

Beginner Programming Fundamentals With Python

Module 1

Introduction to programming

Ali Samanipour

Jan 2022

Module Roadmap

- 1 Introduction
- 2 Hardware and Software
- 3 How Computers Store Data
- 4 How Programs Works
- 5 Using Python

Computers & Programmers

Computers are not designed to do just one job, but to do any job that their programs tell them to do.

A program is a set of instructions that a computer follows to perform a task.
Programs are commonly referred to as software.

A programmer, or software developer, is a person with the training and skills necessary to design, create, and test computer programs

How Programming Works

1 Craft your algorithm

This is where you take the problem or task you want solved and turn it into a high-level recipe, pseudocode, or algorithm that describes the steps that need to be performed by the computer to achieve whatever result you are after.

- 1 Put worm on hook.
- 2 Cast line into pond.
- 3 Watch the bobber until it goes underwater.
- 4 Hook and pull in fish.
- 5 If done fishing, then go home; otherwise, go back to step 1.

This is the step where we map out our solution before doing the hard work of translating it into a programming language.

2 Write your program

Next, you take that recipe and translate it into a specific set of instructions that are written in a programming language. This is the *coding* stage, and we call the result a *program* or just “your code” (or, more formally, the *source code*).

```
def hook_fish():  
    print('I got a fish!')  
  
def wait():  
    print('Waiting...')  
  
print('Get worm')  
print('Put worm on hook')  
print('Throw in lure')  
  
while True:  
    response = input('Is bobber underwater? ')  
    if response == 'yes':  
        is_moving = True  
        print('I got a bite!')  
        hook_fish()  
    else:  
        wait()
```

This is the “coding” step where you turn your algorithm into code (which is shorthand for source code) that is ready to execute in the next step.

3 Run your program

Finally, you take your source code and hand it to the computer, which will start carrying out your instructions. Depending on the language you’re using, this process might be called *interpreting*, *running*, *evaluating*, or *executing* your code.

We often use some of these terms interchangeably as well.



When your source code is complete, you’re ready to execute it. If all goes well, and you designed your code well, you’ll get the result from the computer you were looking for.

