VNUHCM - University of Science

fit@hcmus

Fundamentals of Programming for Artificial Intelligence

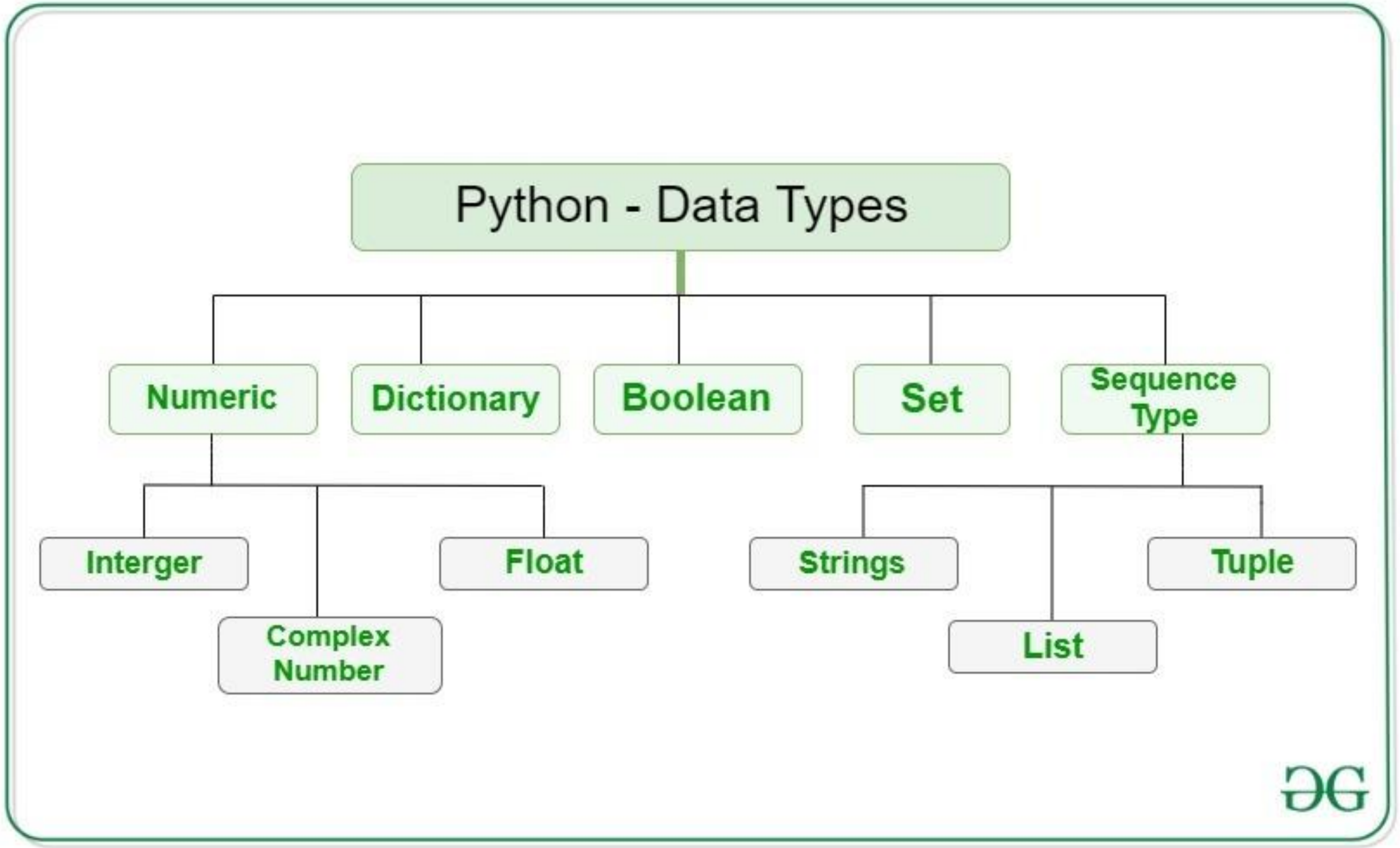
Session 01
Basic Statements

Instructors:
Dr. Lê Thanh Tùng
Dr. Nguyễn Tiến Huy

Content

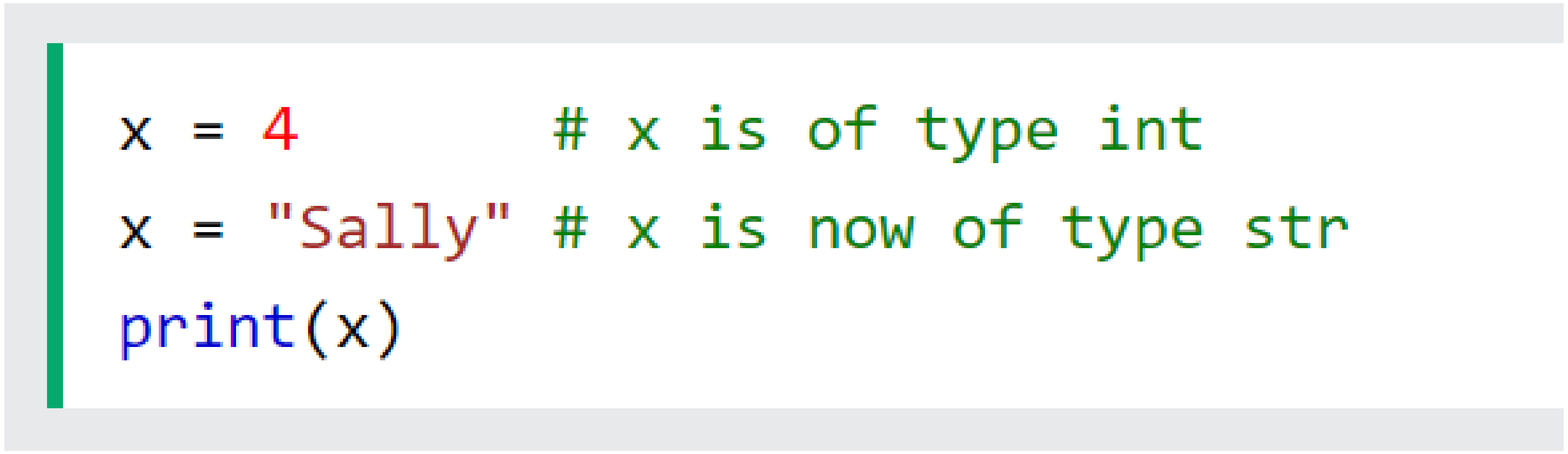
1. Data Types and Expression

Data Types


Python - Data Types
Numeric
Dictionary
Boolean
Set
Sequence Type
Interger
Complex Number
Float
Strings
List
Tuple

Variables

- Variables are containers for storing data values
- Creating variable:
  - Python has **no command for declaring** a variable.
  - A variable is created the moment you first assign a value to it.

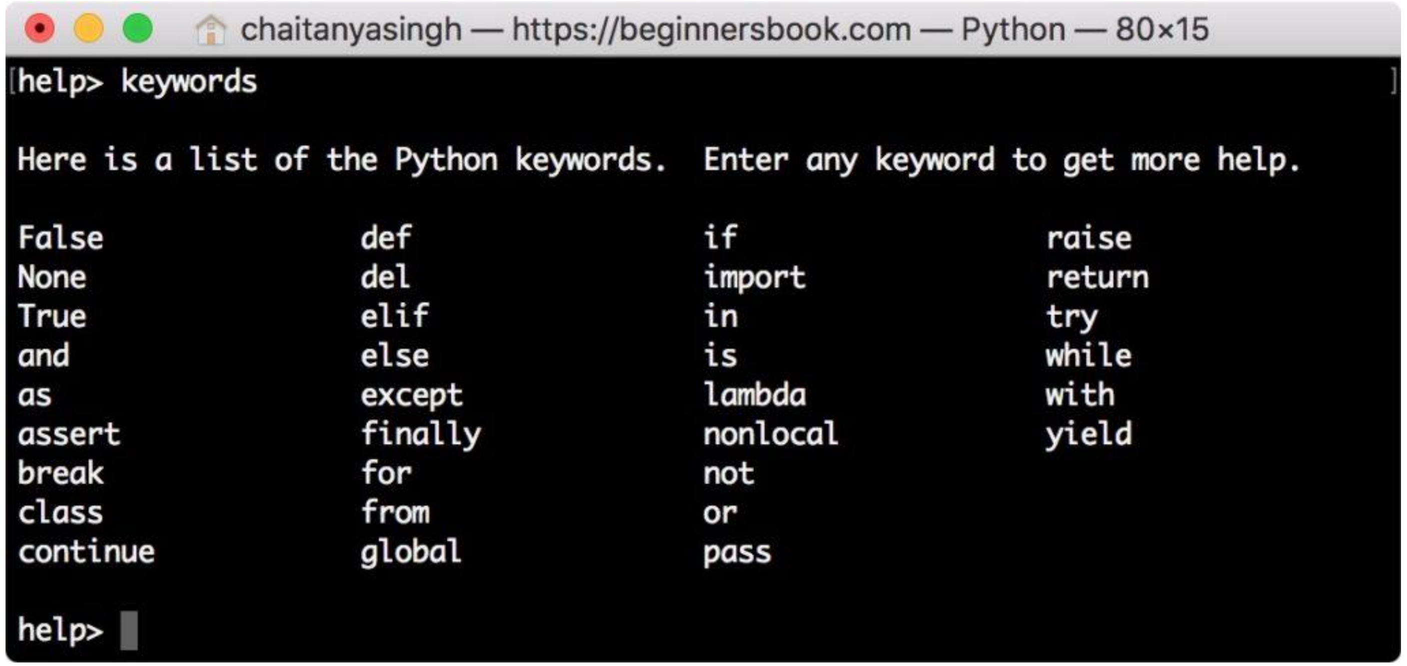


```
x = 4       # x is of type int
x = "Sally" # x is now of type str
print(x)
```

Variables

fit@hcmus

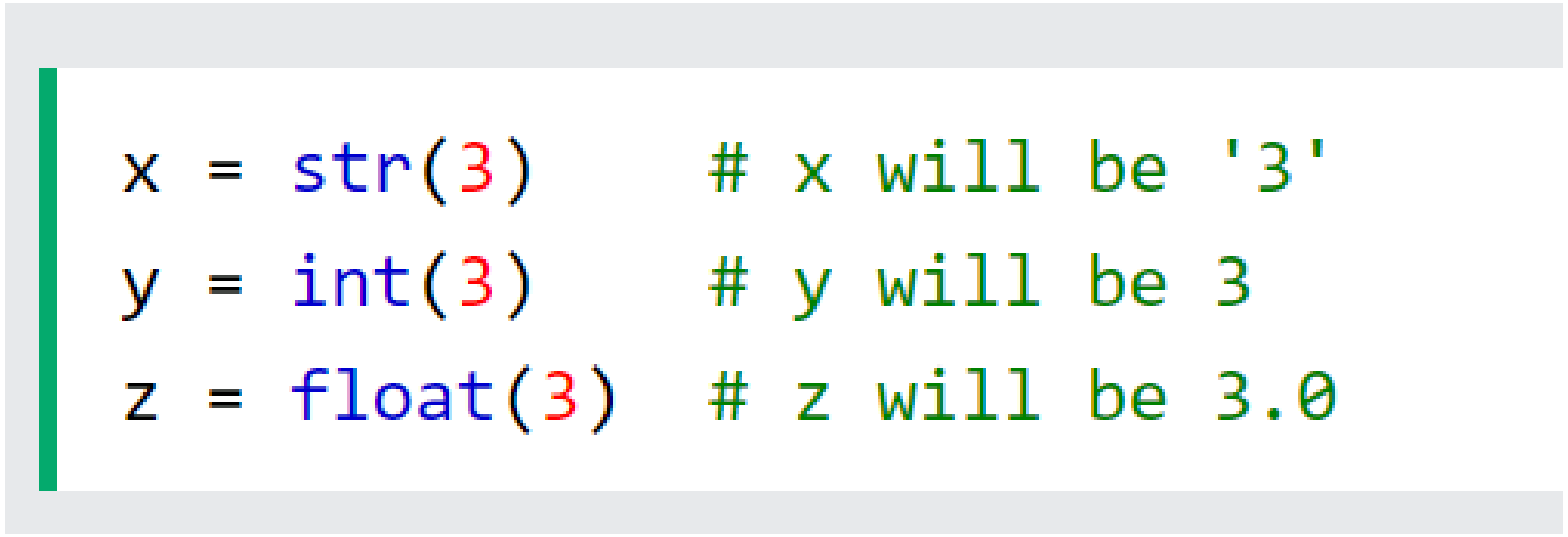
- Rules for Naming Python Variables:
  - Python is **case-sensitive** (i.e: num ≠ Num)
  - Contains Alphabet (upper/lowercase), Digits (0-9), underscore (_)
  - First letter: Alphabet, underscore
  - Avoid Keywords



```
help> keywords

Here is a list of the Python keywords.  Enter any keyword to get more help.

False               def                 if                  raise
None                del                 import              return
True                elif                in                  try
and                 else                is                  while
as                  except              lambda              with
assert              finally             nonlocal            yield
break               for                 not
class               from                or
continue            global              pass

help> 
```

Casting

- If you want to specify the data type of a variable, this can be done with casting

