University of Science - VNU-HCM
Faculty of Information Science
Department of Computer Science

MTH083 - Advanced Programming for Artificial Intelligence

Slot 01-
Python Overview

Advisor:

Dr. Nguyễn Tiến Huy

Dr. Lê Thanh Tùng

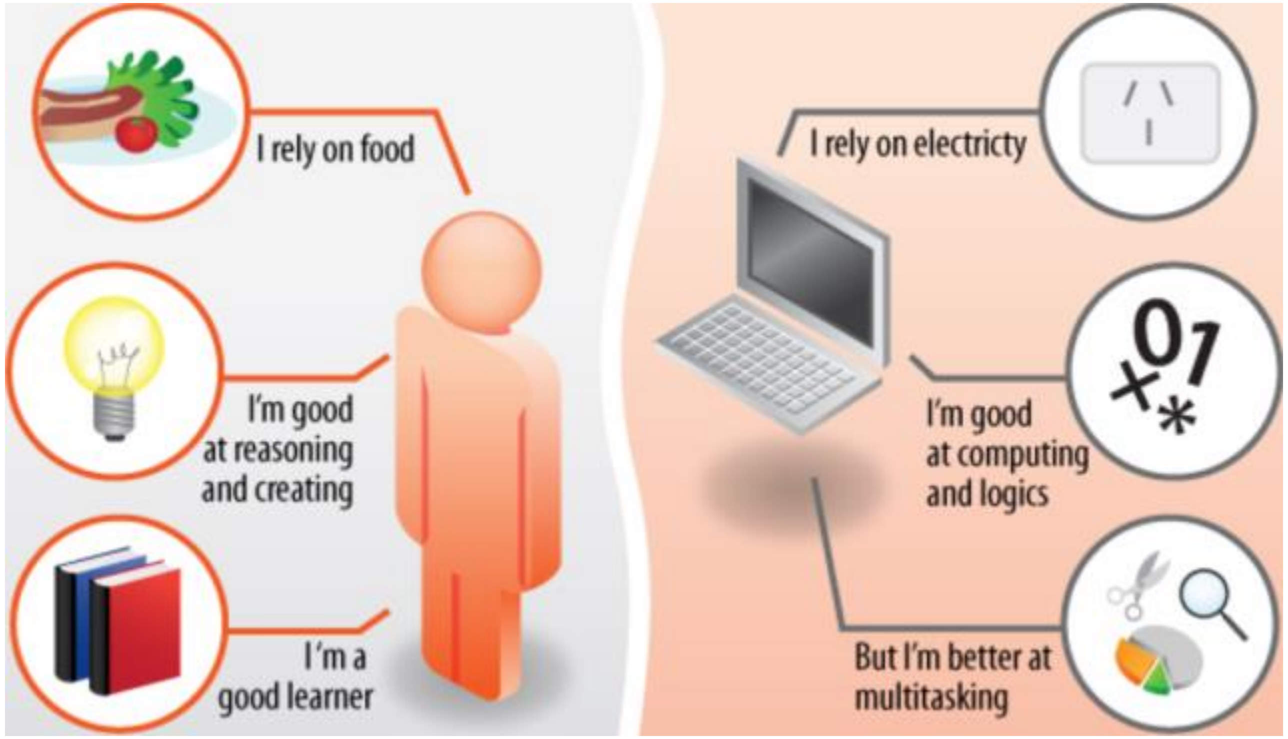
Content

1. Introduction
2. Input/Output Operations
3. Flowchart – Problem Solving

Part 1: Introduction

Introduction

- Computers are built for one purpose - to do things for us
- But we need to speak their language to describe what we want done

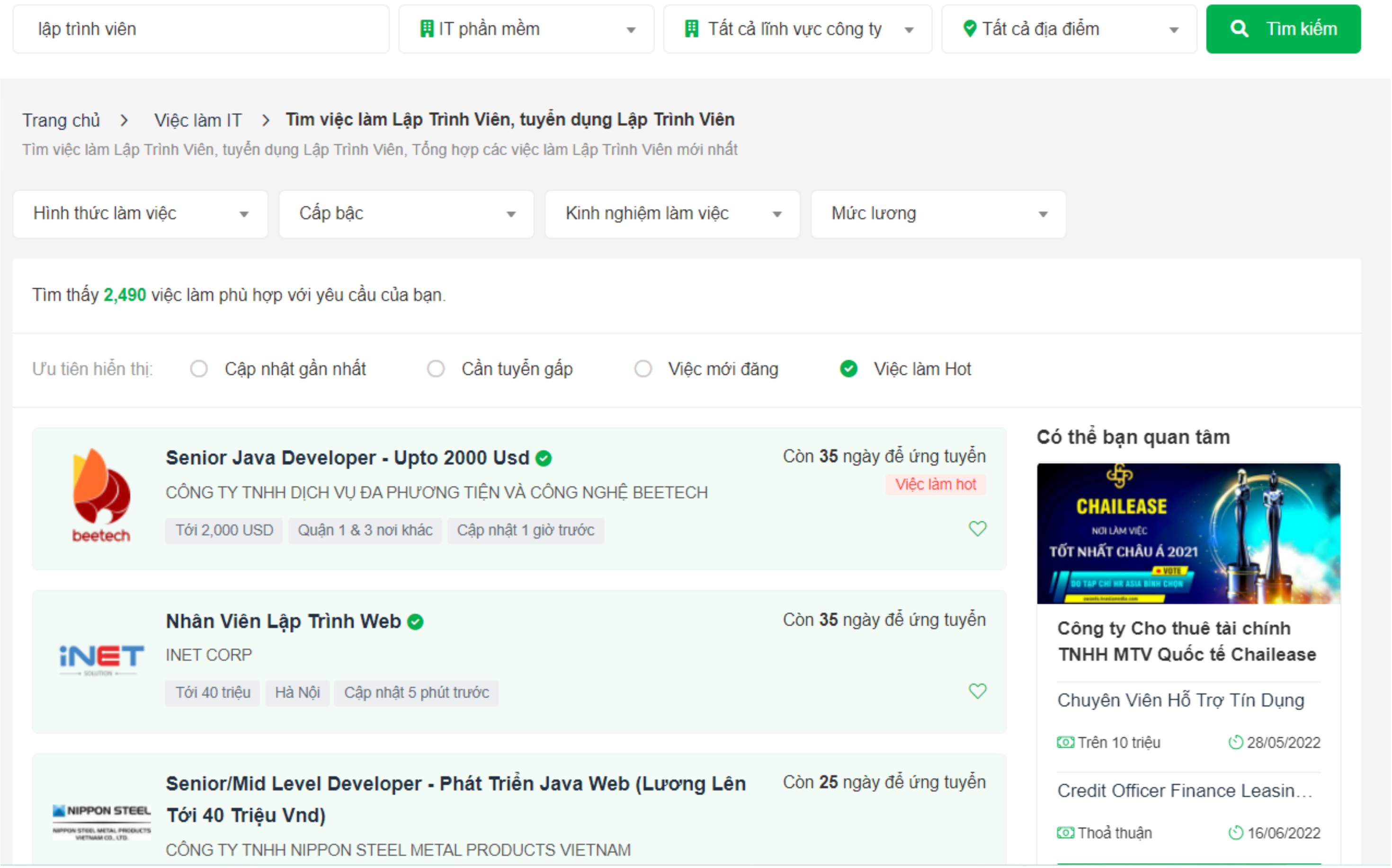

I rely on food
I'm good at reasoning and creating
I'm a good learner
I rely on electricty
I'm good at computing and logics
But I'm better at multitasking
01
x
*

- Users have it easy - someone already put many different programs (instructions) into the computer and users just pick the ones they want to use

Introduction

Programmers Anticipate Needs

- iPhone applications are a market
- iPhone applications have over 3 billion downloads
- Programmers have left their jobs to be full-time iPhone developers
- Programmers know the ways of the program



lập trình viên | IT phần mềm ▾ | Tất cả lĩnh vực công ty ▾ | Tất cả địa điểm ▾ | Tìm kiếm

Trang chủ > Việc làm IT > Tìm việc làm Lập Trình Viên, tuyển dụng Lập Trình Viên

Tìm việc làm Lập Trình Viên, tuyển dụng Lập Trình Viên, Tổng hợp các việc làm Lập Trình Viên mới nhất

Hình thức làm việc ▾ | Cấp bậc ▾ | Kinh nghiệm làm việc ▾ | Mức lương ▾

Tìm thấy **2,490** việc làm phù hợp với yêu cầu của bạn.

Ưu tiên hiển thị: ○ Cập nhật gần nhất ○ Cần tuyển gấp ○ Việc mới đăng ● Việc làm Hot

Senior Java Developer - Upto 2000 Usd ✓
CÔNG TY TNHH DỊCH VỤ ĐA PHƯƠNG TIỆN VÀ CÔNG NGHỆ BEETECH
Tới 2,000 USD | Quận 1 & 3 nơi khác | Cập nhật 1 giờ trước
Còn **35** ngày để ứng tuyển
Việc làm hot

Nhân Viên Lập Trình Web ✓
INET CORP
Tới 40 triệu | Hà Nội | Cập nhật 5 phút trước
Còn **35** ngày để ứng tuyển

Senior/Mid Level Developer - Phát Triển Java Web (Lương Lên Tới 40 Triệu Vnd)
CÔNG TY TNHH NIPPON STEEL METAL PRODUCTS VIETNAM
Còn **25** ngày để ứng tuyển

Có thể bạn quan tâm

Công ty Cho thuê tài chính TNHH MTV Quốc tế Chailease

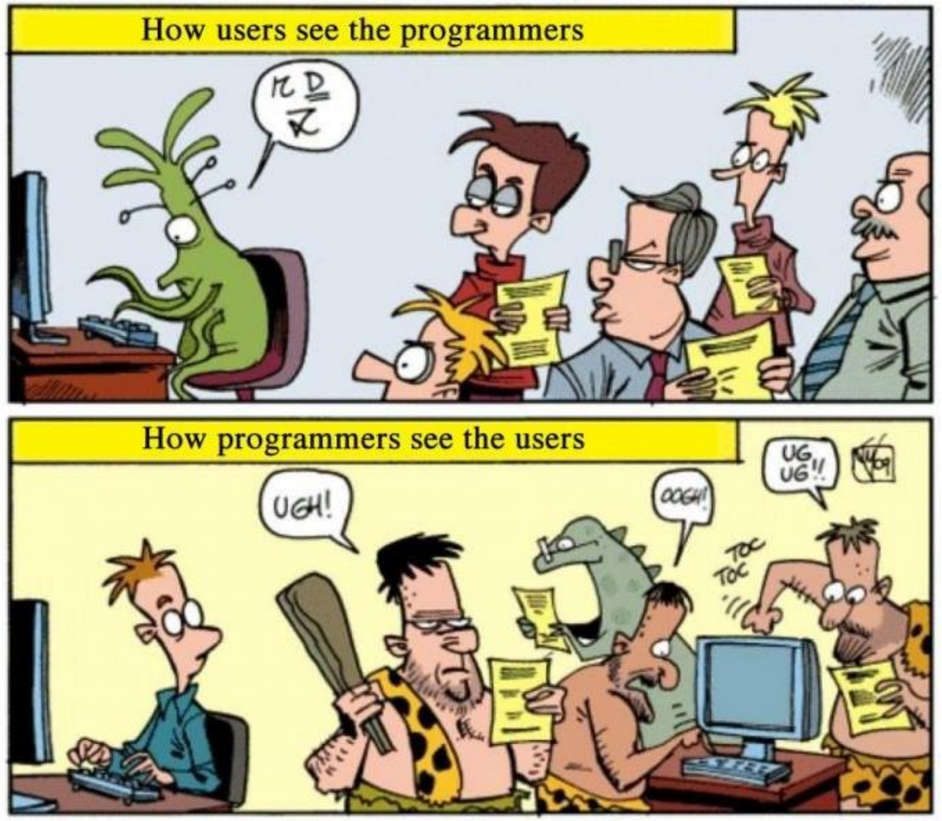
Chuyên Viên Hỗ Trợ Tín Dụng
Trên 10 triệu | 28/05/2022

Credit Officer Finance Leasin...
Thoả thuận | 16/06/2022

Introduction

Users vs. Programmers

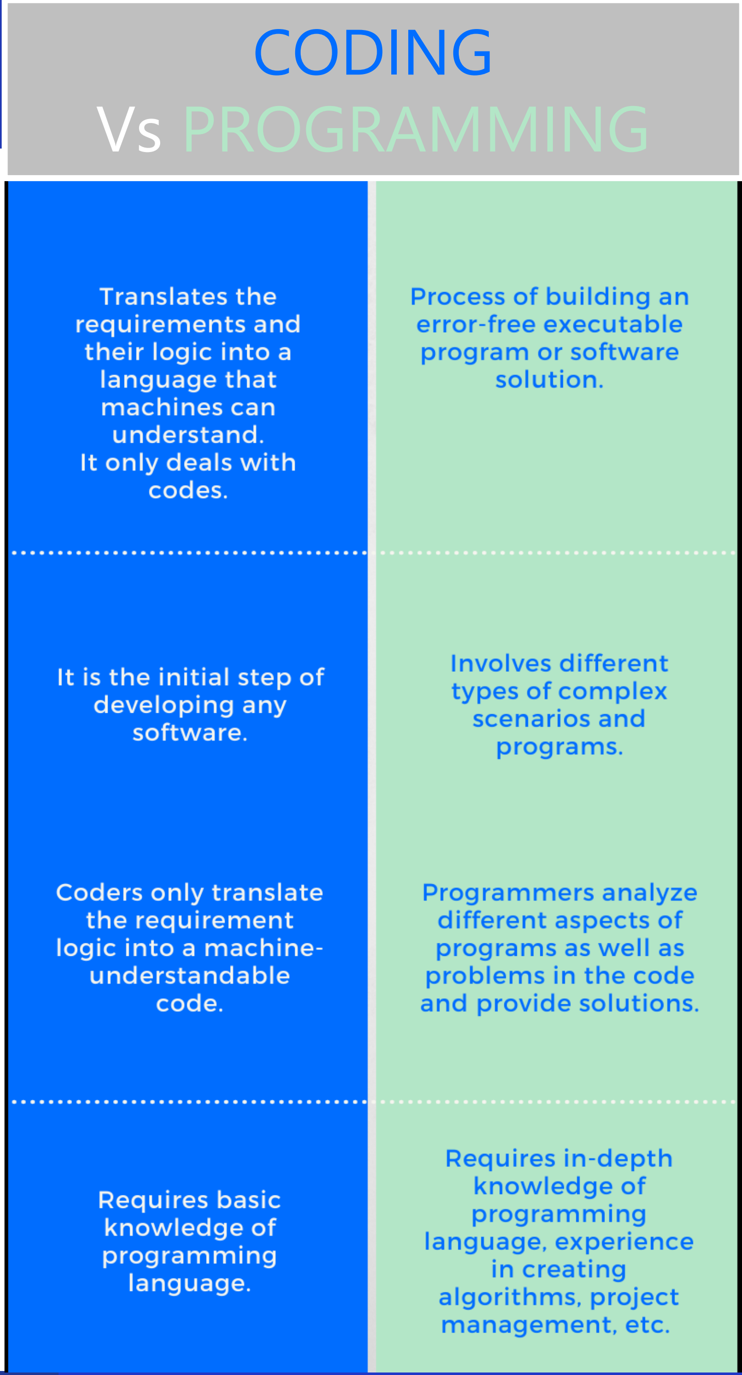
- Users see computers as a set of tools
- Programmers learn the computer "ways" and the computer language
- Programmers have some tools that allow them to build new tools
- Programmers sometimes write tools for lots of users and sometimes programmers write little "helpers" for themselves to automate a task



Introduction

What is Code? Software? A Program?

