# Python Language Fundamentals

### Overview

In this lab, you'll get an opportunity to solidify your understanding of Python language essentials. You'll use various built-in types such as integers, floats, complex numbers, strings, and Boolean conditions.

You'll also get to explore the capabilities of some standard classes and functions in the Python standard library. You'll probably want to consult the online Python documentation to help you out here.

### Source folders

| | |
|---|---|
| Student folder : | `C:\PythonDev\Student\02-PythonLang` |
| Solution folder: | `C:\PythonDev\Solutions\02-PythonLang` |

### Roadmap

There are 4 exercises in this lab, of which the last two exercises are "if time permits". Here is a brief summary of the tasks you will perform in each exercise; more detailed instructions follow later:

1. Using numeric types
2. Using strings
3. Using complex numbers (if time permits)
4. Additional suggestions (if time permits)

## Exercise 1: Using numeric types

Create a new module, and import the `math` module ready to do some mathematical calculations. Ask the user to enter the radius of the circle or sphere, r, as a floating point value. Then perform the following calculations:

- The diameter of a circle (2r).
- The area of a circle ($\pi r^2$)
- The circumference of a circle ($2\pi r$)
- The area of a sphere ($4\pi r^2$)
- The volume of a sphere ($^4/_3\ \pi r^3$). By the way, in Python 2, 4//3 would evaluate to 1 rather than 1.33333 (!).

Output the results nicely. Run the code several times with different values for the radius, to ensure you get the correct results.

.

## Exercise 2: Using strings

Create a new module, and prompt the user to enter the following textual information:

- Honorific, e.g. Mr. or Mrs.
- First name
- Last name

Create a variable to hold their name in the following format, and then output the value:

- Honorific
- A space character
- First letter of their first name, in upper case
- A period, followed by a space
- Last name, ensuring that it starts with a uppercase letter and then all lowercase thereafter

Hint: You might want to look up the `String` class in the Python 3.7 online documentation at `https://docs.python.org/3.8/library/string.html`:

- `format()`
- `title()`
- `upper()`

Once you've got that working to your satisfaction, add some more code to ask the user to input their email address and (fictional!) 4-digit pin code. Get these values as strings, and then validate them as follows:

- The e-mail address must contain an `"@"` character. The easiest way to test if a character is in a string is as follows (we've assumed your email address variable is named `email`):

  ```
  "@" in email
  ```

- The pin code must be 4 characters long. The easiest way to test the length of a string and other sequences is via the built-in `len()` function (we've assumed your pin code variable is named `pincode` here):

  ```
  len(pincode) == 4
  ```

- The pin code must contain just digits. Use the `String` class's `isdigit()` method here:

  ```
  pincode.isdigit()
  ```

If everything is valid, output the details in lowercase. If anything is invalid, output an error. Use an `if: else:` construct here.

## Exercise 3: Using complex numbers (if time permits)

This exercise is quite mathematical – it's based on quadratic equations and how you solve them. If your maths isn't what it used to be, you might want to skip this question and give yourself a different challenge instead (e.g. experiment with functions in the `math` module, explore the `String` class, etc.)

If you're still reading, a quadratic equation is an equation involving $x^2$ such as the following:

$ax^2 + bx + c = 0$

To solve a quadratic equation, you have to find values for x such that when you plug these values into the equation, the result is 0. In general, there are two values for x that solve a quadratic equation. Mathematically, we call these the *roots* of the equation. There's a general formula for finding the roots of a quadratic equation:

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

Your task is to solve a quadratic equation. The first step is to ask the user to enter values for a, b, and c (as integers, let's say). Then you have to plug the values into the formula, to find the two values for x that solve the equation. Note that it's very important whether the value of ($b^2 – 4ac$), known as the *discriminant*, is positive or negative.

- If the discriminant is positive, you can get its square root via the `math.sqrt()` function. You can then add the value and subtract the value as shown in the formula, to get the two roots for the equation.

- If the discriminant is negative, it's a bit more tricky. You can't find the square root of a negative number. In this case, we say that the equation doesn't have real roots. Or to put it another way, the roots will be complex numbers. What you have to do here is convert the discriminant to a positive number (so you can find its square roots), and then treat the result as the imaginary part of a complex number.

Have a go at implementing this in Python. Remember, we've got full solutions in the *Solutions* folder if you need some hints.

## Exercise 4: Additional suggestions (if time permits)

- Create a package containing several simple modules, and write some client code to use these modules.

- Explore regular expressions in Python. For more info about regular expressions, see `https://docs.python.org/3.8/library/re.html`.