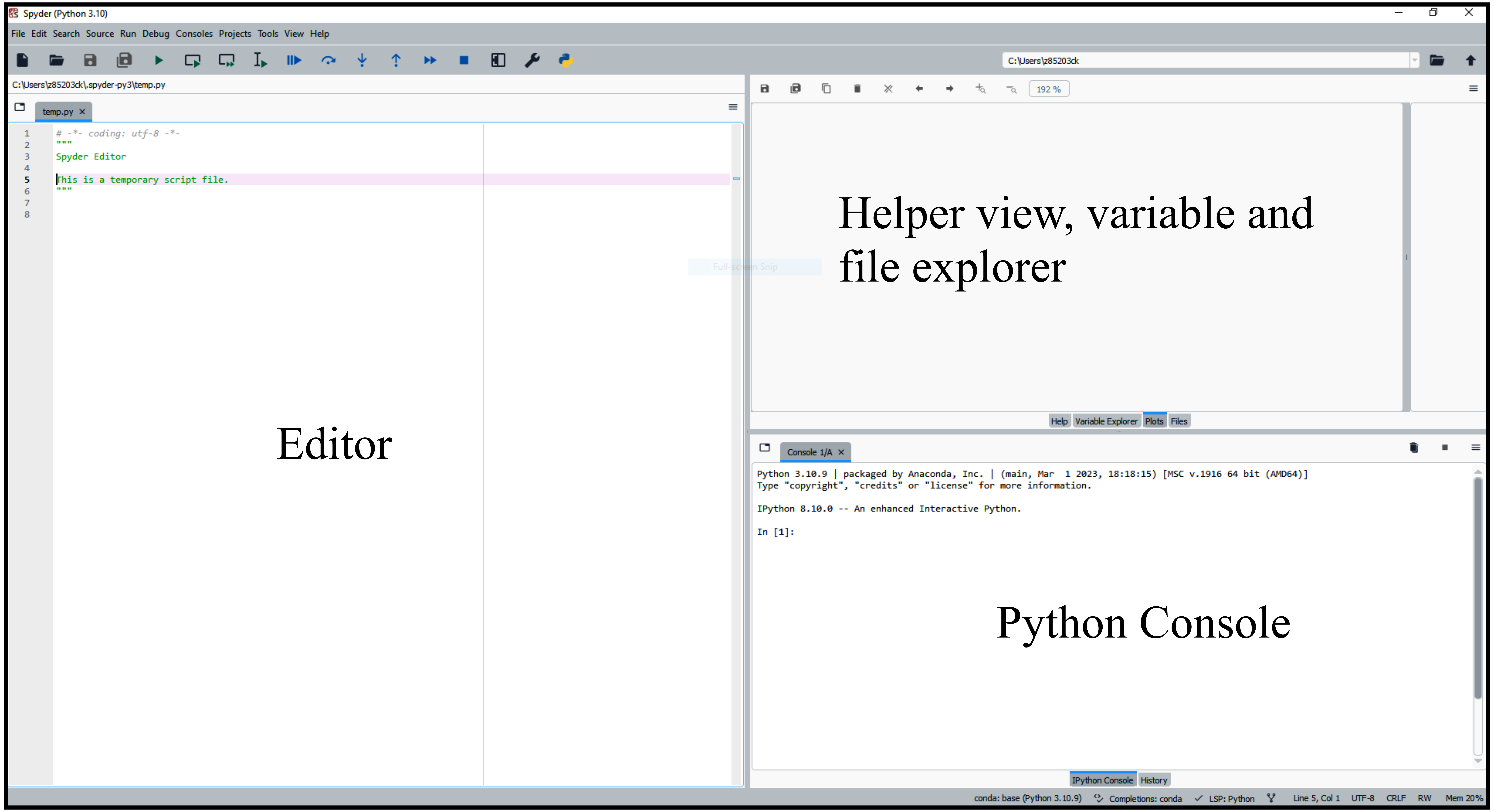
Information

One or more of the settings you changed requires a restart to be applied.

Do you wish to restart now?

Restart it to apply the changes.

Spyder



```
# -*- coding: utf-8 -*-
"""
Spyder Editor

This is a temporary script file.
"""
```

```
Python 3.10.9 | packaged by Anaconda, Inc. | (main, Mar  1 2023, 18:18:15) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.10.0 -- An enhanced Interactive Python.

In [1]:
```

conda: base (Python 3.10.9) Completions: conda ✓ LSP: Python Line 5, Col 1 UTF-8 CRLF RW Mem 20%

Editor

Helper view, variable and file explorer

Python Console

Customised View (changed background theme)