Module Roadmap

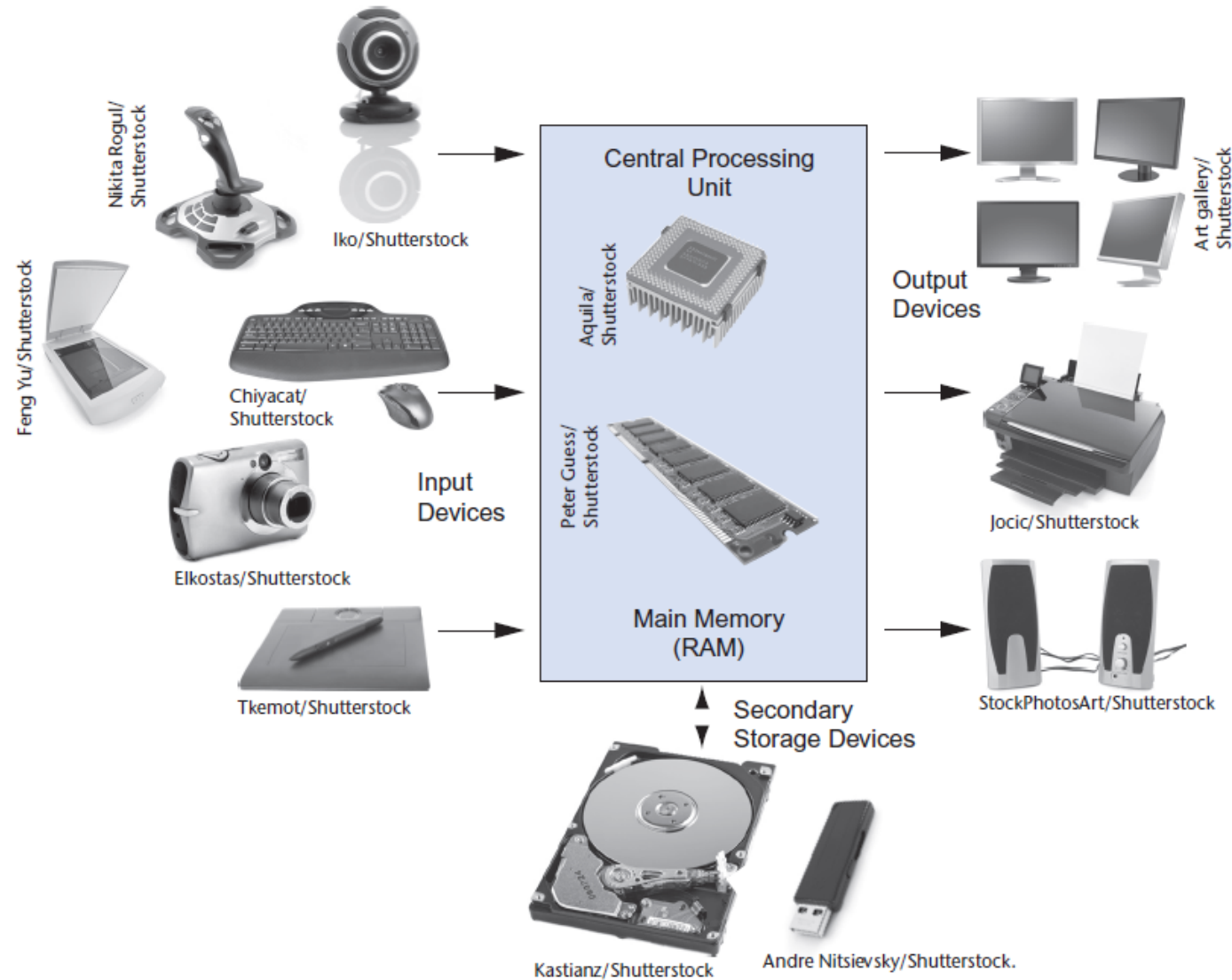
- 1 Introduction
- 2 Hardware and Software
- 3 How Computers Store Data
- 4 How Programs Works
- 5 Using Python

Hardware and Software

CONCEPT

The physical devices of which a computer is made are referred to as the computer's hardware. The programs that run on a computer are referred to as software.

Hardware



CPU

When a computer is performing the tasks that a program tells it to do, we say that the computer is running or executing the program.