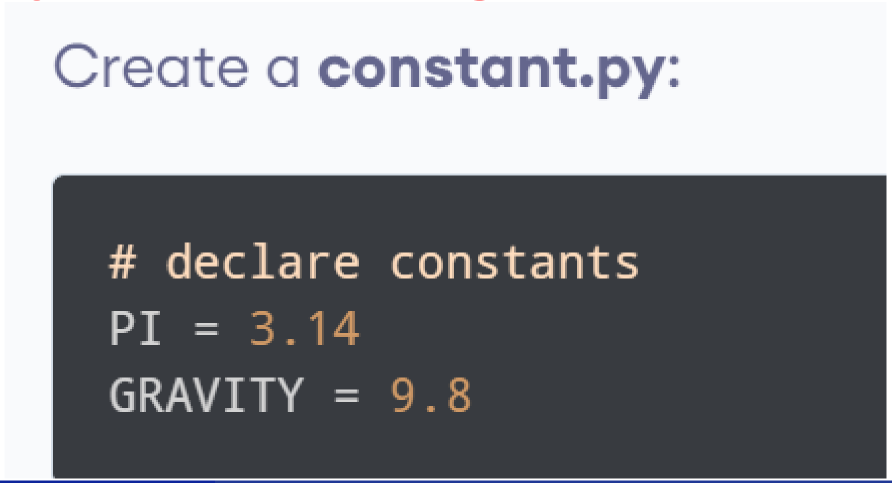


```
x = str(3)    # x will be '3'
y = int(3)    # y will be 3
z = float(3)  # z will be 3.0
```

Constant

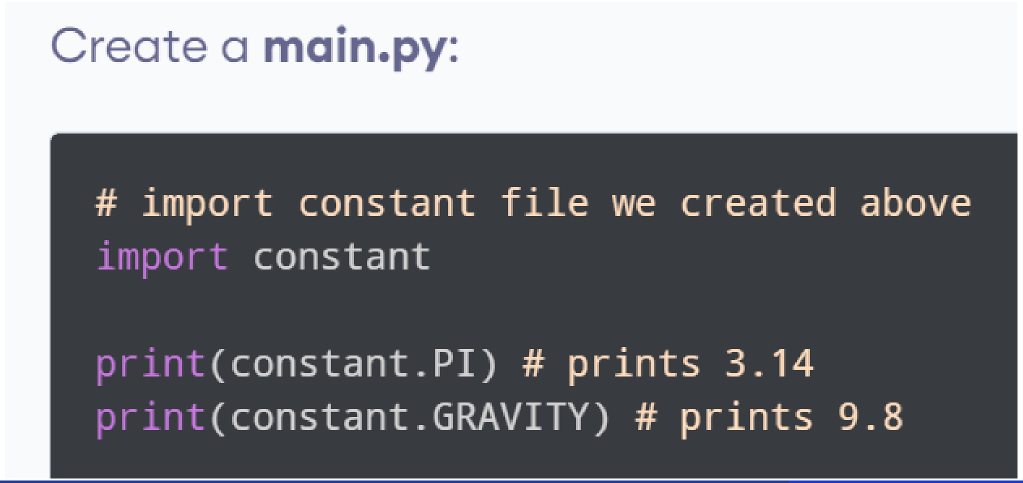
fit@hcmus

- A constant is a special type of variable whose value cannot be changed
- In Python, constants are usually declared and assigned in a module (a new file containing variables, functions, etc which is imported to the main file)
- **Note**: In reality, we don't use constants in Python. Naming them in all capital letters is a convention to separate them from variables, however, it **does not actually prevent reassignment**.



Create a **constant.py:**

```
# declare constants
PI = 3.14
GRAVITY = 9.8
```

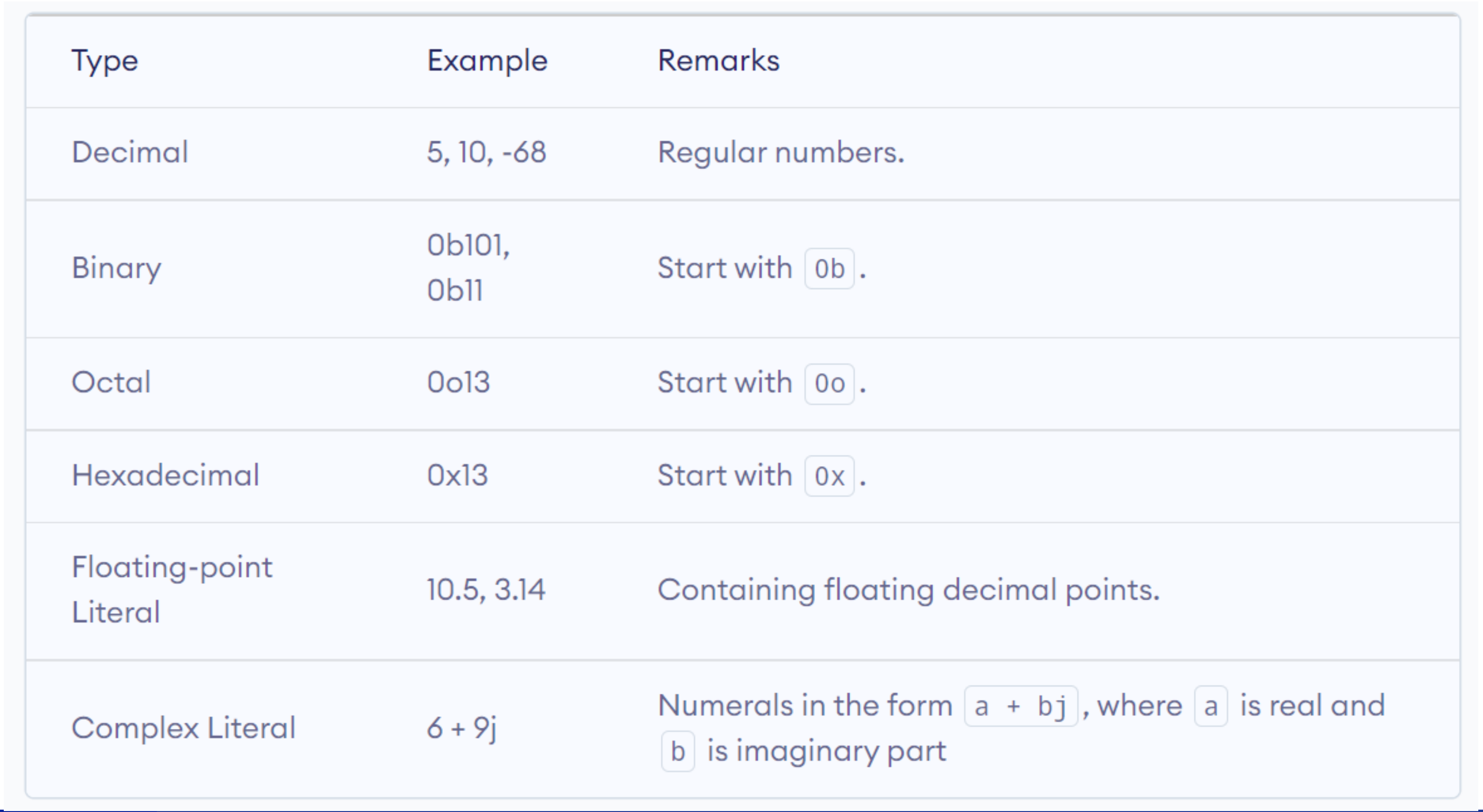


Create a **main.py:**

```
# import constant file we created above
import constant

print(constant.PI) # prints 3.14
print(constant.GRAVITY) # prints 9.8
```

Literals

- Literals are representations of fixed values in a program
- Numeric Literals are immutable (unchangeable). Numeric literals can belong to 3 different numerical types: Integer, Float, and Complex.



Type	Example	Remarks
Decimal	5, 10, -68	Regular numbers.
Binary	0b101, 0b11	Start with `0b`.
Octal	0o13	Start with `0o`.
Hexadecimal	0x13	Start with `0x`.
Floating-point Literal	10.5, 3.14	Containing floating decimal points.
Complex Literal	6 + 9j	Numerals in the form `a + bj`, where `a` is real and `b` is imaginary part

Assign Operator

- Assign operator "**=**": put the value from right-hand side to left-hand side

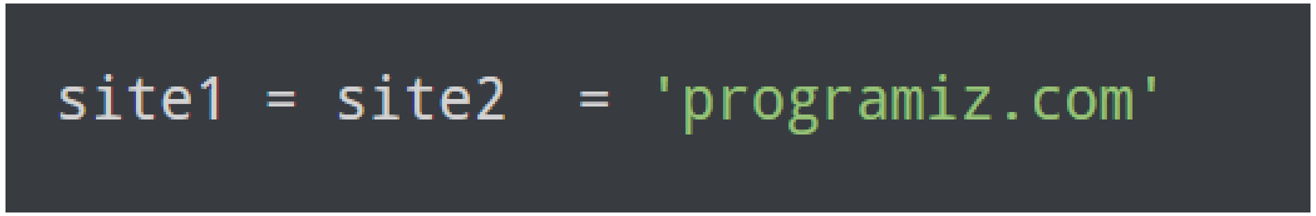
```python
# assign value to site_name variable
site_name = 'programiz.pro'

print(site_name)

# Output: programiz.pro
```

- Assigning multiple values to multiple variables

```python
a, b, c = 5, 3.2, 'Hello'

print(a)  # prints 5
print(b)  # prints 3.2
print(c)  # prints Hello
```

Does it work?

- What is the value of site1, site2 ?

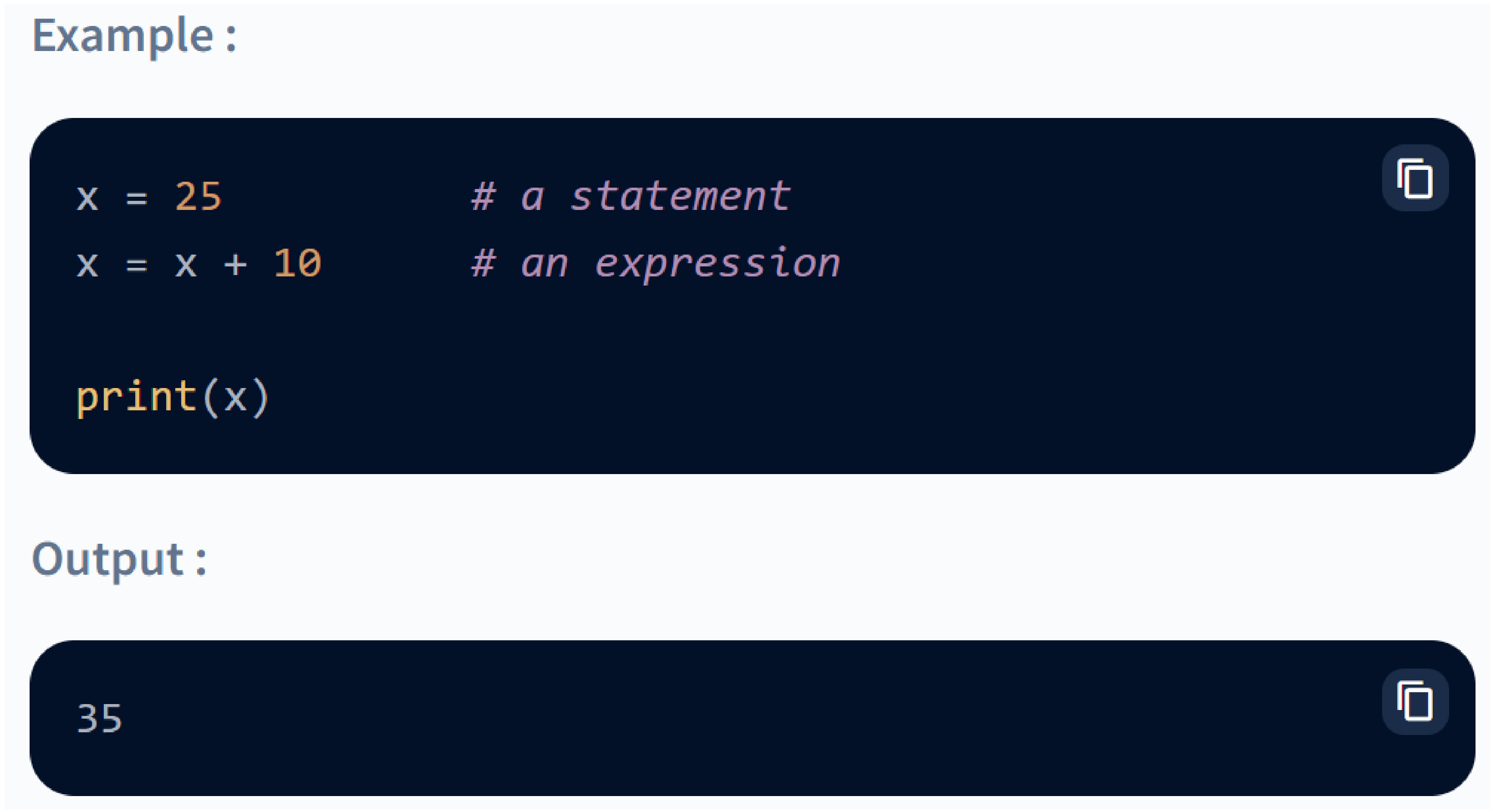


```
site1 = site2  = 'programiz.com'
```

- Separate it into the simple process?

