

Programming Foundations in Python

Prepared by Dr. Hamada I. AbdulWakel

Lecture 1



Introduction to Computer Programming and Python

Lecture 1

Prepared by
Dr. Hamada I. AbdulWakel

Lecture

01

LECTURE MAIN POINTS

- Computer science and information processing relationship.
- The meaning of algorithm and its basic features.
- The hardware and software collaboration in a computer's architecture.
- Reasons of using Python.
- Python's uses.

Lecture

01

LECTURE MAIN POINTS

Cont'd

- Installation environment and the required Integrated Development Environment (IDE).
- Writing a simple Python program.
- Understanding Python code style.

Lecture

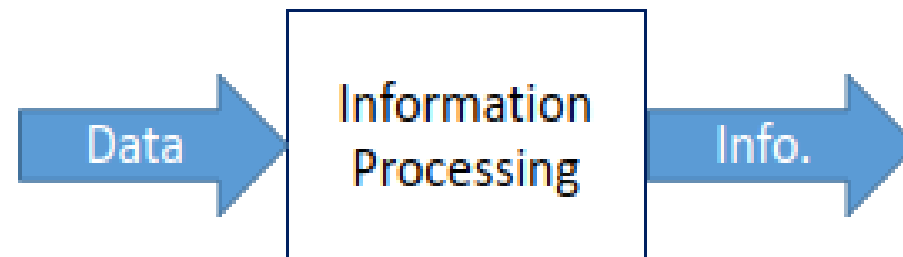
01

COMPUTER SCIENCE

- **Computer Science (CS):** the study of computation and information.
 - Solving problems is the core of CS.
- **Problem-solving:** the process of transforming the description of a **problem** into a **solution** using the problem-solving tools, strategies, and techniques.

INFORMATION PROCESSING

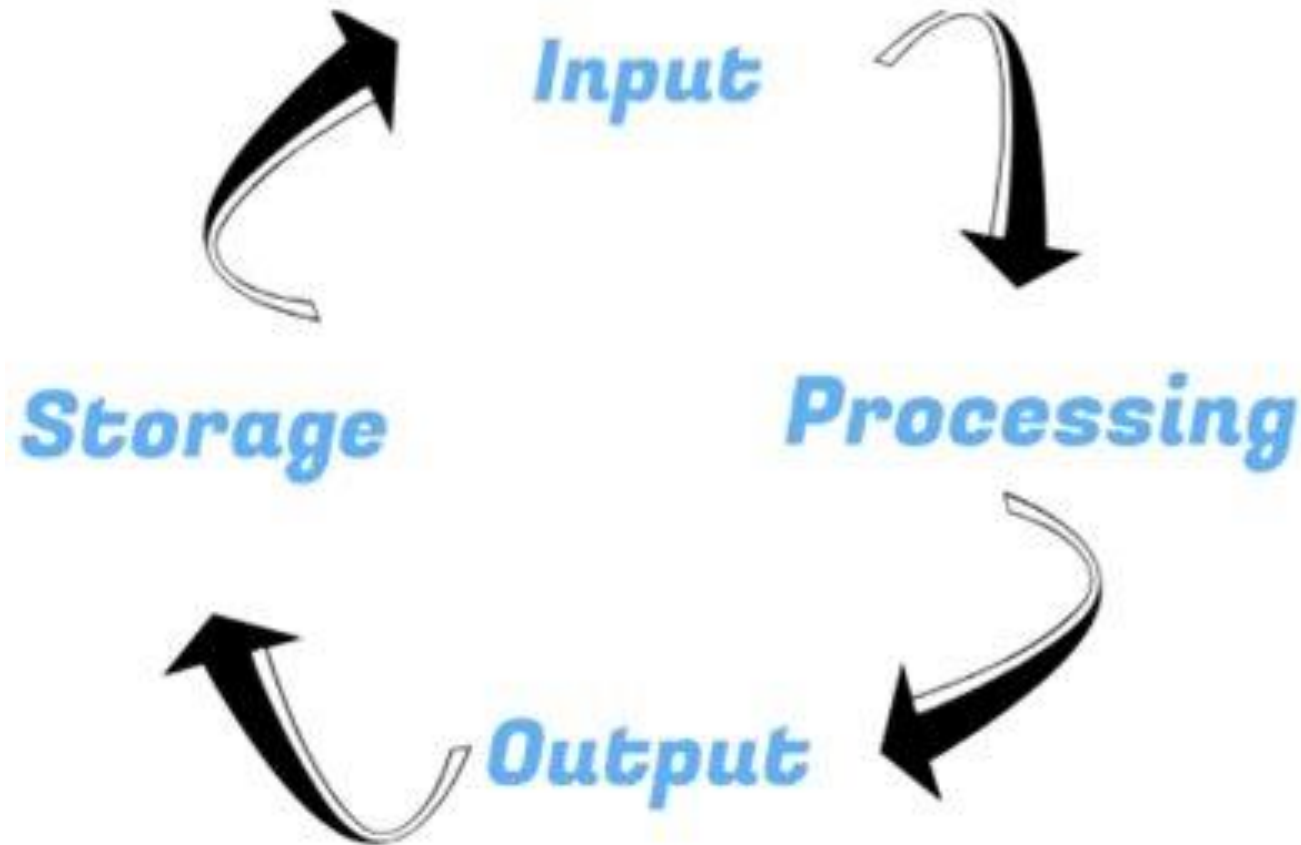
- **Data:** the raw, unorganized facts that can be in the form of text, graphics, audio, or video.
- **Information:** the processed data with a meaningful form.
- **Information processing:** the process of converting raw data into information.