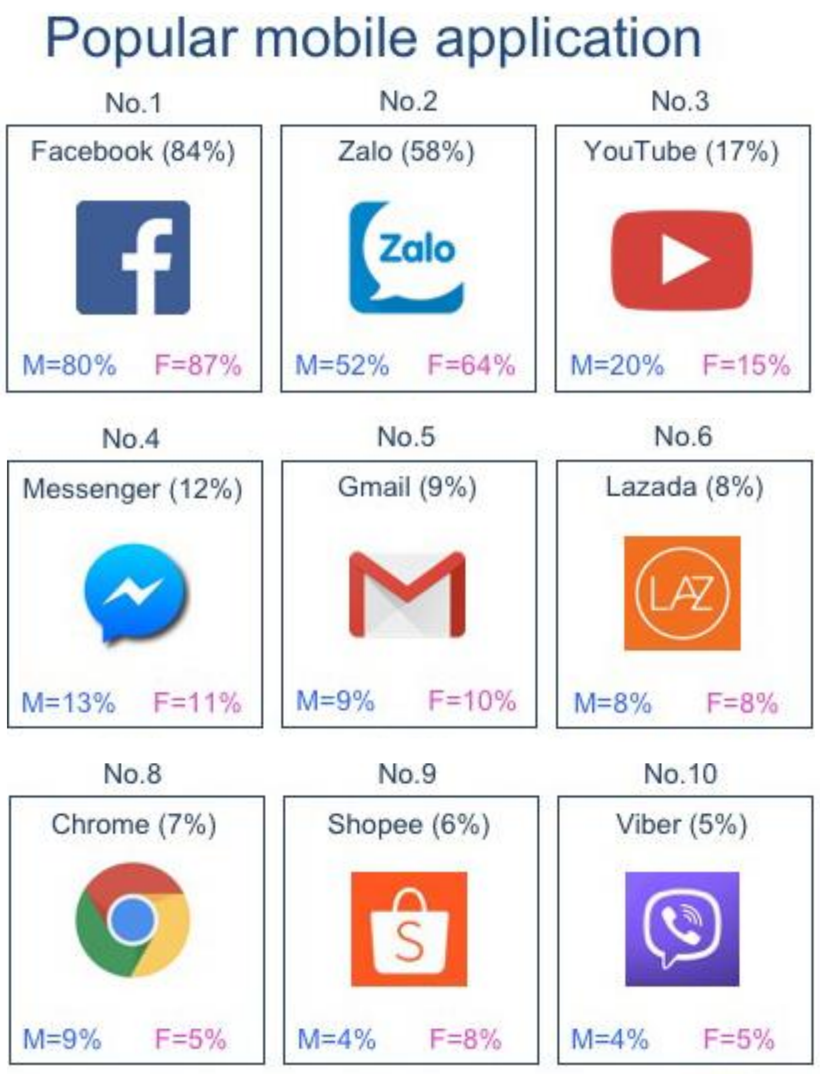
- A sequence of stored instructions
- It is a little piece of our intelligence in the computer
- We figure something out and then we encode it and then give it to someone else to save them the time and energy of figuring it out
- A piece of creative art - particularly when we do a good job on user experience



CODING	Vs PROGRAMMING
Translates the requirements and their logic into a language that machines can understand. It only deals with codes.	Process of building an error-free executable program or software solution.
It is the initial step of developing any software.	Involves different types of complex scenarios and programs.
Coders only translate the requirement logic into a machine-understandable code.	Programmers analyze different aspects of programs as well as problems in the code and provide solutions.
Requires basic knowledge of programming language.	Requires in-depth knowledge of programming language, experience in creating algorithms, project management, etc.

Introduction

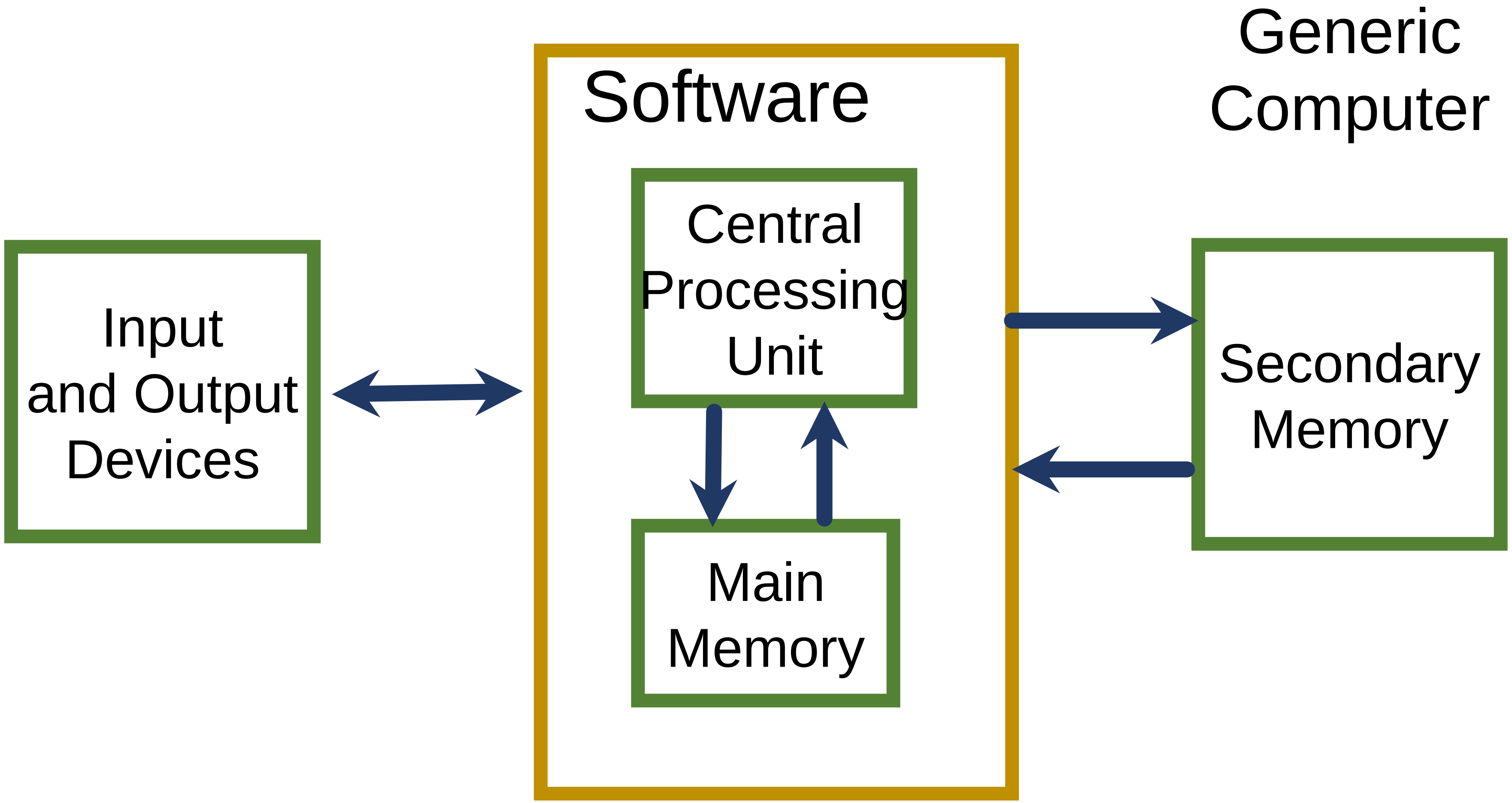
- Why do we program ?
- Computers are our assistances
- They need to be taught how to work
- To teach a computer working, we use a programming language.
- Support our lives



Popular mobile application

No.	Application	M	F
No.1	Facebook (84%)	M=80%	F=87%
No.2	Zalo (58%)	M=52%	F=64%
No.3	YouTube (17%)	M=20%	F=15%
No.4	Messenger (12%)	M=13%	F=11%
No.5	Gmail (9%)	M=9%	F=10%
No.6	Lazada (8%)	M=8%	F=8%
No.8	Chrome (7%)	M=9%	F=5%
No.9	Shopee (6%)	M=4%	F=8%
No.10	Viber (5%)	M=4%	F=5%

Hardware Architecture

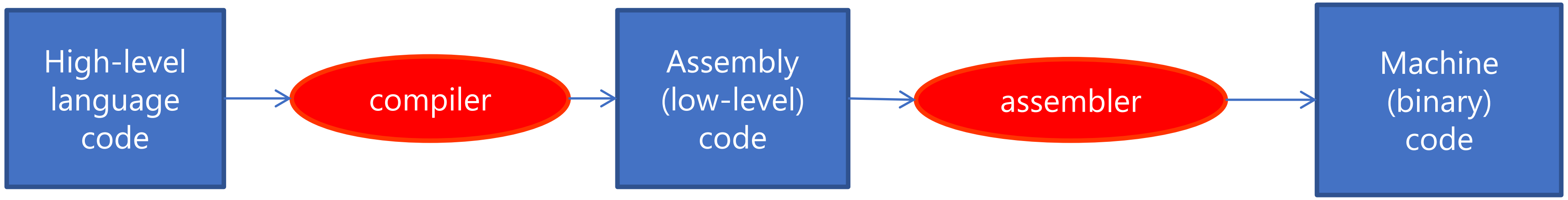


Hardware Architecture

- **Input/Output Devices**:
- **Input Devices**: Keyboard, Mouse, Touch Screen
- **Output Devices**: Screen, Speakers, Printer, DVD Burner
- **Central Processing Unit**: Runs the Program - The CPU is always wondering "what to do next". Not the brains exactly - very dumb but very very fast
- **Main Memory**: Fast small temporary storage - lost on reboot - aka RAM
- **Secondary Memory**: Slower large permanent storage - lasts until deleted - disk drive / memory stick

Python

- **Python** is the language of the Python **Interpreter** and those who can converse with it. An individual who can speak Python is known as a Pythonista. It is a very uncommon skill, and may be hereditary. Nearly all known Pythonistas use software initially developed by Guido van Rossum.



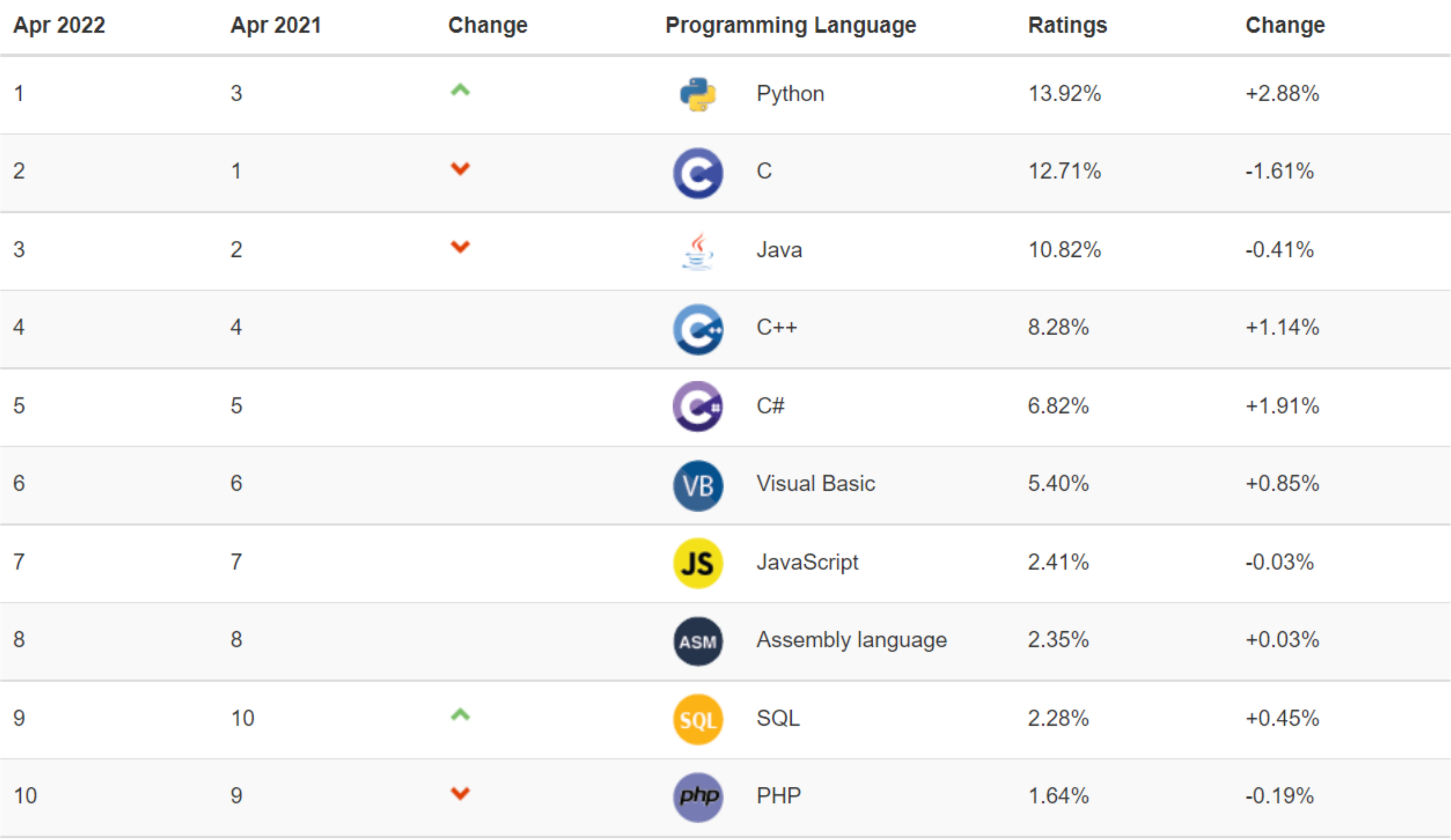
```
swap(int v[], int k)
{int temp;
   temp = v[k];
   v[k] = v[k+1];
   v[k+1] = temp;
}
```

```
swap:
      muli $2, $5,4
      add  $2, $4,$2
      lw   $15, 0($2)
      lw   $16, 4($2)
      sw   $16, 0($2)
      sw   $15, 4($2)
      jr   $31
```

```
00000000101000010000000000011000
00000000100011100001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```

Python

- Why is Python chosen ?



Apr 2022	Apr 2021	Change	Programming Language	Ratings	Change
1	3	▲	Python	13.92%	+2.88%
2	1	▼	C	12.71%	-1.61%
3	2	▼	Java	10.82%	-0.41%
4	4		C++	8.28%	+1.14%
5	5		C#	6.82%	+1.91%
6	6		Visual Basic	5.40%	+0.85%
7	7		JavaScript	2.41%	-0.03%
8	8		Assembly language	2.35%	+0.03%
9	10	▲	SQL	2.28%	+0.45%
10	9	▼	PHP	1.64%	-0.19%

Python

- Program code in a high level language can not run, It must be translated to binary code (machine code) before running.
- 2 ways of translations:
  - **Interpreting**: one-by-one statement is translated then run → **Interpreter**
  - **Compiling**: All statements of program are translated then executed as a whole → **Compiler**
- C translator is a compiler
- Python is an **Interpreter**

Python

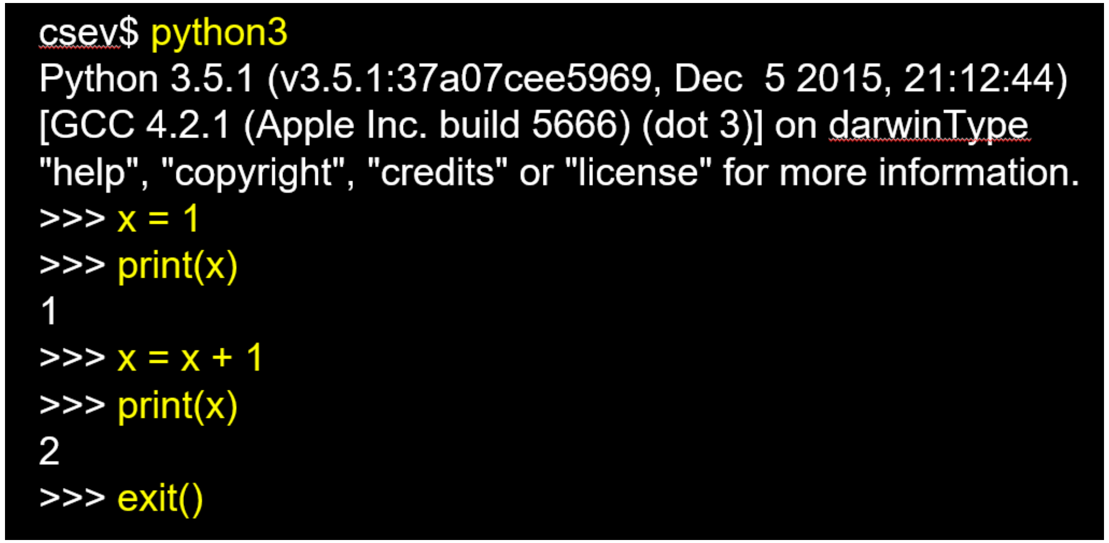
- Python is an **interpreted language**, which means the source code of a Python program is converted into bytecode that is then executed by the Python virtual machine. Python is different from major compiled languages, such as C and C + +, as **Python code is not required to be built and linked** like code for these languages. This distinction makes for two important points:

- Python code is **fast to develop**: As the code is **not needed to be compiled and built**, Python code can be readily changed and executed. This makes for a fast development cycle.

- Python code is **not as fast in execution**: Since the code is not directly compiled and executed and an additional layer of the Python virtual machine is responsible for execution, Python code runs a little slow as compared to conventional languages like C, C + +, etc.

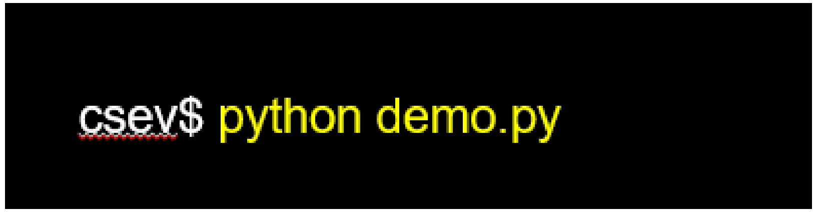
Python

- Two kinds of Python codes:
- **Interactive**: You type directly to Python one line at a time and it responds