Part 3
(Built-in) Data Types and Variables

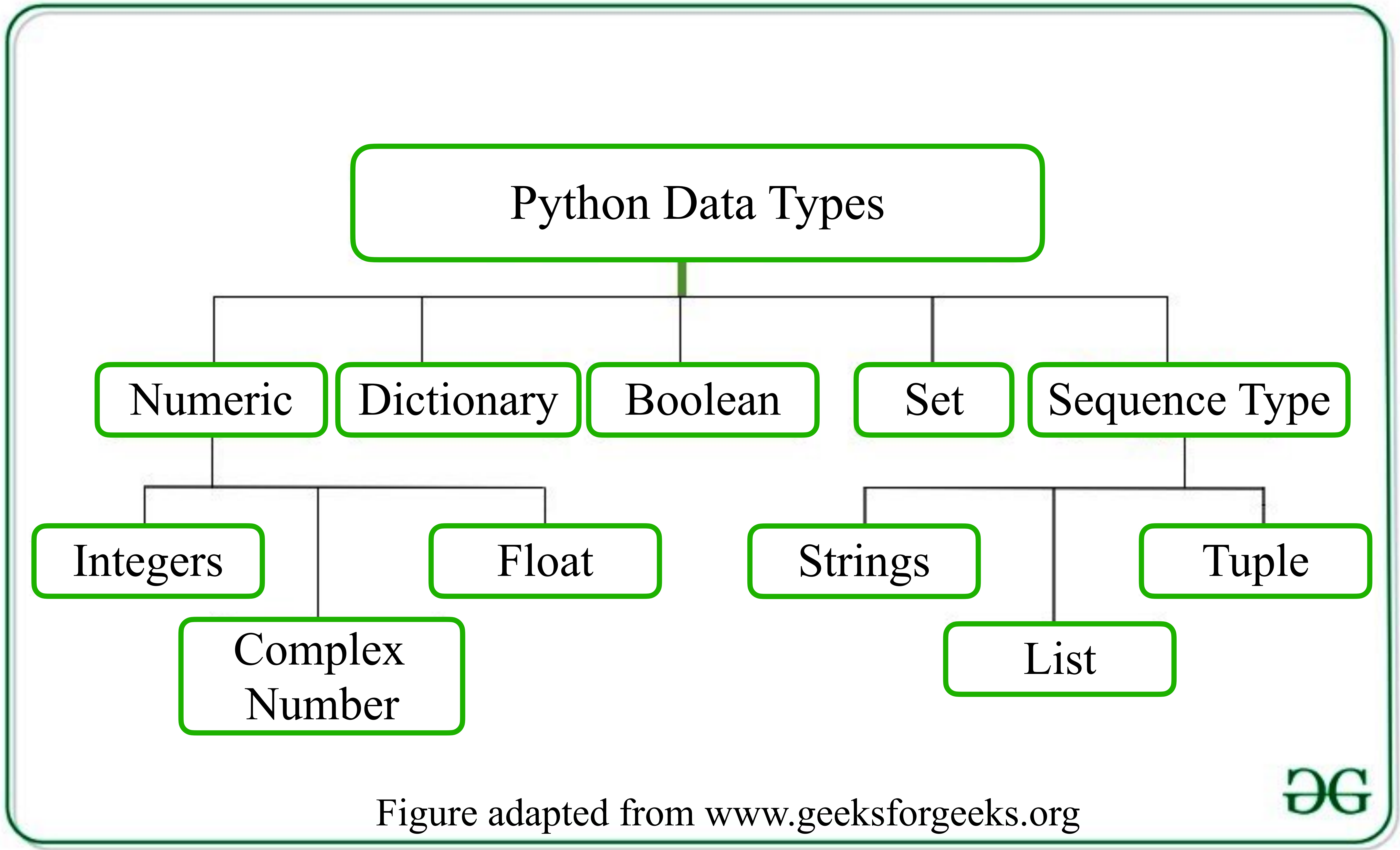


Figure adapted from www.geeksforgeeks.org

We do not need to declare data types of variables in Python.

Numerical Data Storage

- We can define any base 10 number in binary (bits):
  - 0 → 0
  - 1 → 1

0	0	0	0	0	0	0	1

  - 2 → 10
  - 3 → 11
  - 4 → 100
  - 5 → 101

0	0	0	0	0	1	0	1

  - Etc.

Data Types

Data type is a classification of different types of data which decides what type of operations can be performed on that data and how it is stored.

