IN0011 Introduction to Programming with Python

Lab 1: IDE Installation, Variables and Simple Data Types

Reminder: Have you swiped your card? Make sure you see a green light on the reader.

Aim

- Consolidate learning from week 1.
- Installing the Anaconda distribution, understanding the concept of IDE with Spyder IDE and execute a "Hello World" program.
- Executing the file *lab1.py* and modifying it.
- Using python variables and taking user input
- Using python numbers and strings

Note: Feel free to refer to the lecture material if needed. If you encounter any issues or have questions, reach out to your tutor. If you don't have enough time to complete the exercises during the session, it's advised to finish them at home. You are encouraged to also try the optional problem(s). Solutions will be available at 17:00 on September 30.

Part A: IDE installation and running python scripts

Task 1: Installing Anaconda and Launching Spyder

Download the Anaconda installer, specific to your OS (Windows or Mac), from this link: https://www.anaconda.com/download#downloads

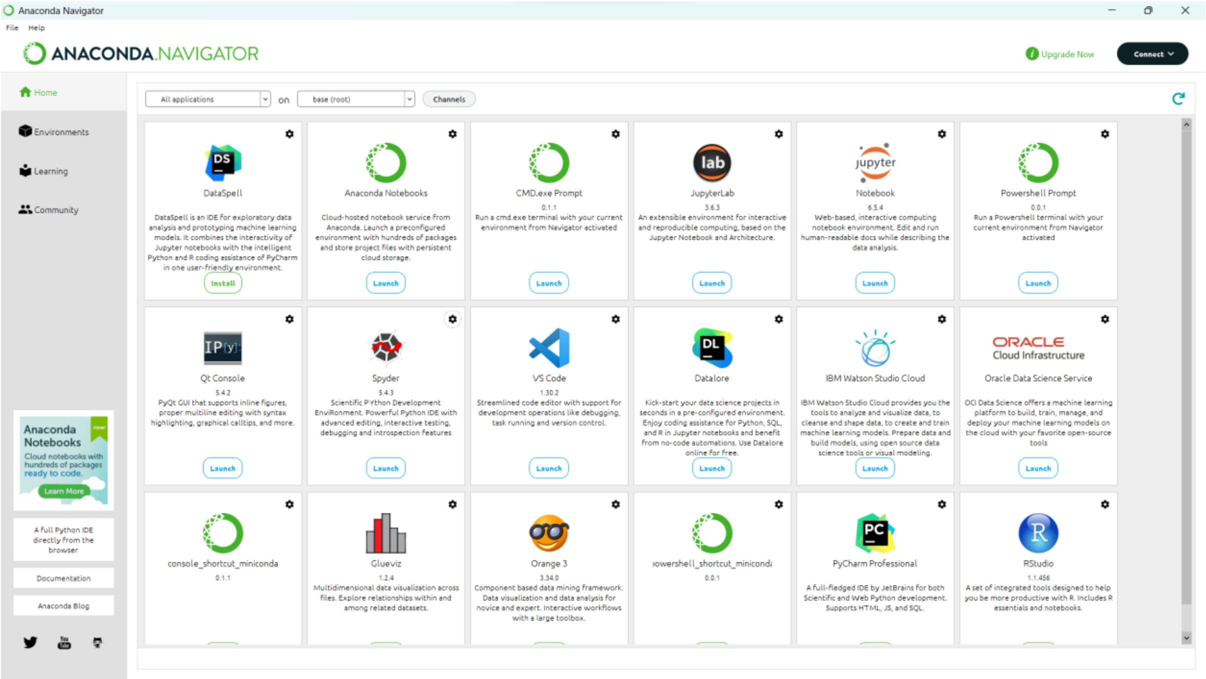
Double click the installer and follow the on-screen instructions. It is advised to follow the recommended options. For further information or troubleshooting you can visit these links:

https://docs.anaconda.com/free/anaconda/install/windows/

https://docs.anaconda.com/free/anaconda/install/mac-os/

Once installed, launch the Anaconda Navigator from the start menu (windows) or launchpad (mac): https://docs.anaconda.com/free/navigator/getting-started/#navigator-starting-navigator

You will see a menu like this:



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.

Anaconda Notebooks

Cloud-hosted notebook service from Anaconda. Launch a preconfigured environment with hundreds of packages and store project files with persistent cloud storage.