```
csev$ python3
Python 3.5.1 (v3.5.1:37a07cee5969, Dec  5 2015, 21:12:44)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwinType
"help", "copyright", "credits" or "license" for more information.
>>> x = 1
>>> print(x)
1
>>> x = x + 1
>>> print(x)
2
>>> exit()
```

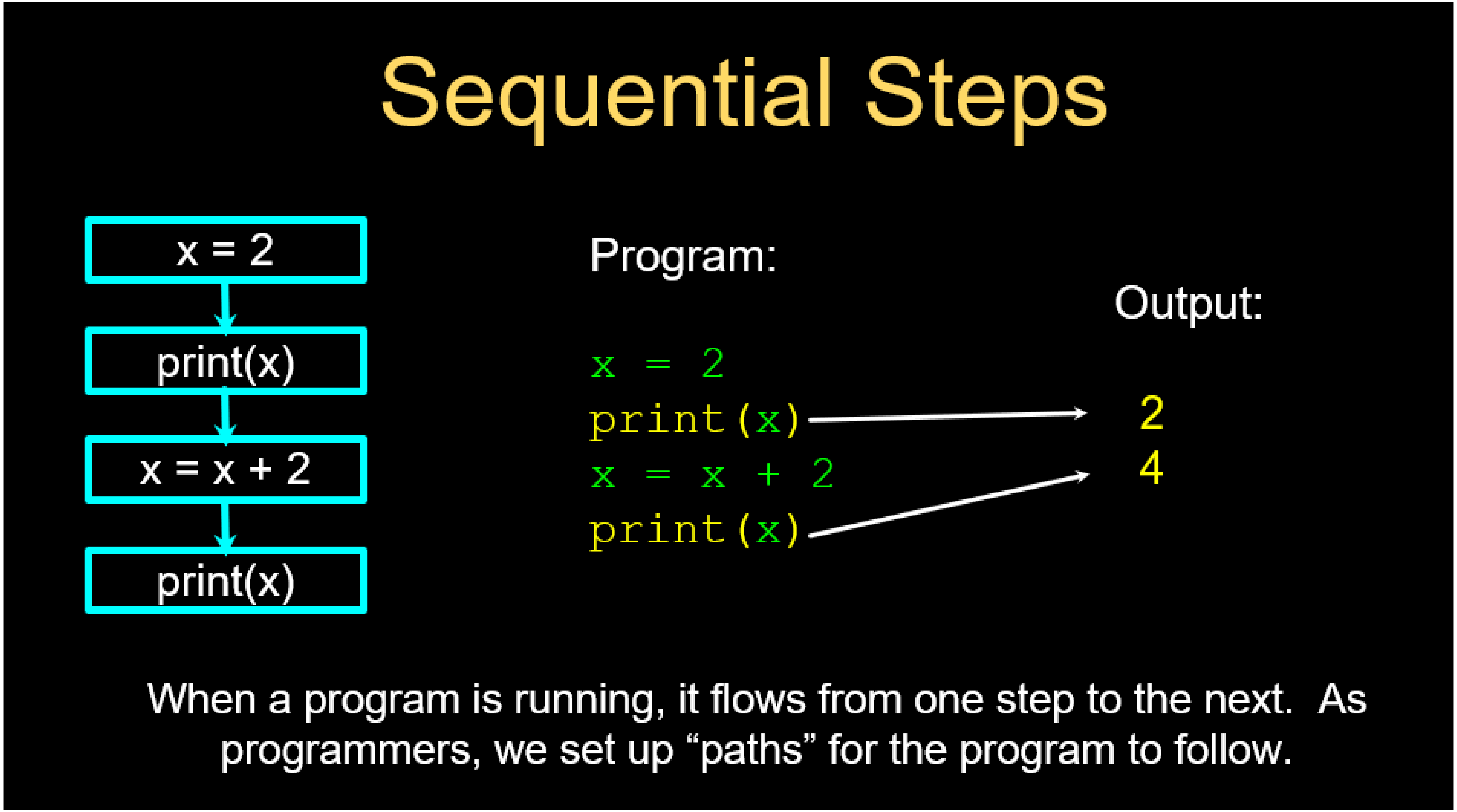
- **Script**: You enter a sequence of statements (lines) into a file using a text editor and tell Python to execute the statements in the file



```
csev$ python demo.py
```

Hardware Architecture

- Like a recipe or installation instructions, **a program is a sequence** of steps to be done in order. (sequential)
- Some steps are conditional - they may be skipped.
- Sometimes a step or group of steps is to be repeated.
- Sometimes we store a set of steps to be used over and over as needed several places throughout the program



Sequential Steps

x = 2

print(x)

x = x + 2

print(x)

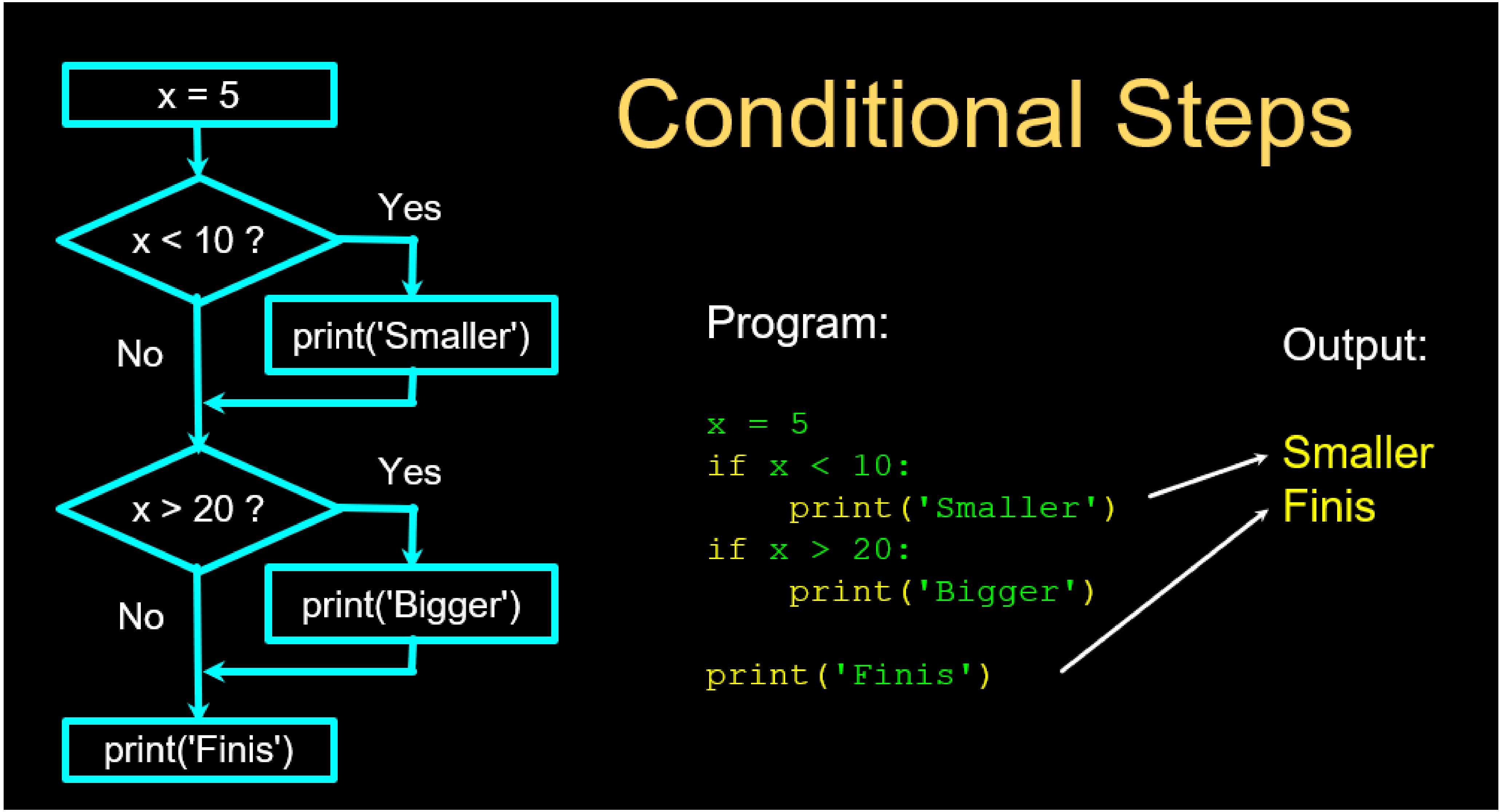
Program:

```
x = 2
print(x)
x = x + 2
print(x)
```

Output:

```
2
4
```

When a program is running, it flows from one step to the next. As programmers, we set up “paths” for the program to follow.



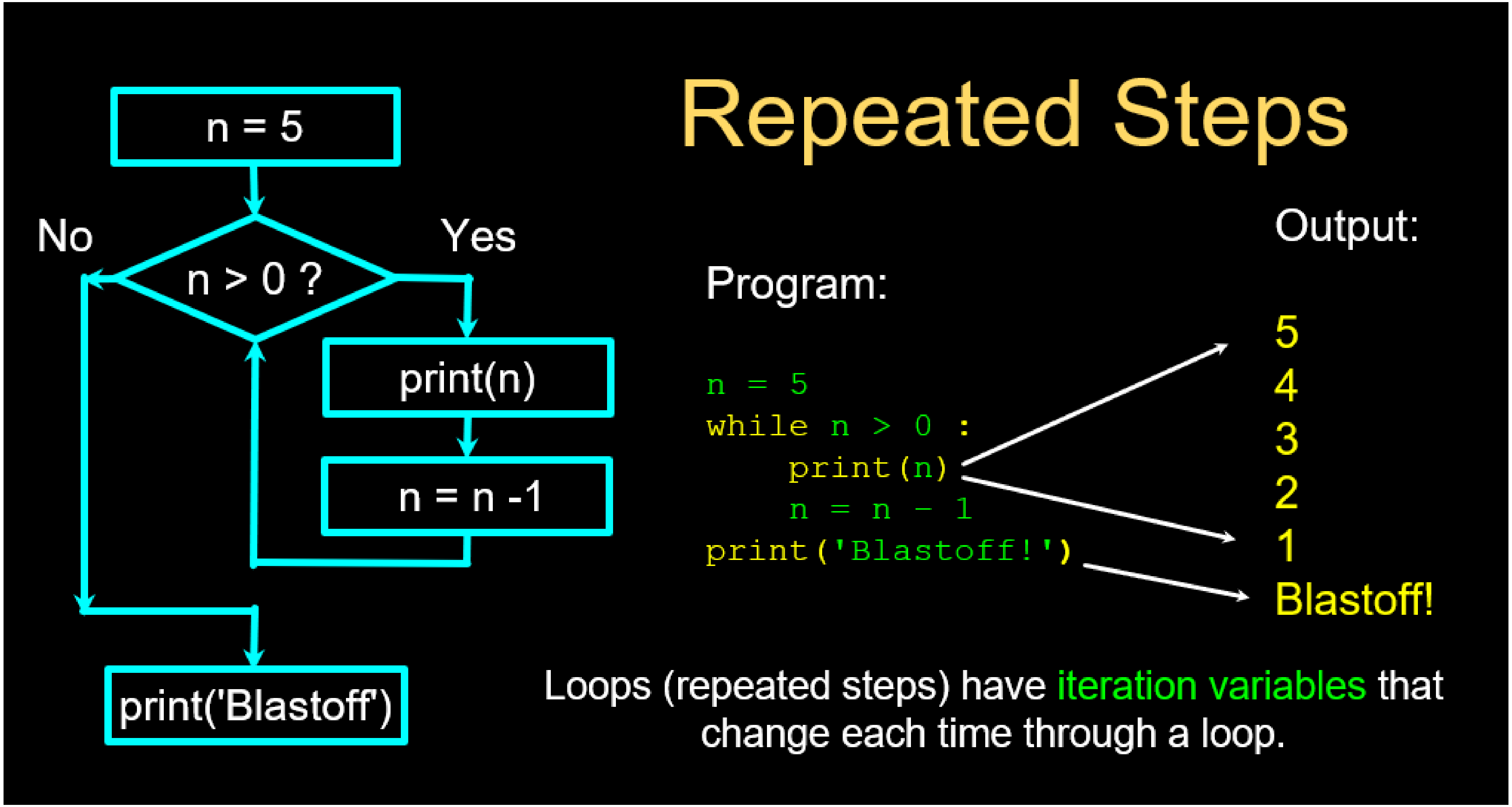
Conditional Steps

Program:

```
x = 5
if x < 10:
    print('Smaller')
if x > 20:
    print('Bigger')

print('Finis')
```

Output:

```
Smaller
Finis
```



Repeated Steps

Program:

