Information Technology Institute

Python

Programming for everyone

Prepared by

Noha Shehab

Agenda

01 Who I am ?

02 Course objective

03 Python History

04 Python syntax

05 Control structure

06 Exersice time

Course objective

- Learning Python basic programming syntax in order to use it later in creating different translation layers that can be used in developing and writing different machine learning and AI plugins.

03

Python History

Python history

- The implementation of Python was started in December 1 Guido Van Rossum at CWI in Netherland.
- In February 1991, Guido Van Rossum published the code (labeled version 0.9.0)
- In 1994, Python 1.0 was released with new features like lambda, map, filter, and reduce.
- Python 2.0 added new features such as list comprehensions, garbage collection systems.
- On December 3, 2008, Python 3.0 (also called "Py3K") was released.
- It was designed to rectify the fundamental flaw of the language.

Why the name Python?

- Guido van Rossum was reading the script of a popular BBC comedy series "Monty Python's Flying Circus". It was late 1970s.

- Van Rossum wanted to select a name which unique, sort, and little-bit mysterious.

- Python is also versatile and widely used in every technical field, such as Machine Learning, Artificial Intelligence, Web Development, Mobile Application, Desktop Application, Scientific Calculation, etc.

Why Python?

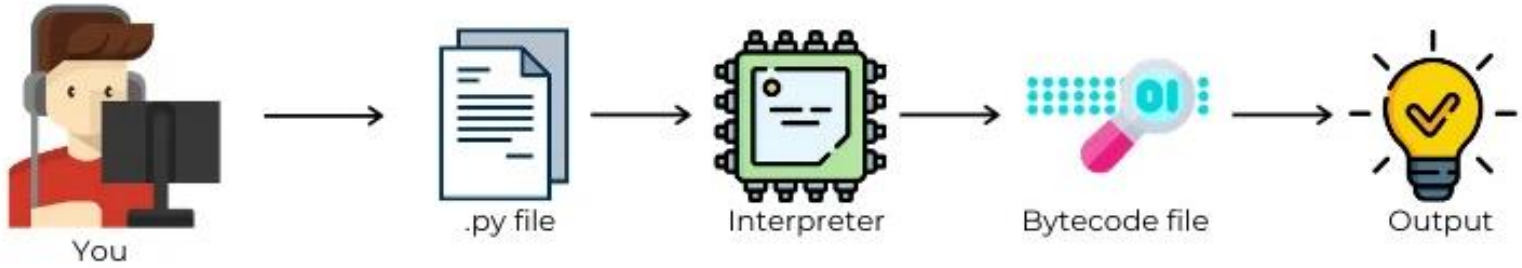
Rapid development

General purpose

Easy to learn

How Python works?

- Python is an interpreted language



04

Python syntax

Python syntax

- Download the python interpreter
  - https://www.python.org/downloads/windows/

- Your first Hello world.

```
1   print("Hello itians ^^")
```

```
Hello itians ^^
```

- Python files should have the **.py** extension.

Python syntax

- Variable is a name that is used to refer to memory location.
- Python variable or identifier and used to hold value.
- Python identifier can be used with variables, functions and classes.
    - The identifier should start with (A-Z, a-z, _)
    - The identifier shouldn’t contain any punctuations characters.
    - The name can have (A-Z, a-z, _ and digits)