Expression

- **Expression: is the combination of operands and operatos**



Example :

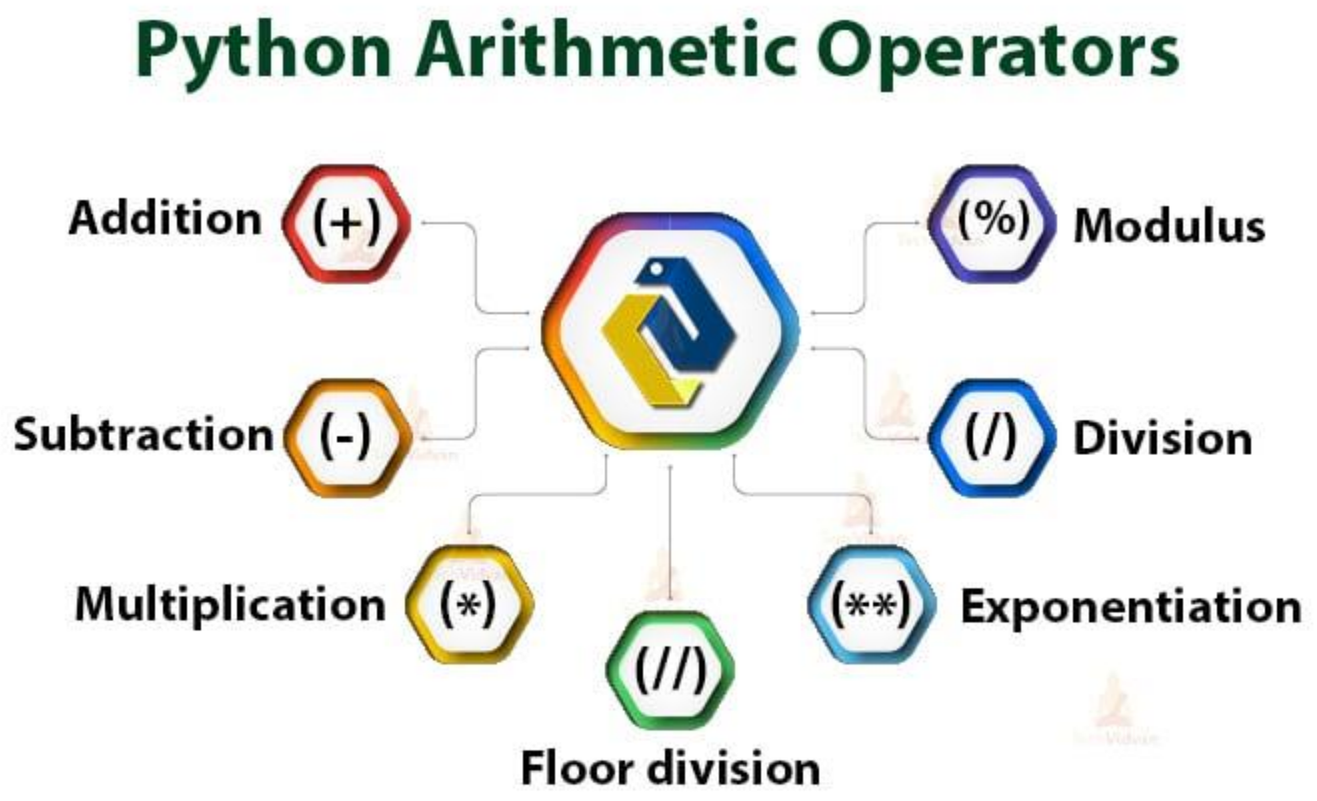
```
x = 25          # a statement
x = x + 10      # an expression

print(x)
```