```
n = 5
while n > 0 :
    print(n)
    n = n - 1
print('Blastoff!')
```

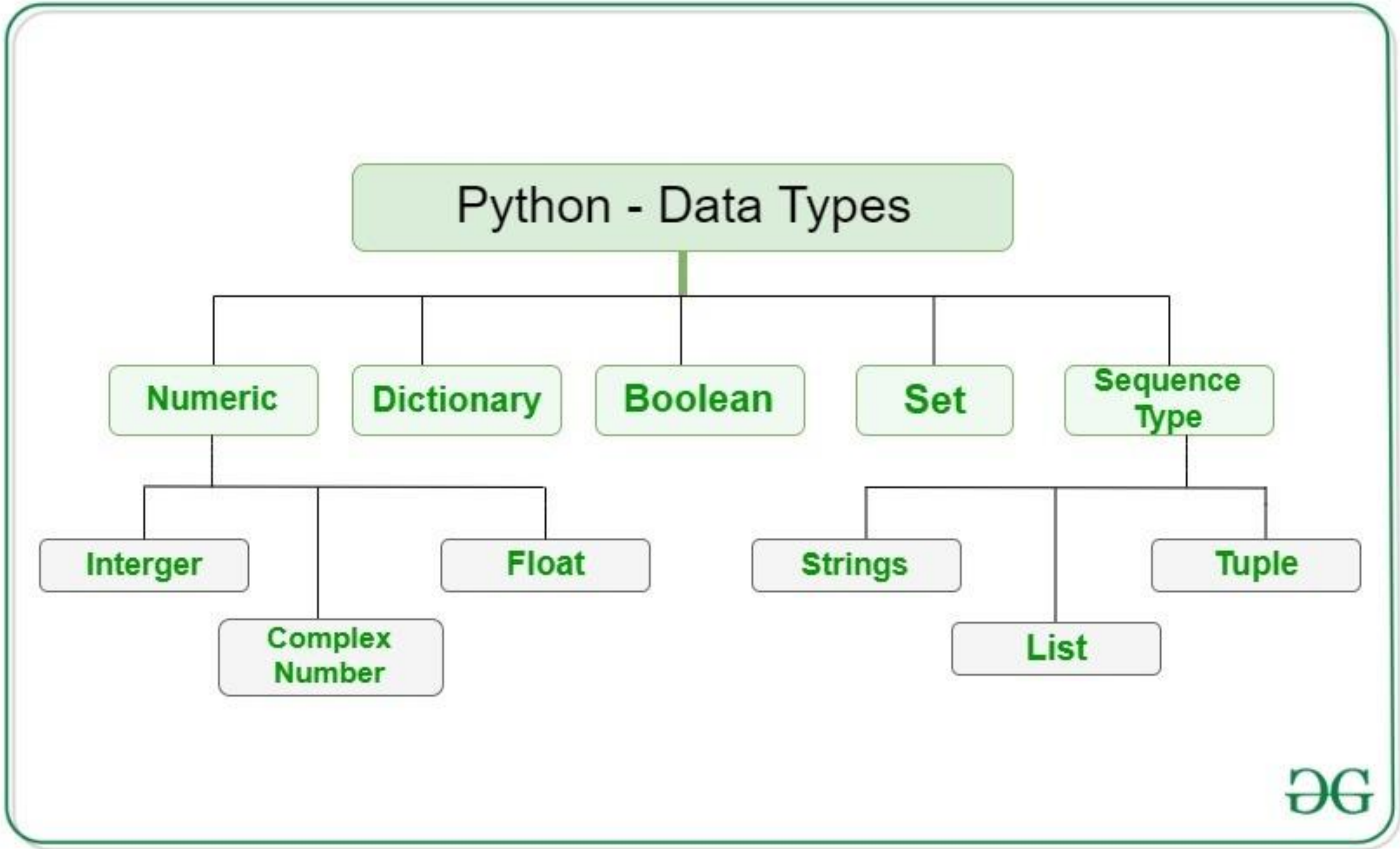
Output:

```
5
4
3
2
1
Blastoff!
```

Loops (repeated steps) have iteration variables that change each time through a loop.

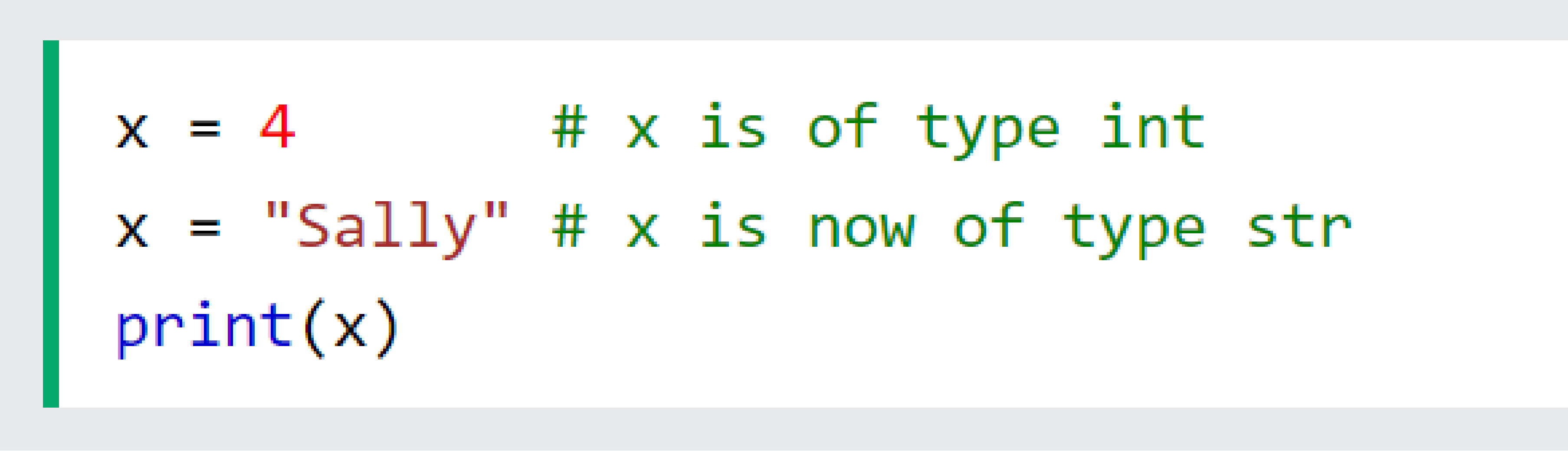
Part 2: Input/Output Operations

Data Types


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

Variables

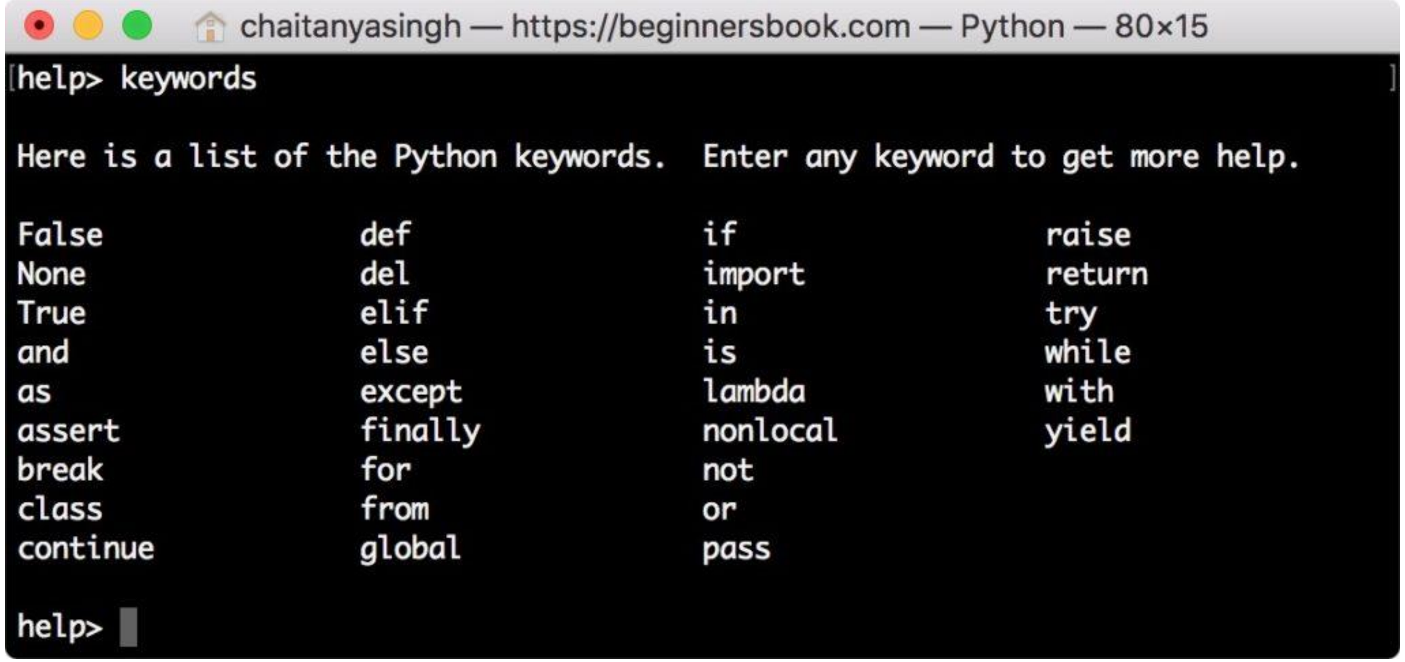
- Variables are containers for storing data values
- Creating variable:
- Python has **no command for declaring** a variable.
- A variable is created the moment you first assign a value to it.



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

Variables

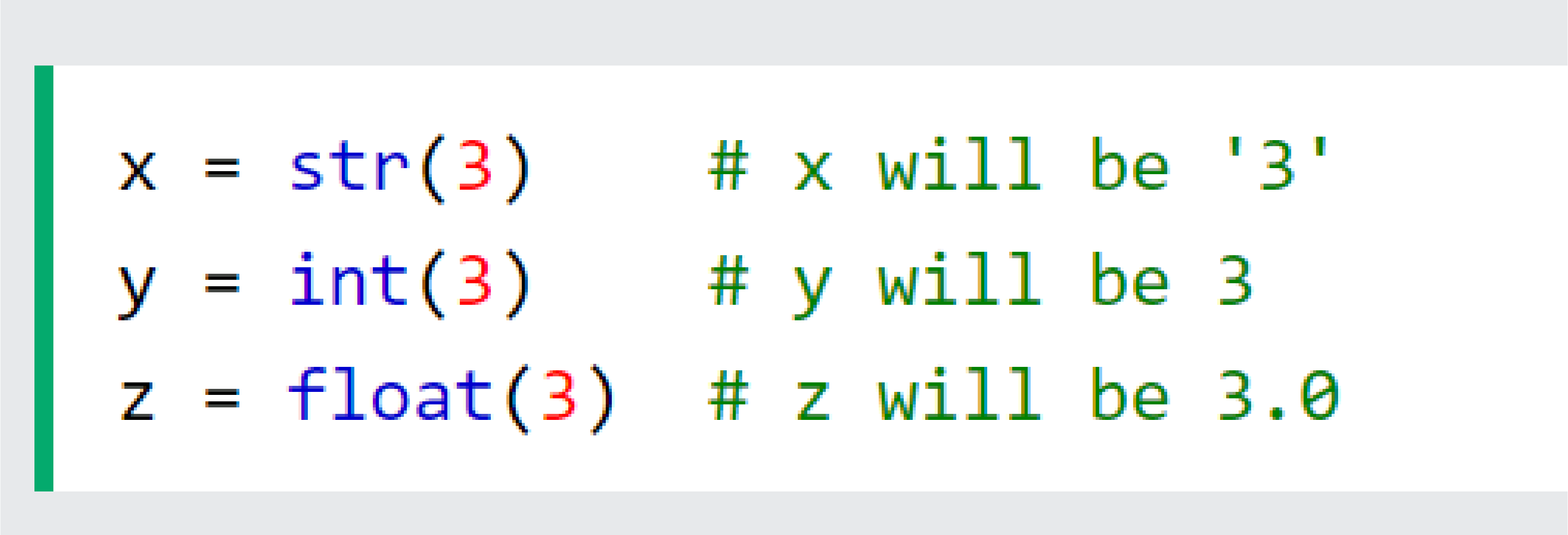
- Rules for Naming Python Variables:
- Python is **case-sensitive** (i.e: num ≠ Num)
- Contains Alphabet (upper/lowercase), Digits (0-9), underscore (_)
- First letter: Alphabet, underscore
- Avoid Keywords



```
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

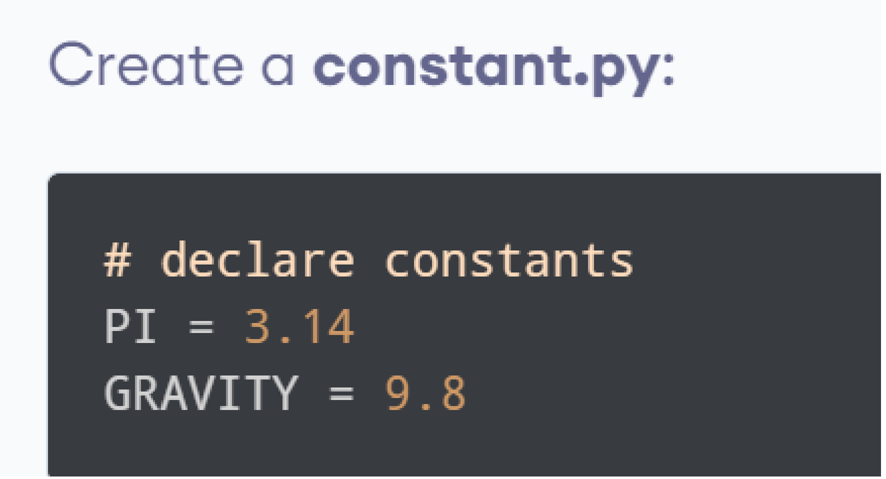
- If you want to specify the data type of a variable, this can be done with casting



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

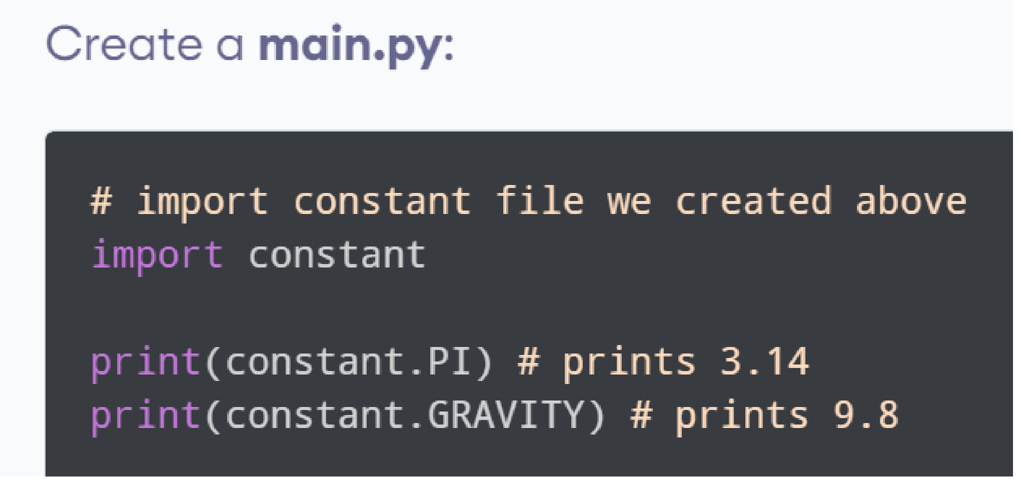
Constant

- 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
```



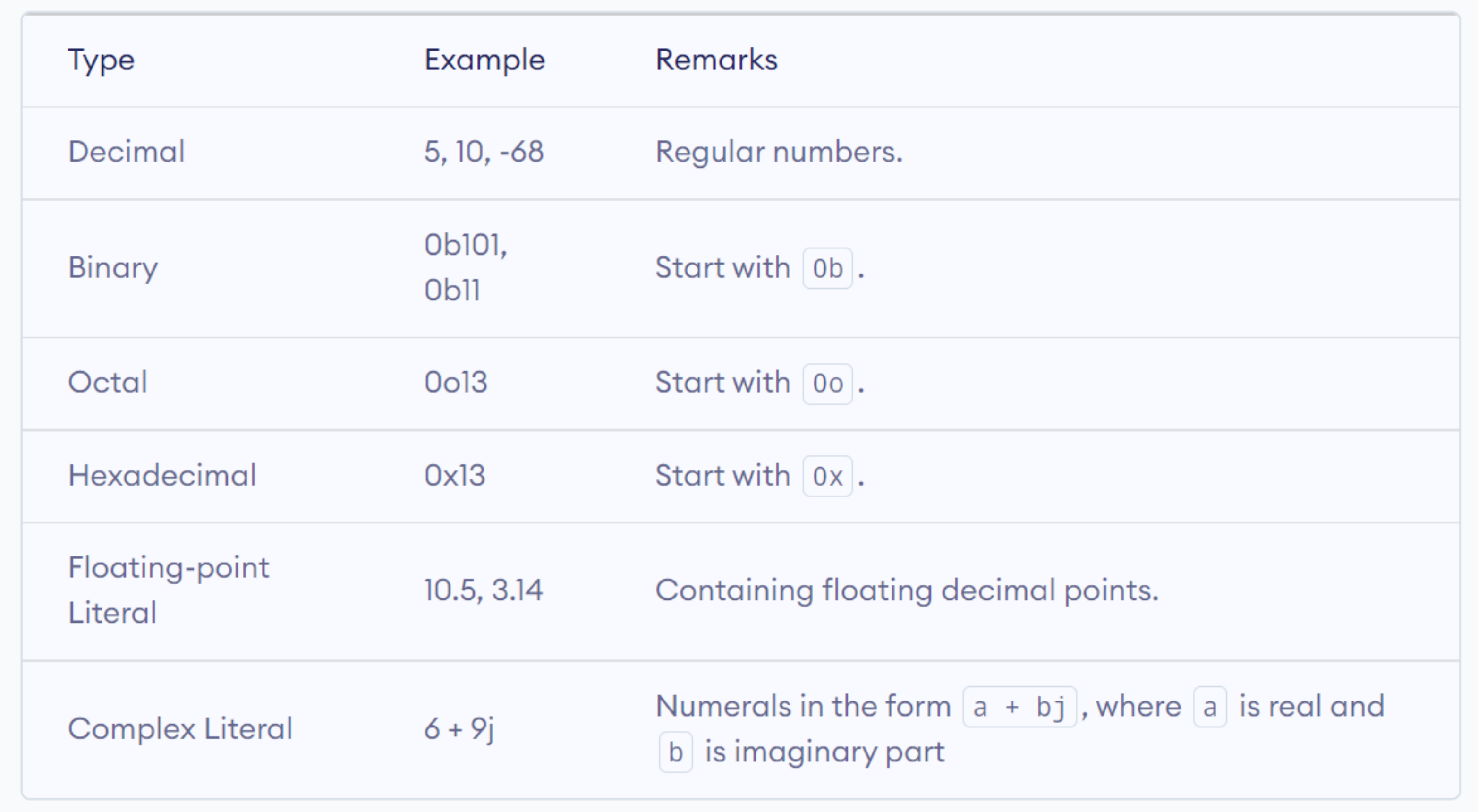
Create a **main.py:**