CMD.exe Prompt
0.1.1

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

JupyterLab
3.6.3

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

Notebook
6.5.4

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
5.4.2

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

Spyder
5.4.3

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

VS Code
1.30.2

Streamlined code editor with support for development operations like debugging, task running and version control.

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.

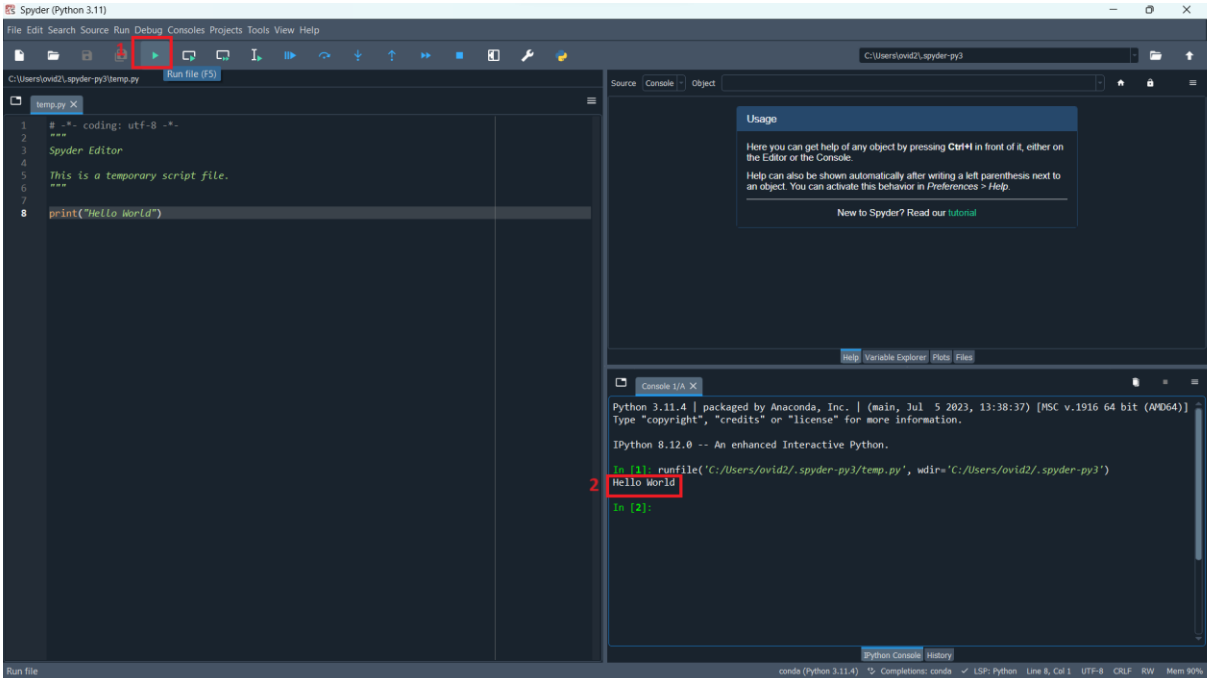
Launch the Spyder IDE from the menu.

Task 2: Run a python script from Spyder

Once Spyder is launched, you will see a *temp.py* file is already created. Add the following line in that file:

```
print(“Hello World”)
```

Click the run button as shown in the figure. You will see the output in the console.



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

This is a temporary script file.
"""

print("Hello World")
```

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

```
Python 3.11.4 | packaged by Anaconda, Inc. | (main, Jul  5 2023, 13:38:37) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.12.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/ovid2/.spyder-py3/temp.py', wdir='C:/Users/ovid2/.spyder-py3')
Hello World

In [2]:
```

Replace the added line with the following lines:

```
import sys

print(sys.version)
```

You will see it prints the version of python that has been installed.

Task 3: Execute and modify the script l*ab1.py*

Download the file lab1.py from Moodle and open it in Spyder. This file contains a Python script that adds two numbers.

1. The script currently adds two predefined numbers. Modify these numbers and observe the output.
2. What do you think about the sections of the file that appear after # or within “ ”? What happens if you modify them? Discuss this with your tutor during the lab. These are called comments, and we will explore them further in our lectures.
3. Can you modify the script to display the result of subtraction, multiplication, and division as well?

Part B: Programming Exercises

Exercise 1: Favourite Quote

Write a program that first prompts the user for their name. Next, it asks the user to input a famous quote, followed by the name of the quote’s author. The program then prints a message like the following:

Hello *Jon Doe*, your favourite quote is “To be, or not to be, that is the question” by *William Shakespeare*.

Note: There is no need to print the names in italic font.

Exercise 2: Favourite Number

Write a program that prompts the user to enter their favourite **even** number between **2 and 100**. Once the number is provided, the program will print three mathematical expressions (addition, subtraction, and division) that result in the user’s favourite number. The expressions should follow this format:

