INTRODUCTION TO

Python

Part 1

At the end of this lecture, you should learn:

- Environment of Python programming
- Python data structures
- Variables
- Comment
- Input & Output Syntax

Introduction

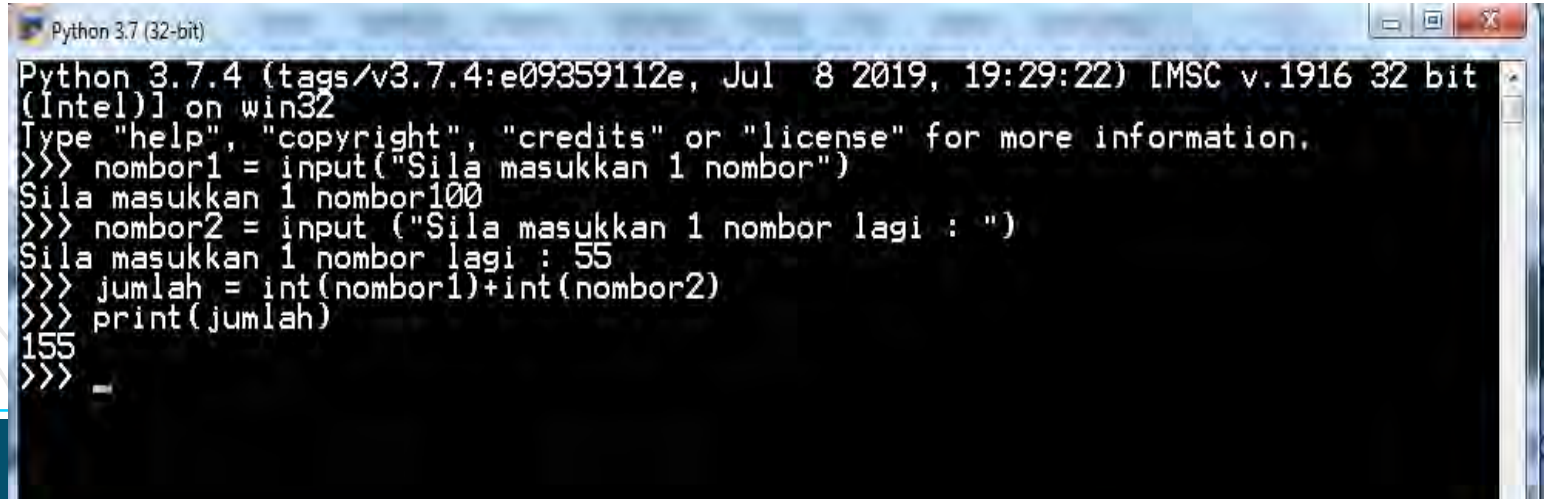
- Python is a powerful multi-purpose programming language created by Guido van Rossum.
- It has simple easy-to-use syntax, making it the perfect language for someone trying to learn computer programming for the first time.
- The main portal to Python and the Python community is http://www.python.org.

Introduction

- You can download Python from the Python and the Python community website for free.

Interact with Python

- **Interactive mode:** In interactive mode, you type instructions to Python one line at a time—much the same way that an operating system (shell) accepts instructions from a command line.
- You can also write short multiline programs or import code from text files or from Python's built-in modules.
- To exit Python's interactive mode, press Control-D (on Mac or UNIX) or Ctrl-Z (on Windows).



Interact with Python

- **IDLE:** The IDLE includes Python's interactive mode and more—tools for writing and running programs and for keeping track of names.

- IDLE is written in Python and shows off Python's considerable abilities.

Example of IDLE text editor

test2.py - C:\Users\Admin\Google Drive\kelas\sem 1 20192020\SPPP2102\nota\Lecture Notes 2019\t...

