3 Days Training on Python3

Day 1 : Module 1

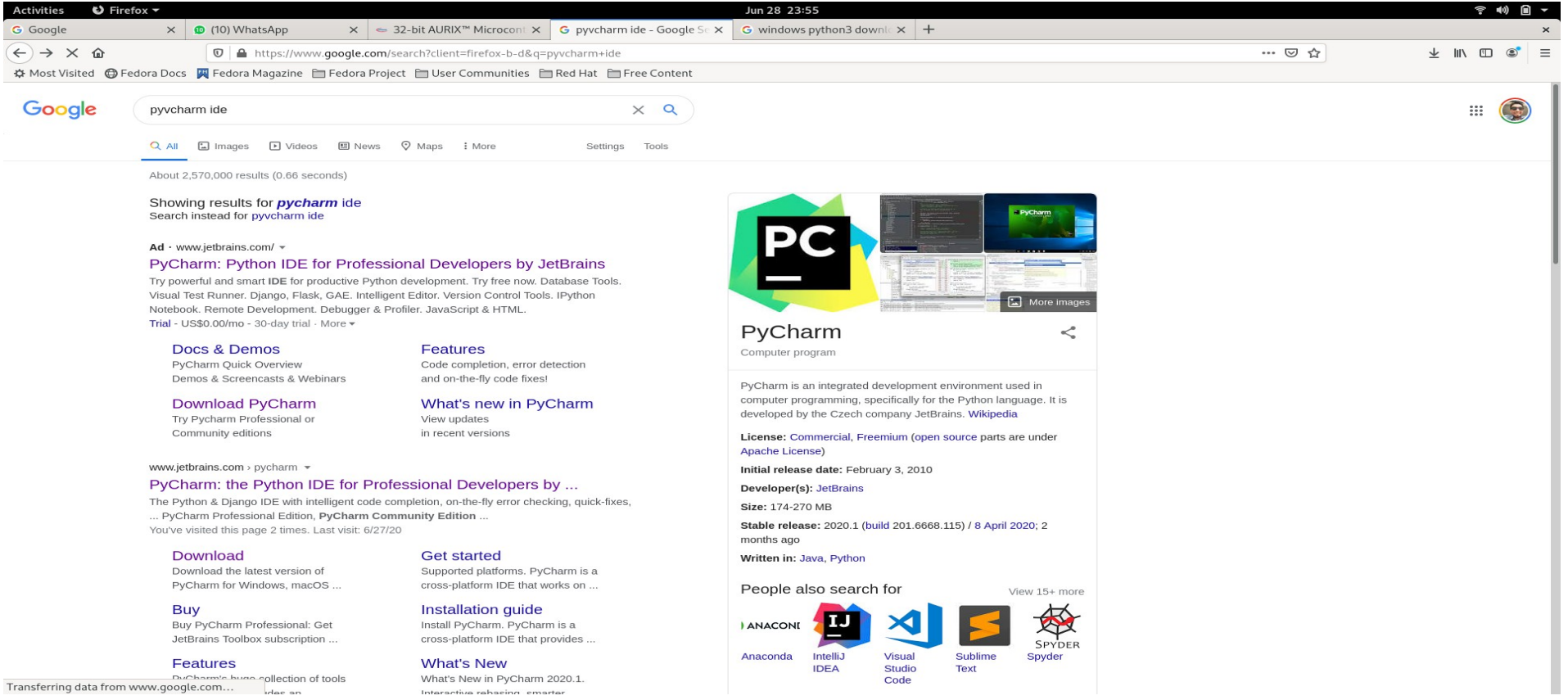
Muhammad Saufy Rohmad

Module 1 (90 minutes)

Objectives

1. Introduction to Python
2. Setting Up the Python Environment
3. Running Salam World
4. Python strings

Download and Install pycharm IDE



1. Introduction to Python

- Python is a general-purpose programming language in a similar vein to other programming languages that you might have heard of such as C++, JavaScript or Microsoft’s C# and Oracle’s Java.
- It has been around for some considerable time having been originally conceived back in the 1980s by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands.
- As a language it has gained in interest over recent years, particularly within the commercial world, with many people wanting to learn the language.