Output :

```
35
```

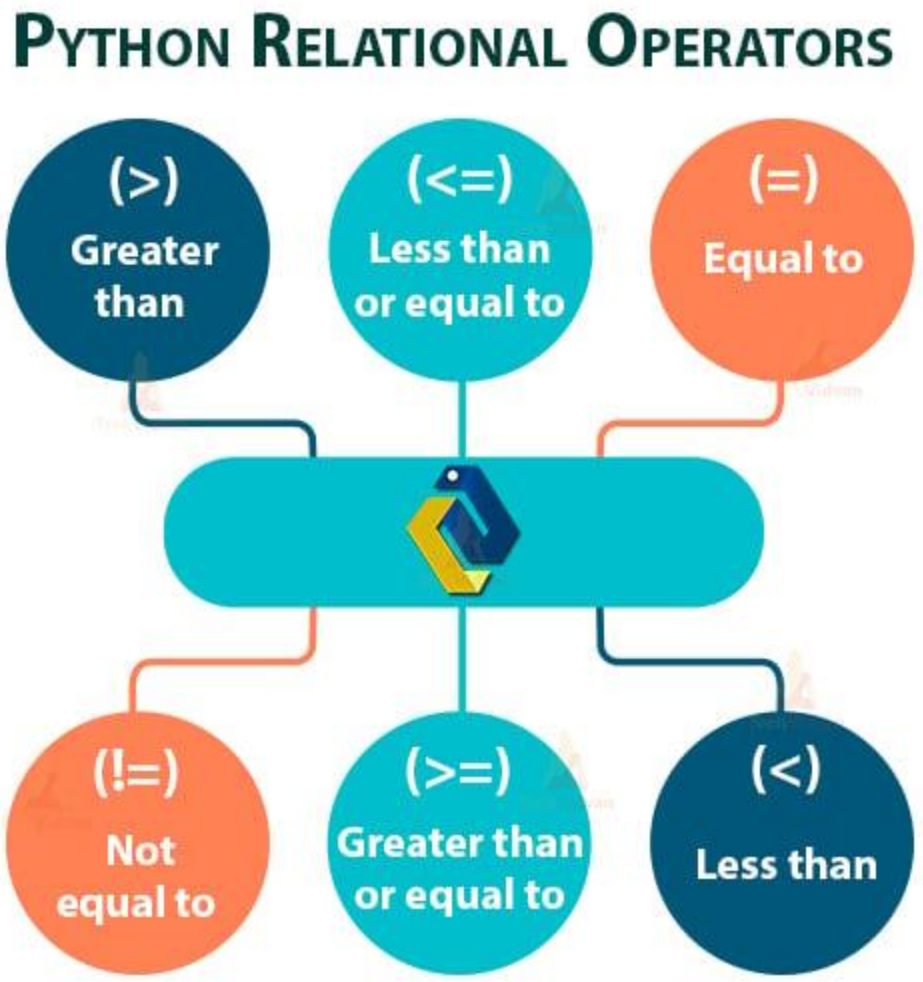
Operators

- **Arithmetic Operators**



Operators

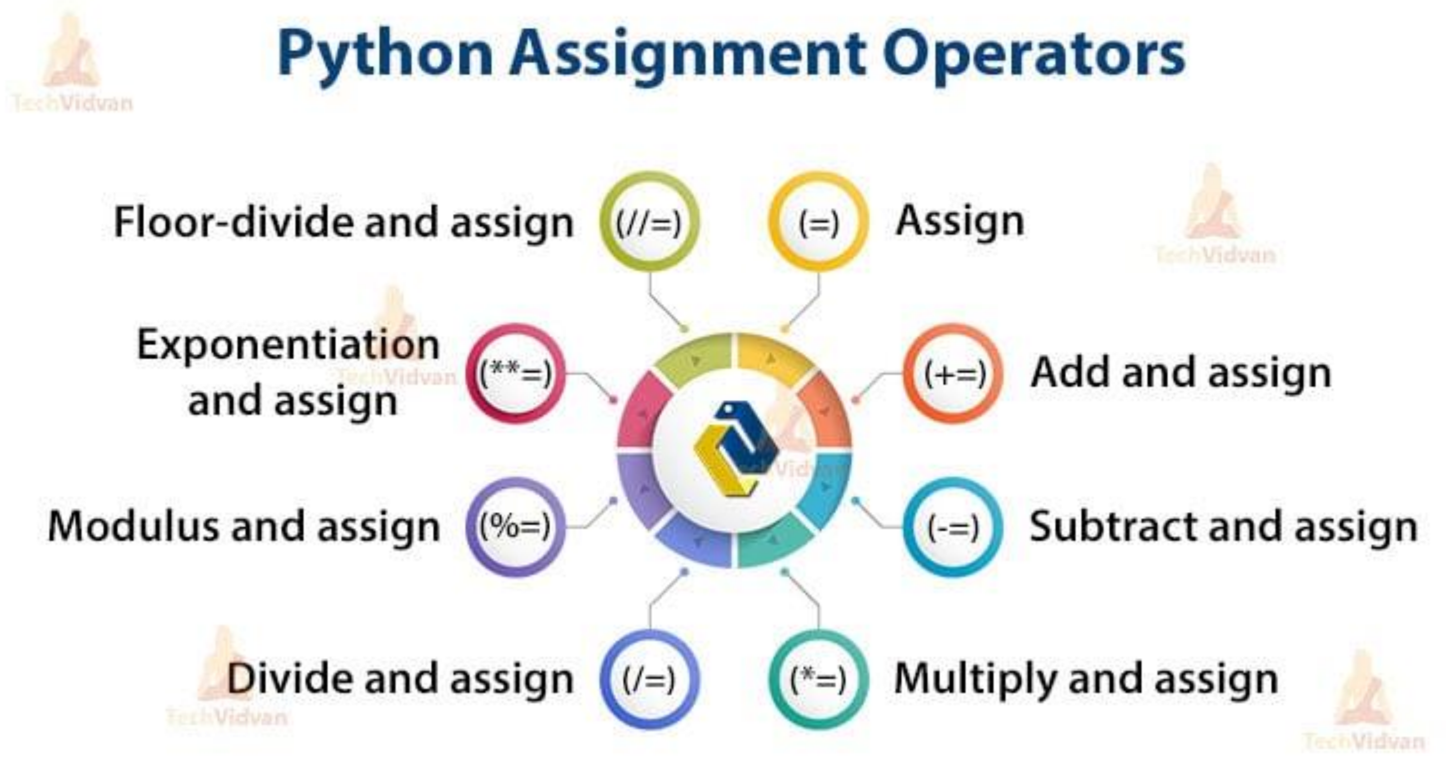
- **Relational Operators**



Operators

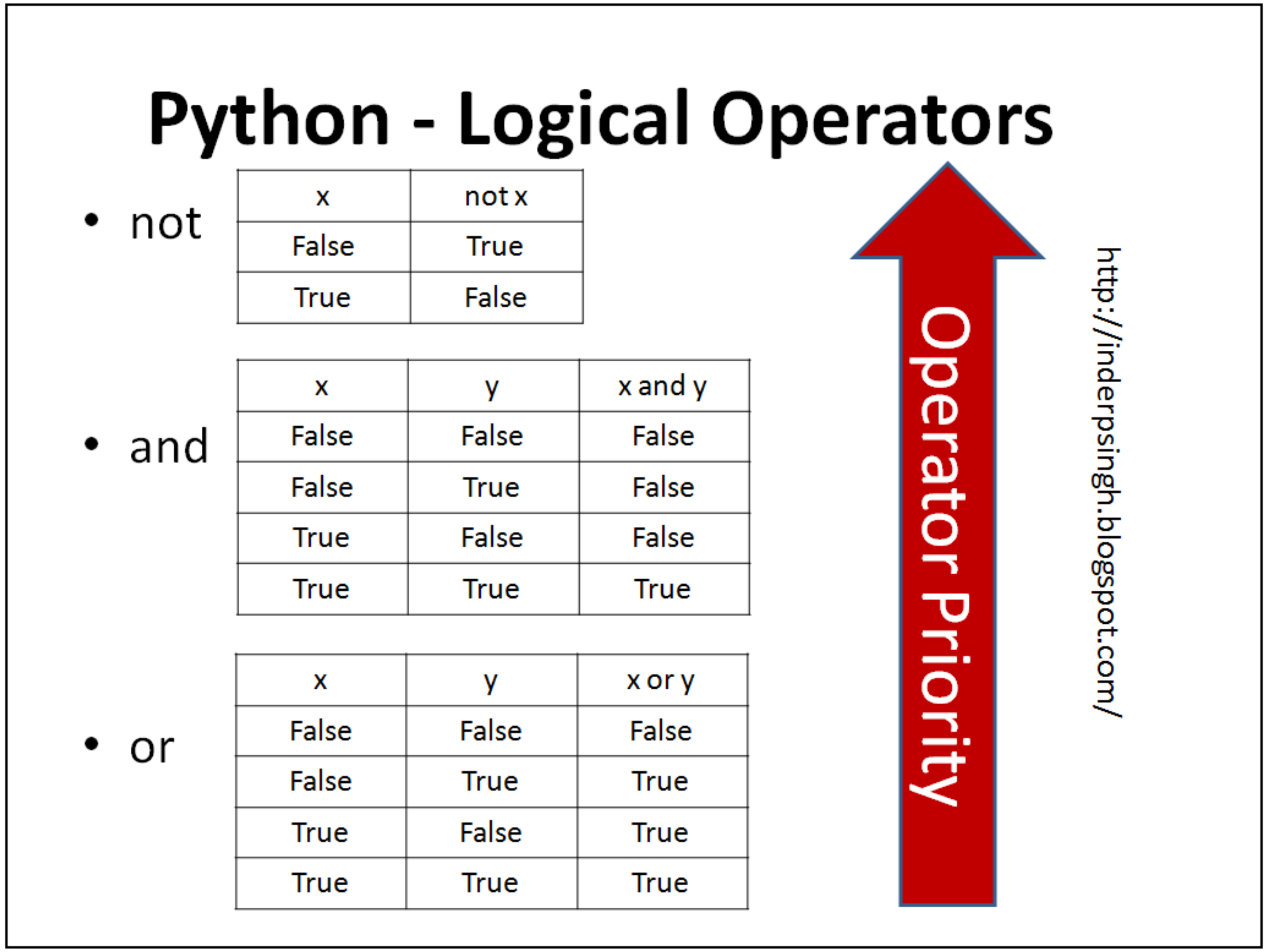
fit@hcmus

- **Assignment Operators**



Operators

- **Logical Operators**



Python - Logical Operators

- not

x	not x
False	True
True	False

- and

x	y	x and y
False	False	False
False	True	False
True	False	False
True	True	True

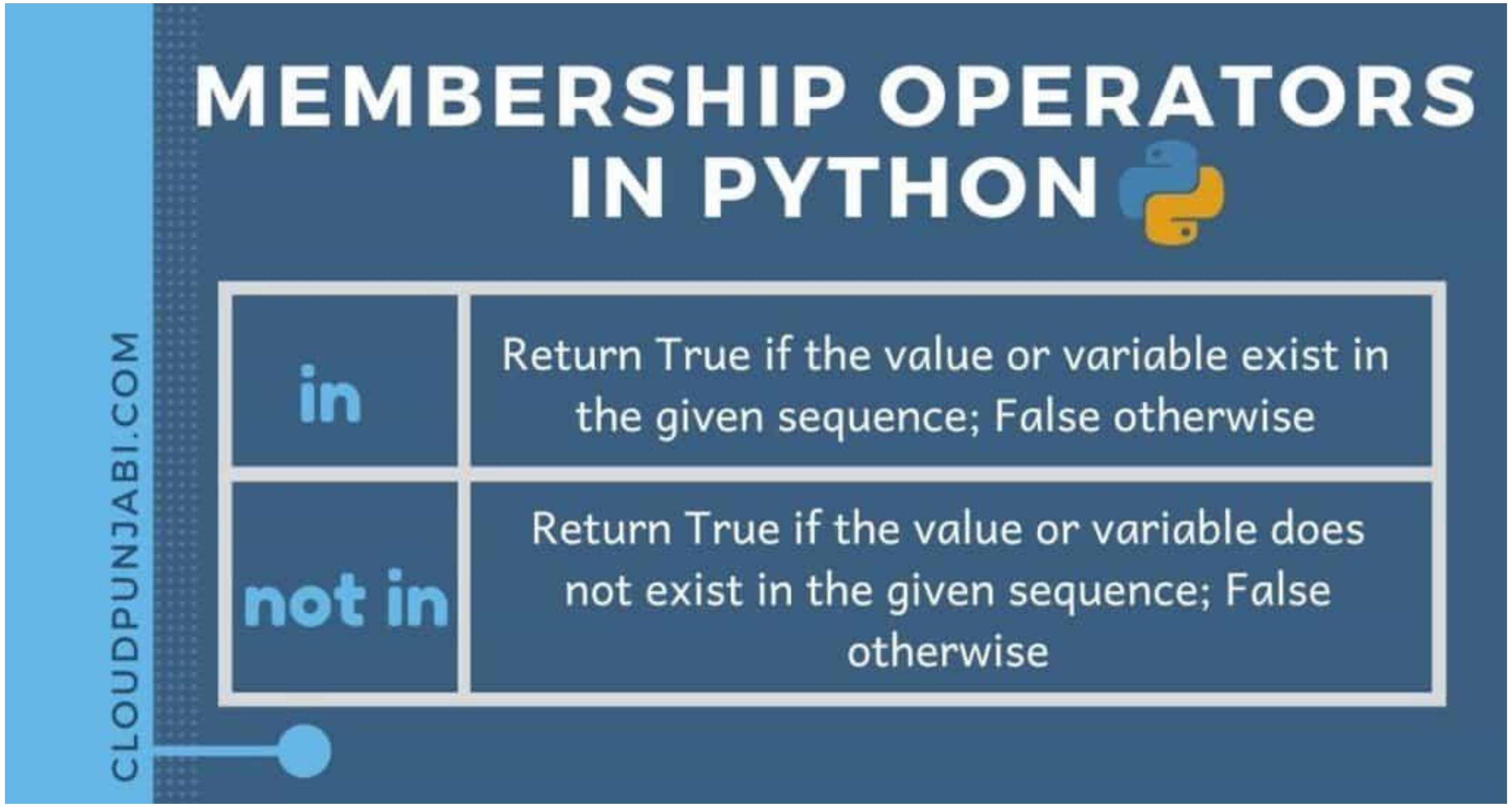
- or

x	y	x or y
False	False	False
False	True	True
True	False	True
True	True	True

Operator Priority
http://inderpsingh.blogspot.com/

Operators

- **Membership Operators**

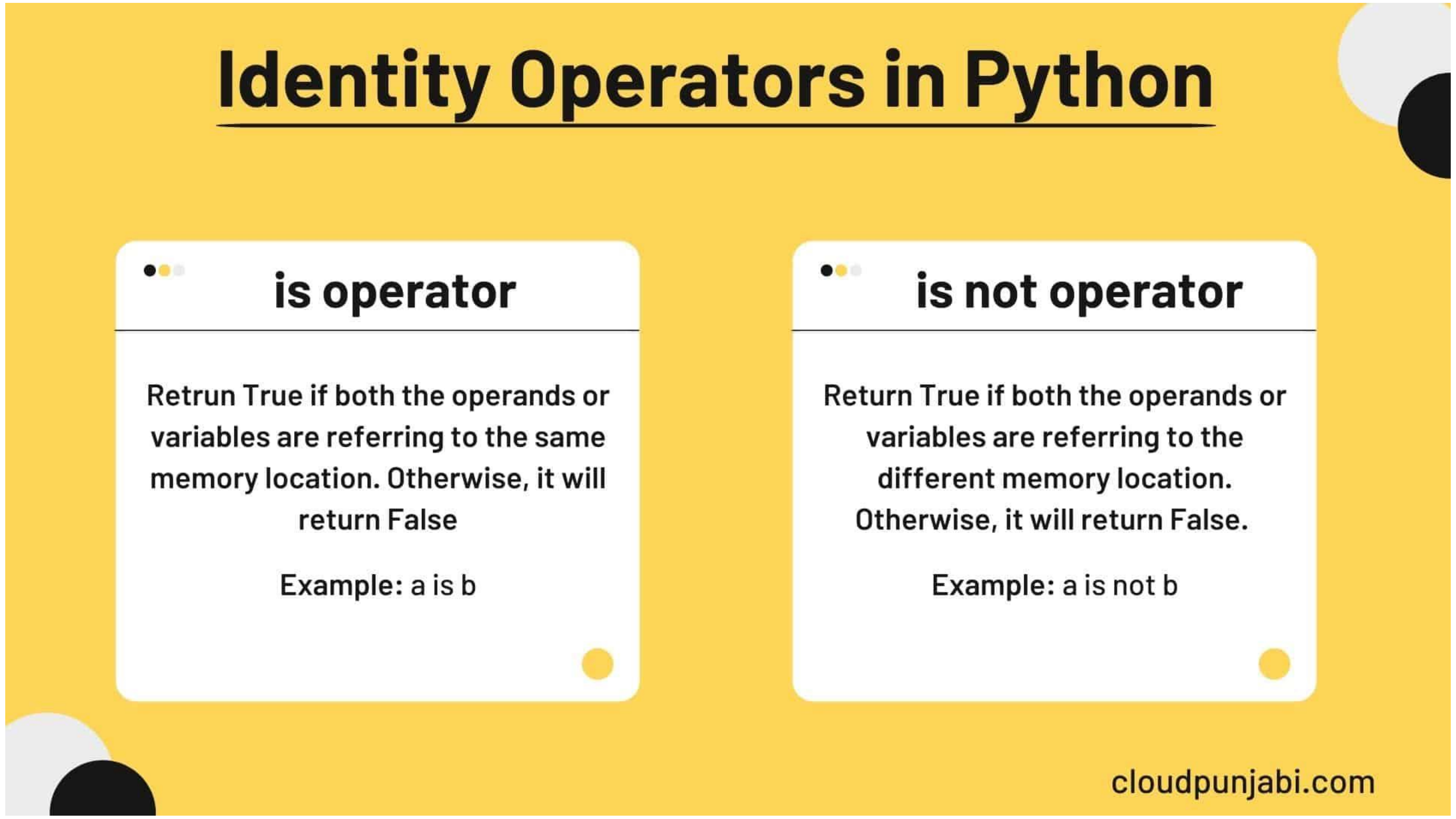

MEMBERSHIP OPERATORS IN PYTHON

in	Return True if the value or variable exist in the given sequence; False otherwise
not in	Return True if the value or variable does not exist in the given sequence; False otherwise

CLOUDPUNJABI.COM

Operators

- **Identity Operators**



Identity Operators in Python

is operator

Retrun True if both the operands or variables are referring to the same memory location. Otherwise, it will return False

Example: a is b

is not operator

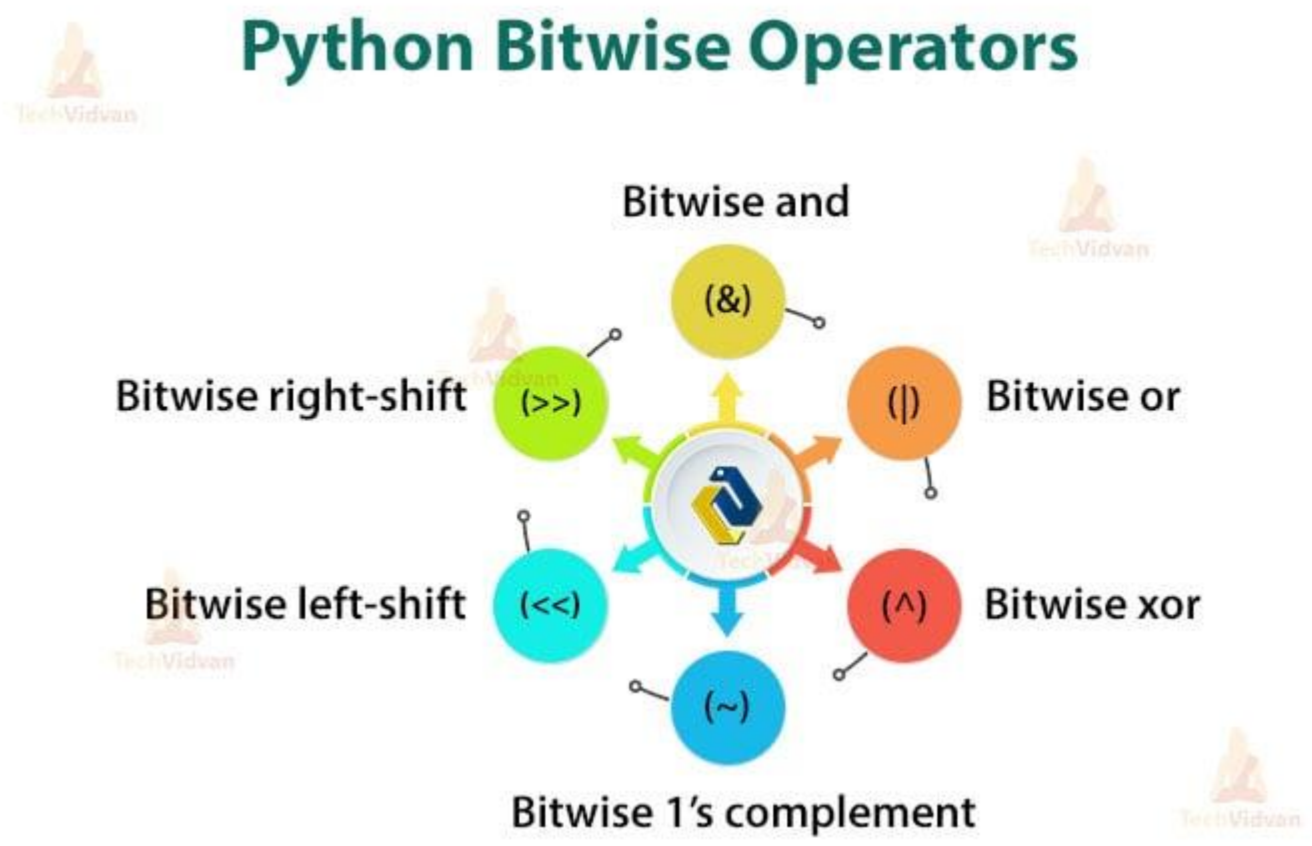
Return True if both the operands or variables are referring to the different memory location. Otherwise, it will return False.