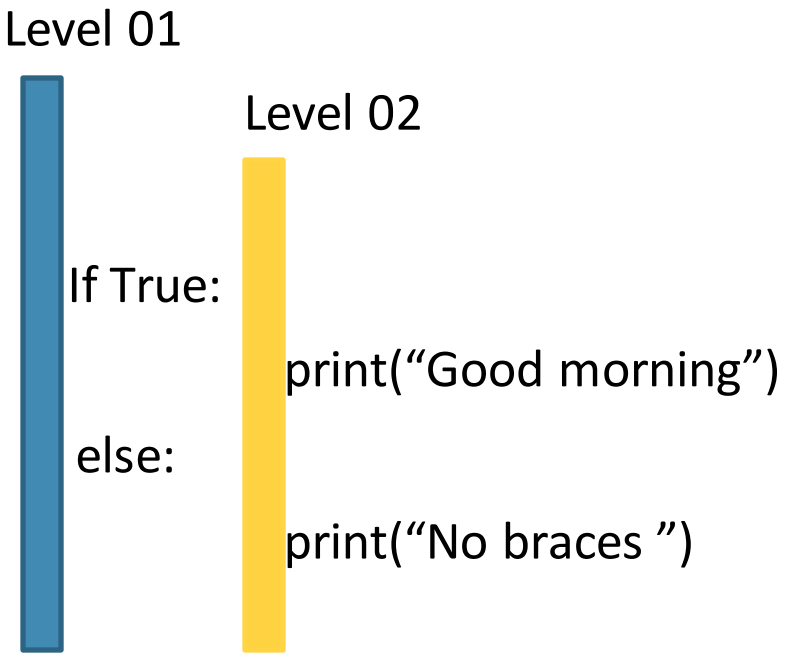
- Python identifiers are **case sensitive**.

Python identifiers

- Some reserved words cannot be used as identifiers.

```
and       exec      not
assert    finally   or
break     for       pass
class     from      print
continue  global    raise
def       if        return
del       import    try
elif      in        while
else      is        with
except    lambda    yield
```

Line indentations



No ;

Quotes and comments

‘ word ’

“ I am a sentences ”

“”””
multi-line
Paragraph
”””””

this a comment

Python Variables

- Python is loosely typed language.
  - No need to define the variable, the interpreter will do everything.

- To define a variable

Variable identifier = Value

```
name = "Noha"
```

```
Year = 2022
```

```
students = True
```

Python data types

- Python supports Primitive and Non-Primitive Data Types.
- Primitive
    - The primitive or basic data structures are the building blocks for data manipulation. They contain pure and simple values of data. In Python.

- Non-Primitive Data Types
    - Non-primitive not just store a value, but rather a collection of values in various formats.

Primitive data types

- Integers
- Strings
- Boolean
- float

Non-primitive data types

- Tuples
- Lists
- Sets
- Dictionaries

Variable types and conversions:

- To check the variable type, call the function type.

```
1   name = "Noha"
2   type(name)
```

```
str
```

```
1   year = 2022
2   type(year)
```

```
int
```

- Type conversions

```
1   year = str(year)
```

```
1   type(year)
```

```
str
```

Python operators

- Arithmetic operators
  - + addition Op
  - - Subtraction Op
  - * Multiplication Op
  - / Division Op
  - % Modulus Op
  - // Division without Fractions
  - ** Exponent Op
- Assignment operators
  - = assign
  - += add and assign
  - -= subtract and assign
  - *= multiply and assign
  - /= divide and assign
  - %= get modulus and assign
  - //= floor divide and assign
  - **= get exponent and assign

Python operators

- Comparison operators
  - == return True if a equals b
  - >= return True if a equals or greater than b
  - <= return True if a equals or less than b
  - != return True if a not equals b
  - <> return True if a not equals b
  - > return True if a greater than b
  - < return True if a lesser than b

Examples

- Assume, Ture = 1 and False = 0

```
1   6 == "6"
```

```
False
```

```
1   True == "True"
```

```
False
```

```
1   False == 0
```

```
True
```

```
1   True == 1
```

```
True
```

```
1   True == 100
```

```
False
```

Python operators

- Logical operators
  - **and** AND Logic Gate

```
True and False
```

```
False
```

  - **or** OR Logic Gate

```
True or False
```

```
True
```

  - **not** Not Logic Gate

```
Not False
```

```
True
```

```
Not (False == 1)
```

```
False
```

```
(False == 2)and (True == 1)
```

```
False
```

Examples

- Logical operations:
  - And
  - Notice the difference..

```
1   1 and 10
```

```
10
```

```
1   10 and 1
```

```
1
```

```
1   10 and 0
```

```
0
```

```
1   0 and 100
```

```
0
```

- Or:

```
1   2 or 10
```

```
2
```

```
1   0 or 100
```

```
100
```

```
1   100 or 0
```

```
100
```

