```
# -*- coding: utf-8 -*-
"""Introduction to Python.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1GqXb-5lqK8CyE8iEmquvl_QlKGlWxUiD

# **Introduction to Colab**


*   Colab is a Python development environment that runs in the browser using Google Cloud
*   For example, to print "Hello World", just hover the mouse over [ ] and press the play button to the upper left. Or press shift-enter to execute.

## Uploading files from your local file system
`files.upload` returns a dictionary of the files which were uploaded.
The dictionary is keyed by the file name and values are the data which were uploaded.
"""

from google.colab import files

uploaded = files.upload()

for fn in uploaded.keys():
  print('User uploaded file "{name}" with length {length} bytes'.format(
      name=fn, length=len(uploaded[fn])))

"""## Downloading files to your local file system

`files.download` will invoke a browser download of the file to your local computer.
"""

from google.colab import files

with open('image001.txt', 'w') as f:
  f.write('some content')

files.download('image001.txt')

"""## Mounting Google Drive locally

The example below shows how to mount your Google Drive on your runtime using an authorization code, and how to write and read files there. Once executed, you will be able to see the new file (`foo.txt`) at [https://drive.google.com/](https://drive.google.com/).

This only supports reading, writing, and moving files; to programmatically modify sharing settings or other metadata, use one of the other options below.

**Note:** When using the 'Mount Drive' button in the file browser, no authentication codes are necessary for notebooks that have only been edited by the current user.
```

```
"""

from google.colab import drive
drive.mount('/content/drive')

with open('/content/drive/My Drive/foo.txt', 'w') as f:
  f.write('Hello Google Drive!')
!cat /content/drive/My\ Drive/foo.txt

drive.flush_and_unmount()
print('All changes made in this colab session should now be visible in Drive.')

"""### Colab is a virtual machine you can access directly. To run commands at the VM's terminal, prefix the line with an exclamation point (!)."""

print("\nDoing $ls on filesystem")
!ls -l
!pwd

"""Create a code cell underneath this text cell and add code to:


*   List the path of the current directory (pwd)
* Go to / (cd) and list the content (ls -l)
"""

!pwd
!cd /
!ls -l
print("Hello")

"""GPU usage is provided free of charge for some hours of usage every day.

**Using GPUs**
* Runtime -> Change runtime type -> Hardware accelerator -> GPU

**Some final words on Colab**
*   You execute each cell in order, you can edit & re-execute cells if you want
*   Sometimes, this could have unintended consequences. For example, if you add a dimension to an array and execute the cell multiple times, then the cells after may not work. If you encounter problem reset your environment:
  *   Runtime -> Restart runtime... Resets your Python shell
  *   Runtime -> Restart all runtimes... Will reset the Colab image, and get you back to a 100% clean environment
* You can also clear the output in the Colab by doing: Edit -> Clear all outputs
* Colabs in this course are loaded from GitHub. Save to your Google Drive if you want a copy with your code/output: File -> Save a copy in Drive.

## **Introduction to Python**

> Python is a high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991
> **Popular Version 2.7 and 3.5 or 3.6 or 3.7**

> Current stable release 3.8.5

> The extension of python file is .py

> To run the file type python finel_name.py
"""

#In google colab : To run any linux command please use "!" befor the command
#To Check the version install in your system
!python3 --version

#Getting Started
# Print Hello world
print("Hello, World!")

"""# > **Python Indentation**

> Indentation refers to the spaces at the beginning of a code line.

> Python uses indentation to indicate a block of code.
"""

# Python indentation example
if 5 > 2:
  print("Five is greater than two!")

"""# **Comment in a code**


"""

# Hi I am a comment line. My identity is '#'

"""# **Variable**

> A variable name must start with a letter or the underscore character

> A variable name cannot start with a number

> A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _ )

> Variable names are case-sensitive (age, Age and AGE are three different variables)
"""

#Legal variable names:
myvar = "John"
my_var = "John"
_my_var = "John"
myVar = "John"
```

```
MYVAR = "John"
myvar2 = "John"

#Illegal variable names:
#2myvar = "John"
#my-var = "John"
#my var = "John"

# Multiple variables
x, y, z = "Orange", "Banana", "Cherry"
print(y)
print(x)
print(z)

#

# Print the value of the variable with a text
print("Hello, the color of x is " + x)

"""# **Build in data types**
**Text Type:   str**

**Numeric Types:      int, float, complex**

**Sequence Types:     list, tuple, range**

Mapping Type:         dict

Set Types:      set, frozenset

**Boolean Type:       bool**

Binary Types: bytes, bytearray, memoryview
"""

# How to know the type of a variables

a = 5
b = 5.5
c = 1j
d = True
print(type(x))
print(type(a))

# Setting the Specific Data Type/Type Conversion

a = float(a)
print(type(a))

# Type casting
x = 5.5
print(int(x))
```

```
# Important in terms of ML
# Random number generation
# Python has a built-in module called random that can be used to make random numbers. We need
to import that module

import random
print(random.randrange(1, 10))

# String with array
a = '''Lorem ipsum dolor sit amet,
consectetur adipiscing elit,
sed do eiusmod tempor incididunt
ut labore et dolore magna aliqua.'''
# Length of the string
print(len(a))
# Print some specific rang of characters
print(a[0:10])
# starting the count from the end of the string:
print(a[-10:-1])

# Remove any whitespace from the beginning or the end of the string
a = " Hello, World! "
print(a.strip())
# returns the string in lower/upper case:
print(a.lower())
print(a.upper())
# Replcaement in a string
print(a.replace("H", "J"))
# Split the string w.r.t some character
print(a.split(","))
# Search a character or phrase in a string
x = "He" in a
print(x)
# String concatanation
b = "ML Lab"
print(b+a)
# add something between two string
print(b + " rrr " + a)

# Combine string and numerical value together
x = 50
#txt = "My name is John, I am " + x
txt = "My name is John, and I am {}"
print(txt.format(x))

"""## Python operators

> +     Addition        x + y
-       Subtraction     x - y
*       Multiplication  x * y
+       Division        x / y
```

```
+	Modulus		x % y
+	Exponentiation		x ** y
And many more (Eg: Check some assignment operators)
"""

# Python list
# List is a collection which is ordered and changeable. Allows duplicate members.
mylist = ["apple", "banana", "cherry"]
print(mylist)
print(mylist[0])
# add item in the list
mylist.append("new member")
print(mylist)

# Tuple
# Tuple is a collection which is ordered and unchangeable. Allows duplicate members, you cn not add or remove item
mytuple = ("apple", "banana", "cherry")
print(mytuple)
print(mytuple[1])

# if else in python
a = 33
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")

# while loop
i = 0
while i < 6:
  i += 1
  if i == 3:
    continue
  print(i)

# For loop
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)

# Function
def my_function(fname):
  print(fname + " Refsnes")

my_function("Emil")
my_function("Tobias")

# Arbitrary Arguments, *args
# If the number of arguments is unknown, add a * before the parameter name:
```

```
def my_function(*kids):
  print("The youngest child is " + kids[2])

my_function("Emil", "Tobias", "Linus")
#my_function("Emil")
```