Example: a is not b

cloudpunjabi.com

Operators

- **Bitwise Operators**



Expression

- Arithmetic Operators

Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y

Expression

- Modulus Operators
  - $a \vdots b$ $\rightarrow$ `a % b == 0`
  - $a \not\vdots b$ $\rightarrow$ `a % b != 0`
  - a is even $\rightarrow$
  - a is odd $\rightarrow$

Type cast

- Type Casting is the method to convert the Python variable datatype into a certain data type in order to perform the required operation by users
- There can be two types of Type Casting in Python:
  - Python Implicit Type Conversion
  - Python Explicit Type Conversion

Type cast

- Implicit Type Conversion
  - Via the operators

```
a = 5         # int
b = 7.0       # float
c = a + b     # float => Implicit Type
```

Type cast

- Explicit Type Conversion
  - Via the function: `int(), float(), str()`

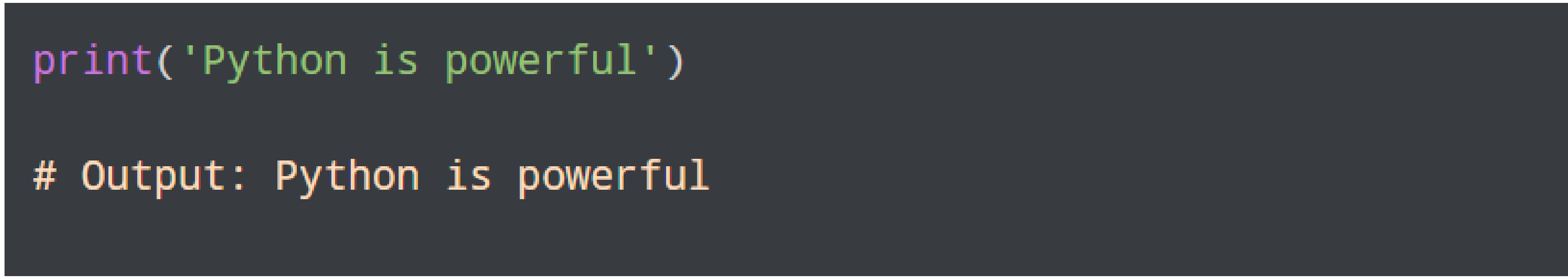
```
# int variable
a = 5.9
# typecast to int
n = int(a)
print(n) # 5
```

2. Input/Output Statements

Input/Output

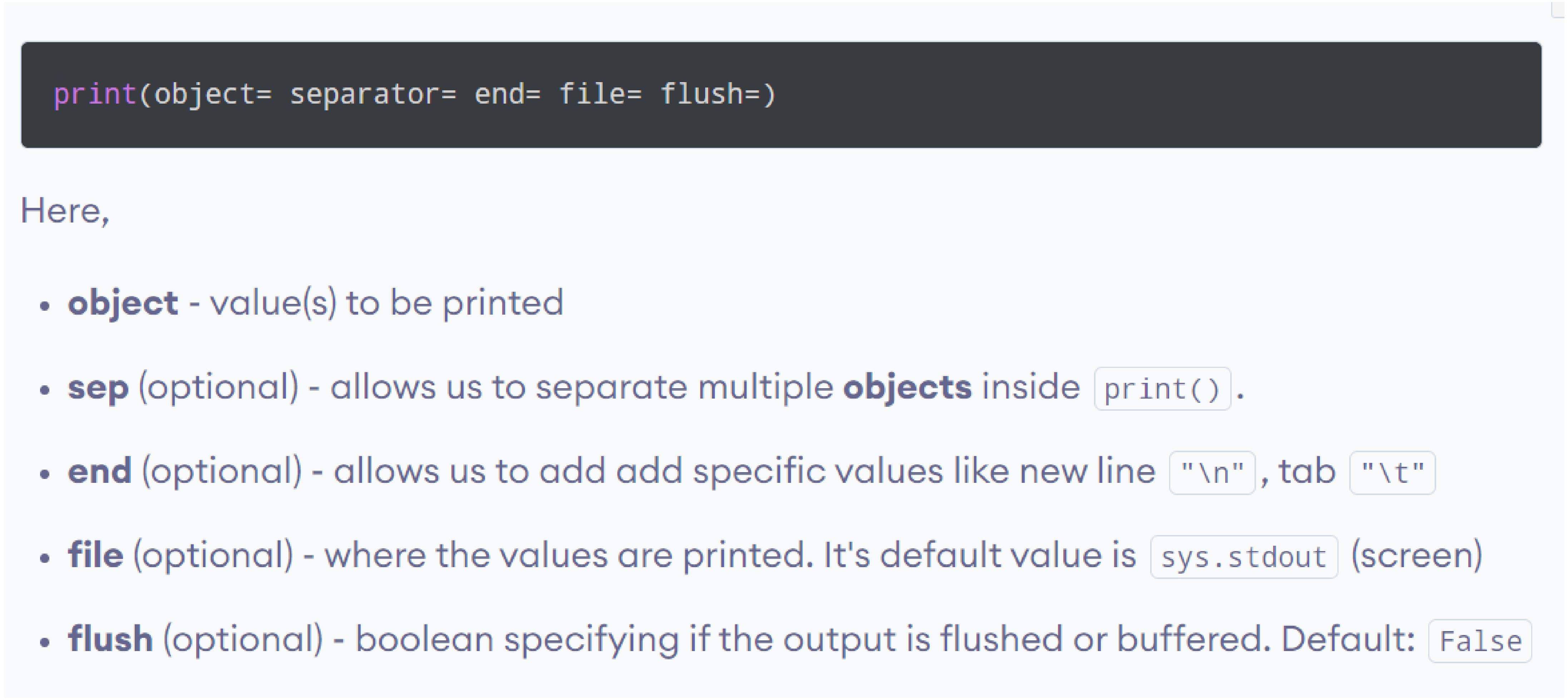
- Output: display the data to users (Console/File)

print(<value>):



```
print('Python is powerful')

# Output: Python is powerful
```

print()

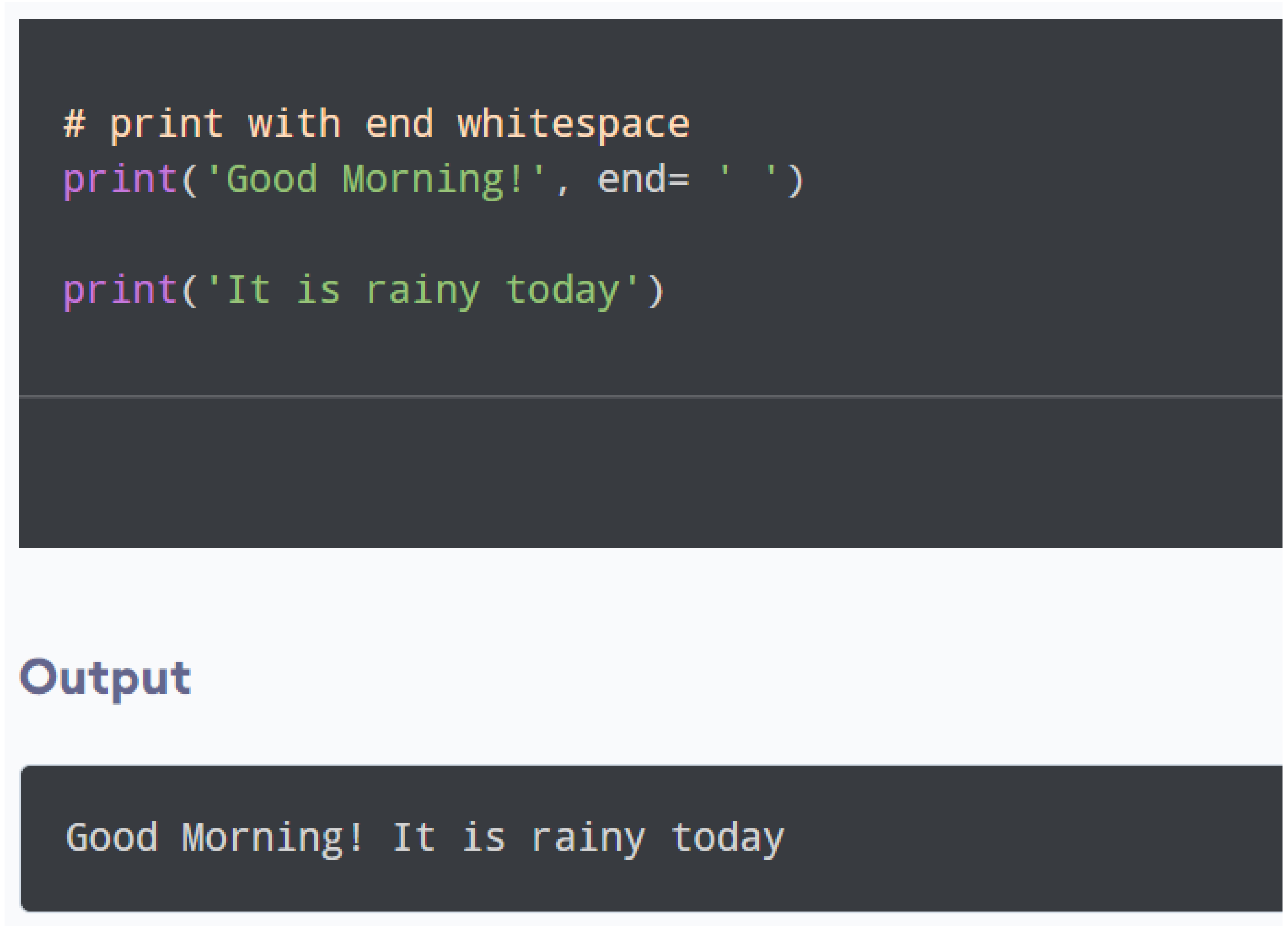


```
print(object= separator= end= file= flush=)
```

Here,

- **object** - value(s) to be printed
- **sep** (optional) - allows us to separate multiple **objects** inside `print()`.
- **end** (optional) - allows us to add add specific values like new line `"\n"`, tab `"\t"`
- **file** (optional) - where the values are printed. It's default value is `sys.stdout` (screen)
- **flush** (optional) - boolean specifying if the output is flushed or buffered. Default: `False`

print()

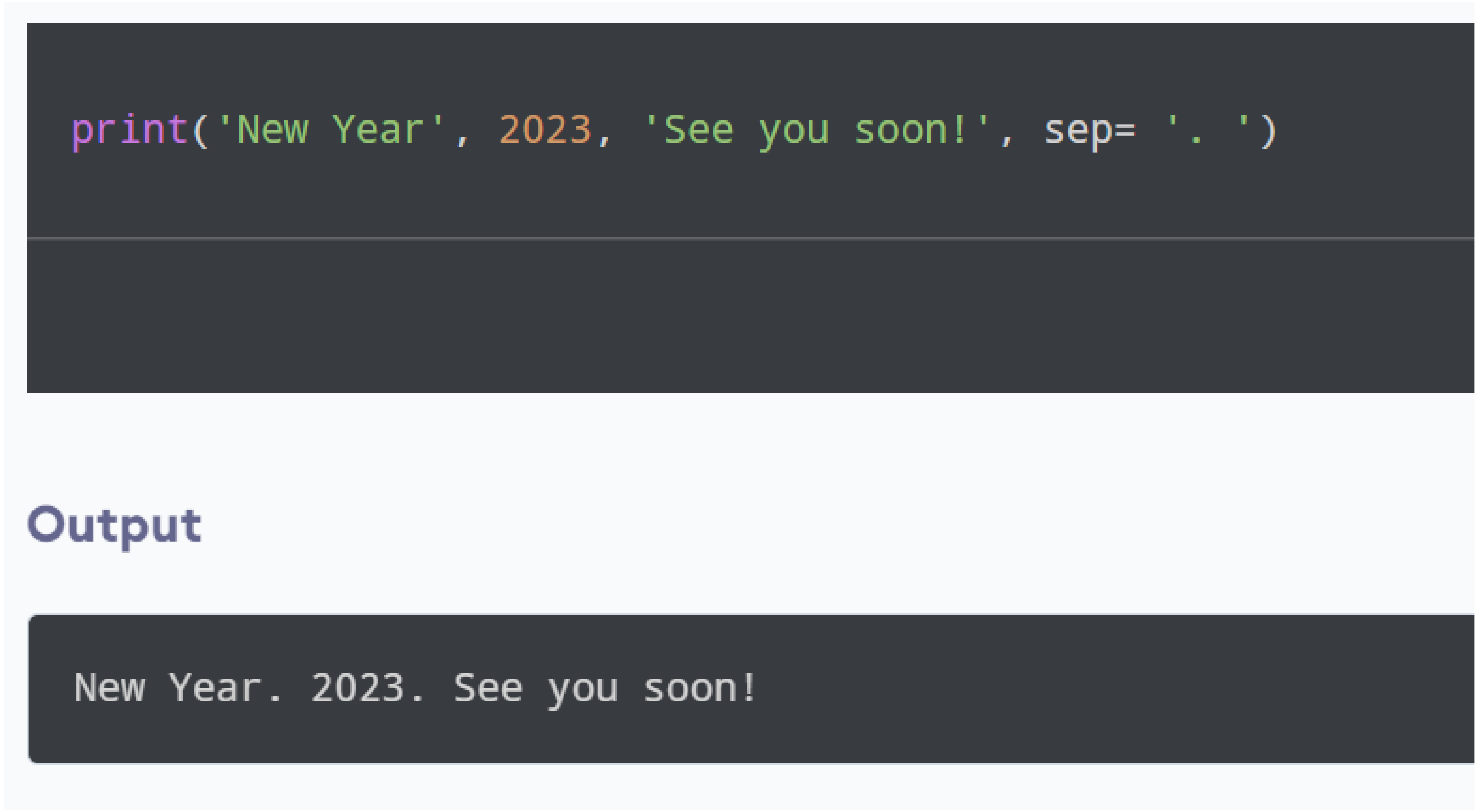


```
# print with end whitespace
print('Good Morning!', end= ' ')

print('It is rainy today')
```