- Not

```
1   not "sample text"
```

```
False
```

```
1   not 10
```

```
False
```

```
1   not ""
```

```
True
```

Falsy values

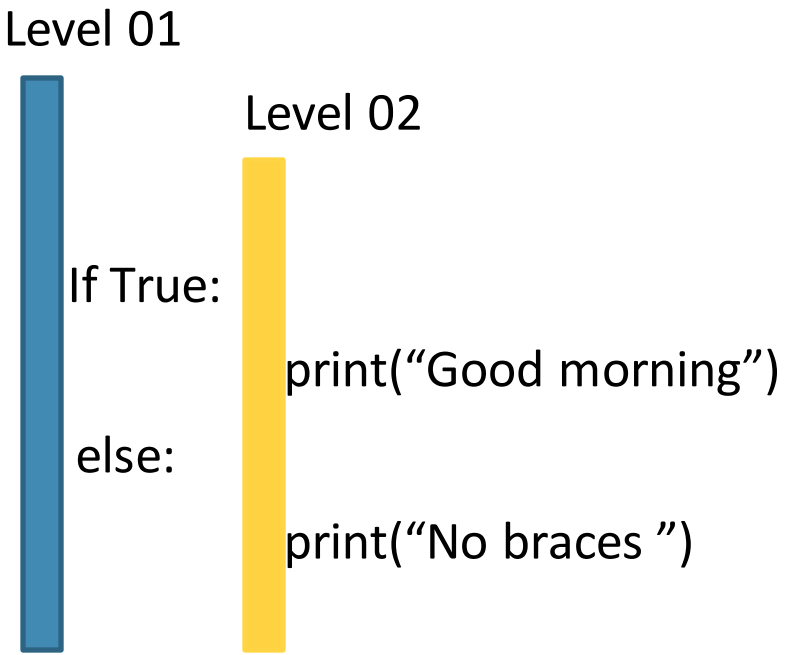
- Falsy values are values that evaluate to False in a boolean context.
- Falsy values include empty sequences (lists, tuples, strings, dictionaries, sets), zero in every numeric type, None , and False

- None
- {}
- “”
- ()
- 0
- False
- Empty collections

05

Control structure

If statement




```
day = "Sunday"

if day == 'Sunday':
    print("Wish you a happy week ^^ ")
elif day == "Thursday":
    print("Enjoy your weekend")
else:
    print("It seems impossible until it is done ;) ")
```

Wish you a happy week ^^

02

Strings

Strings

- A string is a sequence of characters.

```
name = 'Noha'
work = "Information Technology Institute"
```

- String is treated like an array
  - access string part according index starts from 0
- You can calculate its length

```
print(len(name)) # 4
```

- Access certain char at some position.

```
print(work[10]) # n

print(work[2:8]) # ?

print(work[-2]) # t

print(work[100]) # ?
```

- Count chars
  - work.count("j")

String functions

- Interpolation:

```
name = "Noha Abd El-Hady Abd El-Hady Shehab"

fname = "Noha "
mid = "Abd El-Hady "
lastname = "Shehab"

fullname = fname + mid * 2 + lastname
```

- Capitalize:

```
x = "people develop countries, we develop people"
print(x.capitalize())
```

```
People develop countries, we develop people
```

- Replace:

```
# define a pattern
greet = "Welcome to your first python course provided by @ "
# replace certain parts with others
print(greet.replace("@", "iti"))
```

```
Welcome to your first python course provied by iti
```

String functions

- isdigit:

```
x = "10"
# check the value inside the string is digit
print(x.isdigit())  # True
```

- isalpha:
- isnumeric:

```
name, email = "noha", "nshehab@iti.gov.eg"
# #is_alpha
# check all symbols value inside the string are characters
print(email.isalpha())
# is_numeric
print(name.isnumeric())
```

- Check capital or small

```
name = "itI"
#  ---> islower ---> conver .lower()
print(name.islower())
```

https://docs.python.org/3/library/string.html#

String functions

- Strip, lstrip , rstrip:

```
# string strip
# string.strip , lstrip , rstrip
greet = "     welcome to day02        "
print(len(greet))
greet = greet.rstrip()
print(len(greet))
```

