technoteach

technocamps

Llywodraeth Cymru
Welsh Government

Prifysgol Abertawe
Swansea University

CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

PRIFYSGOL BANGOR UNIVERSITY

Cardiff Metropolitan University | Prifysgol Metropolitan Caerdydd

University of South Wales
Prifysgol De Cymru

Cyngor Cyllido Addysg Uwch Cymru
Higher Education Funding Council for Wales
hefcw

Prifysgol Cymru Y Drindod Dewi Sant
University of Wales Trinity Saint David

PRIFYSGOL ABERYSTWYTH UNIVERSITY

Prifysgol Glyndŵr Wrecsam | Wrexham Glyndŵr University

institute of CODING in wales
technocamps

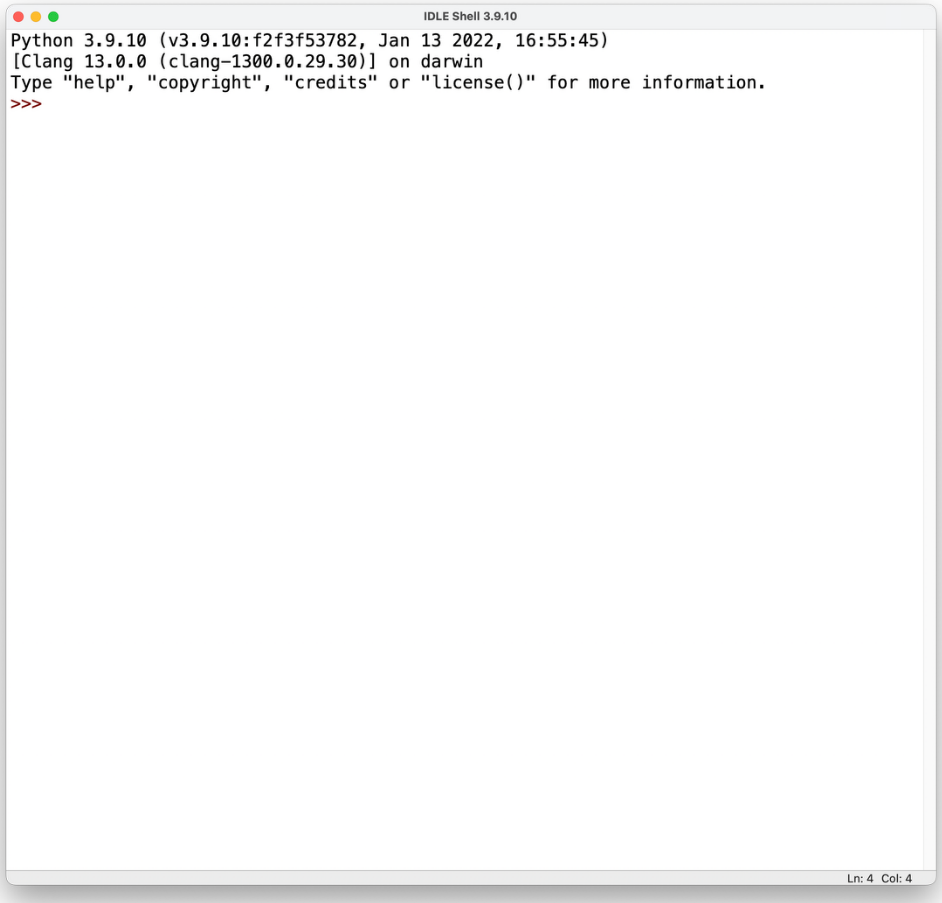
Python: Introduction

IDLE Shell

Opening IDLE will launch the IDLE shell.

The shell is where your basic programs will print out any information, as well as where we will input any information as our programs become more complex.

The IDLE shell can be a useful tool for performing quick calculations.



```
Python 3.9.10 (v3.9.10:f2f3f53782, Jan 13 2022, 16:55:45)
[Clang 13.0.0 (clang-1300.0.29.30)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>>
```

IDLE Shell

While the shell is a useful (and essential) tool, it runs line by line.

This means that if we want to do more complicated things, we must write our programs somewhere else.

Go to File > New File to create a new Python File.

First Program

To begin we will once again make use of the print function.

Write out the following statements exactly as show below:

```
print(“1 + 2”)
print(1+2)
```

Save Your File

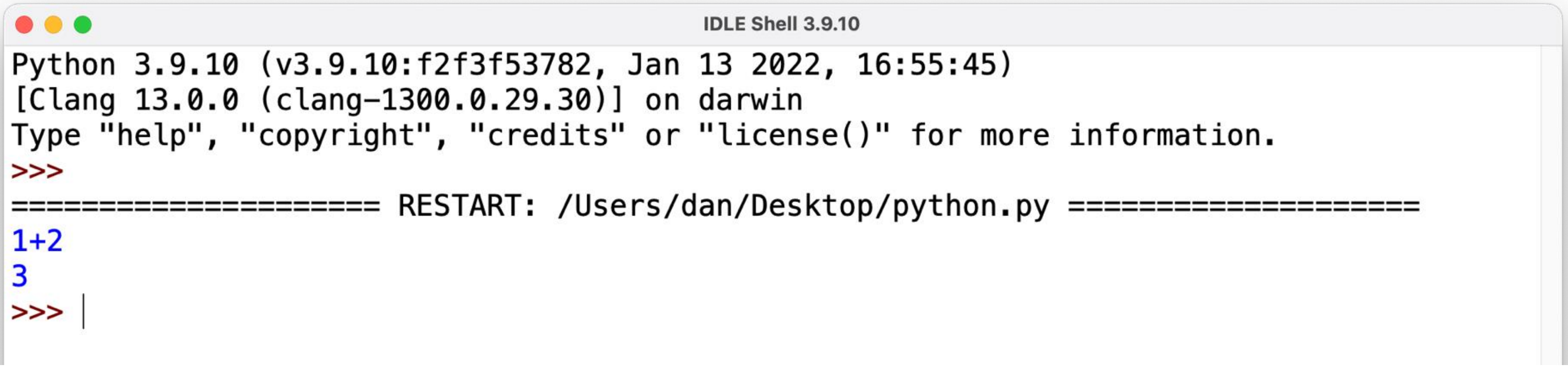
Before Python runs any program, it must first be saved!

