More Environment Set-Up

Add PIP to PATH

What's it mean? The package manager for Python.

Why do I need it? Tell your OS where to look for PIP (the package manager for Python).

How do I do it? Windows key + R → type 'sysdm.cpl' → hit 'OK' → 'Advanced' → 'Environment Variables' → 'System Vairables' → 'Path' → 'Edit' → Add the path to your Python Scripts (typically looks something like: `C:\Users\[name]\AppData\Local\Programs\Python\Python313\Scripts`)

Libraries

Helpful Libraries

Native Libraries (a.k.a. Built-In or Standard Libraries)

Name	Use
`math`	basic number manipulations
`random`	randomising data and generating random values
`re`	quick text manipulations and pattern matching

Learn more about the Python native library here.

External Libraries (a.k.a. Third-Party Libraries)

Name	Use
`numpy`	more convenient number operations (incl. advanced stuff)
`pandas`	handling tables and databases
`matplotlib`	plotting data

Explore more Python external libraries here.

How to Run Native Libraries

1. In your .py file, import library under an arbitrary alias by typing `import [library_name] as [alias]`.
2. Use desired functions by calling `[alias].[function]`.

How to Run External Libraries

1. In your VSCode terminal (ctrl + '), run `pip install [library_ name]`.
2. In your .py file, import library under an arbitrary alias by typing `import [library_name] as [alias]`.
3. Use desired functions by calling `[alias].[function]`.

Errors & Debugging

Common Errors and Exceptions

Type	Occurs When
`SyntaxError`	Python cannot compile script
`NameError`	you call a variable you haven't named yet
`ZeroDivisionError`	you try to divide by zero
`IndexError`	you iterate out of the bounds
`TypeError`	an operation or a function is fed an argument of the wrong type

Learn more about Python built-in exceptions (and exception hierarchies!) here.

Catching Errors

`print`

```
...
print([expression you want to double-check])
...
```

`try-except-else-finally`

```
try:
    [this block will try to execute first]
except ##[opt_exception_name]:
    [this block will execute if the above exception occurs]
    ...
##else:
    ##[this block will execute if no errors occur]
##finally:
    ##[this block will execute regardless of whether or not errors occur]
```

Learn more about `try-except` statements here.

`unittest`

```
import unittest

[all_functions]

class TestAddFunction(unittest.TestCase):
    def [test_name](self):
        self.[assert_statement]([assert_statement_args])

if __name__ == '__main__':
    unittest.main()
```

Outputs `.` (a literal dot) if test passed and `F` if test failed. Learn more about Python `unittest` here. Learn more about `assert` statements here.

Functions: General Tips.

1. **Sometimes you may wish to rewrite more instances at the same time.** Hold the ‘alt’ key and click on all instances you’d like to rewrite simultaneously to insert multiple cursors.