- Format string

```
student = "Zeniab"
mytemp = f"Good morning {student}"
```

```
# format string
name = "Noha"
faculty = "Engineering"
greet = "My name is {0} I graduated from faculty of {1}"  # create common format
print(greet.format(name, faculty))
greet = "My name is {n} I graduated from faculty of {f}"  # create common format
print(greet.format(f=faculty, n=name))
```

```
My name is Noha I graduated from faculty of Engineering
```

03

Numbers

Numbers

- Integers
- Float
- Long: removed from python3
- Complex

```
a = 200  # int
y = 3.15  # float
z = 4 + 5j  # complex
number
c = complex(4, 5)
```

Functions

- Round
- Min
- max

```
a, b, c, d = 10, 66.66, 76.3, 100.54
round(b)
round(c)
min(a, b, c, d)
max(a, b, c, d)
```

https://docs.python.org/3/library/numbers.html

04

Data structures

-Lists-

Data structures in python

- The most basic data structure in python is called sequence.
- A sequence, collection of elements, each element is assigned to a number, starts form 0 , represent its position or index.
- Python has different datatypes like lists, sets, tuples and dictionaries.
- Most common sequences are lists and tuples.

Lists

- Lists: the versatile datatype available in python.
- A collection of various data types.

```
l = []
```

- To define a list:

```
l = list([5, 6, 7])
l2 = ["iti", "3DFX", "python", "databases"]
```

- Lists can hold different values, with different datatypes.

```
l = list([5, 6, 7])

l3 = ["a", 5, "test", True, l]

print(l3)
```

- Lists can hold different lists also.

List operations

- Get items at certain index:

```
z = ["abc", 55, 67]
print(z[2])
```

- Lists are mutable data types, means that the values can be updated in the run time.

```
z = ["abc", 55, 67]
print(z[2])
z[2] = "updated item"
z[3] = "new item added"
```

- You can only update items at existing indices.

- You can sort the list items.

```
organic = ['kiwi', 'orange', 'apple', 'tomato', 'Carrot']
organic.sort()  # sort ascending
organic.reverse()  # sort descending
```

List functions

- Pop:

```
l = list([5, 6, 7])
l3 = ["a", 5, "test", True, l]
# pop the last item
print(l3.pop())  # return with removed it
print(l3)
# pop item at certain index
l3.pop(2)
print(l3)
```

- Append

```
# append
l3.append("iti")
print(l3)
```

- Insert

```
# insert into certain
index
l3.insert(4,"python")
print(l3)
```

List functions

- Remove

```
names = ["Mohamed", “Ahmed", “Ali"]
# # remove item
names.remove(“Ali")
```

- Extend:

```
# extend
l4 = [“Mostafa”, “Omar", "Mohamed"]
l3.extend(l4)
print(l3)
```

- Len

```
print(len(l3))
```

- Concatenation

```
m = ["abc", True, 40 ,66]
n = ["python", "test", "iti", 88]
z = m + n
print(z)
```

List functions

- Membership

```
# membership
l4 = [“Python", “Maya", “c#"]
print(“Mohamed" in l4)
```

- Iterations === Looping using for

```
# for loop
for item in l4:
    print(item)
```

- Min, max

```
x = [5,66,77]
print(min(x))
```

```
x = [5,66,77]
print(max(x))
```

- Empty lists are falsy values

```
# empty lists
l = []
if l:
    print(“non empty list")
else:
    print(“Empty list ")
```