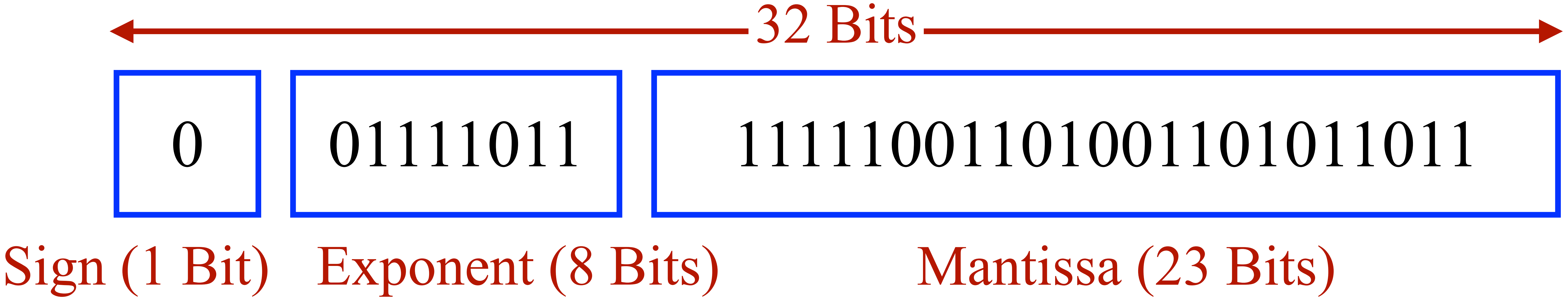
- Numeric data type: `int, float`
- Sequence type: strings (`str`), `list` (video 6)

`int` type

- Integers are whole numbers such as 1, 2 , -5, 123445 etc.
- Typically, integer number is composed of 32 bits (4 bytes).
- Python will update the size allocated to it automatically if needed.

`float` type

- To work with decimals, we make use of the single-precision floating-point format, or float.
- We can write any number as mantissa x base$^{\text{exponent}}$.
- Example: $(0.12345)_{10} = (1.9752000570297241 \times 2^{-4})_2$
- These occupy 32 bits of memory which are



32 Bits

0 | 01111011 | 11111001101001101011011

Sign (1 Bit) | Exponent (8 Bits) | Mantissa (23 Bits)

- These will be accurate to approx 7 decimal places.
- For more accuracy, we can use double which occupies 64 bits and is accurate up to approx. 16 decimals. This is the default in Python.
- Speed of floating point operations is given by FLOPS, which is characteristic of the speed of the computer system.

String Data Type (`str`)

- Typically any alphanumeric character type is stored as a char type of 1 byte.
- Computer then uses an encoder (ASCII, UTF-8, etc.) to relate machine language (1s & 0s) to a character (alphanumerical, greek, mathematical, etc.).
- Python instead just uses an array of chars called string.
- So a char in Python would be a string with one element.
- Strings are represented by either single or double-quotes.

What are Variables?

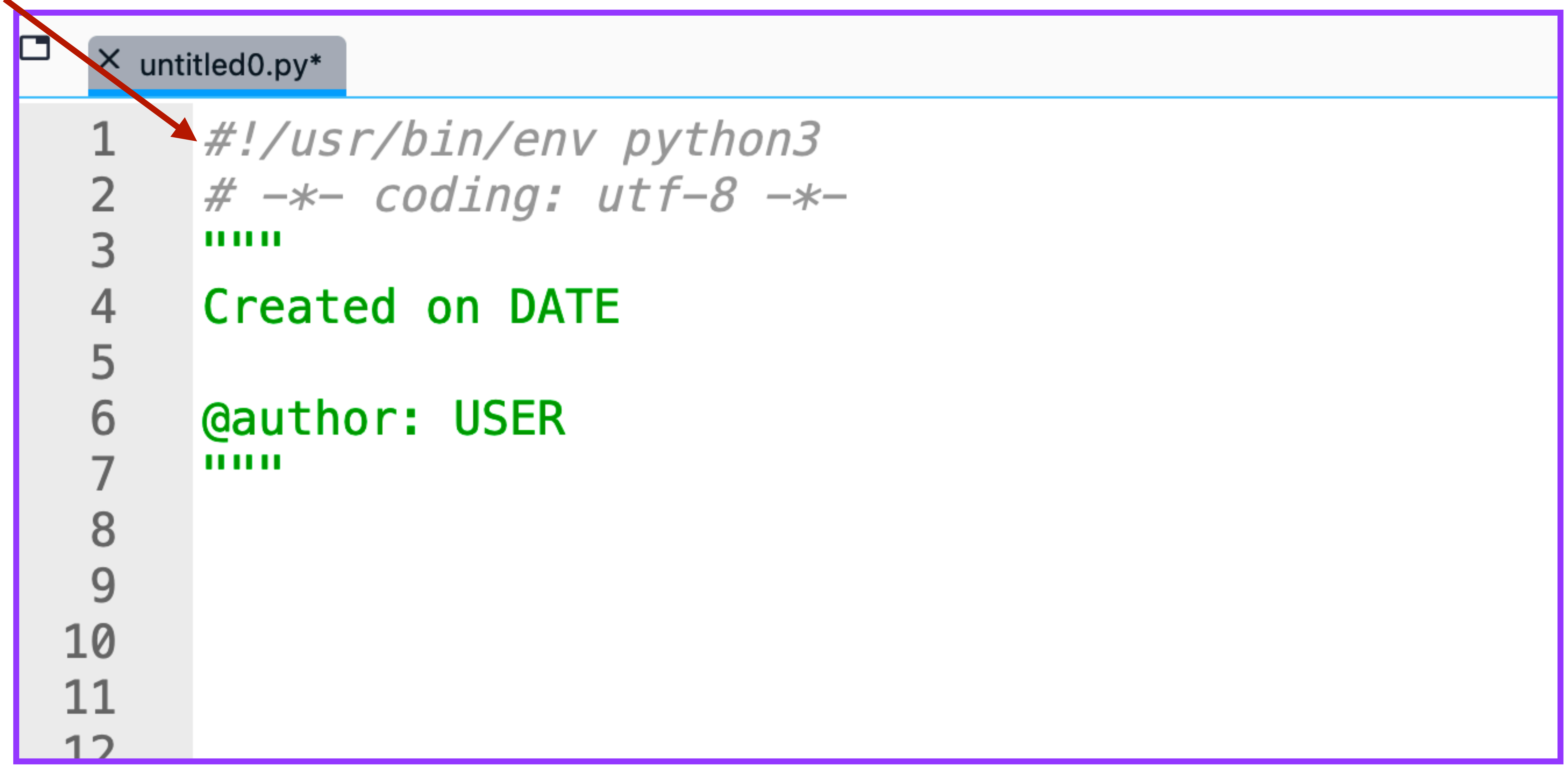
- When a number is called, the computer allocates some memory to store it.
- Variable is a named location in the memory.
- Variable names (general rules, more on this in the coming weeks):
  - Must start with a letter or the underscore character.
  - Must not start with numbers and can't be the reserved keywords.
  - Names are case sensitive, e.g. NUMBER, Number, and number are different variables.