Lecture

01

INFORMATION PROCESSING CYCLE



INFORMATION PROCESSING CYCLE

Cont'd

- **Input:** the stage where the computer is fed with the data from its user by the use of the input devices.
- **Processing:** after the unorganized facts (data) have been sent (input) into the computer's system, the computer processes and organize them to make them meaningful.

INFORMATION PROCESSING CYCLE

Cont'd

- **Output:** the step where the computer brings out information in the form understood to its users and displays it using the output devices.
- **Storage:** the organized data (information) is kept for future use. It can be stored on the internal hard disk or external on storage media.

Lecture

01

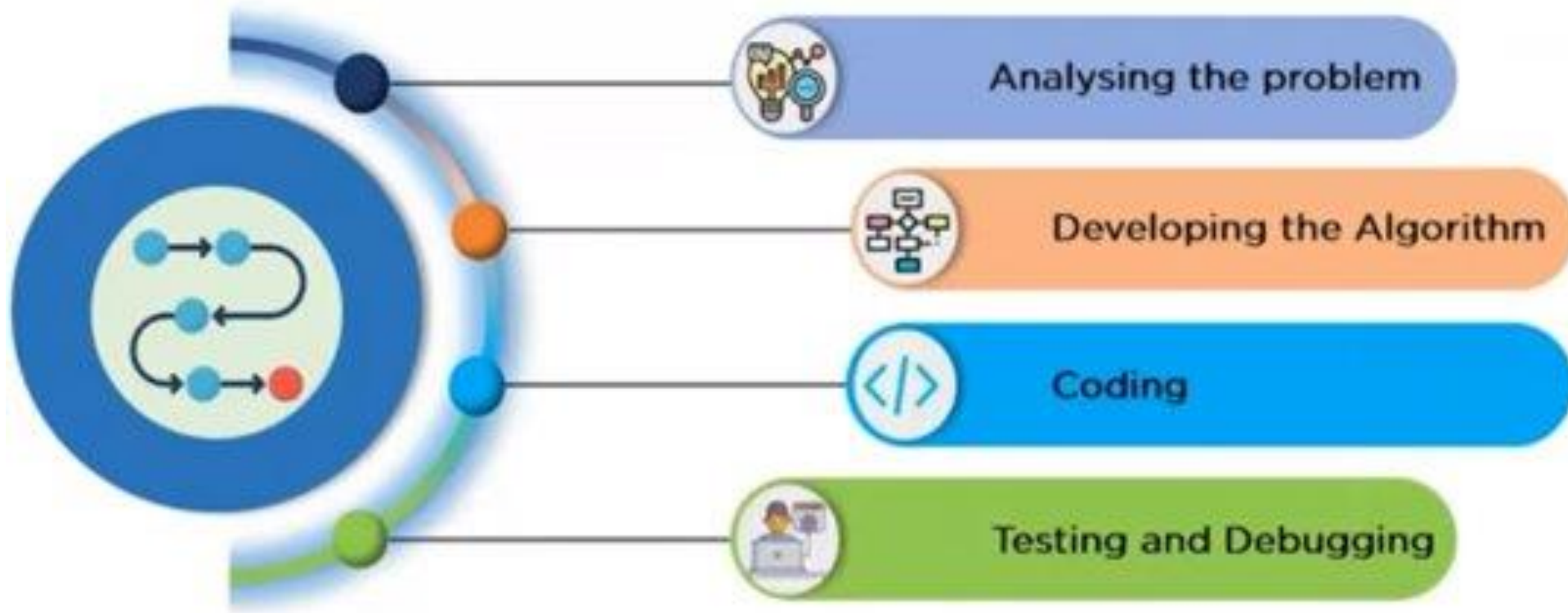
COMPUTER SCIENCE & INFORMATION PROCESSING RELATIONSHIP

- In order to solve a problem on a computer, you must:
 - Know how to **represent** data describing the problem.
 - Determine the steps to **transform** the information from one representation to another.