05

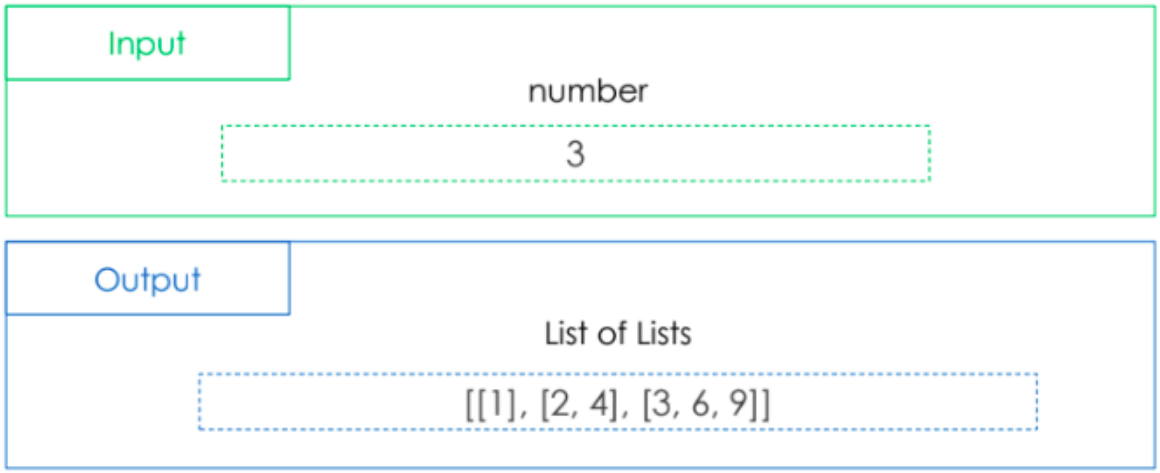
Time for practice

Lab01

- Install python interpreter on your machine.
- Write a program that counts up the number of vowels [a, e, i, o, u] contained in the string.
- Fill an array of 5 elements from the user, Sort it in descending and ascending orders then display the output.
- Write a program that prints the number of times the string 'iti' occurs in anystring.
- Write a program that remove all vowels from the input word and generate a brief version of it.

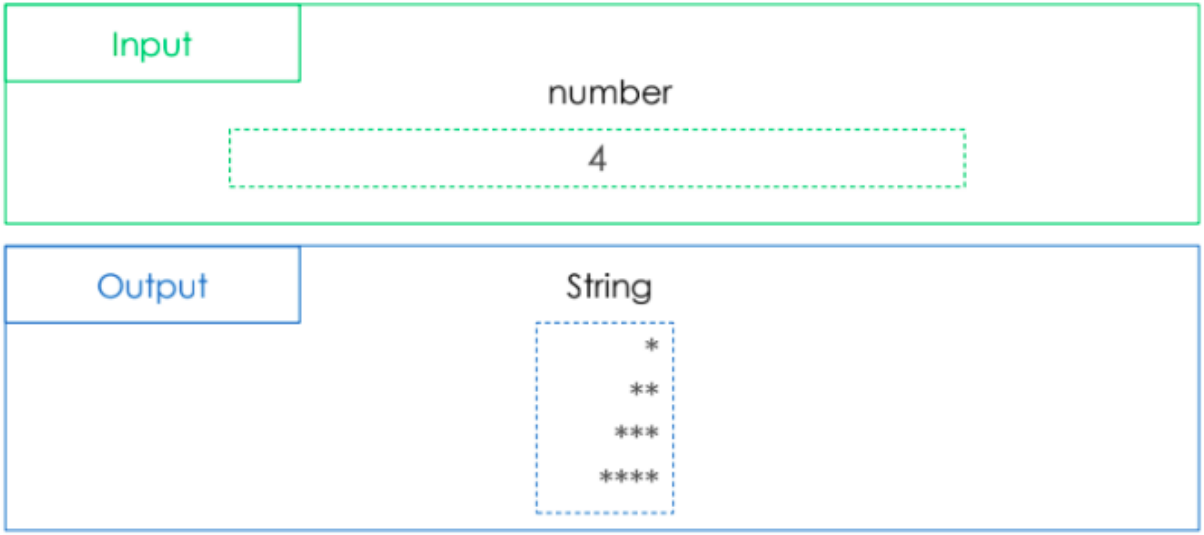
Lab01

- Write a program that prints the locations of "i" character in any string you added.
- Write a program that generate a multiplication table from 1 to the number passed.



Lab01

- Write a program that build a Mario pyramid like below:



“It seems impossible until it is done”

Thank you

nshehab@iti.gov.eg