File Edit Format Run Options Window Help

```
print ("************************")
print ("My first try using Python 3")
print ("************************")
nama = input ("Siapa nama anda : ")
kp = input ("No Kad pengenalan anda : ")
alamat = input ("Masukkan alamat  anda : ")
dob = int( input ("Masukkan tahun lahir : ") )
usia = 2019 - dob
print ("Nama : " ,nama)
print ("KP : " , kp)
print ("Alamat : " , alamat)
print ("Usia anda : " , usia, "tahun")
```

Ln: 1 Col: 0

Interact with Python

- IDLE stands for *Integrated Development and Learning Environment.*
- IDLE is an editing program written entirely in Python by Guido van Rossum.
- IDLE is installed when you install any recent version of Python

Example of IDLE – Output Modules

Python 3.7.4 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit
(Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\Admin\Google Drive\kelas\sem 1 20192020\SPPP2102\nota\Lecture
 Notes 2019\test2.py
************************
My first try using Python 3
************************
Siapa nama anda : NORAH MD NOOR
No Kad pengenalan anda : 123456789
Masukkan alamat  anda : TAMAN MUTIARA RINI, JOHOR BAHRU
Masukkan tahun lahir : 1976
Nama :  NORAH MD NOOR
KP :  123456789
Alamat :  TAMAN MUTIARA RINI, JOHOR BAHRU