- Be organised!
- Your 'source code' is stored in '*.py*' files
- Create one folder per program
  - This can have many .py files
- Be sure you know where your IDE stores your files!
- Backup your work.

Go to Run > Run Module to run your program.

First Program

Your output should look something like this:



```
Python 3.9.10 (v3.9.10:f2f3f53782, Jan 13 2022, 16:55:45)
[Clang 13.0.0 (clang-1300.0.29.30)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==================== RESTART: /Users/dan/Desktop/python.py ====================
1+2
3
>>> 
```

Python will read any text in *"speech marks"* as a string, anything outside of speech marks it sees as a variable.

Variables

If we try to run the following code we will get an error:

```
print(“Hello”)
print(hello)
```

Can you work out why?

Variables

If instead we run the following code:

```
hello = “I don’t want to say hello world!”
print(hello)
```

Errors in Python

1) **Compile-time Errors**
   - Syntax Errors
     - Spelling, capitalization, punctuation
     - Correct order, matching of parenthesis, quotes…
   - No executable program is created by the compiler
   - Correct first error listed, then compile again.
     Repeat until all errors are fixed

2) **Run-time Errors**
   - Logic Errors
   - Program runs, but produces unintended results
   - Test your program regularly to make sure it is
     producing expected results

Syntax in Python

To use, or call, a function in Python you need to specify:

- The name of the function that you want to use (in the previous example the name was `print`)
- Any values (arguments) needed by the function to carry out its task (in this case, `"Hello World!"`).
    - Arguments are enclosed in parentheses and multiple arguments are separated with commas.

Syntax in Python

What happens if you…

• Misspell a word:	`primpt("Hello World!")`
• Use wrong case:	`Print("Hello World!")`
• Leave out quotes:	`print(Hello World!)`
• Mismatch quotes:	`print("Hello World!’)`
• Don’t match brackets	`print(‘Hello’`

Try it to see what error messages are generated

Variables

Most computer programs hold temporary values in named storage locations.

Programmers name them for easy access

There are many different types of storage to hold different things

You ‘define’ a variable by telling the compiler:

- What name you will use to refer to it
- What is the initial value to be stored in it

Defining Variables

To define a variable, you must specify an initial value.

We use the ‘assignment statement’ (with an '=') to place a new value into a variable:

```
a = 6
amount = 60
number_of_students = 150
```

This copies the value on the right side into the variable on the left side

Naming Variables

Variable names should describe the variables' purpose i.e. ‘canVolume’ is better than ‘cv’

Use These Simple Rules:

1) Variable names must start with a lower case letter or the underscore (_) character. You may then continue with letters (upper or lower case), digits or the underscore
2) You cannot use other symbols (? or %...) or spaces
3) Separate words with ‘camelCase’ or 'snake_case’ notation
    - Be consistent! Choose a style and stick with it.
4) Avoid using Python reserved words

Using Variables

The print command can accept many comma separated arguments.

Create and amend the following program:

```
name = “Dan”

age = 30

print(“My name is”, name, “and I am”, age, “years old!”)
```

Python Maths

The symbols used to perform mathematical operations in Python are not those we use day to day but are fairly standard across programming languages. The primary ones are:

Addition	+	Multiplication	*
Subtraction	-	Division	/
Exponent	**		

Using Maths

We can write the following program to calculate the relationship of two numbers:

```
number_1 = 2
number_2 = 4

print("The sum is = ", number1 + number2)
print("The product is = ", number1 * number2)
print("The difference is = ", number1 - number2)
print("The ratio is 1:", number1 / number2)
```

Our Problem

Soft drinks are sold in cans and bottles. A store offers a six-pack of 12-ounce cans for the same price as a two-litre bottle. Which should you buy?

List of variables:

- Number of cans per pack
- Ounces per can
- Conversion rate (fl. oz to litres)

CONSTANTS

A constant is a variable whose value should not be changed after it's assigned an initial value.

```
OUNCE_TO_LITRE = 0.028
```

It is good style to use named constants to explain numerical values to be used in calculations. Which is clearer?

```
can_vol_litres = can_vol_ounces * 0.028
```

or

```
can_vol_litres = can_vol_ounces * OUNCE_TO_LITRE
```

A programmer reading the first statement may not understand the significance of the number 0.028. Additionally, if the constant is used in multiple places and needs to be changed, you only need to change it in one place.

Naming Constants

It is customary to use all UPPER_CASE letters for constants to distinguish them from variables.

```
BOTTLE_VOLUME = 2           #Constant
MAX_SIZE = 100              #Constant
tax_rate = 5                #Variable
```