2 - Numbers

### General Teacher Notes

#### Key Concepts/Vocab/Misconceptions

This set of exercises will take us through several features within Python that let us deal with numbers, logic and arithmetic.

Each feature works in a fairly unique way.

As such, I’ve put the specific notes in the ‘Teacher Notes’ section for each task. They seem to fit better there.

### 1 - Data Types

#### TEACHER NOTES

The first feature is **data types.** This doesn’t strictly deal with only numbers, but it fits best here.

A data type is a setting for a variable that tells it what sort of data to accept. The three data types we will use in these exercises are:

* String - Text
* Int - Whole numbers
* Float (also called *Real* in other coding languages) - numbers with decimals.

There are more data types, but we will focus on these three for now.

Be careful, all data in string variables is treated as text, even if it’s numeric characters.

It is good practice to use the same data type when comparing variables in a condition.

We can use == and != operators in conditions that compare strings, ints or floats.

The string data type cannot be used with arithmetic operators (+,-,\*,/), or > & < comparisons.

The **type()** function is used to return the data type of some data or a variable.

#returns int but does not output anything

type(6)

#returns float

type(6.0)

#outputs float

print(type(6.0)

#As above, but using a variable as the parameter

num1 = 6.0

print(type(num1))

One data type can be converted into another using a technique called **casting.** We have come across this before in the ‘Intro To Python’ unit.

#Converts the parameter to an integer

int(variable)

#Converts the parameter to an float

float(variable)

#Converts the parameter to a str

str(variable)

All input is treated as strings. If we want to use our input as integers or floats (eg to do maths or logical comparisons) we have to cast it.

#Gets user input, casts it to an integer and stores in the num1 variable.

Num1 = int(input(“Enter a number”))

#### Tasks

##### Predict And Run

Task and instructions - <https://repl.it/@MrAColley/2201-Type-and-Cast-Predict-Run>

Example solution - <https://repl.it/@MrAColley/2201-Type-and-Cast-Predict-Run-Solution>

##### Investigate And Modify

Task and instructions - <https://repl.it/@MrAColley/2202-Type-and-Cast-Investigate-and-Modify>

Example solution - <https://repl.it/@MrAColley/2202-Type-and-Cast-Investigate-and-Modify-Solution>

##### Make

Task and instructions - <https://repl.it/@MrAColley/2203-Type-and-Cast-Make>

Example solution - <https://repl.it/@MrAColley/2203-Type-and-Cast-Make-Solution>

### 

### 

### 2 - Random Numbers

#### TEACHER NOTES

Python has a great feature that automatically generates random numbers.

This can be useful for lots of applications, such as rewards in games that have rarity - generate a random number between 1 and 100. IF the number is 1-50 then you get common reward, 51-75 gets uncommon, 76-90 rare etc…

* Before we can use random numbers, we have to import the random **library** (a set of pre written functions that we can use to generate random numbers).
* You only have to import a library once, then you can use it as many times as necessary.
* The function we are going to use is **randint(x,y)** which generates random integers.
* randint(x,y) takes two parameters. x is the lowest possible random number to be picked, y is the largest.

#Import the random library - common convention is to put imports at the top of your code.

import random

# Generate a random number between 1 and 5

random.randint(1,5)

# As above but assign the returned value to a variable

num1 = random.randint(1,5)

# As above but output the result instead of assign the returned value to a variable

print(random.randint(1,5))

#### Tasks

##### Predict And Run

Task and instructions - <https://repl.it/@MrAColley/2204-Random-Predict-and-Run>

Example solution - <https://repl.it/@MrAColley/2204-Random-Predict-and-Run-1>

##### Investigate And Modify

Task and instructions - <https://repl.it/@MrAColley/2205-Random-Investigate-and-Modify>

Example solution - <https://repl.it/@MrAColley/2205-Random-Investigate-and-Modify-Solution>

##### Make

Task and instructions - <https://repl.it/@MrAColley/2206-Random-Make>

Example solution -

### 3 - Modulus

#### TEACHER NOTES

Modulus performs integer division and returns the **remainder only**.

It uses the % operator.

For example:

3 % 1 = 3 remainder 0, so the value returned would be 0.

5 % 2 = 2 remainder 1, so the value returned would be 1.

14 % 4 = 3 remainder 2, so the value returned would be 2.

It is really useful for working out whether a number is divisible by another one - if the remainder returned is 0 then the number is exactly divisible. So to calculate if a number is in the 2 times table you would mod it by 2, if the remainder is 0 then the number is in the 2 times table.

#### Tasks

##### Predict And Run

Task and instructions - <https://repl.it/@MrAColley/2207-Modulus-Predict-and-Run>

Example solution - <https://repl.it/@MrAColley/2207-Modulus-Predict-and-Run-Solution>

##### Investigate And Modify

Task and instructions - <https://repl.it/@MrAColley/2208-Modulus-Investigate-and-Modify>

Example solution - <https://repl.it/@MrAColley/2208-Modulus-Investigate-and-Modify-Solution>

##### Make

Task and instructions - <https://repl.it/@MrAColley/2209-Modulus-Make>

Example solution - <https://repl.it/@MrAColley/2209-Modulus-Make-Solution>