Usia anda :  43 tahun
>>>
```

Ln: 16 Col: 4

Interact with Python

- IDLE is a decent IDE for learning as it's lightweight and simple to use. However, it's not for optimum for larger projects.

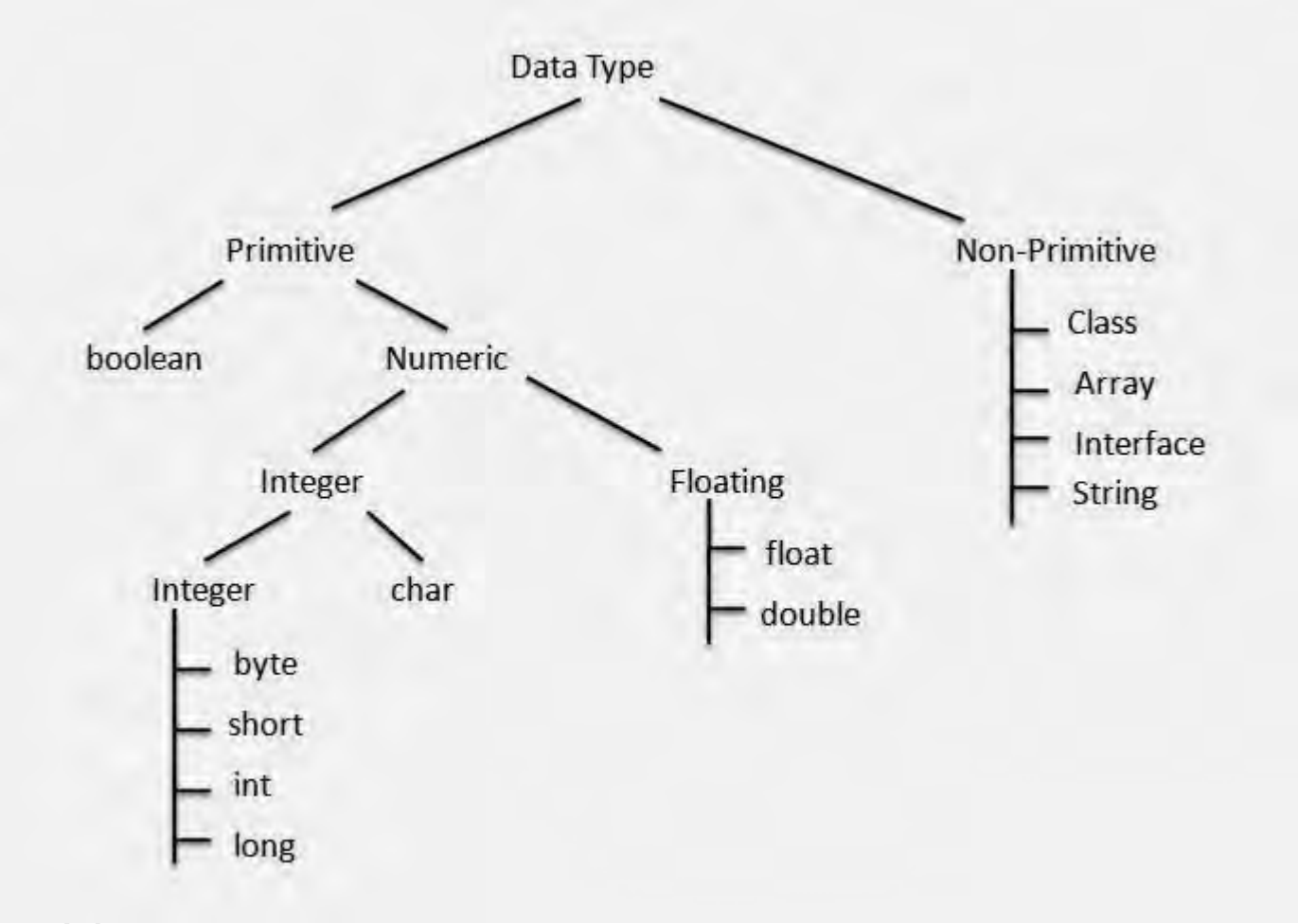
- It integrates editing activity, compiling, debugging and testing in a single environment

- There are many other IDE for python such as **Sublime Text**, **PyCharm**, **Visual Studio Code**, **Spyder** **and etc**

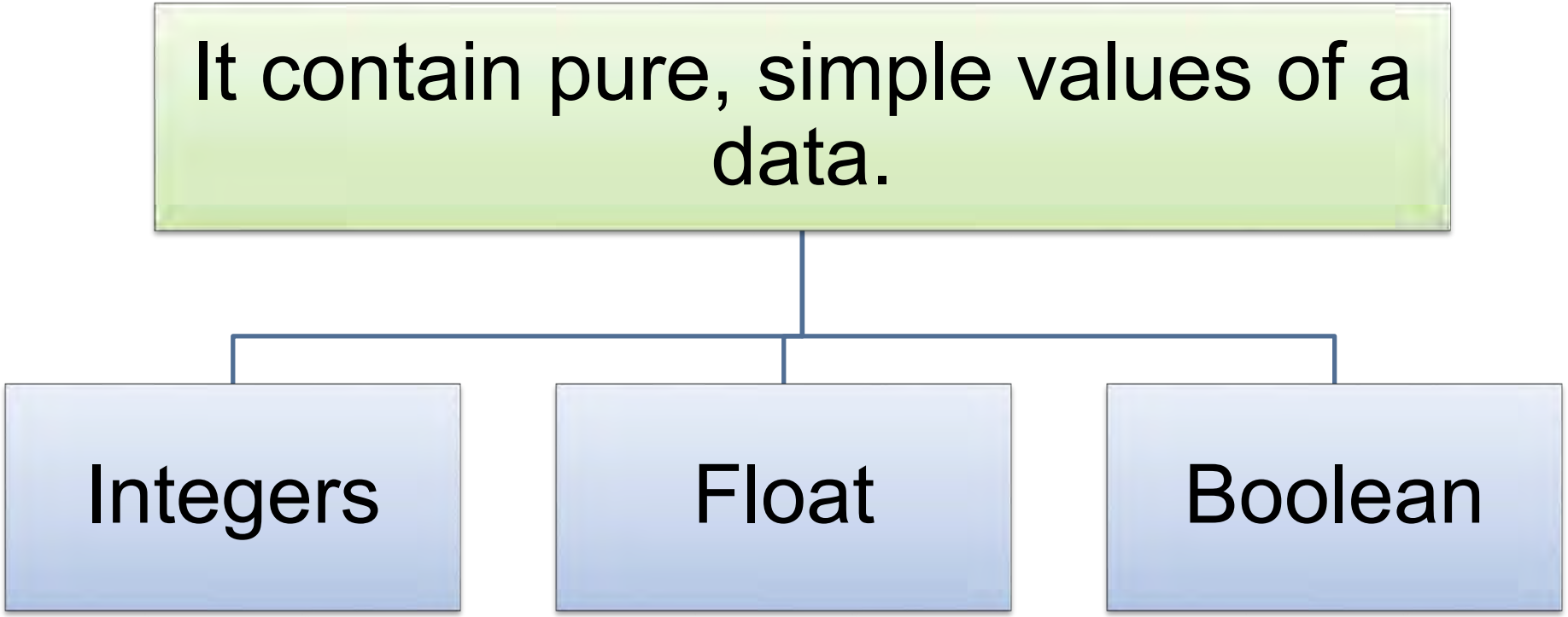
Data Structure

- Data structures can be divided into two categories in computer science:
    - **primitive and non-primitive data structures.**
- The former are the simplest forms of representing data, whereas the latter are more advanced: they contain the primitive data structures within more complex data structures for special purposes.

Primitive Data Structures



Primitive Data Structures



Primitive Data Structures

- Integers
  - You can use an integer represent numeric data, and more specifically, whole numbers from negative infinity to infinity, like 4, 5, or -1.
- Float
  - "Float" stands for 'floating point number'. You can use it for rational numbers, usually ending with a decimal figure, such as 1.11 or 3.14.

- Any number you enter in Python will be interpreted as a number.
