### Programming languages

A programming language is a formal set of rules and syntax used to communicate instructions to a computer. It allows programmers to write code that the computer can interpret and execute to perform specific tasks or solve problems

# Python

 - is a high-level programming language

#### IDLE ( Integrated DeveLopment Environment )

### Python is an interpreted language

- a programm language that executes its statements line by line

# Immutable and mutable

\*\*Immutable\*\*: strings, int, float, bool, bytes, tuple (cannot change)

\*\*Mutable\*\*: list, dictionary (can change, remain the same object)

For example

```

For immutable, for a tuple

x = (1,2)

y = x

x = (4,5)

print (x, y)

--> (4,5) (1, 2)

For mutable for a list

x = [1,2]

y = x

x = [4,5]

print (x, y)

--> [4,5], [4,5]

```

# List comprehensions

```

x = [i for i in range(10)]

print(x)

--> [0,1,2,3,4,5,6,7,8,9]

```

# Branching

Branching refers to the use of \*\*conditional statements\*\* that allow the program to choose between different paths or actions based on certain conditions

## break Statement

Exits a loop and skips the rest of the block

---

# Strings

- immutable

- operations : concatenation, search

```

name = "Zara"

s = "hello"

conc1 = s + name

conc2 = s + " " + name

print(conc1) --> helloZara

print(conc2) --> hello Zara

```

- last element

```

s = "HELLO CMF MATE"

print(s[-1]) --> E

print(s[-2]) --> T

```

- \*\*Slicing and search\*\*

```

s = "hello there"

s[0] --> ’h’

s[2:5] --> ’llo’

s[2:] --> ’llo there’

s[:2] --> ’he’

s[-1] --> ’e’

s.find("l") --> 2

s.rfind("l") --> 3

```

---

# Tuples

- immutable

---

# Lists

- mutable ( elements of a list can be modified )

- Operations:

• Create (manually, using range)

• Access (index, len) and modify elements

• Remove (pop) and insert (insert) elements

• Slicing and encapsulating

• Using as stacks (append, pop)

#### !! BY doing this : \*\*second = first[:]\*\* we create a new list. The lists are independent

Example

```

I

first = ["red", "yellow", "blue"]

second = first[:] # Creates a copy of the list

second.append("green") # Adds "green" to the second list

print(first) # ['red', 'yellow', 'blue']

print(second) # ['red', 'yellow', 'blue', 'green']

II

first = ["red", "yellow", "blue"]

second = first

second.append("green")

print(first)

print(second)

```

the first snippet, `second = first[:]` creates a \*\*shallow copy\*\* of the list, so modifying `second` does not affect `first`. In the second snippet, `second = first` makes `second` a reference to the same list as `first`, so modifying `second` also modifies `first`.

---

# Dictionaries

sequence of unordered pairs of data (key, value) – with unique keys

- mutable

- Operations:

• Creation

• Access the value for given key

• Add/modify/delete a given pair (key, value)

• Verify if a key exists

- Values:

• Can be duplicates

• Any type

- Keys:

• Must be unique

• Immutable type

```

#empty dictionary

d = {}

#create a dictionary

a = {'num': 1, 'denom': 2}

print(a)

#get a value for a key

a['num'] --> 1

```

---

\*Procedural programming\* – each program is formed by several procedures (functions)

# Function

- A block of statements that can be reused

- Are run in a program only when they are called

```

def complicated\_function(x, y): # parameters

print(x,y)

pass

complicated\_function(1, 2) # arguments

```

## Variable scope

PAG(31)

---

# Programming paradigms

### Procedural programming

is a style of programming where you write a list of instructions (called functions or procedures) to perform tasks in a specific order. It focuses on how to do things step by step

### OOP (Object oriented programming)

---

# TEST DRIVEN DEVELOPMENT (TDD)

- create tests before writing the code of the functions

---

# Modular programming

A design method in software development that breaks a program into smaller, manageable modules. Each module encapsulates specific functionality and can communicate with others

\*\*Module\*\* - a file that contains Python statements and definitions