Part 4
First Python Program

`first_program.py`

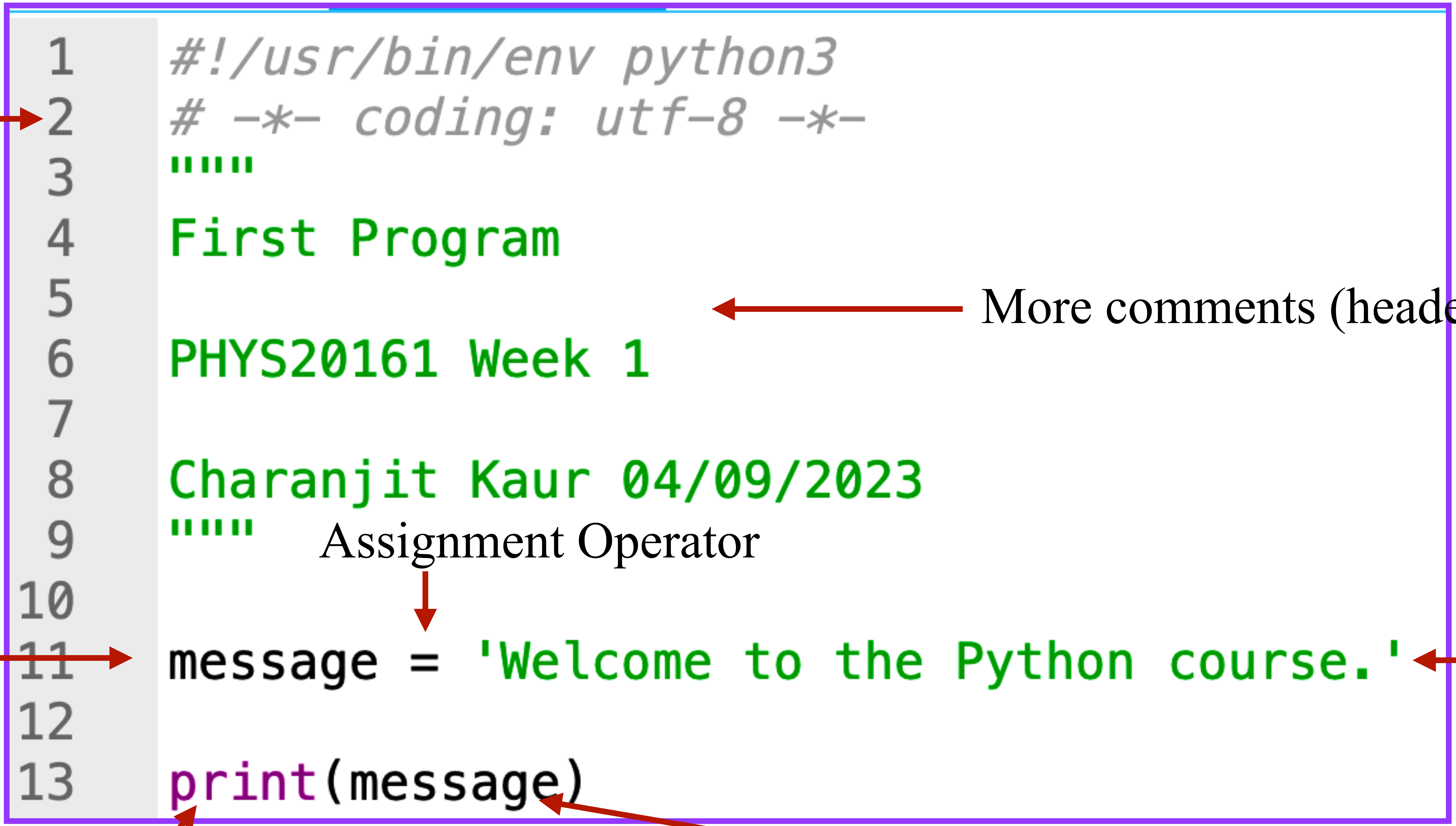
Spyder : Editor View

Shebang (#!)



```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""

Created on DATE

@author: USER
"""
```

A Simple Python Program



```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
First Program

PHYS20161 Week 1

Charanjit Kaur 04/09/2023
"""

message = 'Welcome to the Python course.'

print(message)
```

Comment

More comments (header)