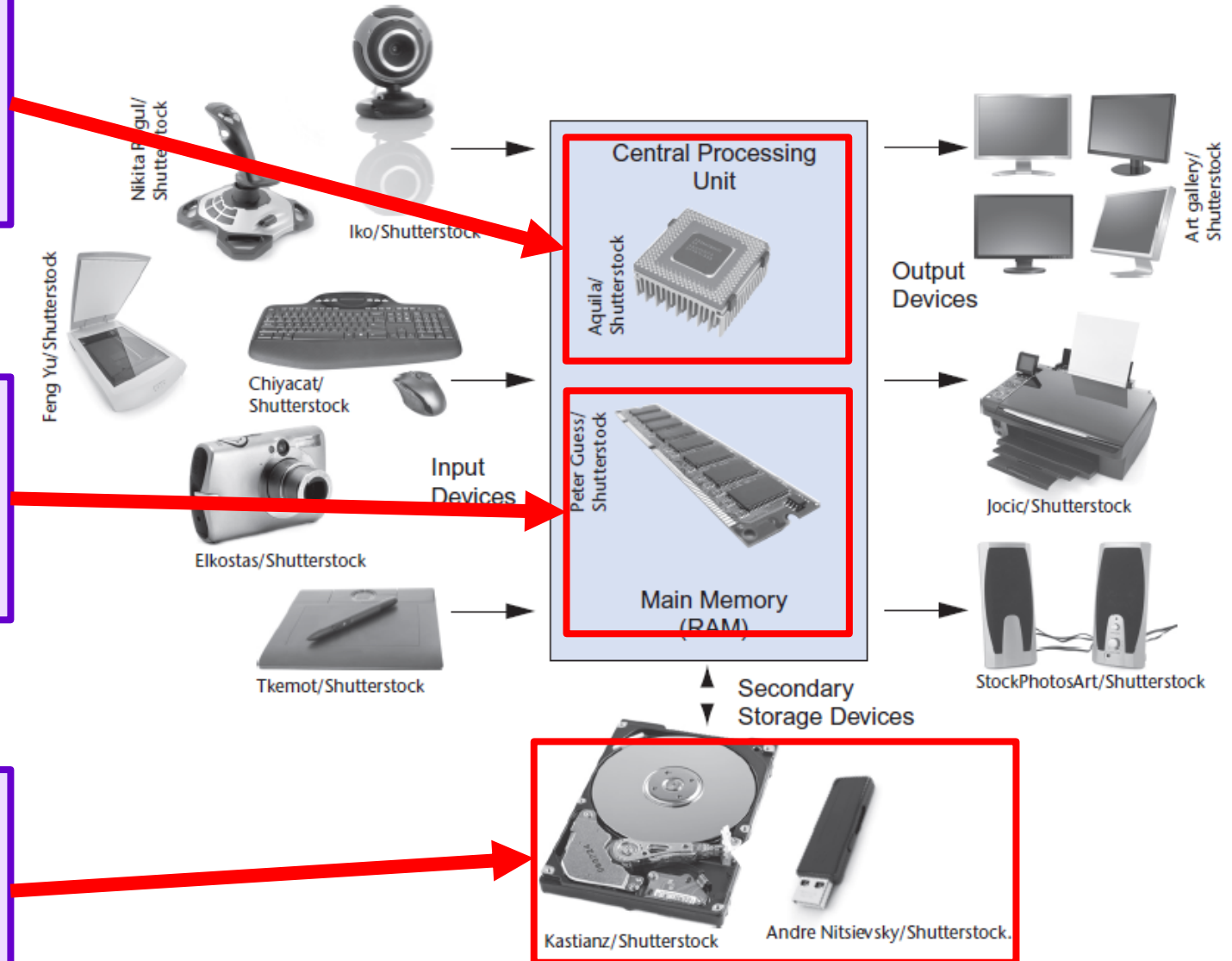
Main Memory

This is where the computer stores a program while the program is running, as well as the data that the program is working with.

Secondary Storage

Secondary storage is a type of memory that can hold data for long periods of time, even when there is no power to the computer.