Output

```
Good Morning! It is rainy today
```

print()

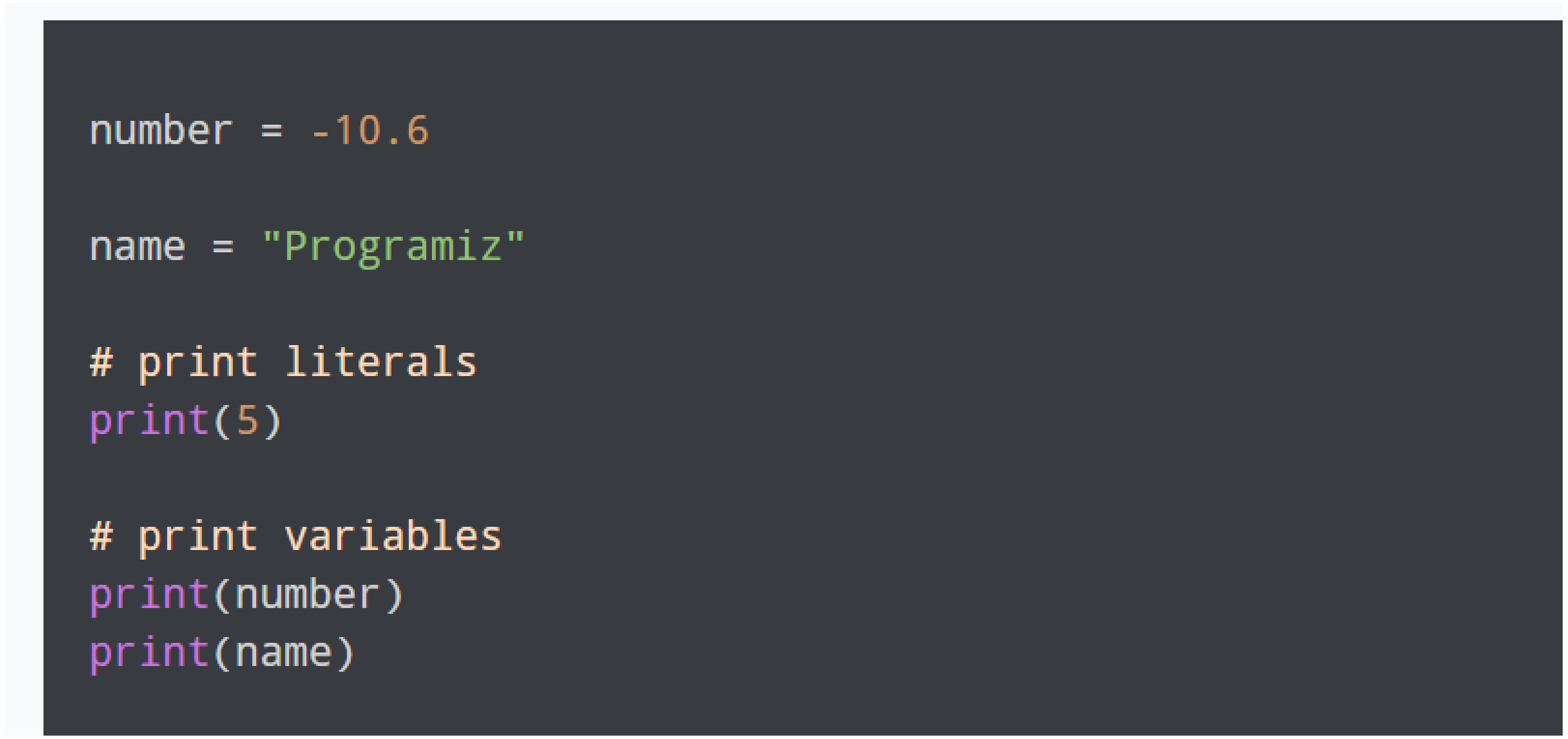


```
print('New Year', 2023, 'See you soon!', sep= '. ')
```

Output

```
New Year. 2023. See you soon!
```

print()



```python
number = -10.6

name = "Programiz"

# print literals
print(5)

# print variables
print(number)
print(name)
```

Input/Output

fit@hcmus

- **Output: display the data to users (Console/File)**

print(<value>):

```python
print('Python is powerful')

# Output: Python is powerful
```

- **Input: get the data from users (Console/File)**

input([prompt])

```python
# using input() to take user input
num = input('Enter a number: ')

print('You Entered:', num)

print('Data type of num:', type(num))
```

  - However, all input value is string

3. Flowchart

Problem Solving

fit@hcmus

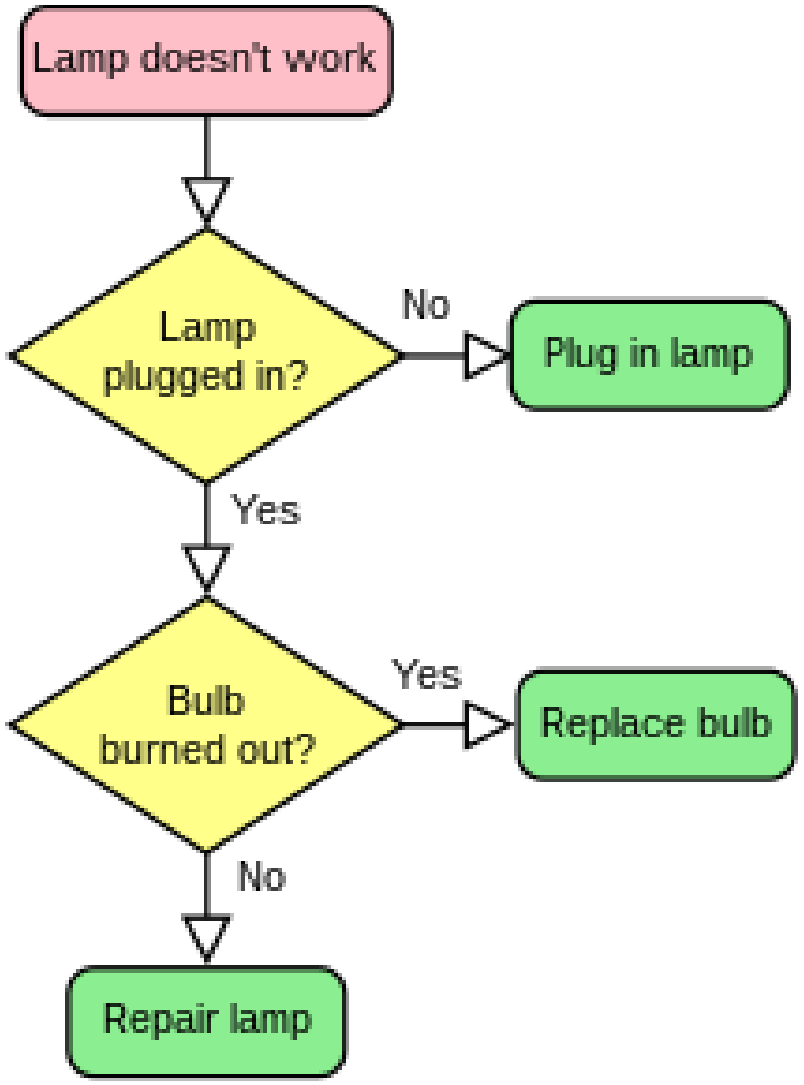
- For problem in programming, we need to identify:
    - Input
    - Output
    - Algorithm: In Computer Science, an algorithm is a list set of instructions, used to solve problems or perform tasks, based on the understanding of available alternatives.

Problem Solving

- I want to check whether 2023 is a leaf year or not. What is the input/output of this problem

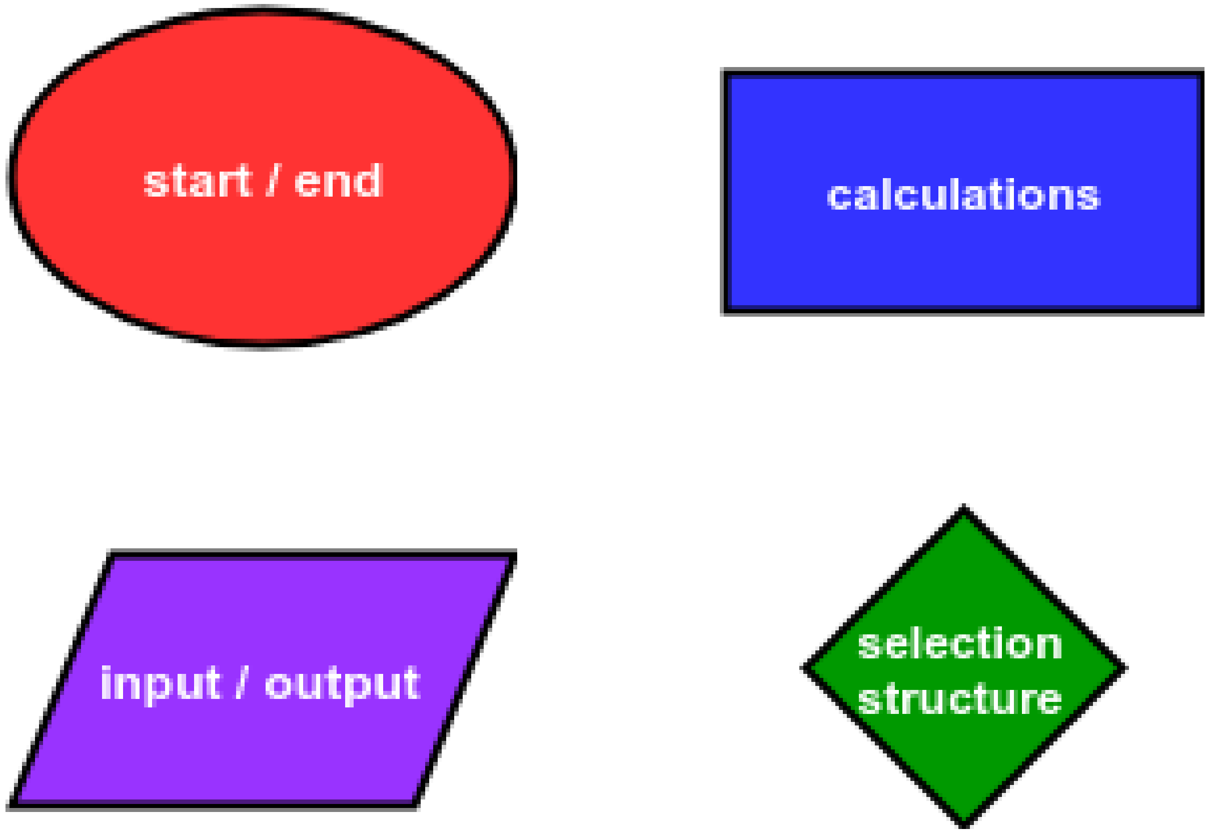
Flowchart

- Python is sequentially programming language
- A flowchart is a type of diagram that represents a workflow or process
- Use flowchart to visualize the algorithm



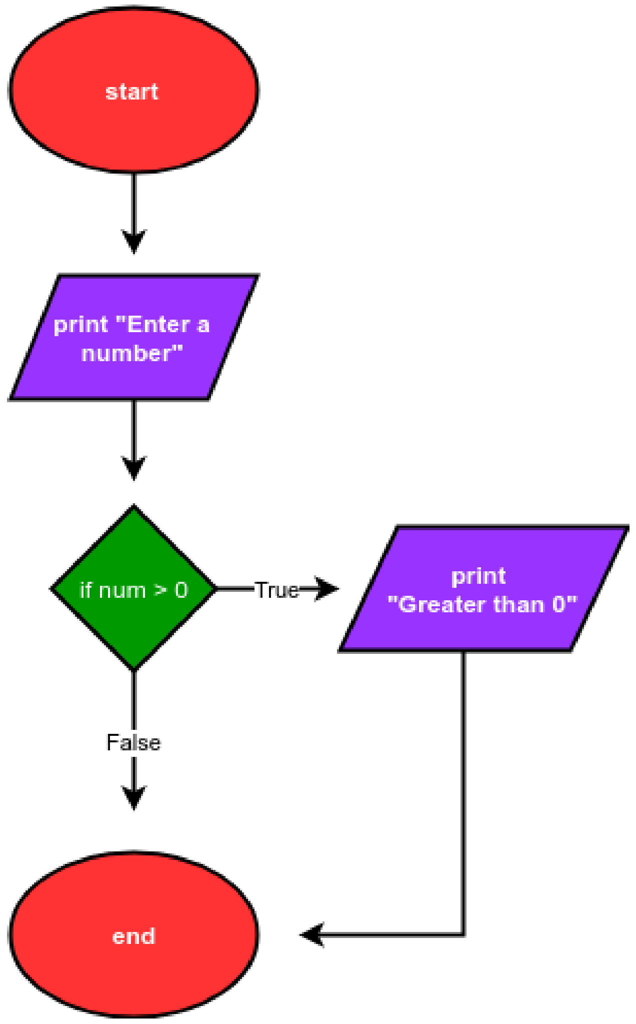
Flowchart

- Four basic shapes in flowchart:
  - oval: start / end
  - parallelogram: input / output
  - rectangle: calculations
  - diamond: selection structures