```
# 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

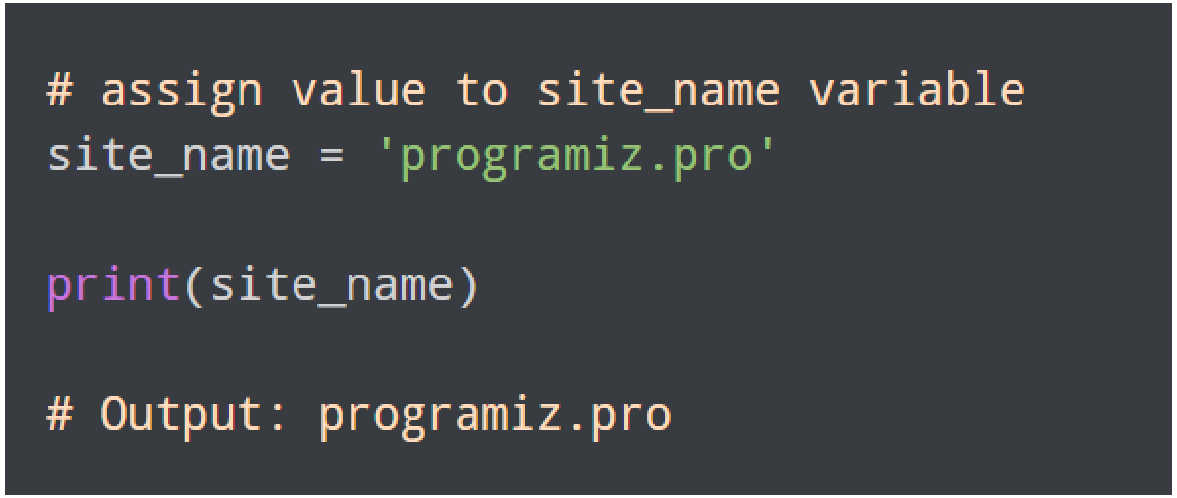
- Numeric Literals are immutable (unchangeable). Numeric literals can belong to 3 different numerical types: Integer, Float, and Complex.



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

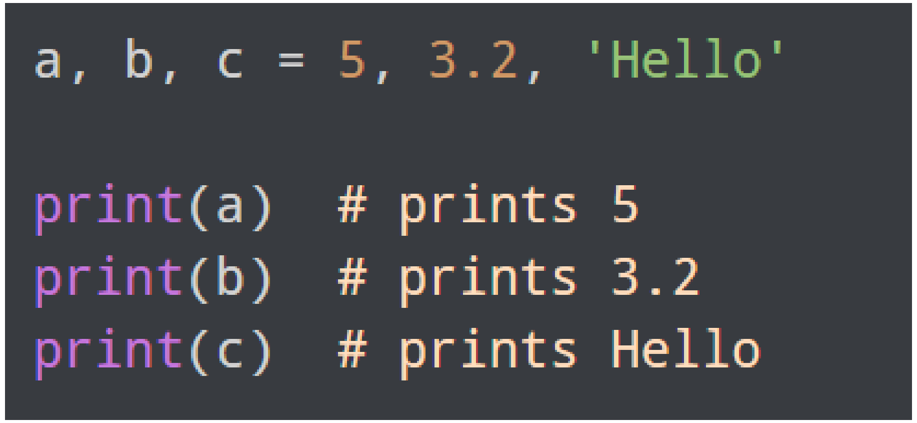
Assign Operator

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



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

print(site_name)

# Output: programiz.pro
```

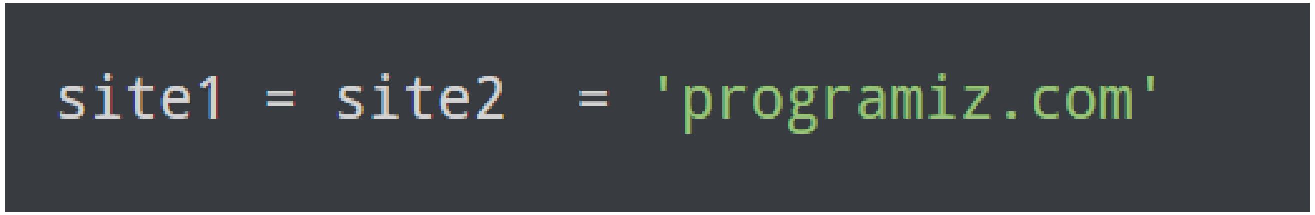
- Assigning multiple values to multiple variables



```
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 ?



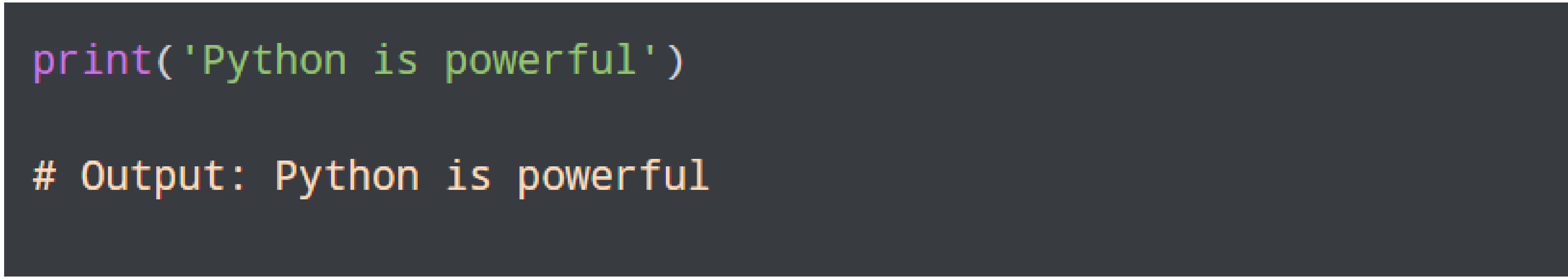
```
site1 = site2  = 'programiz.com'
```

- Separate it into the simple process?

Input/Output

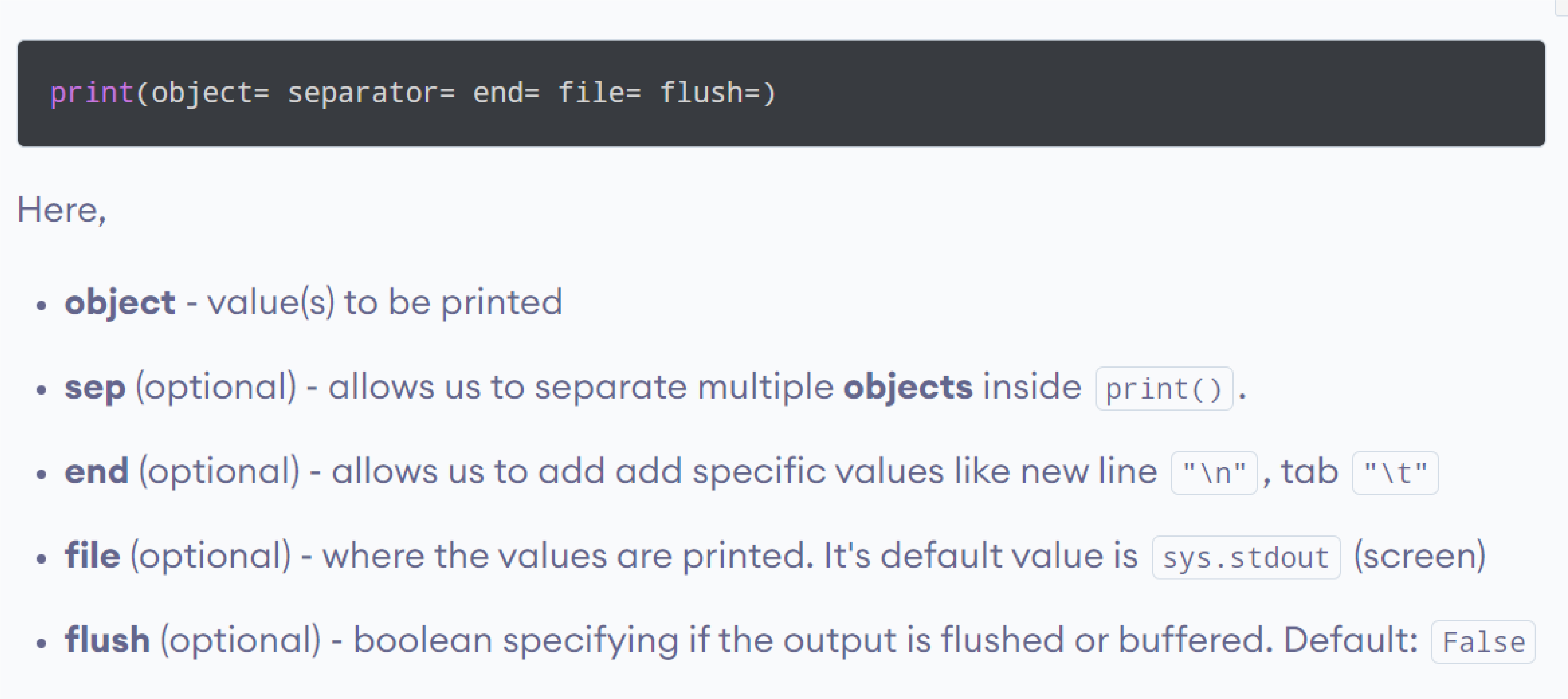
- Output: display the data to users (Console/File)

print(<value>):



```
print('Python is powerful')

# Output: Python is powerful
```

print()

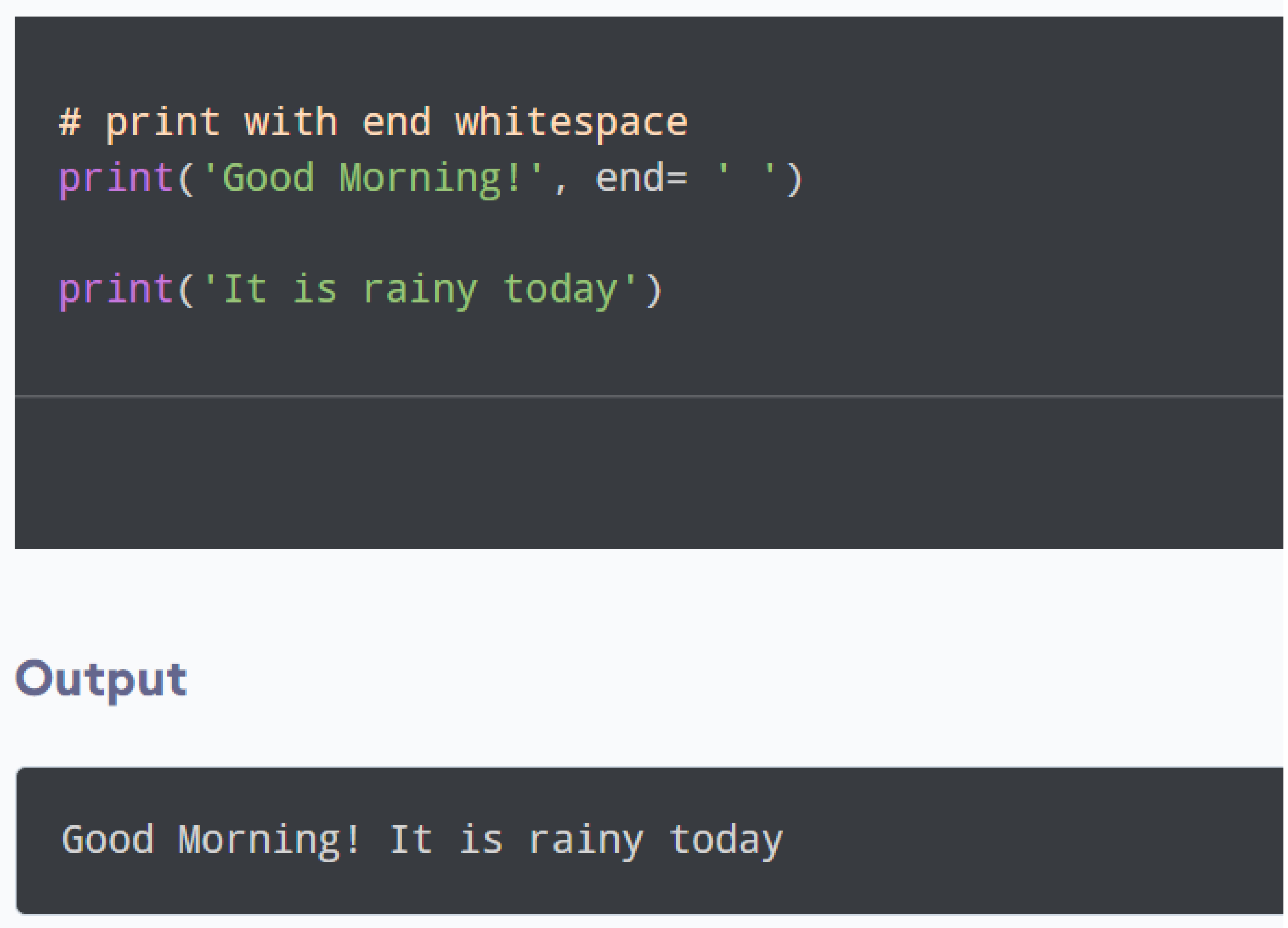


```
print(object= separator= end= file= flush=)
```

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()