Hardware



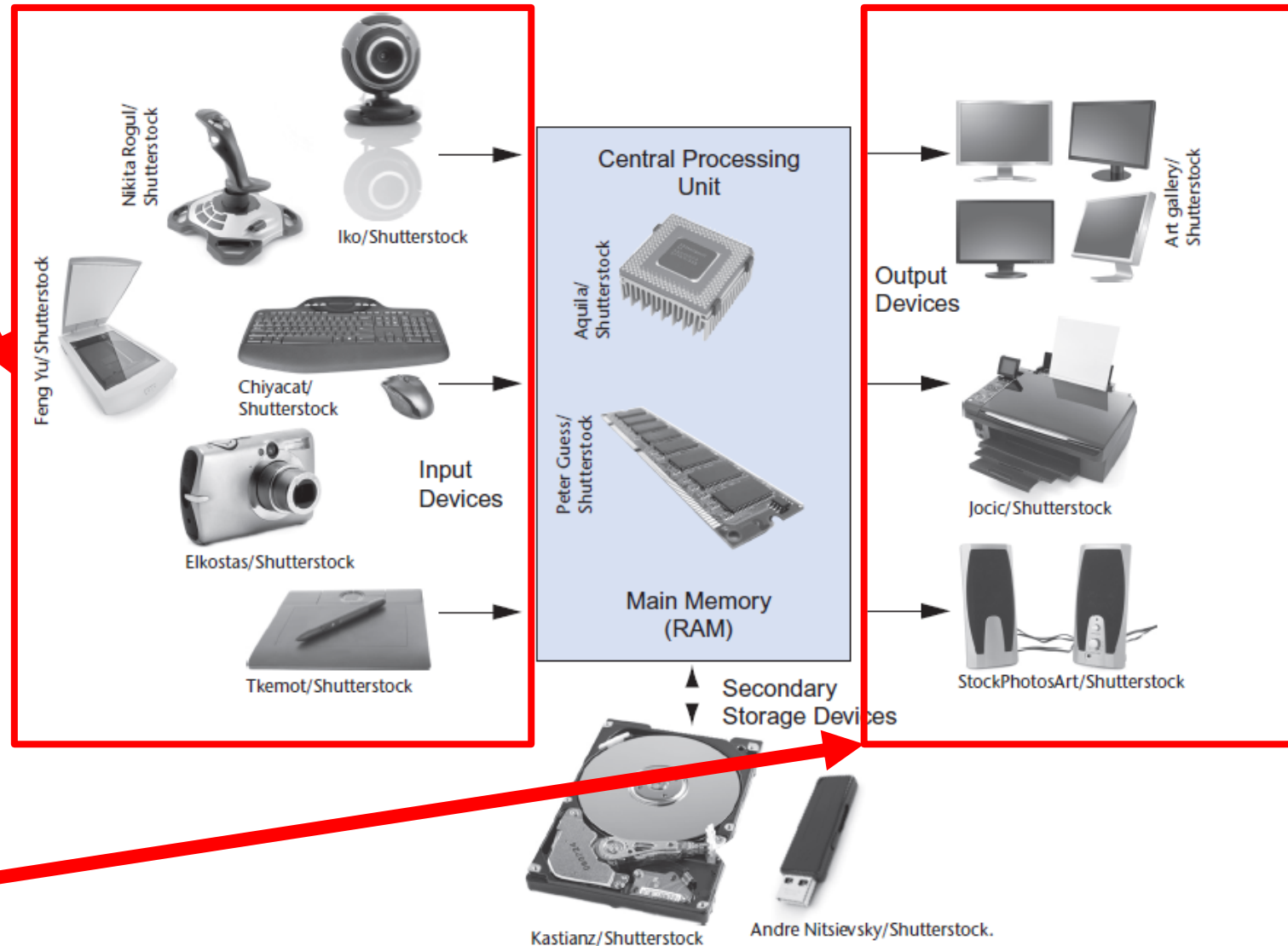
Hardware

Input Devices

The component that collects the data and sends it to the computer is called an input device.

Output Devices

The data is sent to an output device, which formats and presents it.



Software

System Software

The programs that control and manage the basic operations of a computer

- Operating Systems
- Utility Programs
- Software Development Tools

Application Software

Programs that make a computer useful for everyday tasks are known as application software. These are the programs that people normally spend most of their time running on their computers.