Flowchart

- Explain the flow of the following figure



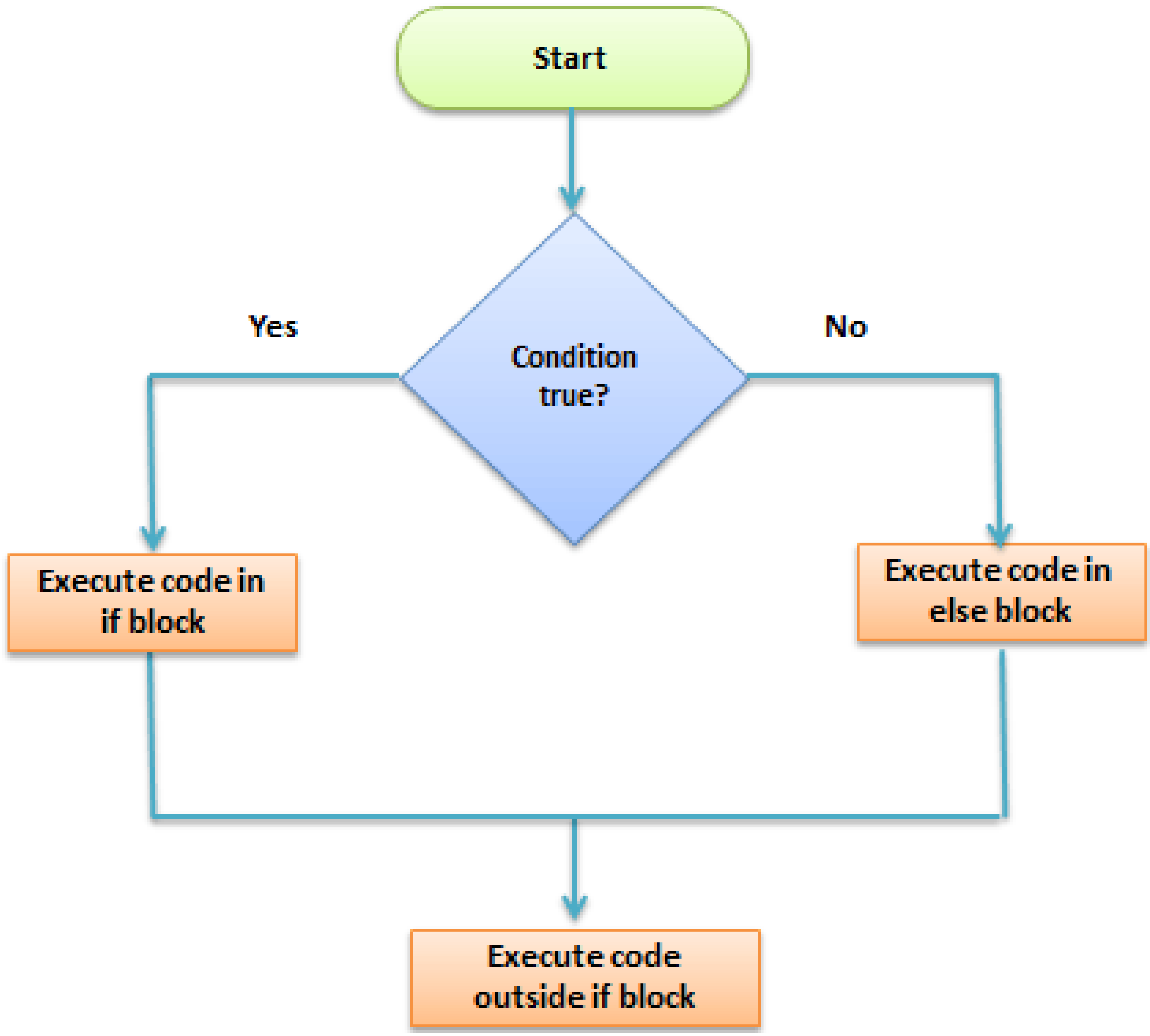
Flowchart

- Draw a flowchart to display the absolute value of integer x

4. Conditional Statements

Condition

- Condition or selection, need to detect:
  - A condition
  - True action
  - False action (optional)



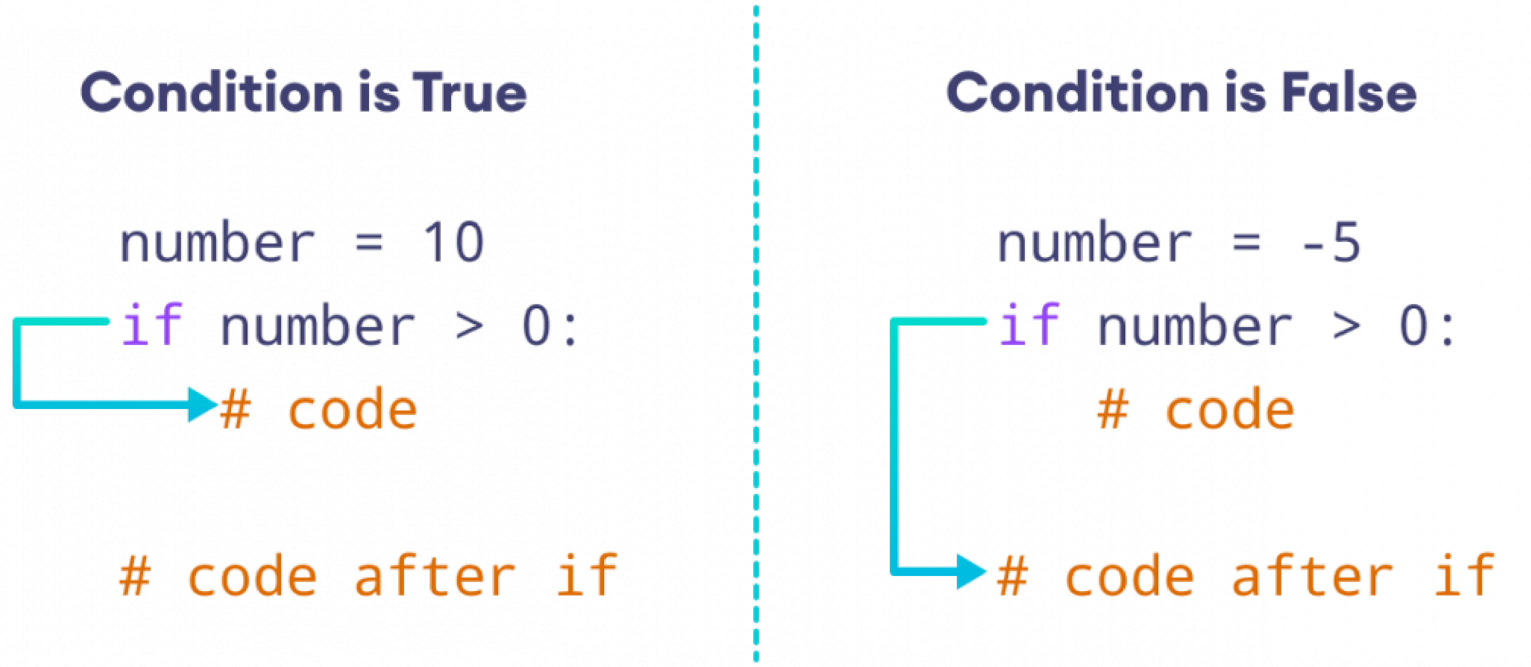
Condition

- Obtain an integer score. If score is bigger than 5.0, display “Pass”, otherwise “Not pass”. Draw a flowchart to solve this problem

Condition: if

- In python, condition is represented by:

```
if condition:
    # body of if statement
```

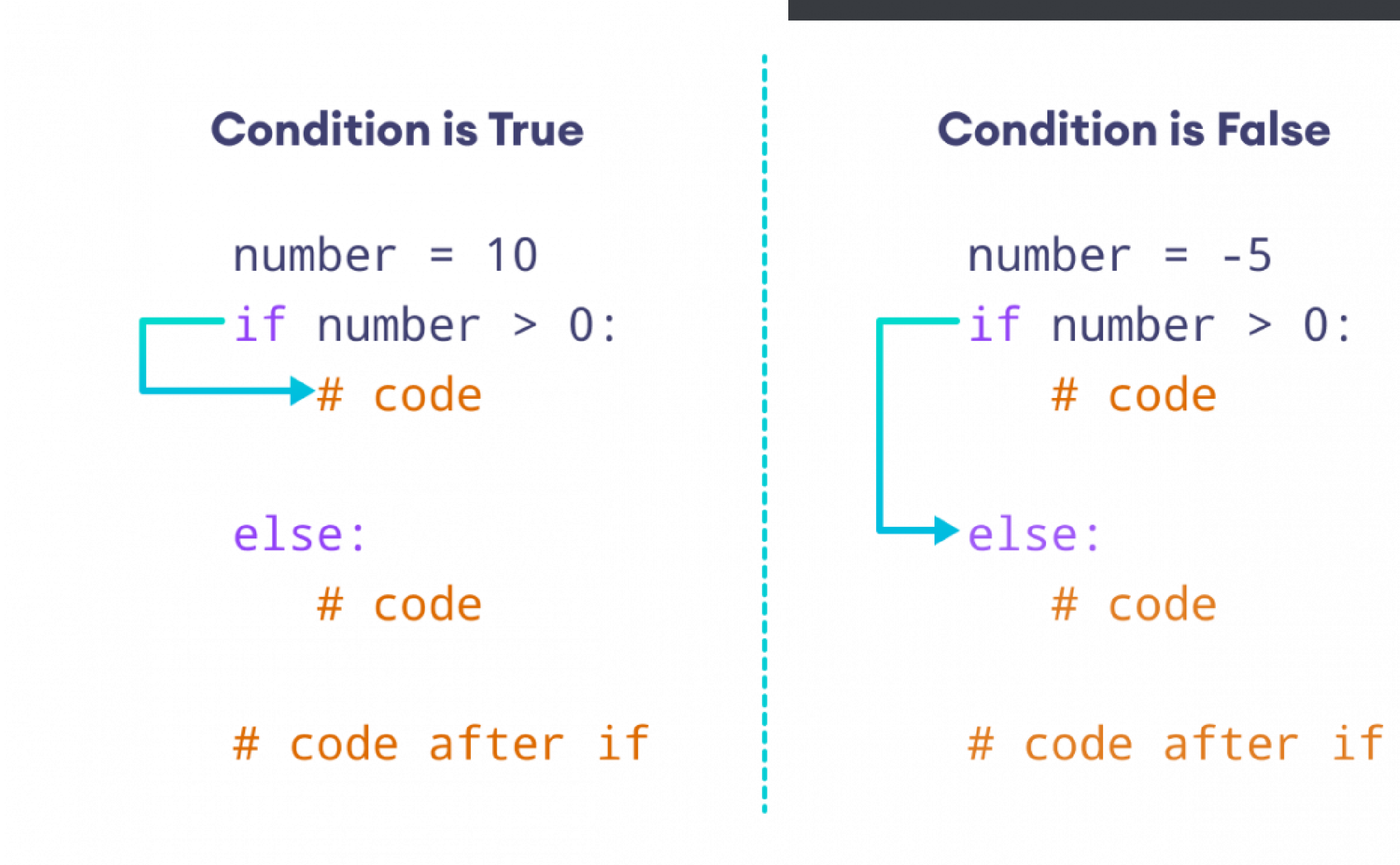


```
number = 10
if number > 0:
    # code

# code after if
```

```
number = -5
if number > 0:
    # code

# code after if
```

Condition: if … else

```
if condition:
    # block of code if condition is True

else:
    # block of code if condition is False
```



Condition is True

```
number = 10
if number > 0:
    # code

else:
    # code

# code after if
```

Condition is False

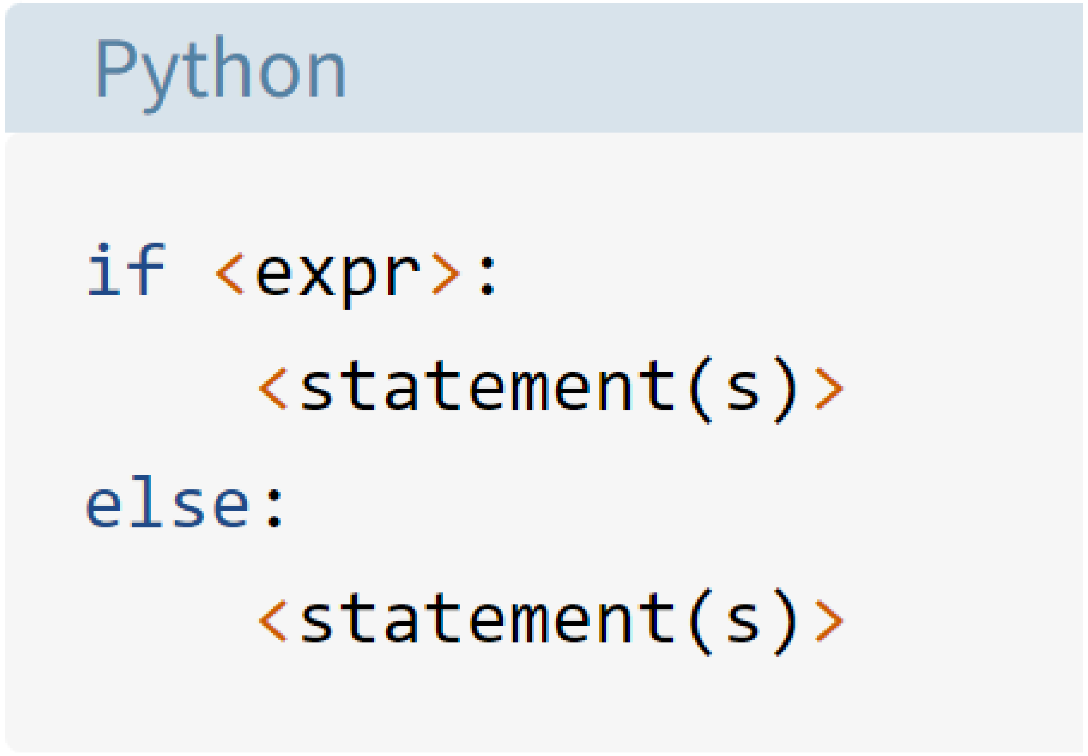
```
number = -5
if number > 0:
    # code

else:
    # code

# code after if
```