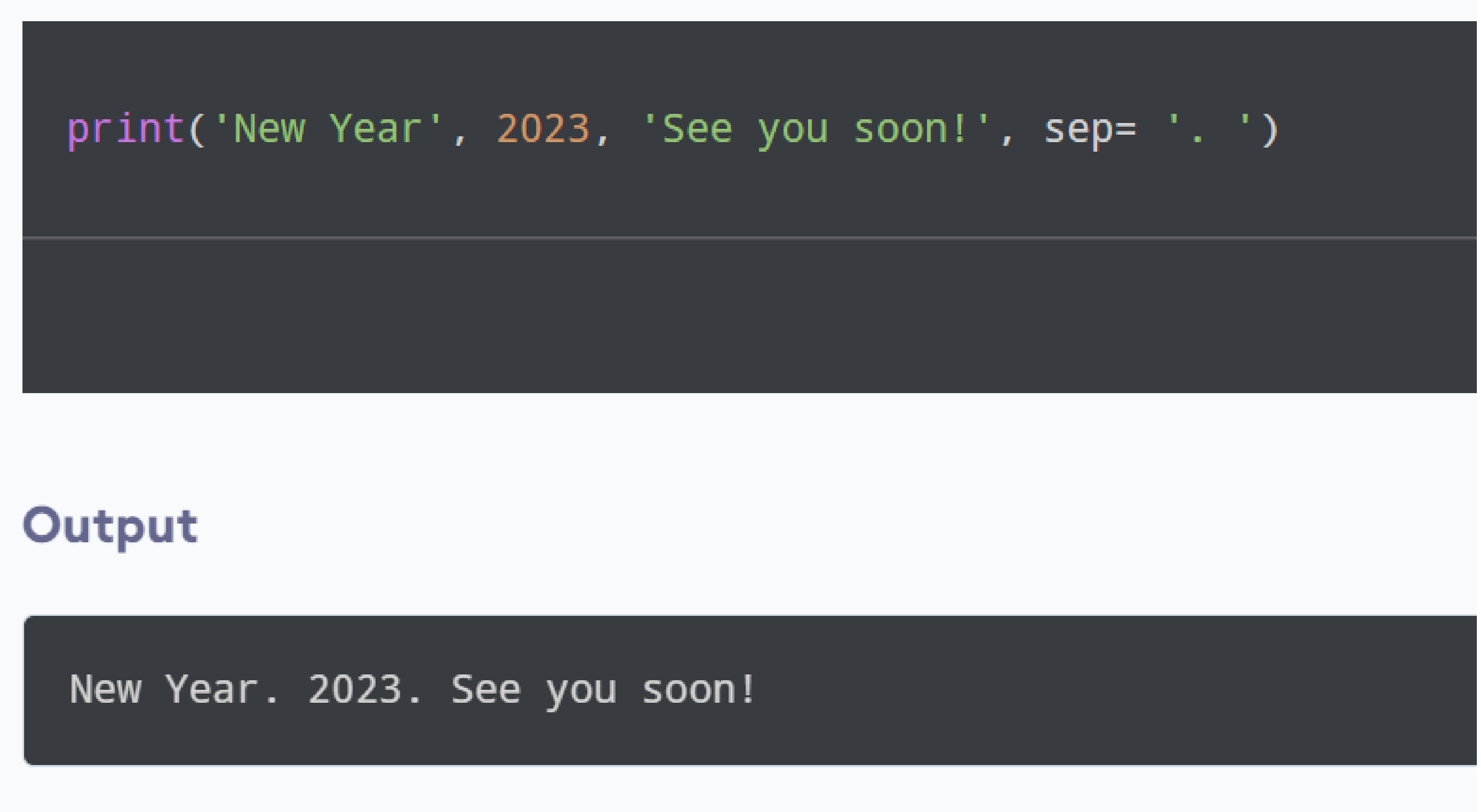


```
# print with end whitespace
print('Good Morning!', end= ' ')

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

Output

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

print()

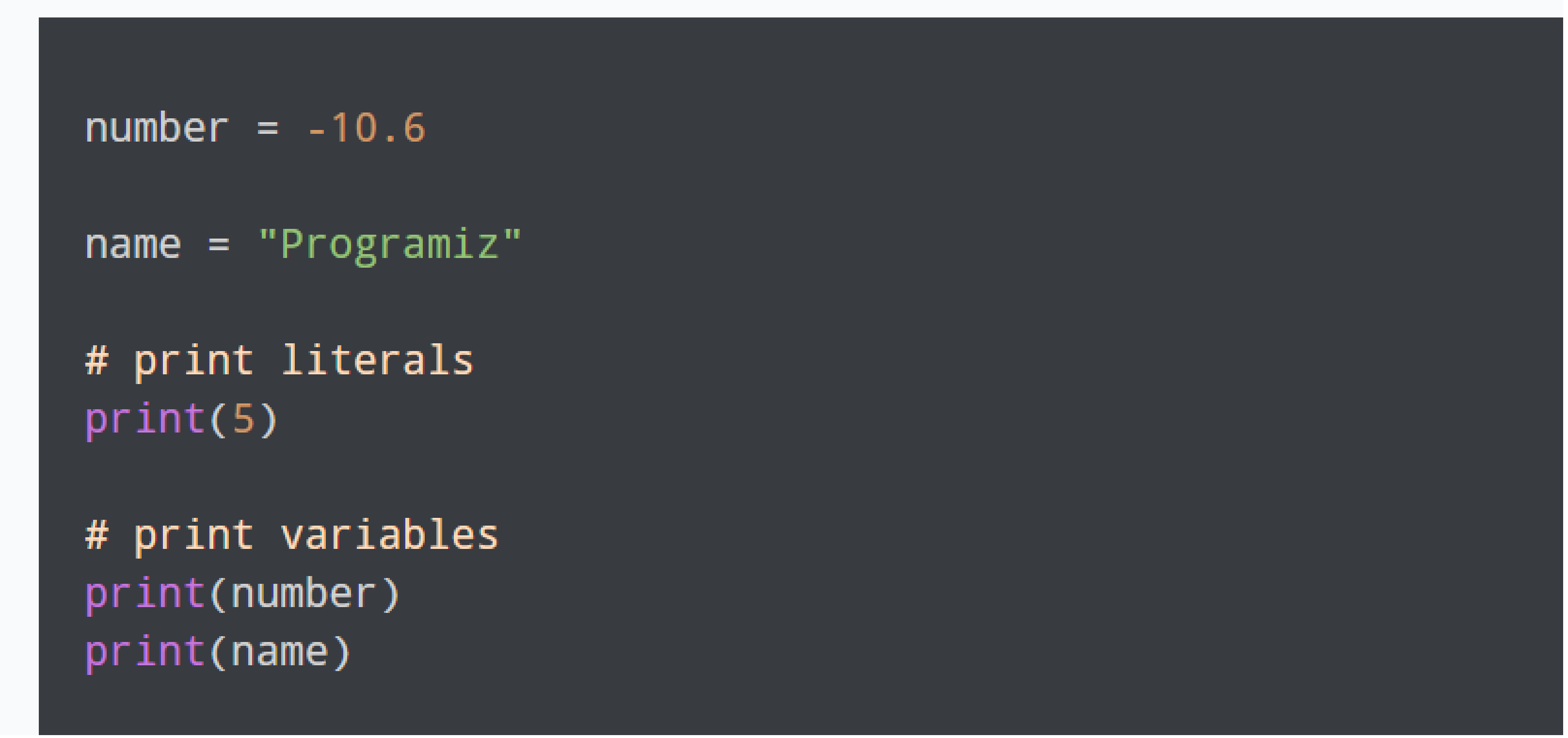


```
print('New Year', 2023, 'See you soon!', sep= '. ')
```

Output

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

print()

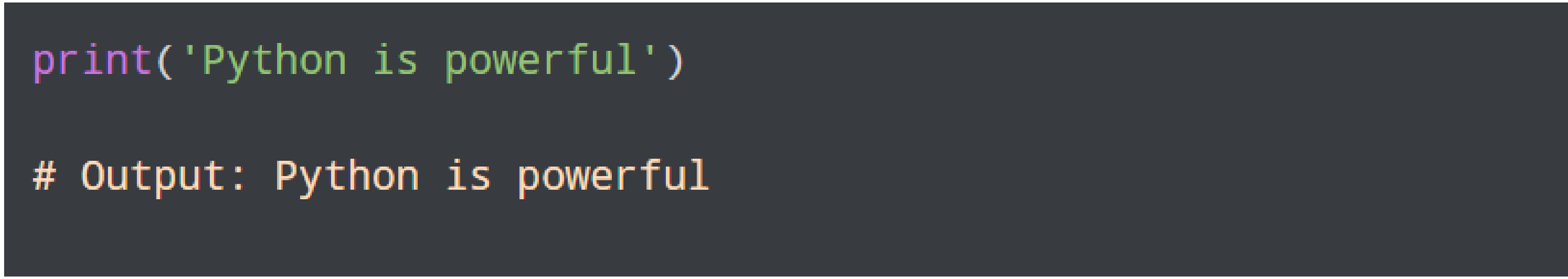


```python
number = -10.6

name = "Programiz"

# print literals
print(5)

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

Input/Output

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

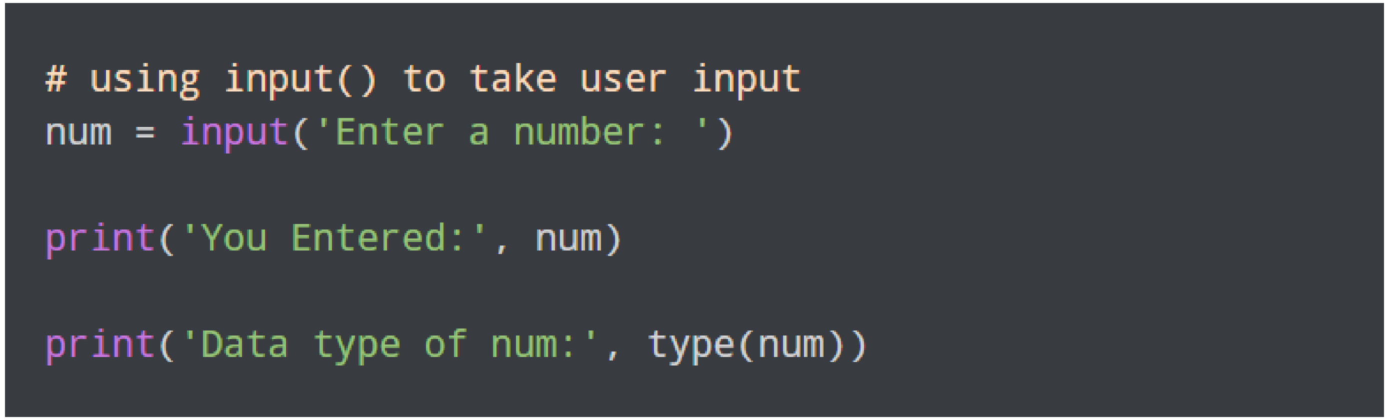
print(<value>):



```
print('Python is powerful')

# Output: Python is powerful
```

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

input([prompt])

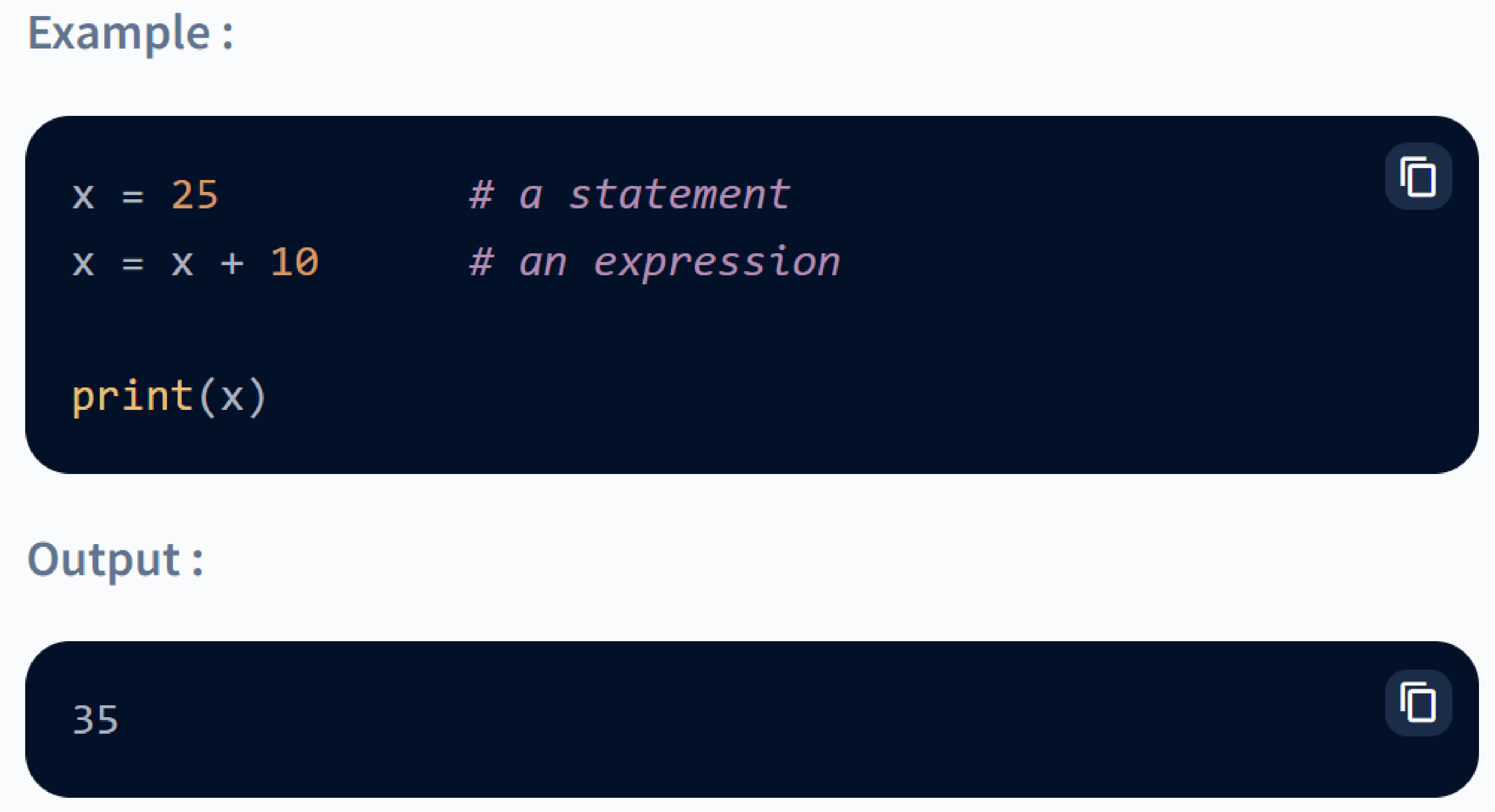


```
# using input() to take user input
num = input('Enter a number: ')

print('You Entered:', num)

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

- However, all input value is string

Expression

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



Example :

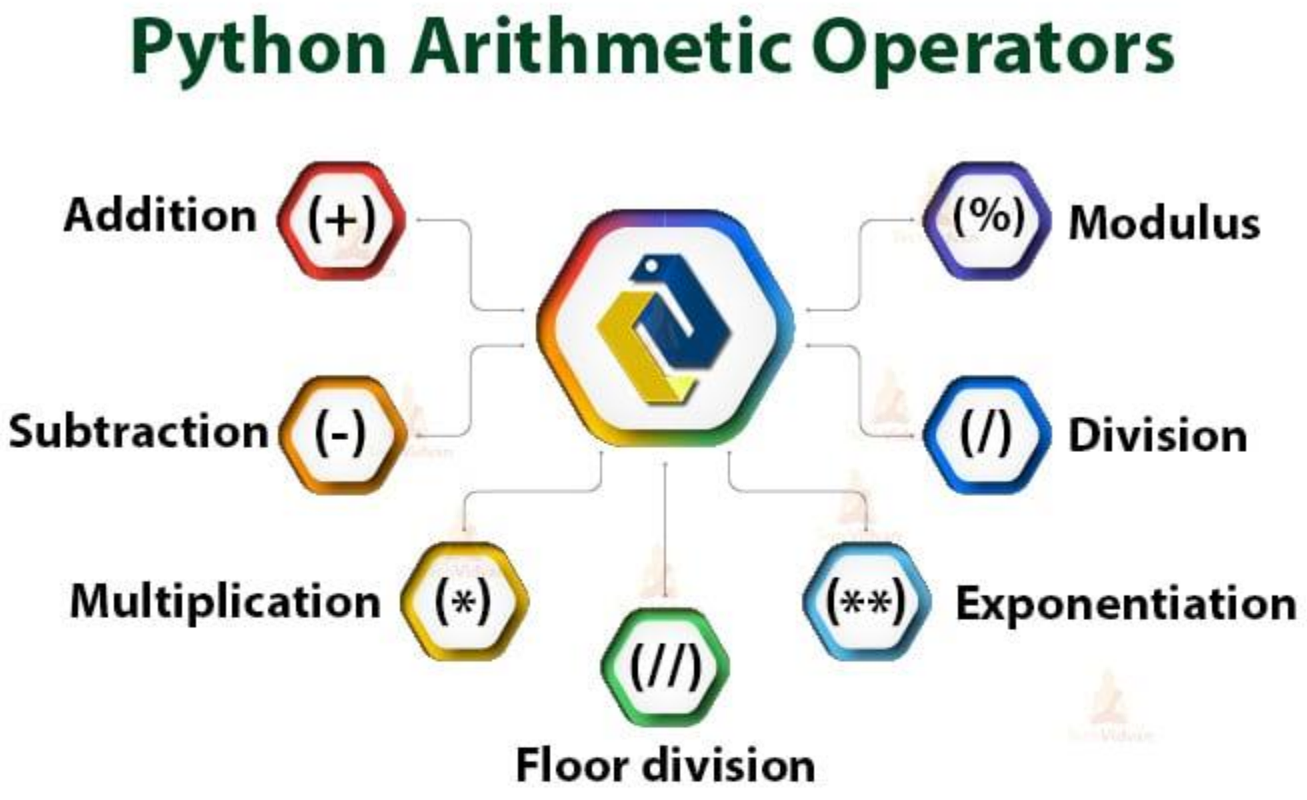
```
x = 25          # a statement
x = x + 10      # an expression