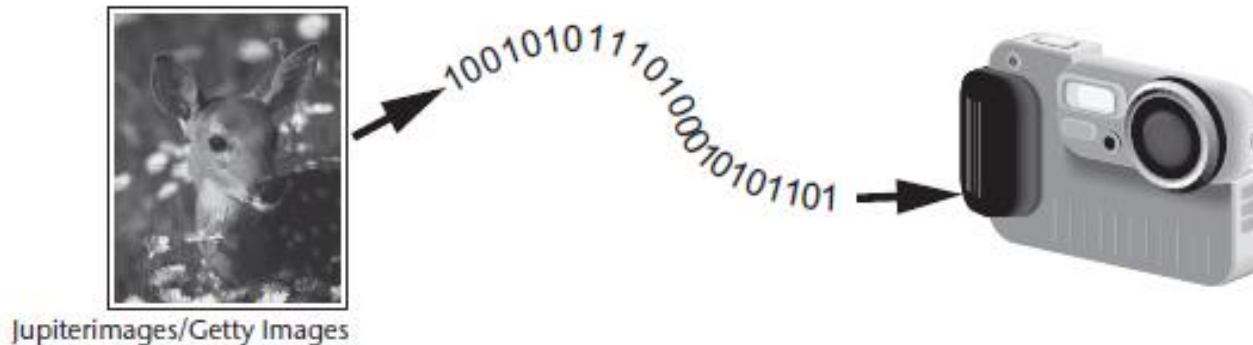
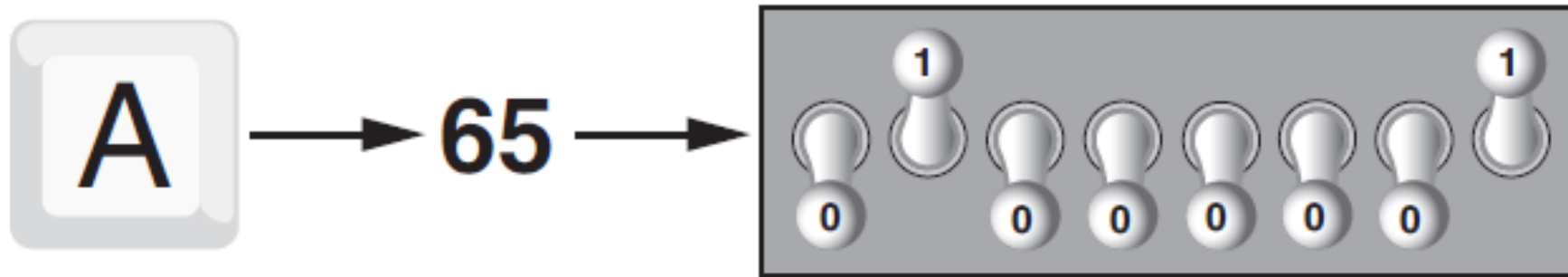
Module Roadmap

- 1 Introduction
- 2 Hardware and Software
- 3 How Computers Store Data
- 4 How Programs Works
- 5 Using Python

How Computers Store Data

CONCEPT

All data that is stored in a computer is converted to sequences of 0s and 1s.
(bits and bytes)



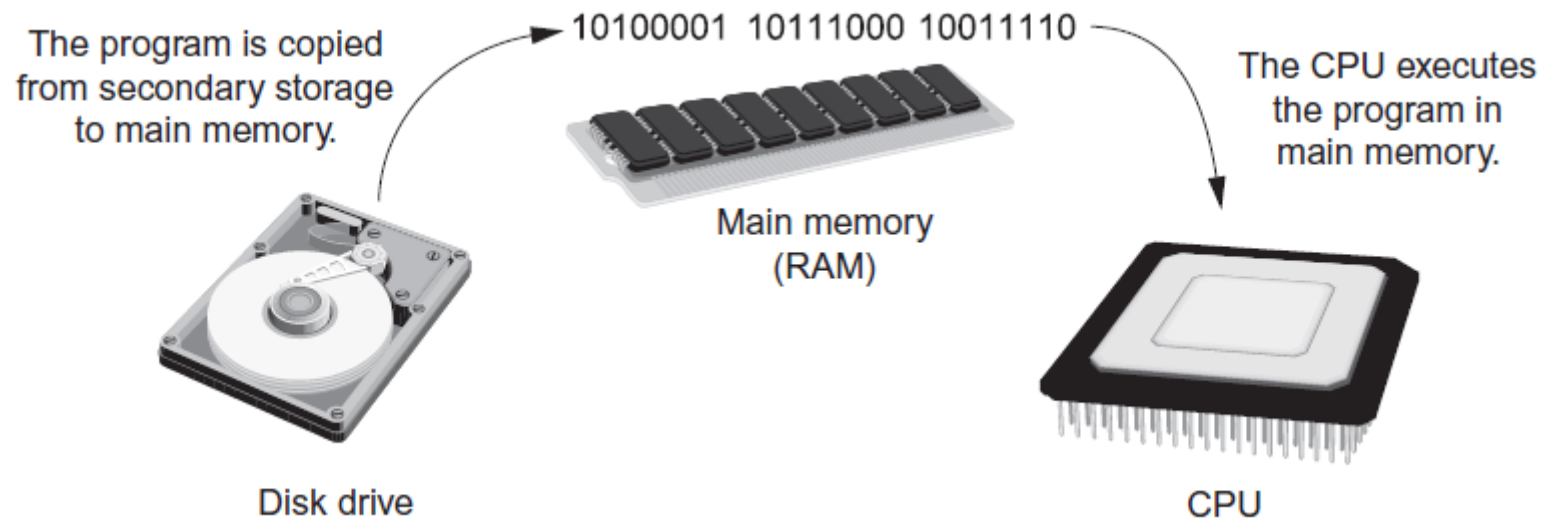
Module Roadmap

- 1 Introduction
- 2 Hardware and Software
- 3 How Computers Store Data
- 4 How Programs Works
- 5 Using Python