1. Introduction to Python(2)

- The increased interest in Python is driven by several different factors:
  - Its flexibility and simplicity which makes it easy to learn.
  - Its use by the Data Science community where it provides a more standard programming language than some rivals such as R.
  - Its suitability as a scripting language for those working in the DevOps field where it provides a higher level of abstraction than alternative languages traditionally used.
  - Its ability to run on (almost) any operating system, but particularly the big three operating systems Windows, MacOS and Linux.
  - The availability of a wide range of libraries (modules) that can be used to extend the basic features of the language.
  - It is Free!

1. Introduction to Python(3)

- Currently there are two main versions of Python called Python 2 and Python 3.
- Python 2 was launched in October 2000 and has been, and still is, very widely used.
- Python 3 was launched in December 2008 and is a major revision to the language that is not backward compatible.
- The issue between the two versions can be highlighted by the simple print facility:
  - In Python 2 this is written as print 'Hello World'
  - In Python 3 this is written as print ('Hello World')
- Python is an example of a hybrid programming language as it allows you to write very procedural code, to use objects in an object oriented manner and to write functional programs.

1. Introduction to Python(4)

- There are several ways in which you can run a Python program, including
  - Interactively using the Python interpreter
  - Stored in a file and run using the Python command
  - Run as a script file specifying the Python interpreter to use within the script file
  - From within a Python IDE (Integrated Development Environment) such as PyCharm.

2. Setting Up Python Environment

- Open your cmd and type python
- 1st step : Downloading python www.python.org
- 2nd step : Running the Installer
- 3rd step : Verify the installation with python command
- 4th step : Run print (‘Hello World’)
- 5th step : Exit with exit() or Ctrl-Z

3. A First Python Program

- Create hello.py
- ***print (‘Hello World’)***
- Run with python hello.py
- Interactive hello world

  print('Hello, world')

  user_name = input('Enter your name: ')

  print('Hello ', user_name)
- Check Each Line what is does

3. A First Python Program(2)

- Variables

print('Hello, world')

name = input('Enter your name: ')

print('Hello', name)

name = input('What is the name of your best friend: ')

print('Hello Best Friend', name)

3. A First Python Program(3)

```
my_variable = 'John'
print(my_variable)
my_variable = 42
print(my_variable)
my_variable = True
print(my_variable)
```

3. A First Python Program(4)

- Python accepted style
  - my_name, your_name, user_name, account_name,count, total_number_of_users, percentage_passed, pass_rate,where_we_live, house_number,is_okay, is_correct, status_flag
- Python not accepted style
  - A, Aaaaa, aaAAAaa,Myname, myName, MyName or MYName,WHEREWELIVE

3. A First Python Program(5)

- Python assignment operator
  - user_name = input('Enter your name: ')
- Python statement
  - print('Hello', user_name)
- Python comment
  - # This is a comment
  - name = input('Enter your name: ')
  - print(name) # this is a comment to the end of the line

4. Python Strings

- In Python a string is a series, or sequence, of characters in order.
- In this definition a character is anything you can type on the keyboard in one keystroke, such as a letter 'a', 'b', 'c' or a number '1', '2', '3' or a special characters such as '\', '[', '$' etc.
- It should also be noted that strings are immutable. Immutable means that once a string has been created it cannot be changed.
- 'Hello'
- 'Hello World'
- 'Hello Andrea2000'
- 'To be or not to be that is the question!'