print(x)
```

Output :

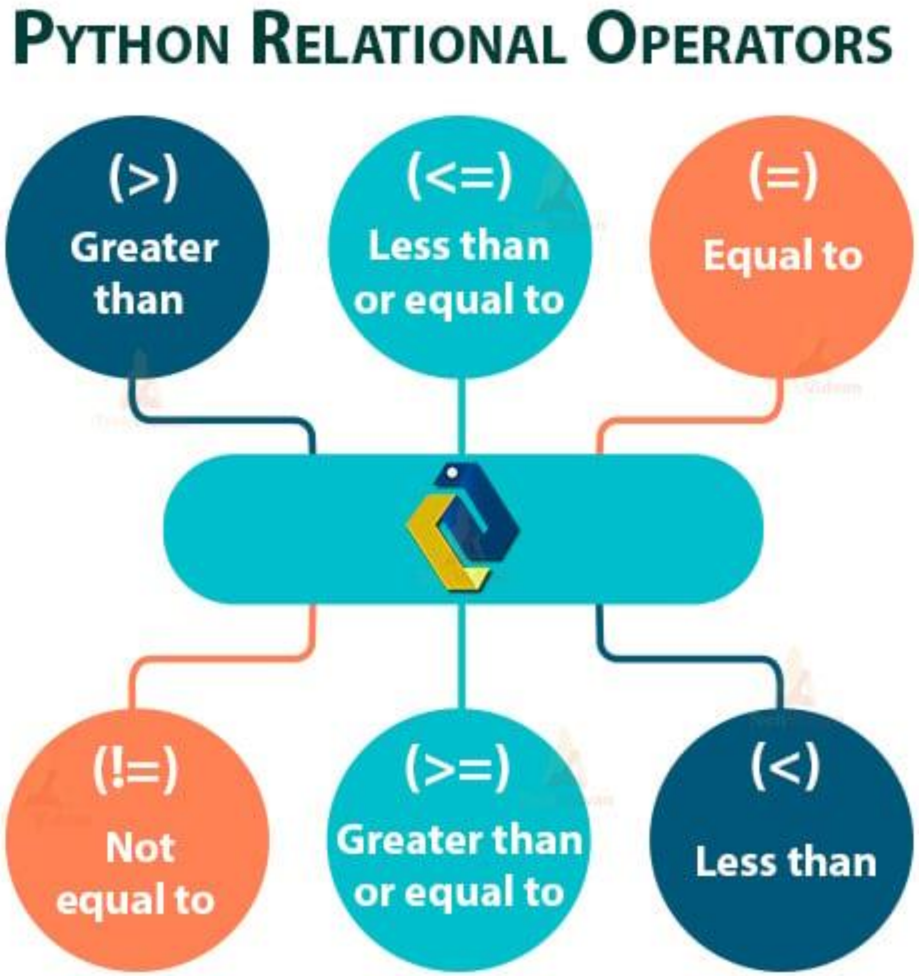
```
35
```

Operators

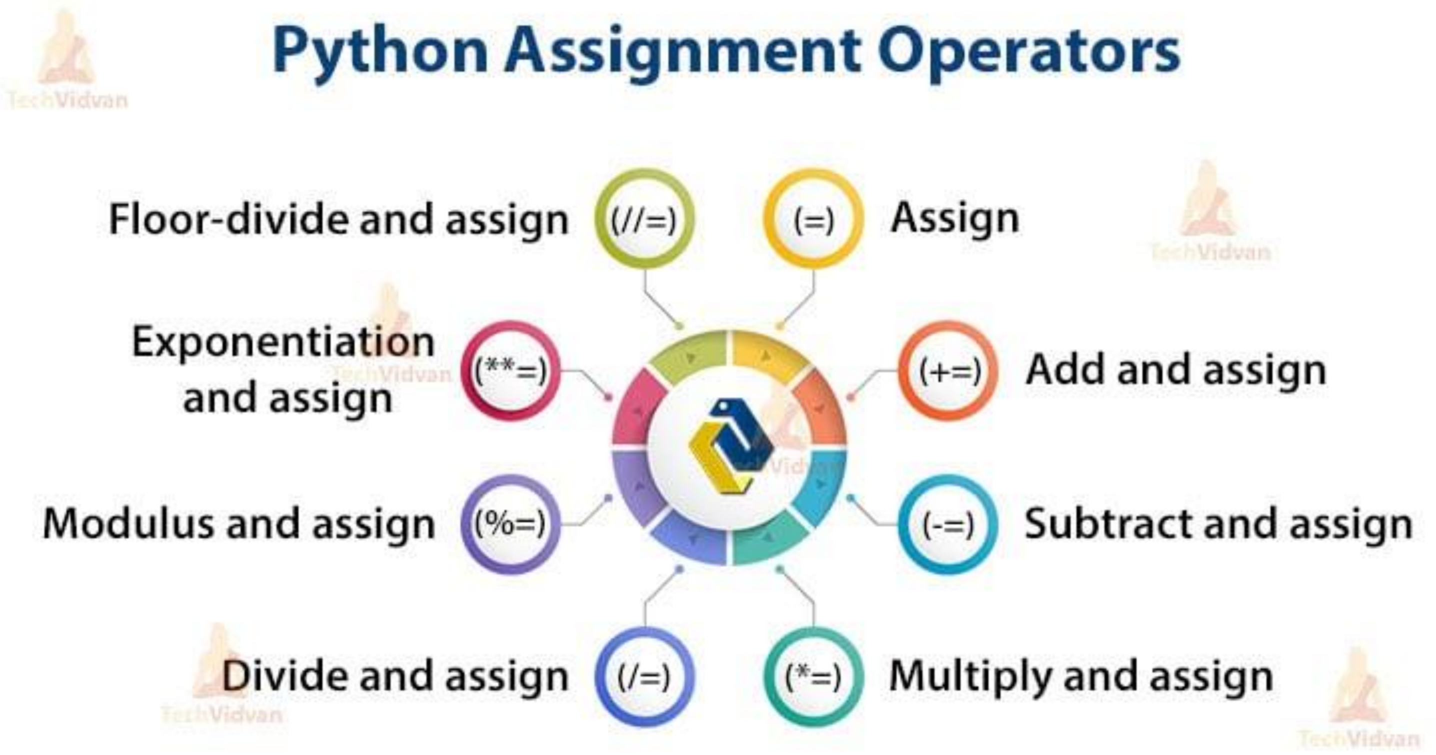
- **Arithmetic Operators**



- **Relational Operators**

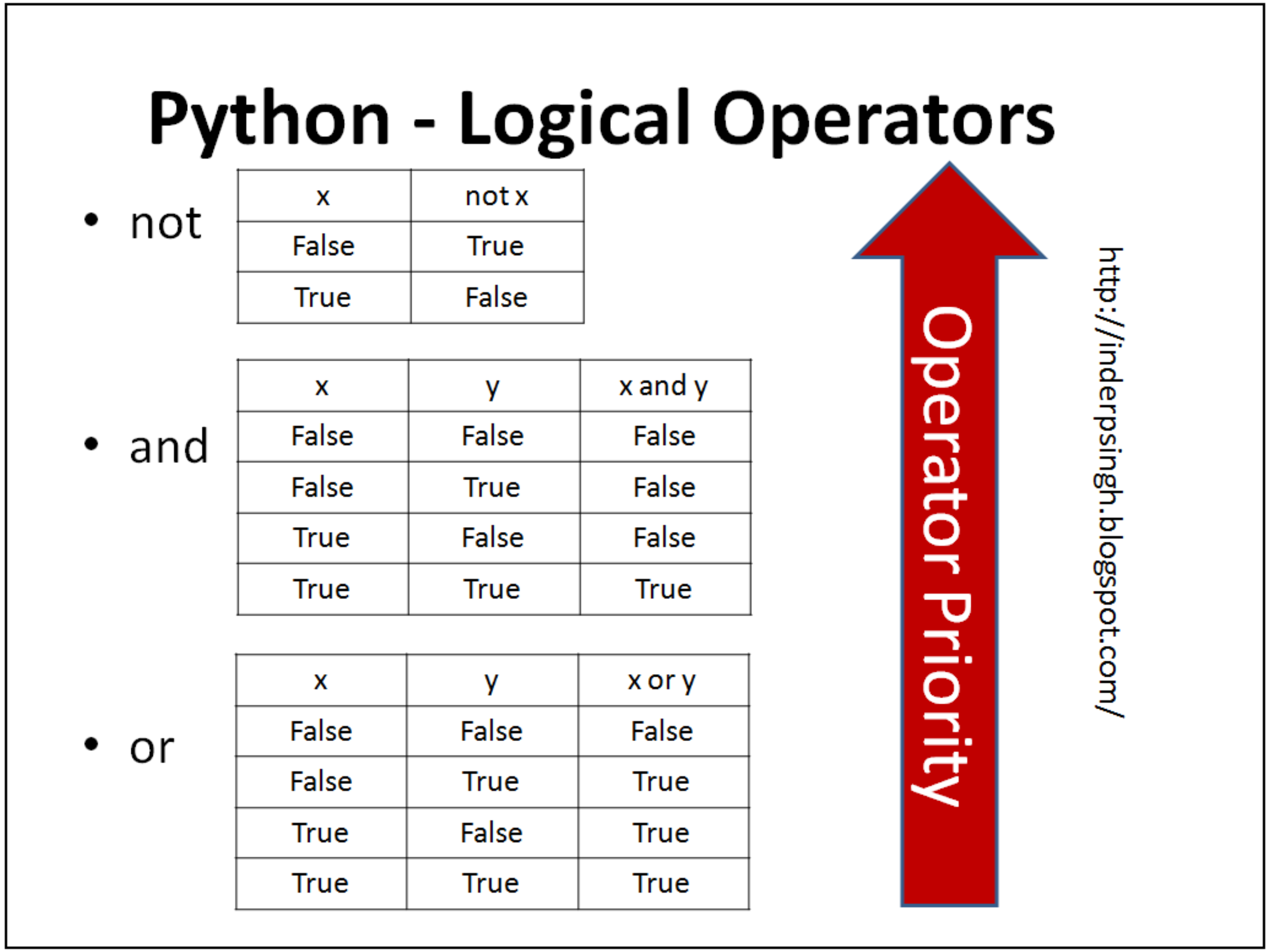


- **Assignment Operators**



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

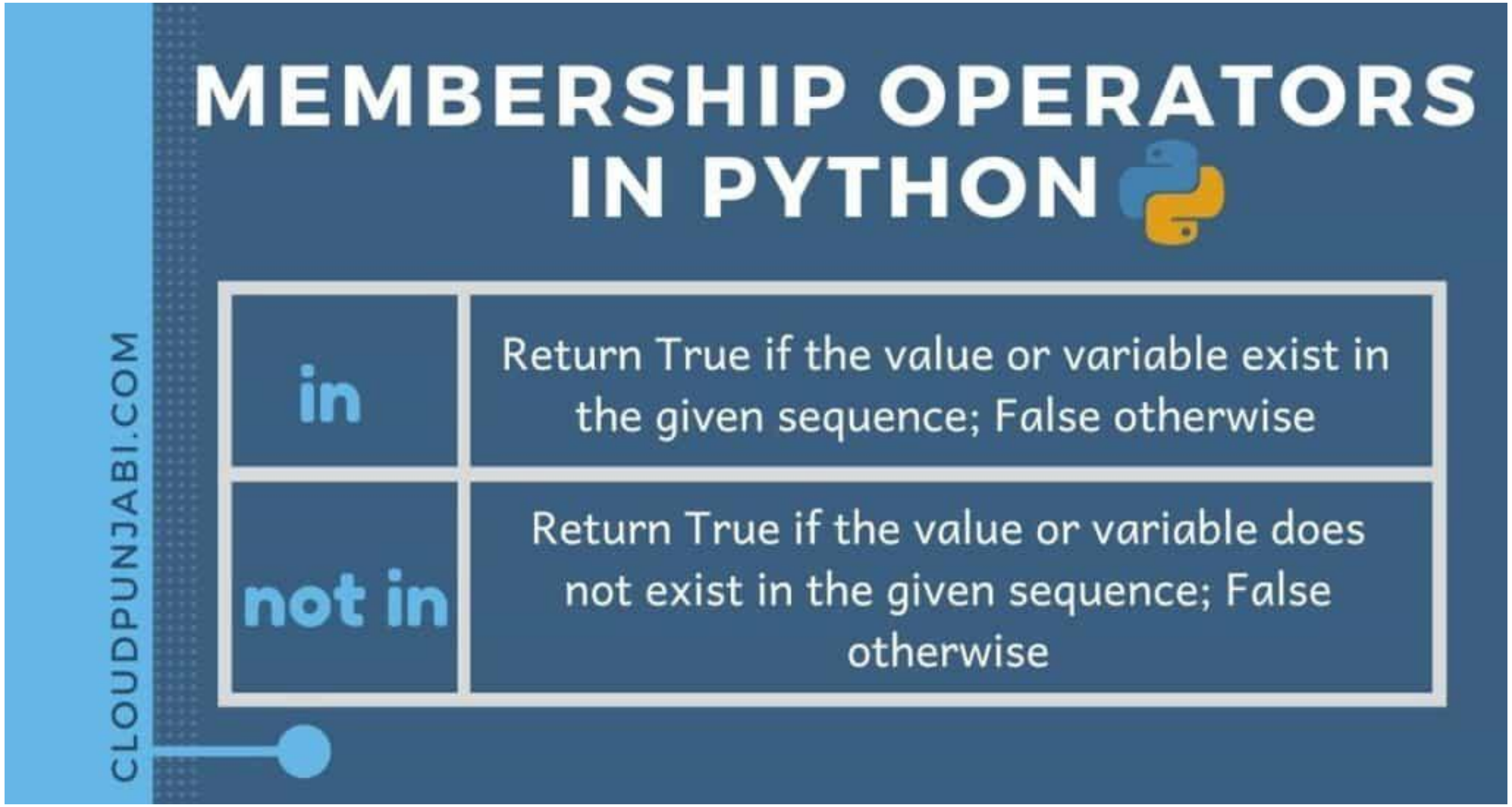
x	y	x or y
False	False	False
False	True	True
True	False	True
True	True	True

Operator Priority

http://inderpsingh.blogspot.com/

Operators

- **Membership Operators**



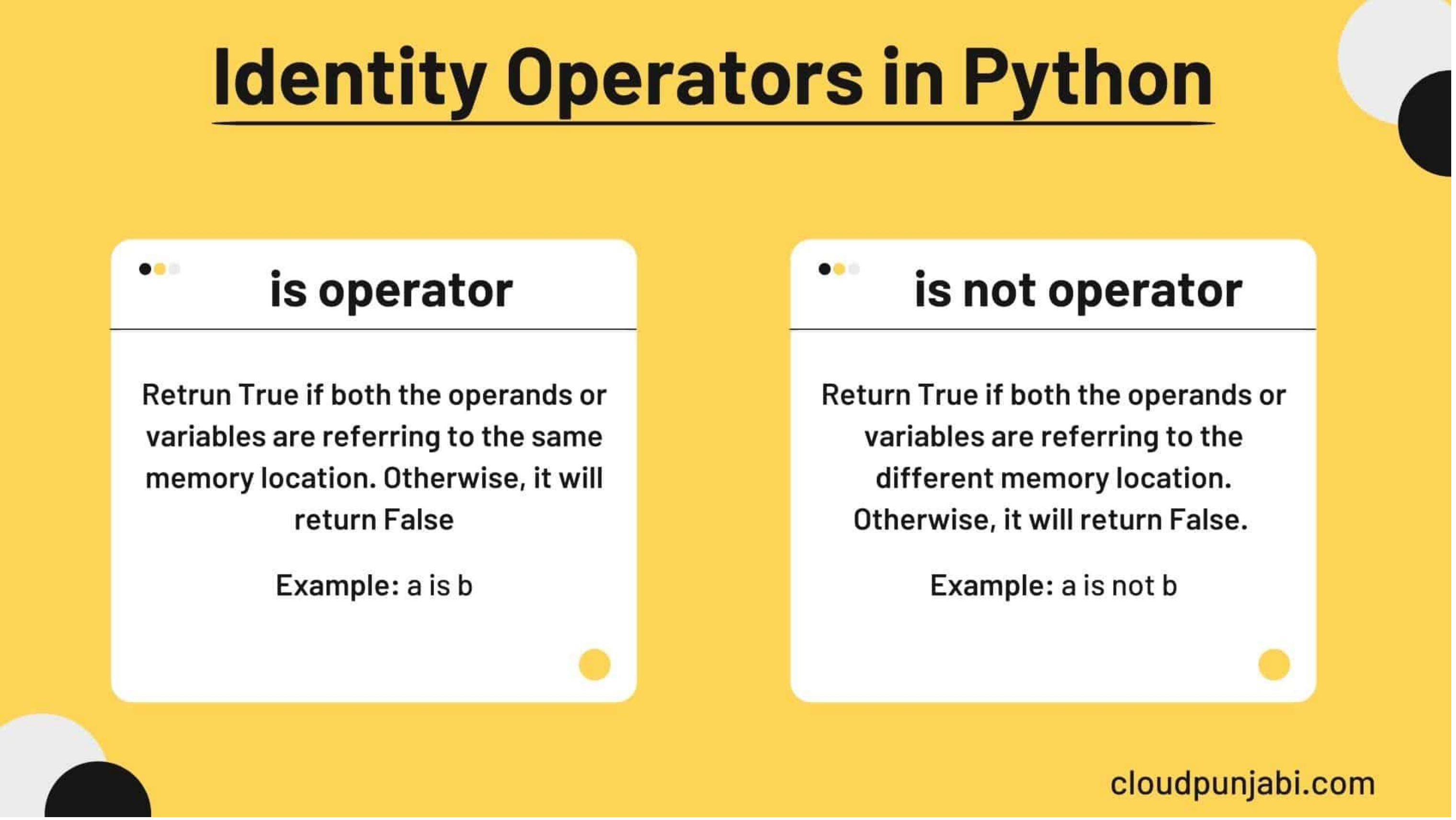
MEMBERSHIP OPERATORS IN PYTHON

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

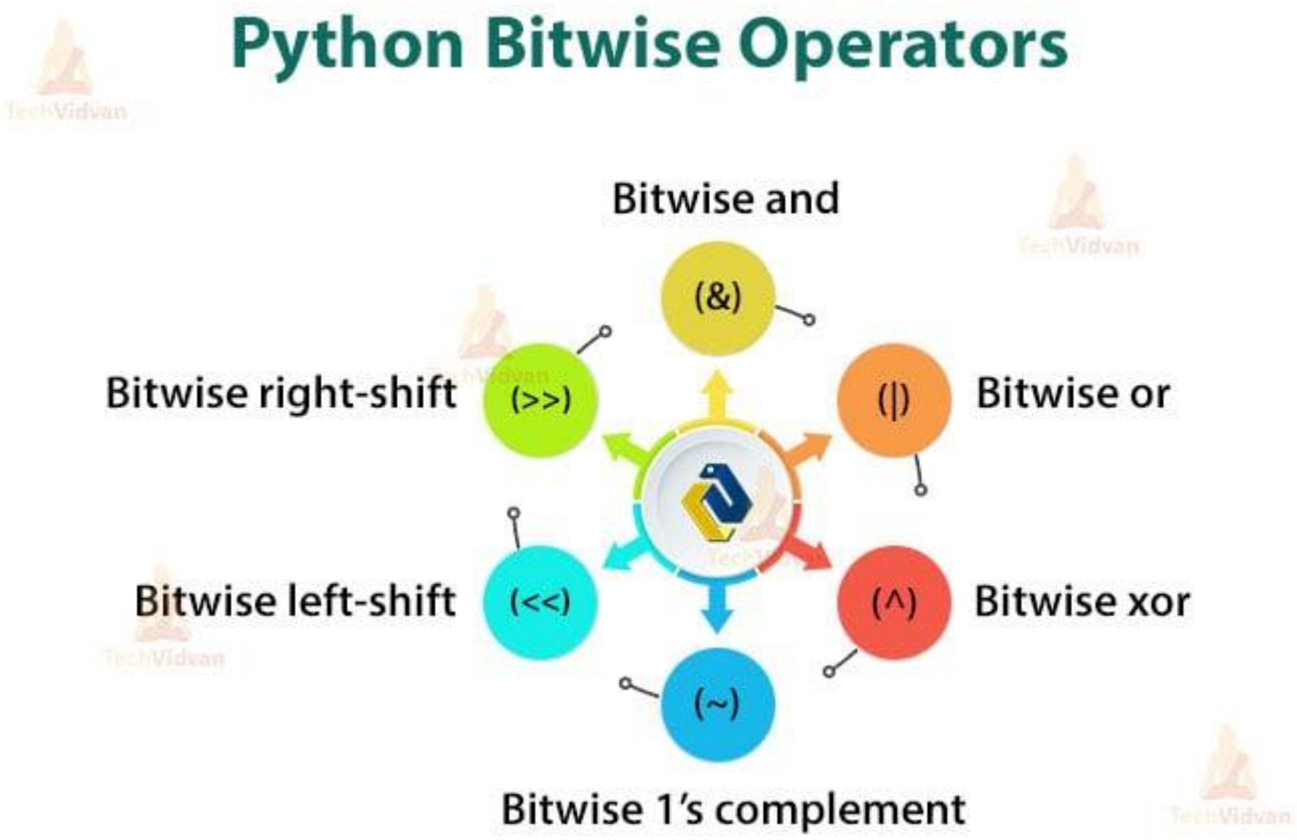
is not operator