- Python will consider any number written without decimals as an integer (as in 138) and any number written with decimals as a float (as in 138.0).

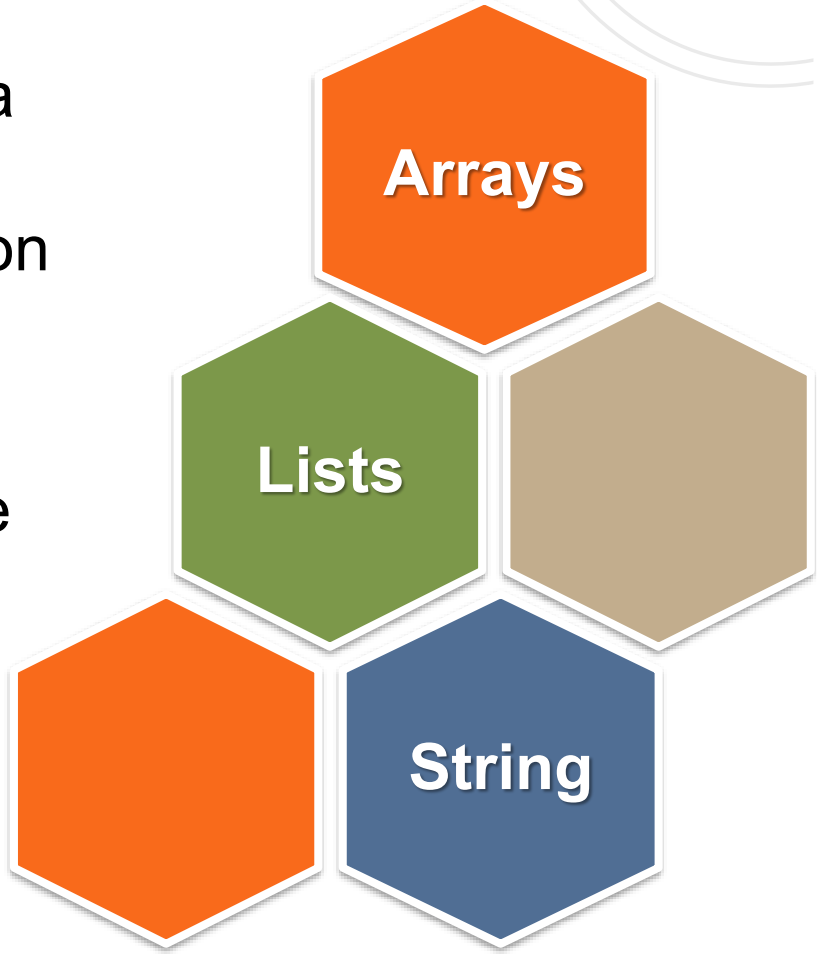
Primitive Data Structures

- Boolean
  - This built-in data type that can take up the values: True and False, which often makes them interchangeable with the integers 1 and 0. Booleans are useful in conditional and comparison expressions.

Non-Primitive Data Structures

- Non-primitive types are the sophisticated members of the data structure family. They don't just store a value, but rather a collection of values in various formats.

- In the traditional computer science world, the non-primitive data structures are divided into:



Non-Primitive Data Structures

- Strings
  - Strings are collections of alphabets, words or other characters. In Python, you can create strings by enclosing a sequence of characters within a pair of single or double quotes. For example: 'cake', "cookie", etc.

Non-Primitive Data Structures

- Array
  - Collection of basic data types of the same data type.
  - In Python, arrays are supported by the array module and need to be imported before you start initializing and using them.

  - Code example