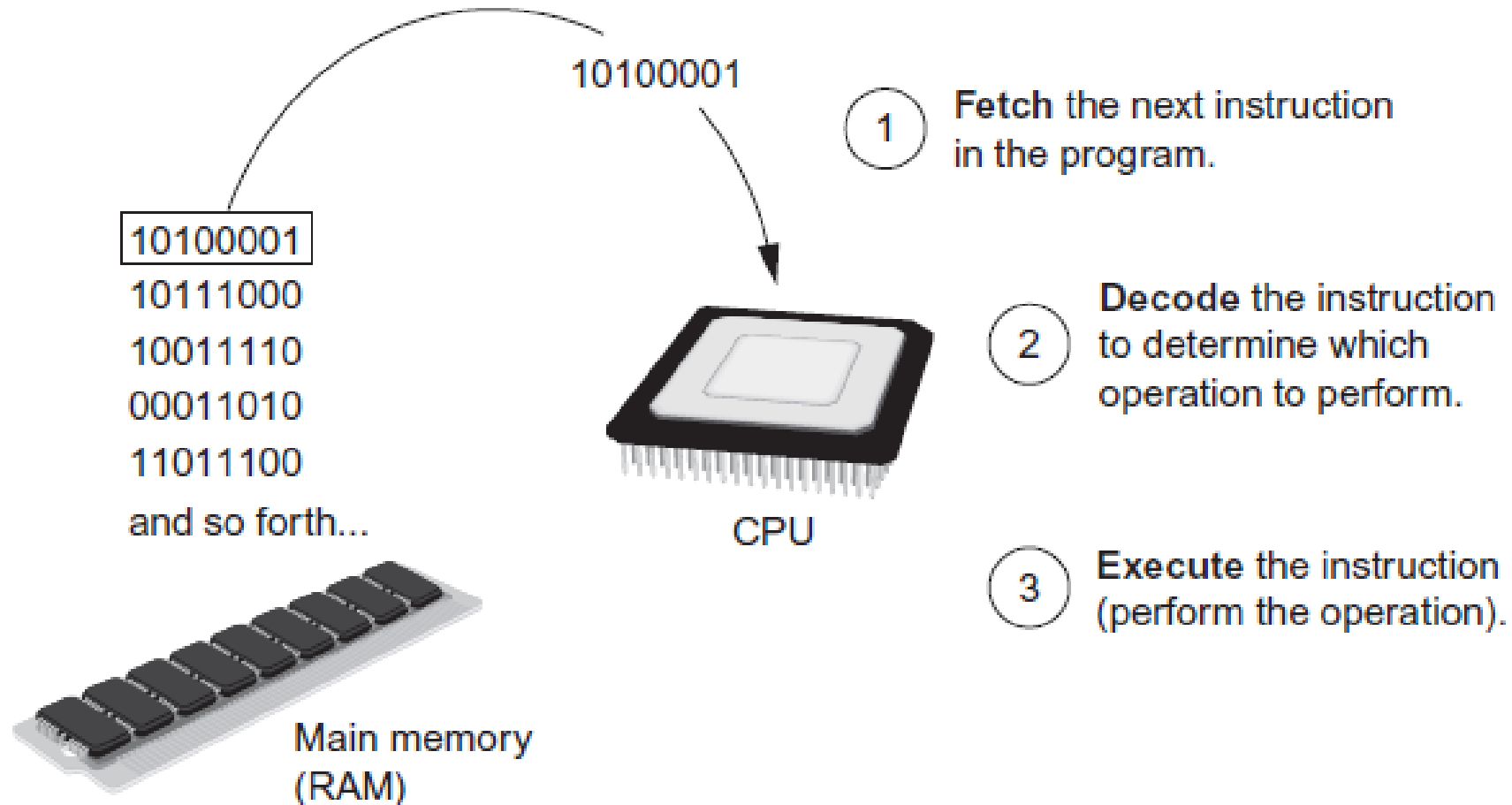
How a Program Works

CONCEPT

A computer's CPU can only understand instructions that are written in machine language. Because people find it very difficult to write entire programs in machine language, other programming languages have been invented.