4. Python Strings(2)

- 'Hello World' or "Hello World" can be used to represent strings
- 'Hello World" and "Hello World' # This is illegal

my_variable = 'Bob'

print(type(my_variable))

4. Python Strings(3)

- String operation – String concatenation

```
string_1 = 'Good'
string_2 = " day"
string_3 = string_1 + string_2
print(string_3)
print('Hello ' + 'World')
```

4. Python Strings(4)

- String operation – Length of a String

  print(len(string_3))

4. Python Strings(5)

- String operation – Accessing a Character (start with zero)

  my_string = 'Hello World'

  print(my_string[4])

4. Python Strings(6)

- String operation – Accessing a Subset of Characters

  my_string = 'Hello World'

  print(my_string[4]) # characters at position 4

  print(my_string[1:5]) # from position 1 to 5

  print(my_string[:5]) # from start to position 5

  print(my_string[2:]) # from position 2 to the end

4. Python Strings(7)

- String operation – Repeating Strings

  print('*' * 10)

  print('Hi' * 10)

4. Python Strings(8)

- String operation – Splitting Strings

  title = 'The Good, The Bad, and the Ugly'

  print('Source string:', title)

  print('Split using a space')

  print(title.split(' '))

  print('Split using a comma')

  print(title.split(','))

4. Python Strings(9)

- String operation – Counting Strings

  my_string = 'Count, the number of spaces'

  print("my_string.count(' '):",
  my_string.count(' '))

4. Python Strings(10)

- String operation – Replacing Strings

  welcome_message = 'Hello World!'

  print(welcome_message.replace("Hello", "Goodbye"))

4. Python Strings(11)

- String operation – Finding Sub Strings

  print('Edward Alun Rawlings'.find('Alun'))

4. Python Strings(12)

- String operation – Converting other type into strings

  msg = 'Hello Lloyd you are ' + str(21)

  print(msg)

4. Python Strings(13)

- String operation – Comparing String

  print('James' == 'James') # prints True

  print('James' == 'John') # prints False

  print('James' != 'John') # prints True

4. Python Strings(14)

```
some_string = 'Hello World'
print('Testing a String')
print('-' * 20)
print('some_string', some_string)
print("some_string.startswith('H')",
some_string.startswith('H'))
print("some_string.startswith('h')",
some_string.startswith('h'))
print("some_string.endswith('d')", some_string.endswith('d'))
print('some_string.istitle()', some_string.istitle())
print('some_string.isupper()', some_string.isupper())
print('some_string.islower()', some_string.islower())
print('some_string.isalpha()', some_string.isalpha())
print('String conversions')
print('-' * 20)
print('some_string.upper()', some_string.upper())
print('some_string.lower()', some_string.lower())
print('some_string.title()', some_string.title())
print('some_string.swapcase()', some_string.swapcase())
print('String leading, trailing spaces', "xyz".strip())
```

4. Python Strings(15)

- String Formatting

format_string = 'Hello {}!'

print(format_string.format('Phoebe'))

name = "Adam"

age = 20

print("{} is {} years old".format(name, age))

4. Python Strings(16)

- String Formatting

  # Can specify an index for the substitution

  format_string = "Hello {1} {0}, you got {2}%"

  print(format_string.format('Smith', 'Carol', 75))

  #Can use named substitutions, order is not significant

  format_string = "{artist} sang {song} in {year}"

  print(format_string.format(artist='Paloma Faith',

  song='Guilty', year=2017))

4. Python Strings(17)

- String Formatting

  print('|{:<25}|'.format('left aligned')) # The default

  print('|{:>25}|'.format('right aligned'))

  print('|{:^25}|'.format('centered'))

4. Python Strings(18)

- String Formatting

  # Can format numbers with comma as thousands separator

  print('{:,}'.format(1234567890))

  print('{:,}'.format(1234567890.0))

4. Python Strings(19)

- String Templates

  import string

  # Initialise the template with ¢variables that

  # will be substitute with actual values

  template = string.Template('$artist sang $song in $year')

  print(template.substitute(artist='Freddie Mercury', song='The Great Pretender', year=1987))