Lecture

01

PROBLEM-SOLVING STEPS



PROBLEM-SOLVING STEPS

Cont'd

- **Analyzing the problem (Step 1):**
 - We can figure out the inputs and outputs to be carried out.
- **Developing the algorithm (Step 2):**
 - It is required to decide a solution before writing a program.

PROBLEM-SOLVING STEPS

Cont'd

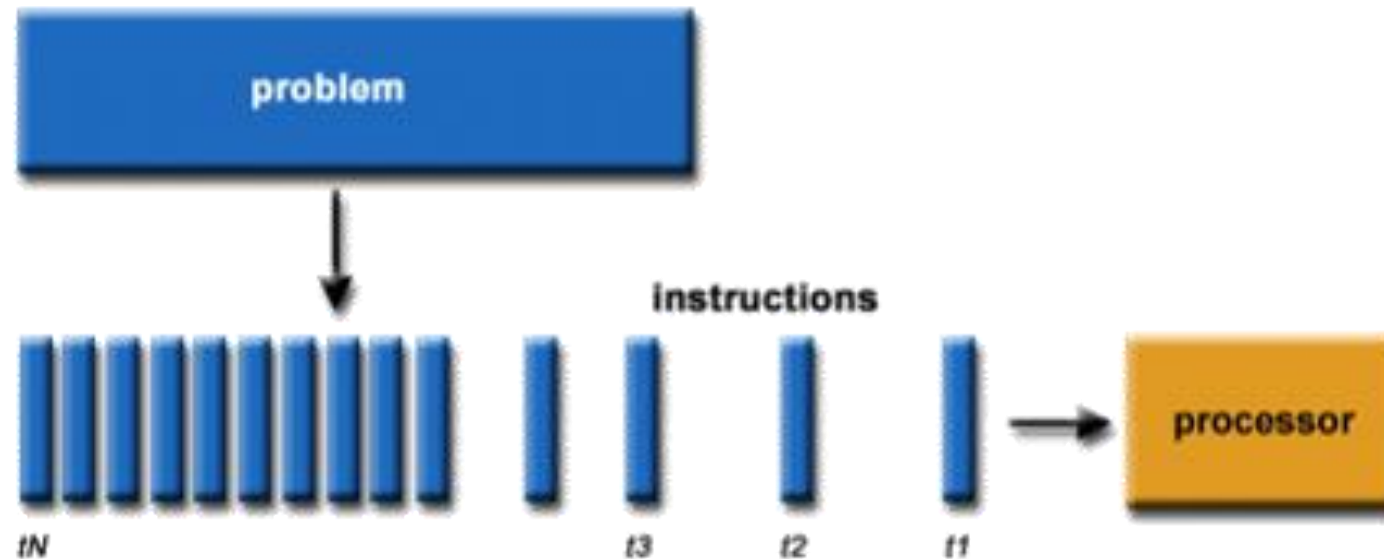
- **Coding (Step 3):**
 - We must convert the decided algorithm into a code.
- **Testing and debugging (Step 4):**
 - The developed code undergoes several rigorous tests.

Lecture

01

ALGORITHM

- The sequence of steps that describes each of the computational processes.