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

- **Bitwise Operators**



Part 3: FlowChart & Problem Solving

Problem Solving

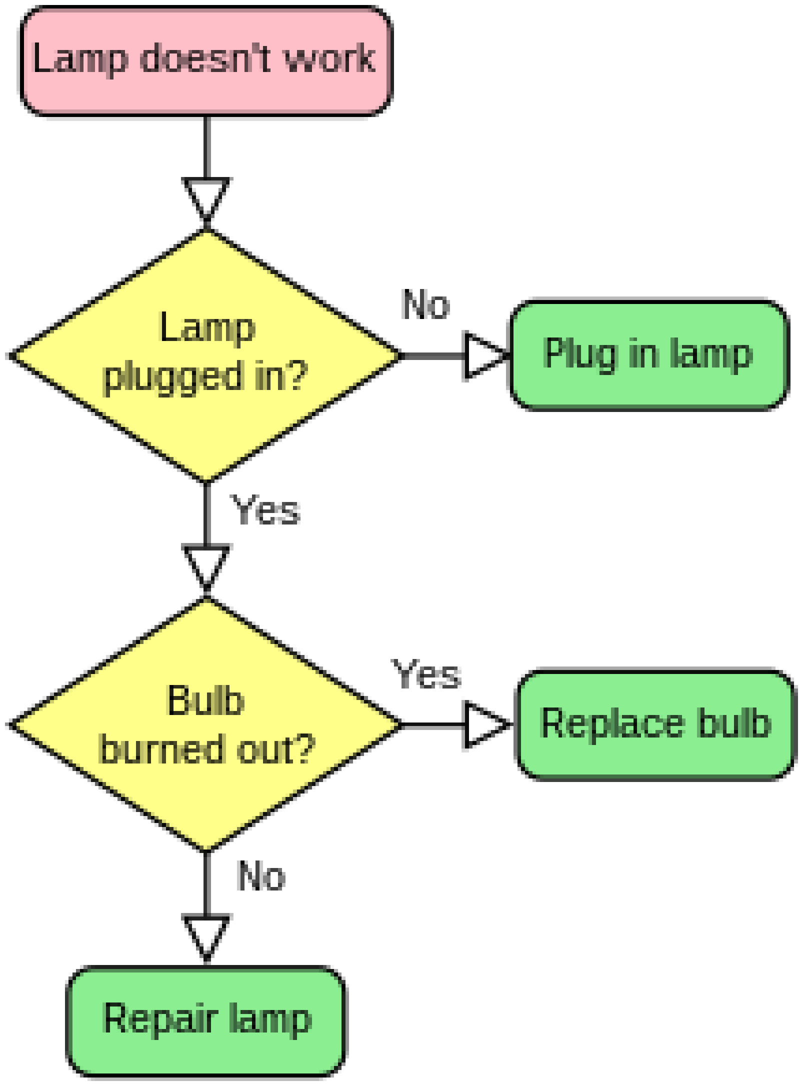
- For problem in programming, we need to identify:
- Input
- Output
- Algorithm: In Computer Science, an algorithm is a list set of instructions, used to solve problems or perform tasks, based on the understanding of available alternatives.

Problem Solving

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

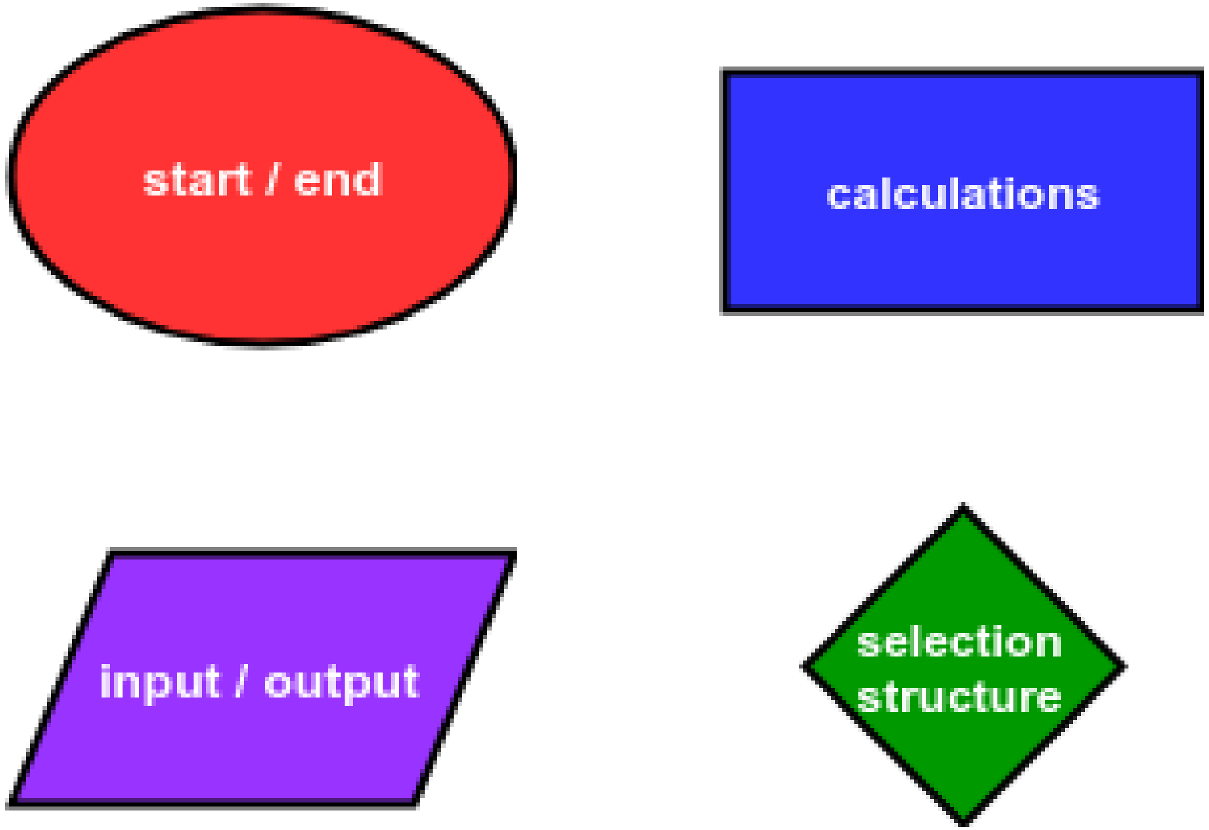
Flowchart

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



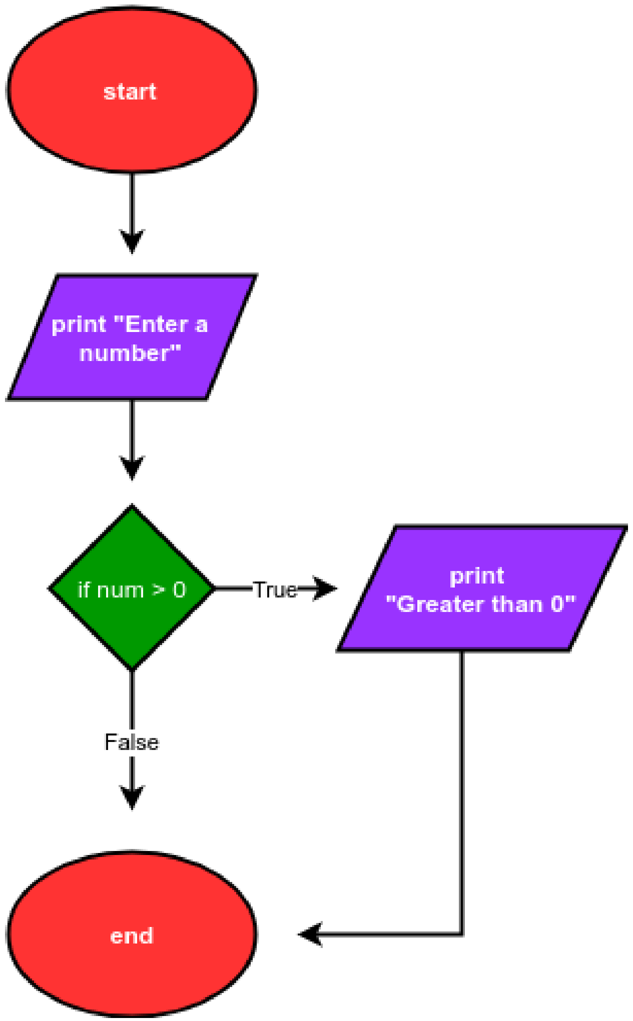
Flowchart

- 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

THANK YOU

for YOUR ATTENTION