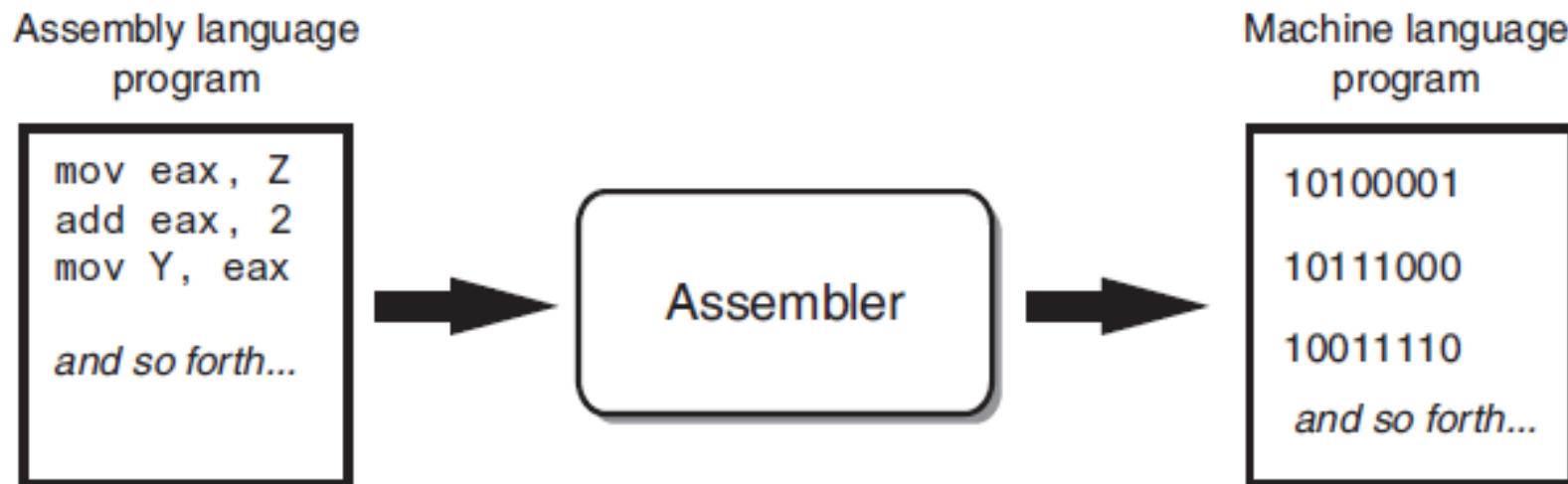
Fetch-Decode-Execute Cycle



From Machine Language to Assembly Language

Computers can only execute programs that are written in machine language


Programming in machine language would also be very difficult, because putting a 0 or a 1 in the wrong place will cause an error.



High-Level Languages

Assembly language also requires that you write a large number of instructions for even the simplest program and also it requires that you know a lot about the CPU.

A *high-level language* allows you to create powerful and complex programs without knowing how the CPU works and without writing large numbers of low-level instructions. In addition, most high-level languages use words that are easy to understand.



```
1 DISPLAY "Hello World"
```



```
1 print("Hello World")
```

Keywords, Operators, and Syntax: An Overview

Each high-level language has its own set of predefined words that the programmer must use to write a program

The words that make up a high-level programming language are known as ***keywords*** or ***reserved words***.

Each keyword has a specific meaning, and cannot be used for any other purpose.