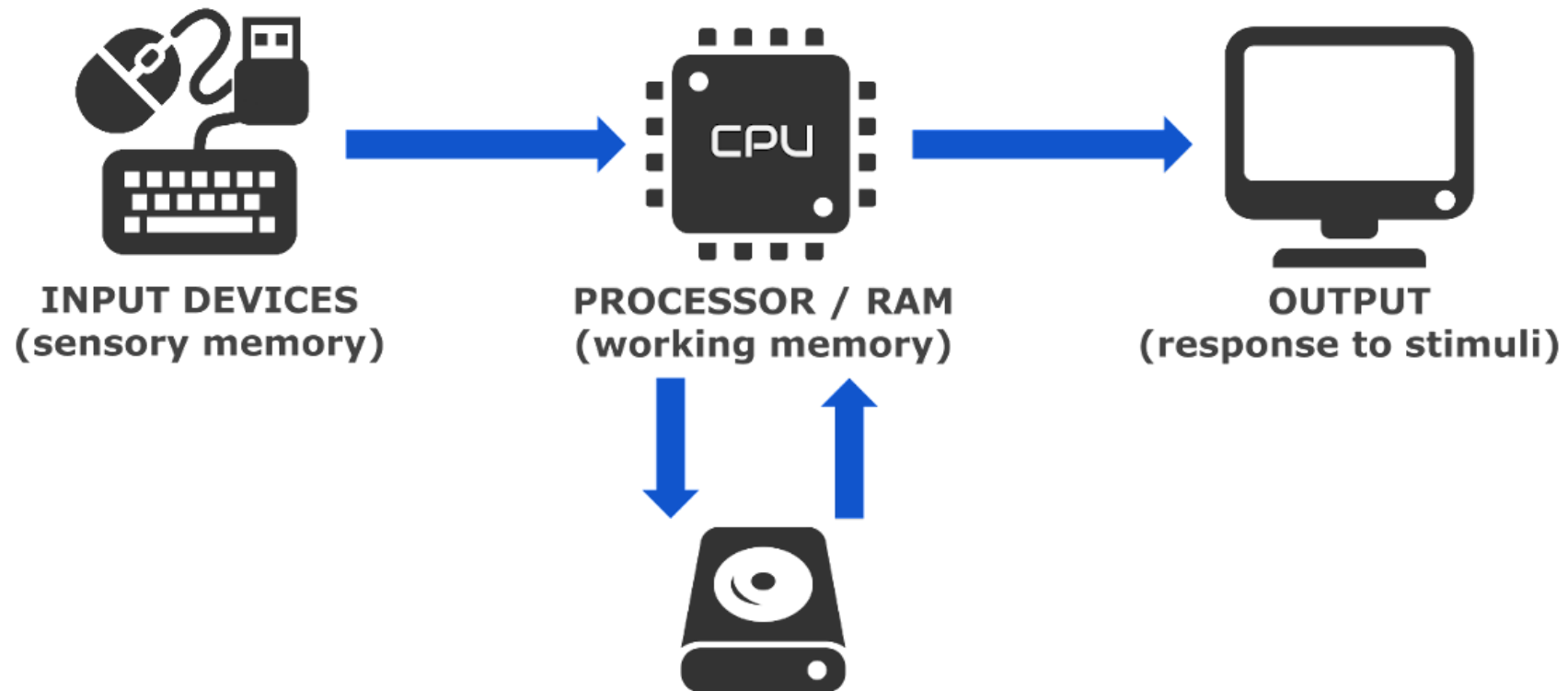
Lecture

01

ALGORITHM FEATURES

- It must consist of a **finite number** of instructions.
- Each algorithm instruction is well defined and **not ambiguous**.
- It should take **zero or more input**.
- It should **terminate** after a finite time and produce at least **one output**.
- It should be **deterministic** means giving the same output for the same input case.

HARDWARE & SOFTWARE COLLABORATION



Lecture

01

HARDWARE & SOFTWARE COLLABORATION

Cont'd

- A program (i.e., an algorithm) stored in computer memory must be represented in binary digits, or **machine code**.
- Each **loader** takes a set of machine language instructions as input and loads them into the appropriate memory locations.
- In CPU units performs the required computational, logical, and controlling the loaded instructions.

Lecture

01

PROGRAMMING & PROGRAMMING LANGUAGES

- **Programming:** the design of software using a set of basic instructions.
- **Programming language:** abstract hardware instructions into basic statements, which are much easier to understand and handle.
- Using a programming language, one can write complex software, achieving tailored functionalities by combination of basic statements.

Lecture

01

PYTHON PROGRAMMING LANGUAGE

- Famous and fastest growing language.
- Open-source language.
 - Mostly open so that **anyone** can modify them, and they are usually not proprietary.
- Multi-purpose language.
 - It's a **general-purpose** language, you can use Python to do and create many different things.

Lecture

01

PYTHON PROGRAMMING LANGUAGE

Cont'd

- Scripting and interpreted language.
 - **Compiler:** scans the whole program at once and converts it into machine code.
 - **Interpreter:** converts the program into machine code one line at a time.
- Simple, expressive, and easy to learn language.

C#: `str.Substring(0, 6);` JavaScript: `str.substr(0, 6);` Python: `str[0:6]`

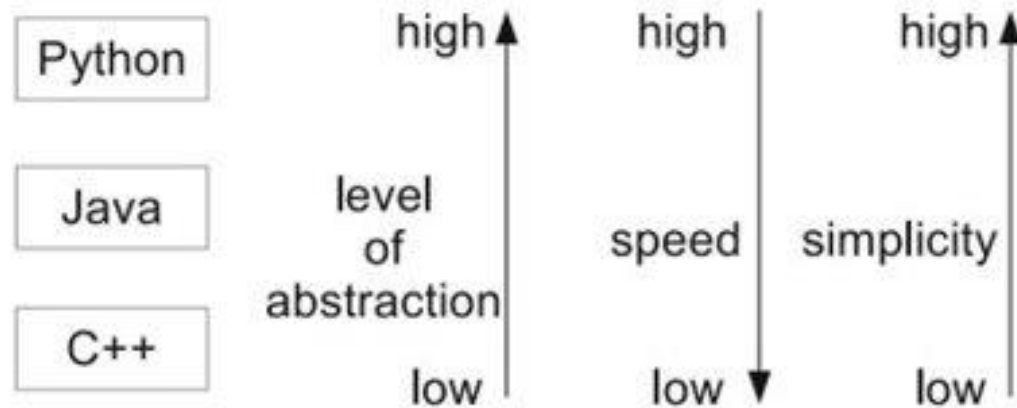
Lecture

01

PYTHON PROGRAMMING LANGUAGE

Cont'd

- High-level language.
- Cross platform language.