```
first_number + second_number = favourite_number

first_number - second_number = favourite_number

first_number / second_number = favourite_number
```

For example, if the user inputs **50**, the program might display:

```
10 + 40 = 50

55 - 5 = 50

100/2 = 50
```

Note: There are infinite combinations that can result in the user's favourite number (*e.g.,* -999 + 1049 = 50 is a valid expression). You do not need to use the same numbers for all three operations (addition, subtraction, and division). The goal is simply to display valid expressions that evaluate to the user's input number.

Exercise 3: Swap Numbers

To swap the values of two variables, it's common to use a third, temporary variable. For example:

```
first_var = 5

second_var = 10

temp_var = first_var

first_var = second_var

second_var = temp_var
```

In this case, the value of `first_var` becomes 10, and `second_var` becomes 5, using `temp_var` to temporarily hold the value.

However, can you write a program that swaps the values of two variables without using a third variable (`temp_var`)?

Exercise 4: Quadratic Solver

The solution for a quadratic equation in the form $ax^2 + bx + c = 0$ is given by:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Write a program that prompts the user to input the values of values of a, b and c, and then calculates the value of x.

Note: Assume that the quadratic equation has a real solution. When testing your program, provide a significantly larger value for b compared to a and c. For example, you might use $b = 40$, $a = 3$, and $c = 2$.

****Hint:** To calculate the square root of a variable, use the `sqrt()` function from the math library. First, include the line `import  math`, then call the function like this: `math.sqrt(my_var)`. For example:

```
import math

my_var = 16

print(math.sqrt(my_var))
```

This will output 4.0.

Exercise 5: Perimeter of an Ellipse

There are no precise formulas for calculating the perimeter of an ellipse. However, the renowned mathematician Ramanujan provided an excellent approximation formula for the perimeter of an ellipse:

$$P = \pi(a+b)\left(1 + \frac{3h}{10 + \sqrt{4 - 3h}}\right)$$

Where $h = \frac{(a-b)^2}{(a+b)^2}$. Here a and b represent the lengths of the major and minor axes of the ellipse, respectively.

Can you write a program that prompts the user to input the values for the major and minor axes and calculates the perimeter?

Note: Consider $\pi = 3.141592$

Exercise 6: String Reverse

Write a program that prompts the user to enter a string that is exactly 10 characters long. The program should then reverse the string and display the reversed version.

For example, if the user inputs **"fried rice",** the output should be **"ecir deirf"**.

Note: You do not need to use any library functions in Python. Instead, focus on using the concepts of indexing that were discussed in the lecture to achieve the desired result.

Exercise 7: Lowercase to Uppercase

Each character in a string has a numeric representation known as its ASCII value. You can find the ASCII value of a character using the `ord()` function. For example:

```
print(ord("h"))
```

This command will display the ASCII value of the character "h". You can also store the ASCII value in a variable like this:

```
ascii_val = ord("h")
```

Additionally, you can print a character using its ASCII value with the `chr()` function. For instance:

```
print(chr(72))
```

This will output “**H**”.

Using the concept of ASCII values, your task is to convert a five-character long lowercase string (without any whitespace) to uppercase. First, prompt the user to input a lowercase string. Then, convert it to an uppercase string using the ASCII values. For example, the input “**hello**” should be converted to “**HELLO**”.

Exercise 8: Elementary School Addition (Optional)

Write a program that accepts two three-digit numbers as input (the numbers should be within the range of 100 to 499) and performs the addition using the method taught in elementary school. For instance, when adding the numbers 327 and 377, the addition should proceed from the rightmost digits, carrying over values as needed to the left.

$$\begin{array}{ccc} {\scriptstyle 1} & {\scriptstyle 1} & \\ 3 & 2 & 7 \\ 3 & 7 & 7 \\ \hline 7 & 0 & 4 \end{array}$$

Note: Ensure that you always input a three-digit number when testing the program. The numbers should not be added directly using $a + b$; instead, you must implement the addition process manually, mimicking the elementary technique, and then display the result.

****Hint:** You will need to use Python's modulo operator for this problem. The modulo operator returns the remainder after division. For example, $a\ = 15\%4$ will assign the value 3 to a.