Condition

- Obtain an integer score. If score is bigger than 5.0, display “Pass”, otherwise “Not pass”
  - Draw a flowchart

Condition

- In python, condition is represented by:
  - Input: int x
  - Output: Pass, Not Pass
  - Condition: x > 5.0
  - True statement: print – Pass
  - False statement: print – Not pass



Python

```
if <expr>:
    <statement(s)>
else:
    <statement(s)>
```

Logical conditions

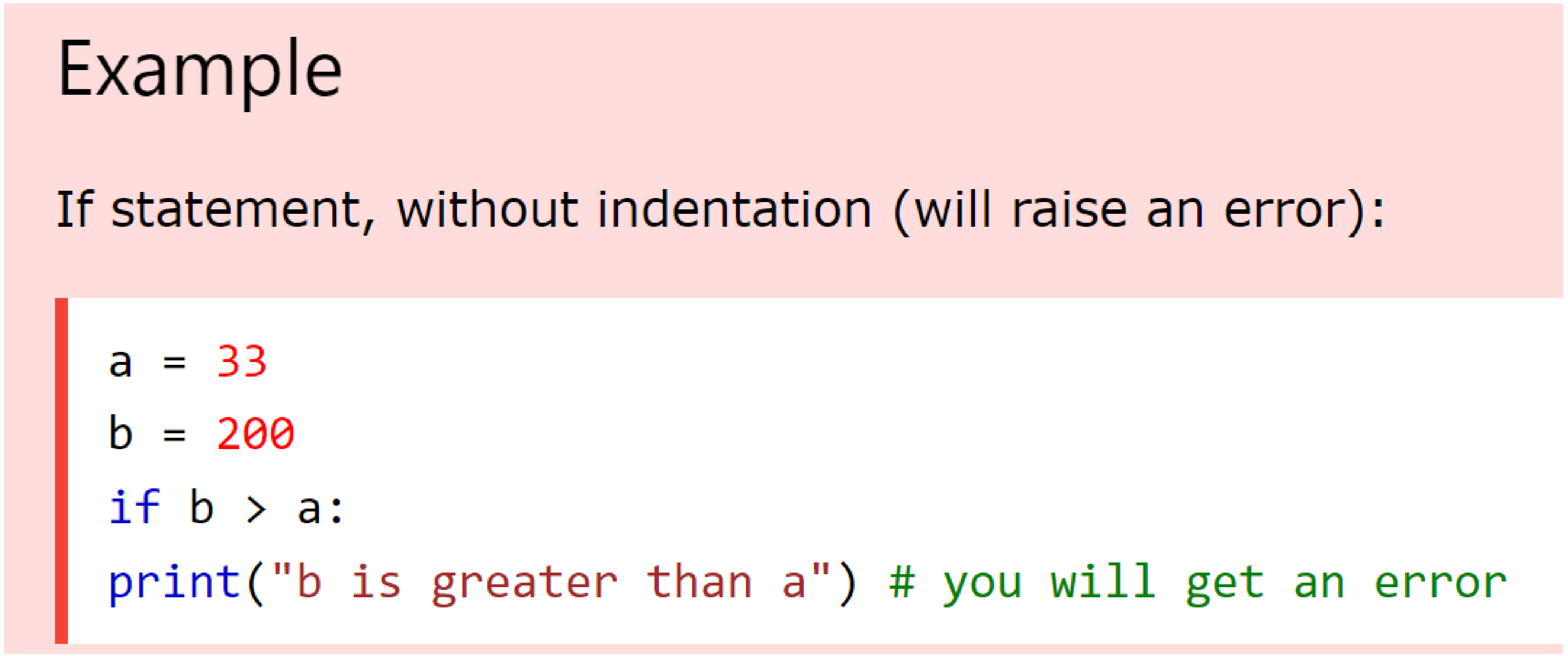
Python

```
if <expr>:
    <statement(s)>
else:
    <statement(s)>
```

- Equals: `a == b`
- Not Equals: `a != b`
- Less than: `a < b`
- Less than or equal to: `a <= b`
- Greater than: `a > b`
- Greater than or equal to: `a >= b`

Indentation

- Indentation is a way to separate the statement's block in Python



Example

If statement, without indentation (will raise an error):

```
a = 33
b = 200
if b > a:
print("b is greater than a") # you will get an error
```

Exercise

- Write a program to calculate the absolute value of integer n

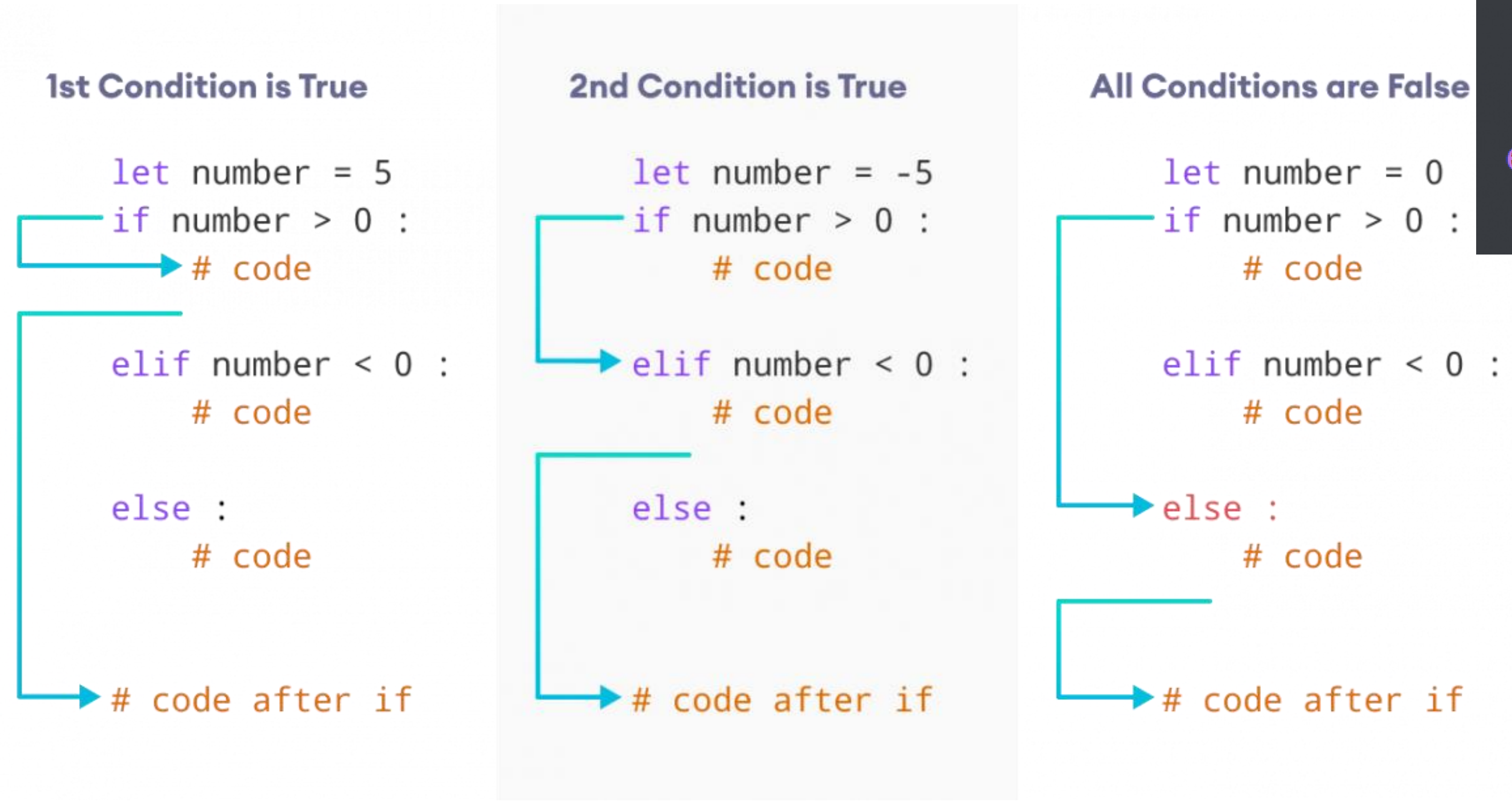
Exercise

- Write a program to find the biggest number between two integers from keyboards

Condition: if … elif … else

```
if condition1:
    # code block 1

elif condition2:
    # code block 2

else:
    # code block 3
```

-



```
let number = 5
if number > 0 :
    # code

elif number < 0 :
    # code

else :
    # code

# code after if
```

```
let number = -5
if number > 0 :
    # code

elif number < 0 :
    # code

else :
    # code

# code after if
```

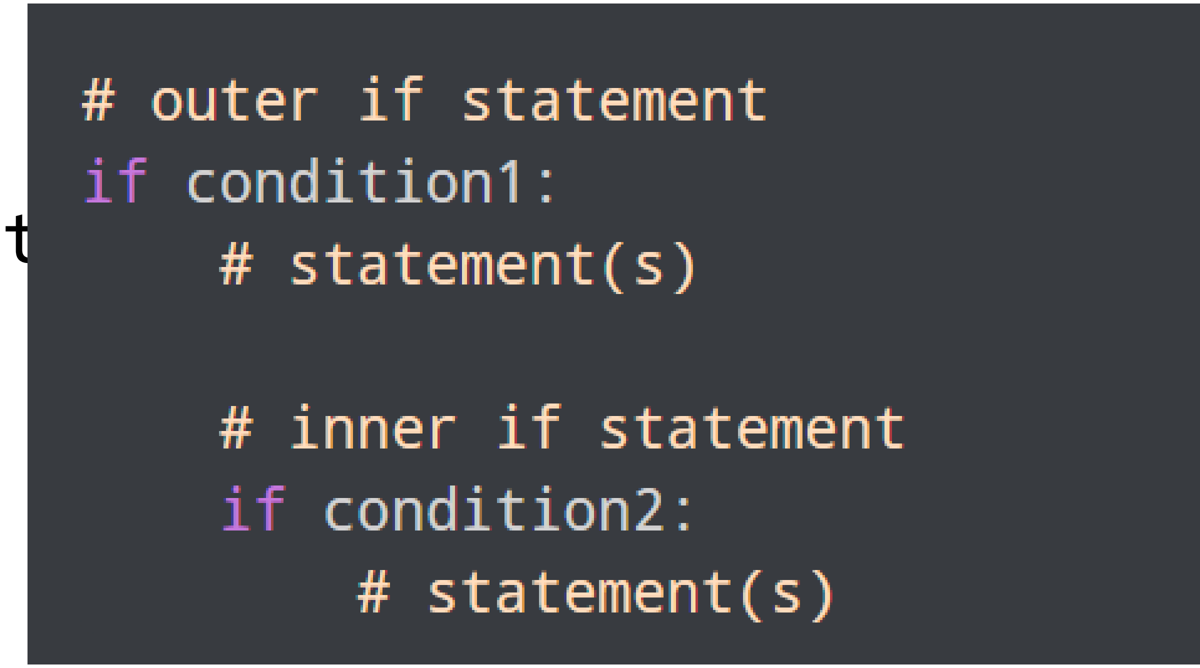
```
let number = 0
if number > 0 :
    # code

elif number < 0 :
    # code

else :
    # code

# code after if
```

Nested Condition

- For statements in `condition2`: it also meet the condition1

- Can be rewritten as:

  if (`condition1` and `condit`



```
# outer if statement
if condition1:
    # statement(s)

    # inner if statement
    if condition2:
        # statement(s)
```

Exercises

fit@hcmus

- Write a program to input a positive integer n. Check whether n is a squared number or not? (A squared number is a number that when taking the square root of 2, the result is an integer)

Exercises

- Write a function that checks for a leap year (leap year: divisible by 4 and not divisible by 100 or divisible by 400)

Exercises

- Enter 1 month and year, show how many days that month (month has 31 days: 1, 3, 5, 7, 8, 10, 12; month has 30 days: 4, 6, 9, 11; February have 28 or 29 days)

Exercises

fit@hcmus

- Print out these 3 numbers in ascending order

Homework 1

- Enter the lengths of 3 sides a, b, c of a triangle. Check whether a,b,c are 3 sides form a triangle or not?
  - Display: “TRUE” $\rightarrow$ if yes
  - Display: “FALSE” $\rightarrow$ if no

Homework 2

- Calculate taxi fare from the number of kilometers entered, knowing:
  - First 1 km costs 15,000 VND,
  - From the 2nd to the 5th km, the price is 13,500 VND,
  - From the 6th kilometer onwards, the price is 11,000 VND,
  - If you go more than 120km, you will receive a 10% discount on the total amount

THANK YOU

for YOUR ATTENTION