Lecture

01

PYTHON USES

- Web Development (Django - Flask).
- Games (PyGame).
- Desktop Apps (PyGUI - Tkinter).
- Hacking.
- Machine Learning and Data Science.

Lecture

01

PYTHON USES

Cont'd

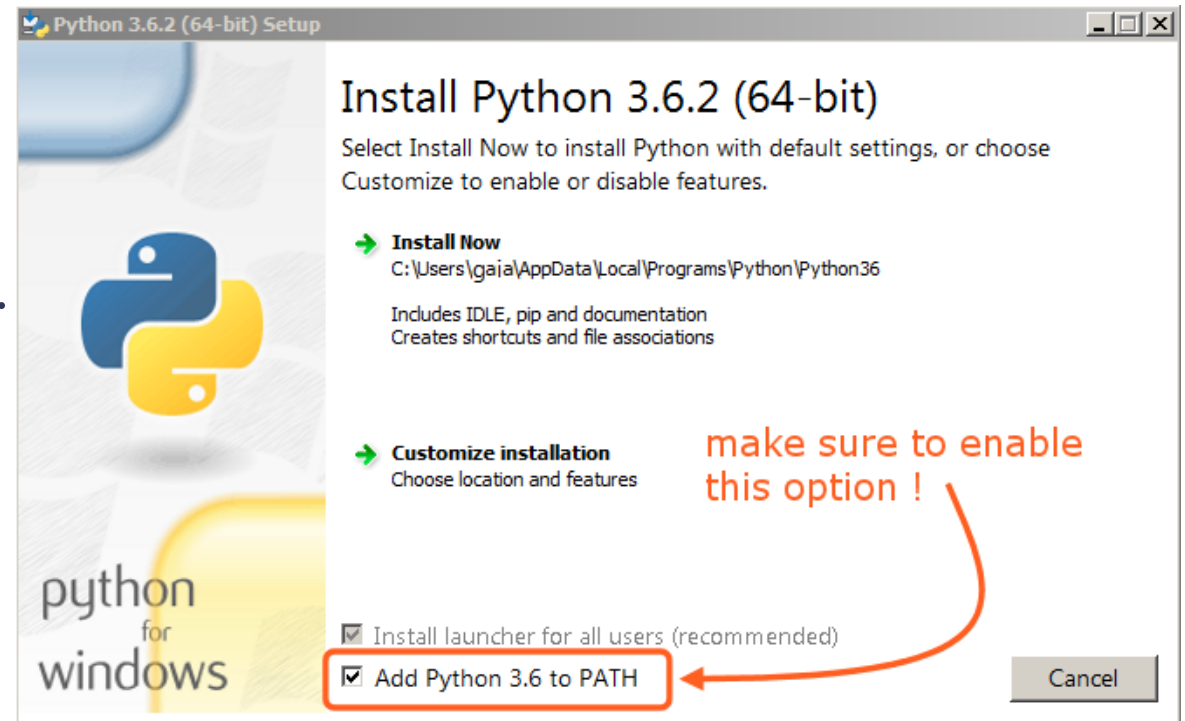
- AI and Robots in Mars.
- Automation.
- Network Programming and Web Scraping.
- Android Games.

Lecture

01

PYTHON INSTALLATION
ENVIRONMENT

- Download Python language.
- On Windows.
 - Follow the Straightforward steps.



Lecture

01

PYTHON INSTALLATION
ENVIRONMENT

Cont'd

- On Mac.
 - Follow the Straightforward steps.



Lecture

01

PYTHON INSTALLATION ENVIRONMENT

Cont'd

- On Linux.

```
sudo apt install python3 python3-pip build-essential python3-dev
```

PYTHON INSTALLATION ENVIRONMENT

Cont'd

- After installation, the executable Python application is the file:
 - On Windows:
`C:\Python_version\python.exe`
 - On Linux and Mac:
 - Python is typically pre-installed, and can be found at:
`/usr/bin/python_version`

Lecture

01

PYTHON INTEGRATED DEVELOPMENT ENVIRONMENT (IDE)

- Pycharm IDE.
 - [Download.](#)
- Visual Studio Code (VScode) IDE.
 - [Download.](#)