Assignment Operator

Variable

String

Function

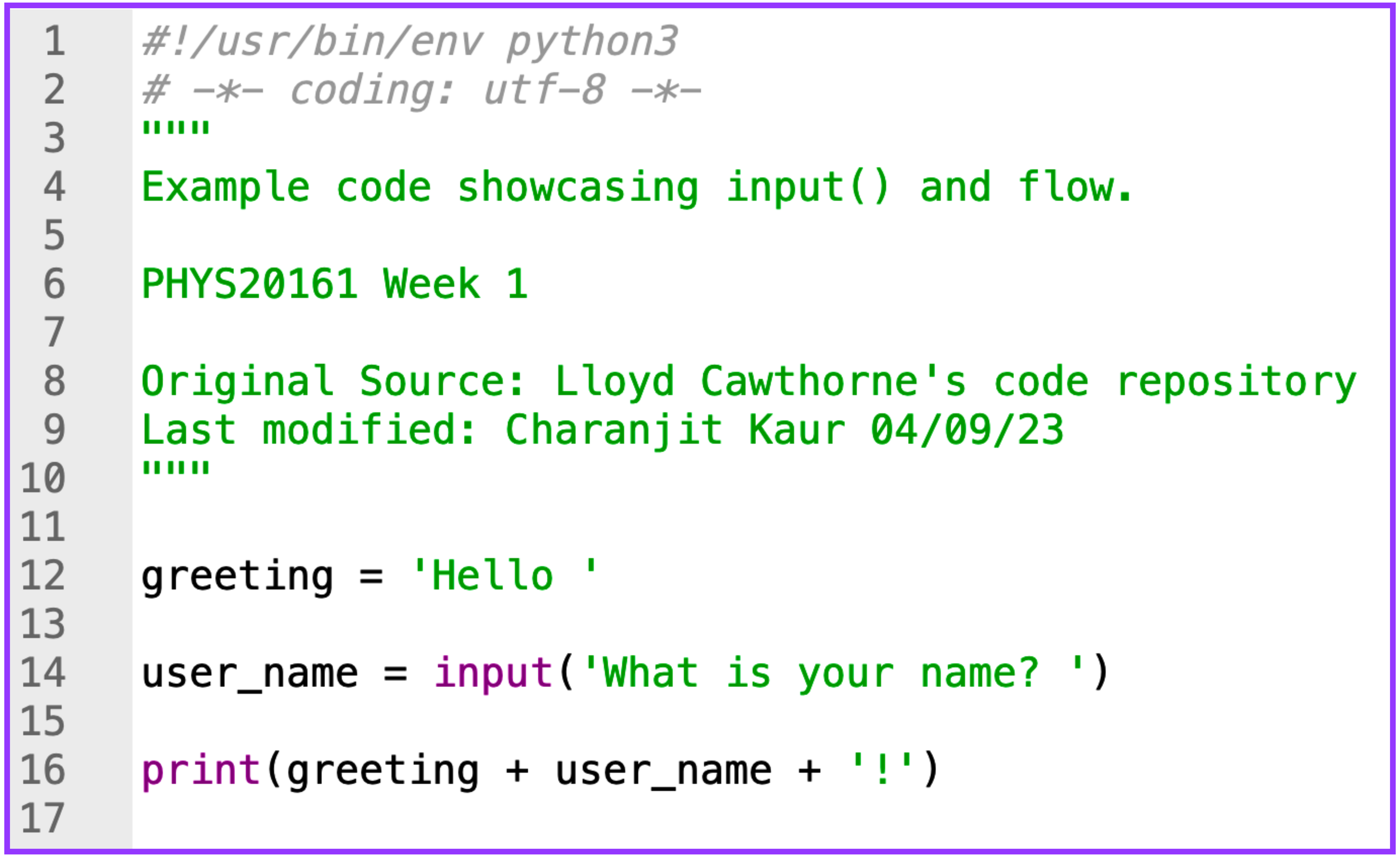
Argument

PL1:first_program.py

Part 5
More Example Program

`hello_user.py`, `number_cube.py`

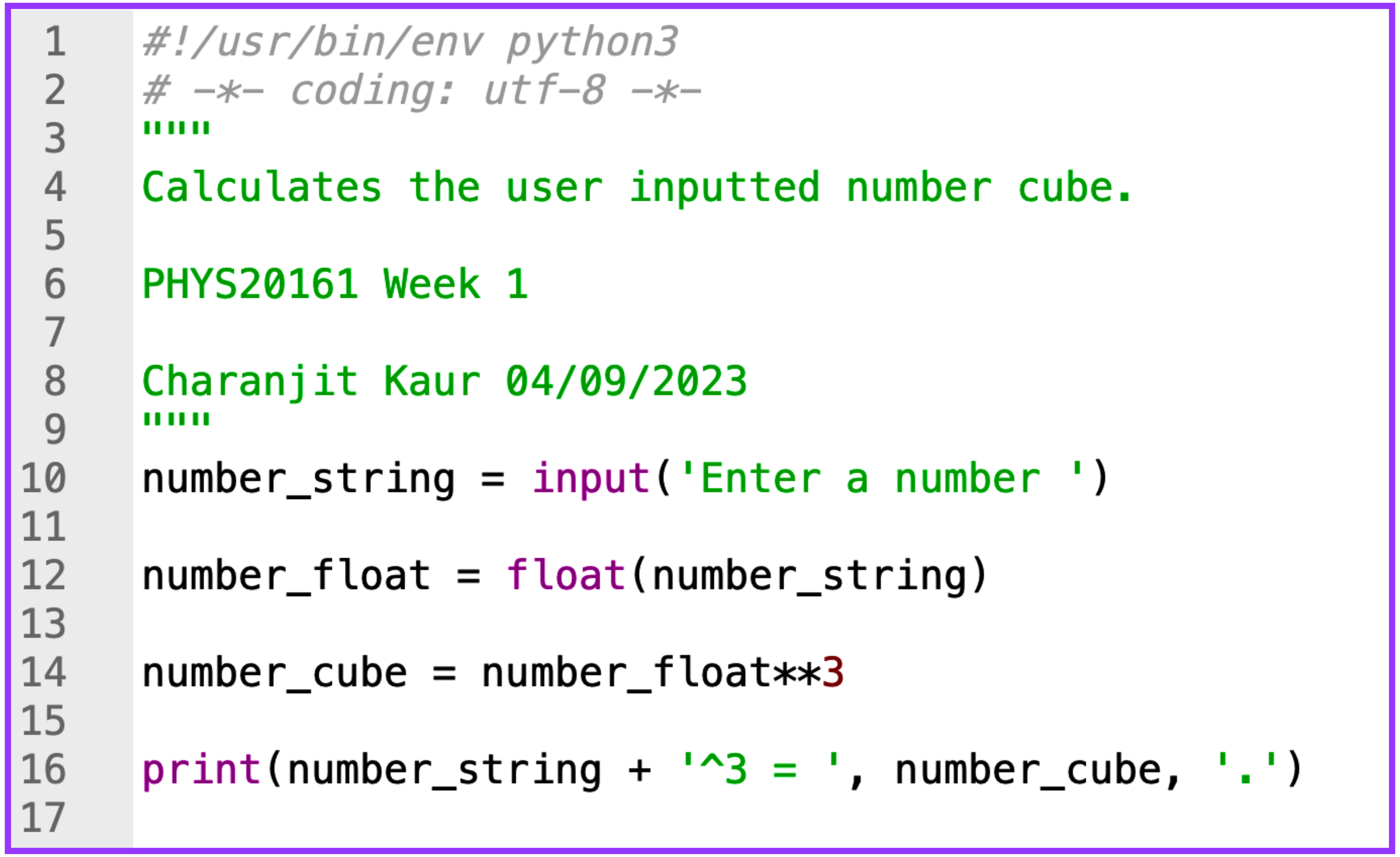
Basic Example 1



```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Example code showcasing input() and flow.

PHYS20161 Week 1

Original Source: Lloyd Cawthorne's code repository
Last modified: Charanjit Kaur 04/09/23
"""

greeting = 'Hello '

user_name = input('What is your name? ')

print(greeting + user_name + '!')

```

PL1:hello_user.py

Basic Example 2



```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Calculates the user inputted number cube.

PHYS20161 Week 1

Charanjit Kaur 04/09/2023
"""
number_string = input('Enter a number ')

number_float = float(number_string)

number_cube = number_float**3

print(number_string + '^3 = ', number_cube, '.')