```
import array as tatasusunan
a = tatasusunan.array("l",[3,6,9])
type(a)
```

Non-Primitive Data Structures

- Lists
  - Lists in Python are used to store collection of heterogeneous items. These are mutable, which means that you can change their content without changing their identity.
  - You can recognize lists by their square brackets [and] that hold elements, separated by a comma ,. Lists are built into Python: you do not need to invoke them separately.

Non-Primitive Data Structures

Listappend.py - /Users/suresh/Documents/Python Programs/Listappend.py (3.7.0)

```python
# Python Append List Items

intList = []

number = int(input("Please enter the Total Number of List Elements: "))
for i in range(1, number + 1):
    value = int(input("Please enter the Value of %d Element : " %i))
    intList.append(value)

item = int(input("Please enter the New Item to append : " ))
print("Original List Items are : ", intList)
intList.append(item)
print("New List Items are      : ", intList)
```

tutorialgateway.org

Python 3.7.0 Shell

```
======= RESTART: /Users/suresh/Documents/Python Programs/Listappend.py =======
Please enter the Total Number of List Elements: 4
Please enter the Value of 1 Element : 10
Please enter the Value of 2 Element : 20
Please enter the Value of 3 Element : 30
Please enter the Value of 4 Element : 60
Please enter the New Item to append : 120
Original List Items are :  [10, 20, 30, 60]
New List Items are      :  [10, 20, 30, 60, 120]
>>>
```

Variables

- Variable is a reserve memory space to hold a values. It is helpful to think of variables as a container that holds data which can be changed later throughout programming.

- You do not need to declare variables before using them in Python, or declare their type. Every variable in Python is an object.

- Python uses the term *name* instead of *variable.*

- The declaration happens automatically when you assign a value to a variable. The equal sign (=) is used to assign values to variables.

Variables

- The operand to the left of the = operator is the name of the variable and the operand to the right of the = operator is the value stored in the variable. For example −

```
counter = 100
miles = 1000.0
name = "John"
print counter
print miles
print name
```

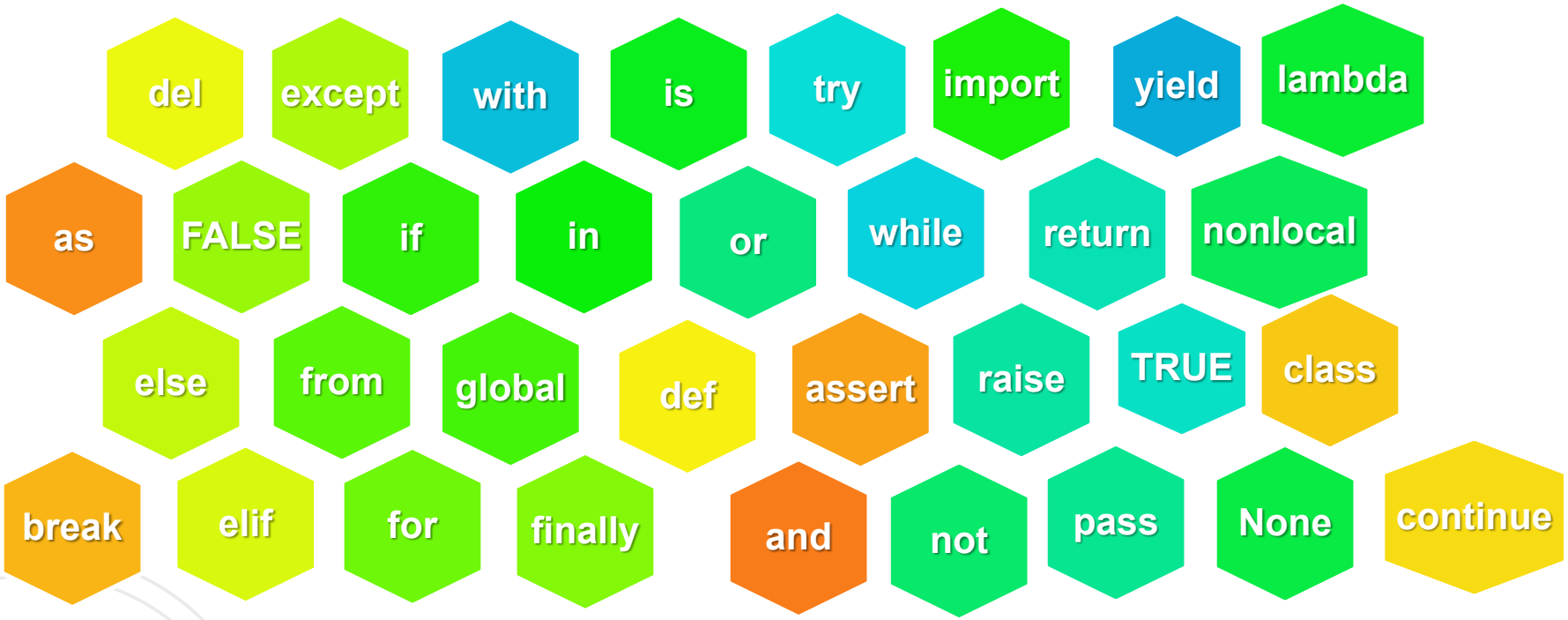
Variables

Variable naming rules

- Never start with number
- Used underscore (_) for spacing
- never use space between character
- Never use special symbol (eg : % $ | & ^ > < : } * / ^ %)
- Case sensitive. num is different from NUM and nUm.
- Never use reserved work (eg : FALSE, TRUE)
- **It's a Good Idea to use meaningful names.** Uniquely on the scope
- Avoid using the lowercase letter ‘l’, uppercase ‘O’, and uppercase ‘I’. Because the l and the I look a lot like each other and the number 1. And O looks a lot like 0.

Variables

- Python has a set of keywords that are reserved words that cannot be used as variable names, function names, or any other identifiers



Constants

- A constant is a type of variable whose value cannot be changed. It is helpful to think of constants as containers that hold information which cannot be changed later.

- Non technically, you can think of constant as a bag to store some books and those books cannot be replaced once placed inside the bag.

Comments

- Comments
  - Writable in any part of the program
  - It will not result in any action by the computer (compilers do not process comments)
  - Used to make the program easier to be read and understand. Also used to explain any part of the program as well as documentation.
  - Written after # as you can observe below:

```
counter = 100 # An integer assignment
kod = "SPPP 2102" # kod subjek
```

Statements

- Instructions that a Python interpreter can execute are called statements. For example, a = 1 is an assignment statement.
- Each line in Python is a single statement
- We can make a statement extend over multiple lines with the line continuation character (\). For example:

```
a = 1 + 2 + 3 + \
4 + 5 + 6 + \
7 + 8 + 9
```

- We could also put multiple statements in a single line using semicolons, as follows

```
a = 1; b = 2; c = 3
```

Input / Output

- **input()** and **print()** are widely used for standard input and output operations respectively.
- We use the **print()** function to output data to the standard output device (screen).
- Example :

```
print (“ Malaysia “)
```

- You can print out “Malaysia” 9 times with the * operator:
- **print(“Malaysia" * 9)**

Input / Output

- To take the input from user, we have the input() function to allow this.

- The syntax for input() is

```
input([prompt])
```

- where prompt is the string we wish to display on the screen.

```
>>> num = input('Enter a number: ')
>>> nama = input("Masukkan nama subjek : ")
```

Escape Characters

- In order to format output, you can use an escape character.
- Escape characters all start with the backslash key (\ ) combined with another character within a string to format the given string a certain way.

Escape Sequence	Description	Example	Result
\\	Backslash (\)	print("\\")	\
\'	Single quote (')	print("\")	'
\"	Double quote (")	print("\"")	"
\a	ASCII Bell (BEL)	print("\a")	**Bell sound
\n	ASCII Linefeed (LF)	print("Hello \n World!")	Hello World!
\t	ASCII Horizontal Tab (TAB)	print("Hello \t World!")	Hello World!

Convert Data Types

Converting Integers to Floats

- Python's method float() will convert integers to floats. To use this function, add an integer inside of the parentheses:

```
float(57)
```

- In this case, 57 will be converted to 57.0.

Converting Floats to Integers

- You can add a floating-point number inside of the parentheses to convert it to an integer:

```
int(390.8)
```

- In this case, 390.8 will be converted to 390.

Convert Data Types

Converting Numbers to Strings

- We can convert numbers to strings through using the str() method.

```
str(12)
```

- We can't print out string and integer values at the same time. We'll have to convert the variable lines to be a string value:

```
user = "Marlissa"
age = 17
print("Happy "+ str(age) + " Birthday, Dear") + user )
```

Convert Data Types

Converting Strings to Numbers

- Strings can be converted to numbers by using the int() and float() methods.

```
int('12')
float("1234.55")
```

You will received error f or this code. What can we do?

```
Current_year = '2019'
Birth_year = '1980'
Age = Current_year - Birth_year
print(Age)
```

Try me

```
name= input("What is your name?")
Greeting = "Hello " + " " + name
print(Greeting)
```

```
one = 1
two = 2
three = one + two
print(three)
```

```
firstname = input("First Name : ")
lastname = input("Surname  : ")

Fullname= firstname + " " + lastname
print(Fullname)
```

That’s all