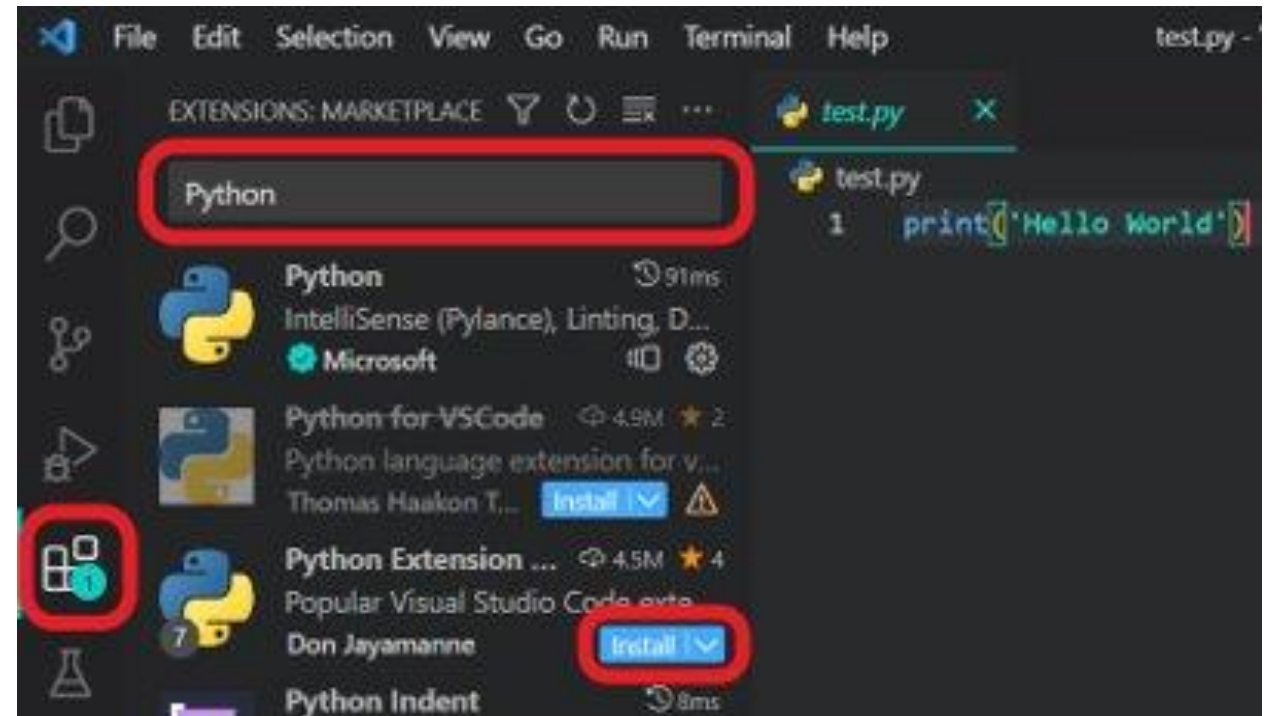
Visual Studio Code



PYTHON INTEGRATED DEVELOPMENT ENVIRONMENT

Cont'd

- In VScode:
 - You need to add Python extension.



Lecture

01

PYTHON INTEGRATED DEVELOPMENT ENVIRONMENT

Cont'd

- Online Editors:
 - [Online Python.](#)
 - [Pynative.](#)
 - [Programiz.](#)

Lecture

01

FIRST PROGRAM IN PYTHON

- In order to write Python scripts (codes) in your machine (**offline**), we **must** install Python language.
- Writing Python scripts using any **online** editors does **not require** to install Python language on your machine.

FIRST PROGRAM IN PYTHON

Cont'd

- Follow these steps in writing codes:
 - Create a code folder in any location on your machine.
 - Create a Python script file.
 - Python has **.py** extension for its script files.
 - Open the script file in your IDE.
 - Write your code.

```
print("Hello World from Python")
```

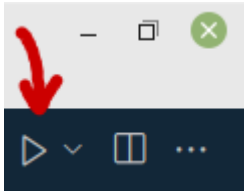
Lecture

01

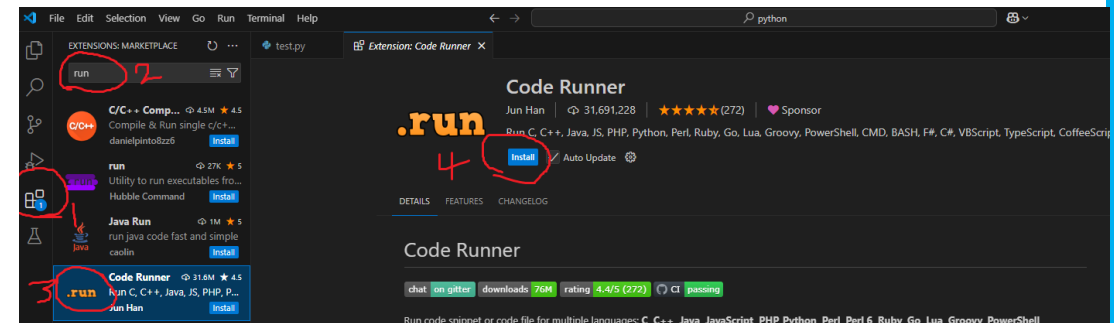
FIRST PROGRAM IN PYTHON

Cont'd

- Follow these steps in writing codes:
 - In VScode, press RUN button on top right corner.