```

PL1: number_cube.py

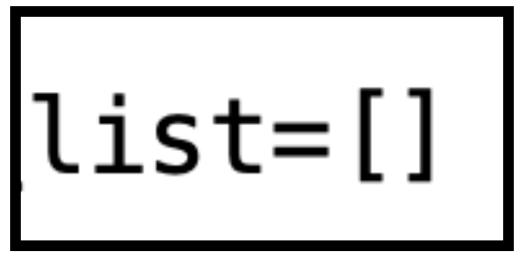
Part 6
Lists

`list_examples.py`

List (`list`)

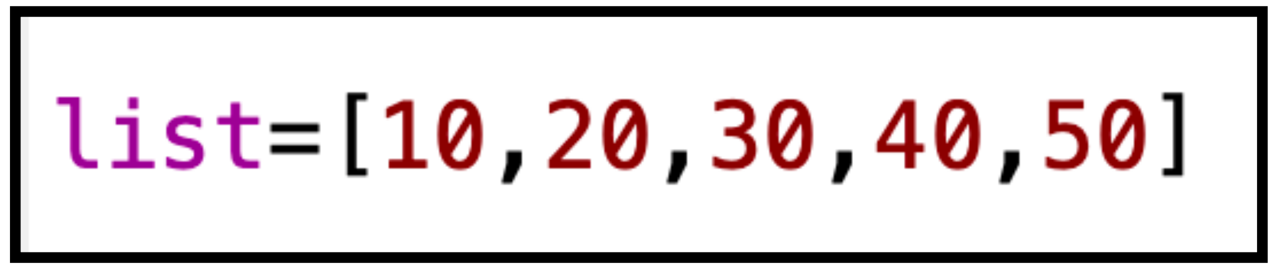
- Ordered sequence of information (data).
- Denoted by a square bracket [ ].
- Elements are accessed with index.
- It can contain elements of same or mixed types.
- Elements of a list can be updated, this is called mutability.
- We can create lists of lists of data or lists of lists of lists of data, etc.