- OR, run your code in cmd/terminal.
`python3 first.py`



`"/path/to/python3" "/path/to/code file/first.py"`

Lecture

01

PYTHON FORMATING STYLE

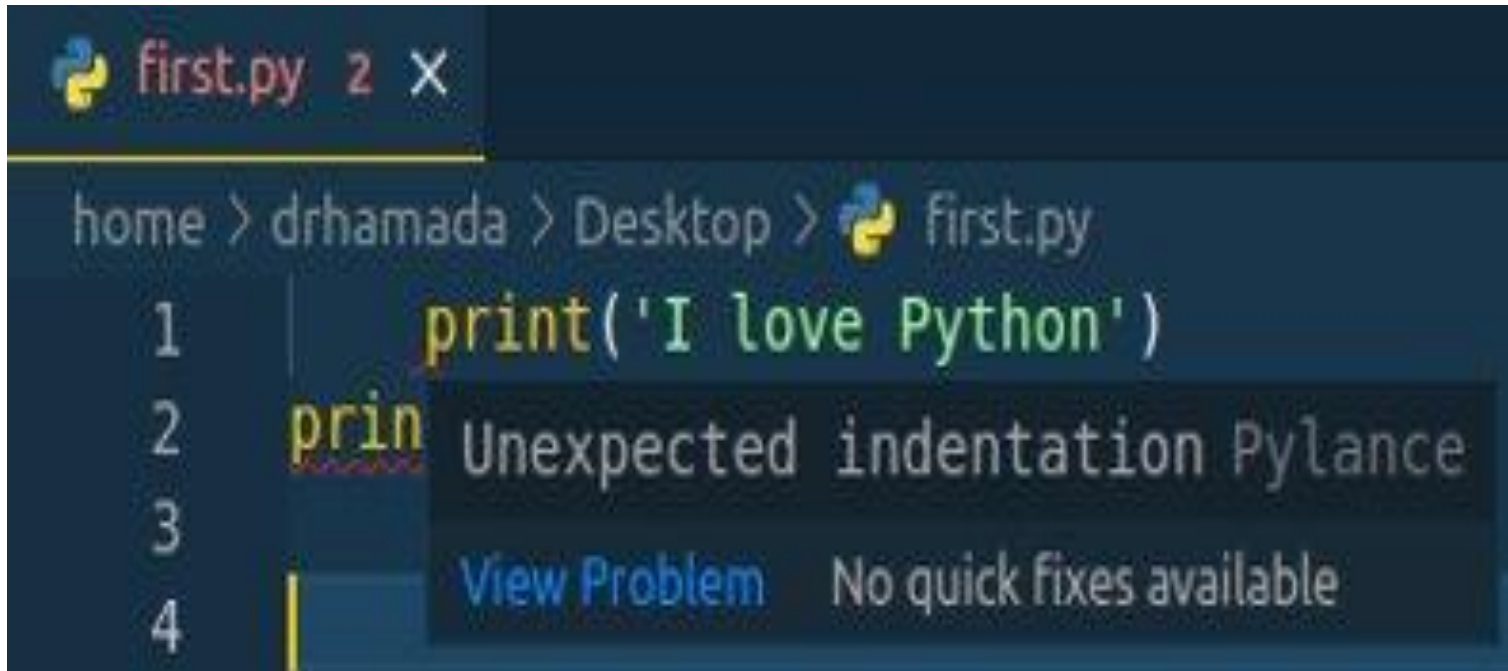
- **Indentation:** the leading whitespace before any statement in Python.
- We can use even if **one space** for code indentation, the default is **four spaces**.
- **A code block:** a group of statements that have the same indentation level (have the same number of leading whitespaces).
- It used to make the code more **readability**.

Lecture

01

PYTHON FORMATING STYLE

Cont'd



The screenshot shows a code editor window titled 'first.py'. The file path is 'home > drhamada > Desktop > first.py'. The code contains two lines: line 1 has 'print('I love Python')' and line 2 has 'prin'. A red squiggly line under 'prin' indicates an error. A tooltip is displayed over the error, showing the text 'Unexpected indentation Pylance' and a 'View Problem' link. The tooltip also states 'No quick fixes available'.

```
1 | print('I love Python')
2 | prin
3 |
4 |
```

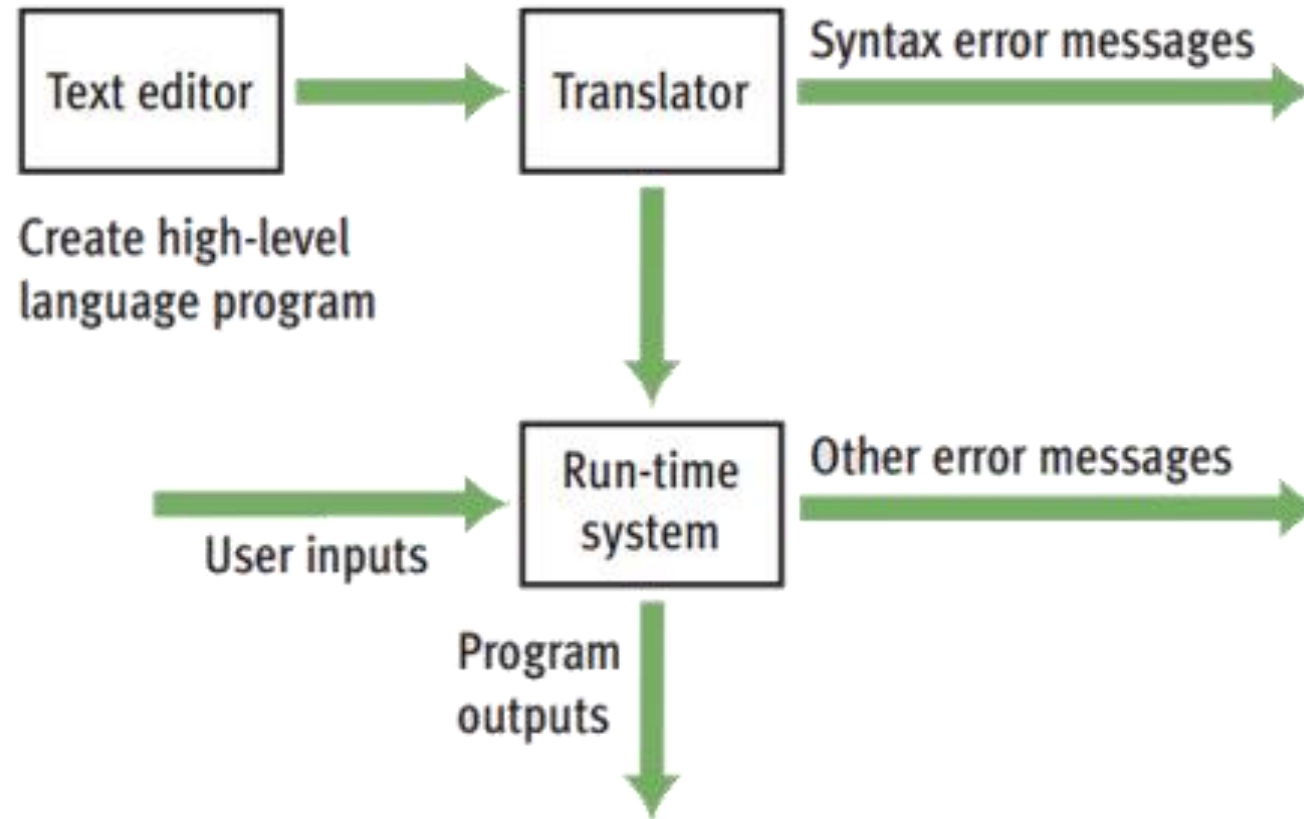
Unexpected indentation Pylance

[View Problem](#) No quick fixes available

Lecture

01

PROGRAMMING SCENARIO



Lecture

01

PROGRAMMING SCENARIO

Cont'd

- A programmer typically starts by writing high-level language statements in a text editor.
- The programmer then runs another program called a **translator** to convert the high-level program code into executable code.
- Because it is possible for a programmer to make grammatical mistakes even when writing high-level code, the translator checks for syntax errors before it completes the translation process.

PROGRAMMING SCENARIO

Cont'd

- The programmer then has to revise the program.
- If the translation process succeeds without a syntax error, the program can be executed by the run-time system.
- The run-time system might execute the program directly on the hardware or run yet another program called an **interpreter or virtual machine** to execute the program.

Lecture

01

QUESTIONS FOR ONLINE DISCUSSION

- Which of the following can be achieved using Python language:
 - Mathematical calculations.
 - A Windows GUI application.
 - Controlling a robotic arm.
 - Web crawling.
 - A machine-learning system that can fly a helicopter.
 - Simulation of large-scale experiments in physics.
 - A game in Android.
 - A operating system from scratch.



Thank You