Lists: Examples

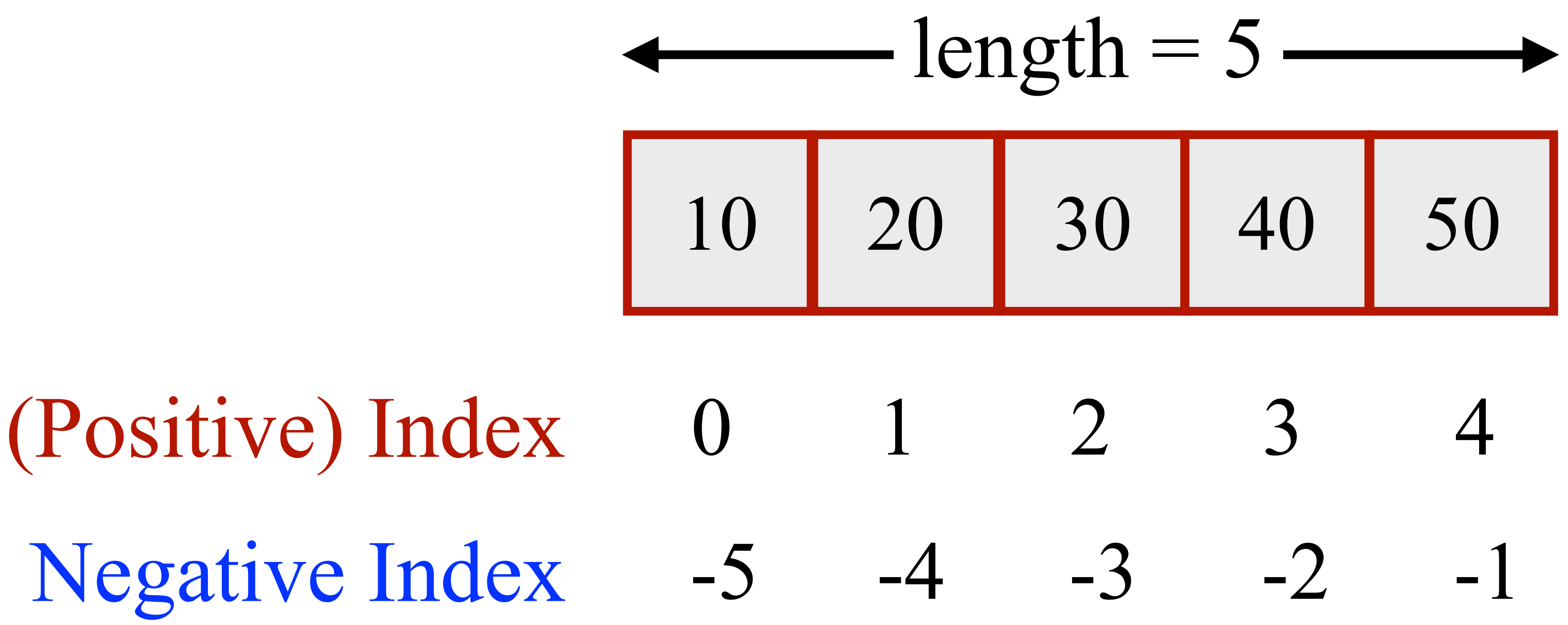


```
list=[]
```

→ Empty List



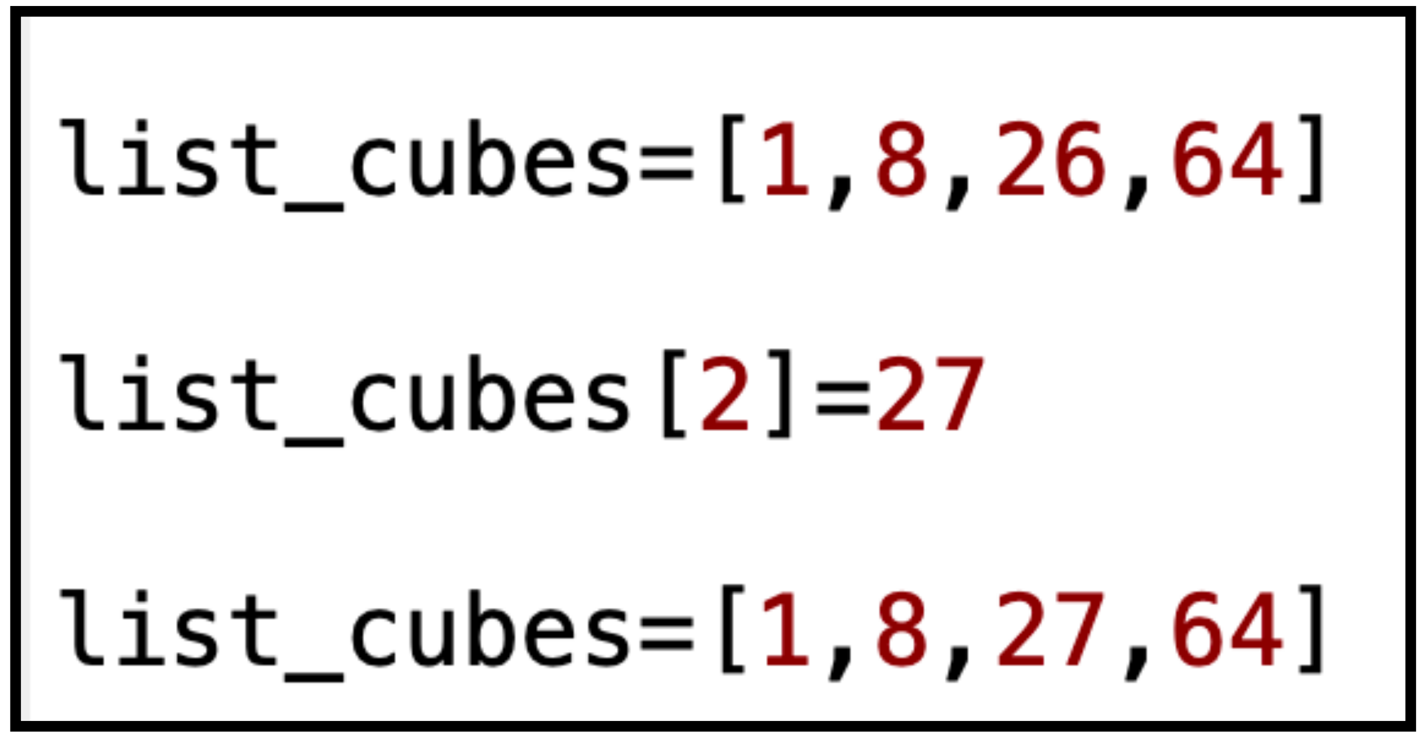
```
list=[10,20,30,40,50]
```



list[0] = 10
list[1] = 20
list[2] = 30
list[3] = 40
list[4] = 50

list[-1] = 50
list[-2] = 40
list[-3] = 30
list[-4] = 20
list[-5] = 10

- Lists are mutable



```
list_cubes=[1,8,26,64]

list_cubes[2]=27

list_cubes=[1,8,27,64]
```

Summary

- Programming is about giving instructions to the computer to perform some tasks.
- Generally programming languages are divided into two categories, compiled and interpreted.
- Python is a general purpose, high-level, interpreted programming language.
- A data type is a classification that specifies which type of value a variable can hold or which operations can be performed on a particular piece of data.
- Python is a dynamically typed language which means we do not need to define the type of variables. It automatically gets determined when variables are used for the first time in the code.
- Key to learn programming is practice.

Happy coding!