The Python keywords

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

Keywords, Operators, and Syntax: An Overview

In addition to keywords, programming languages have ***operators*** that perform various operations on data

In Python, as well as most other languages, the + sign is an operator that adds two numbers.

7 + 12

Keywords, Operators, and Syntax: An Overview

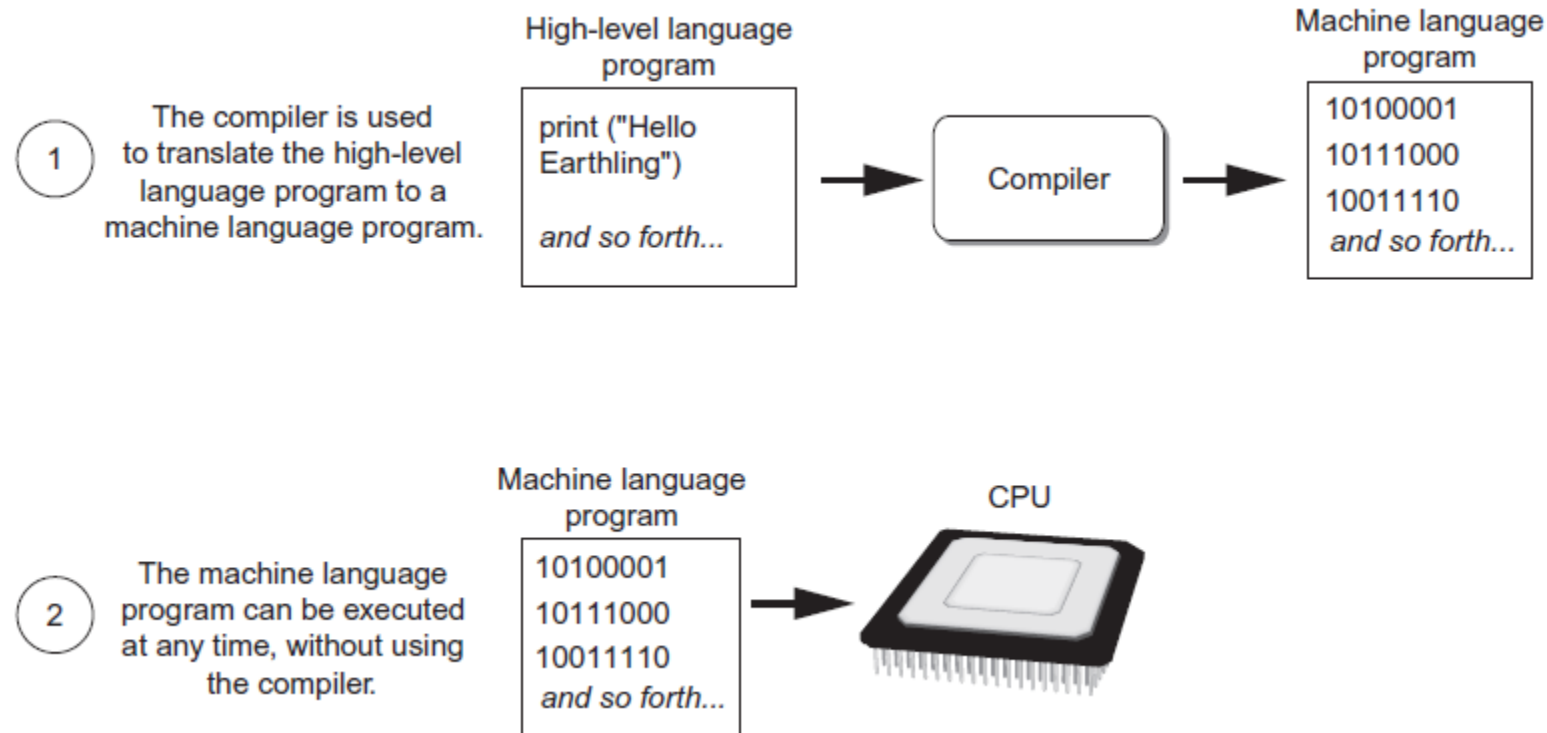
In addition to keywords and operators, each language also has its own ***syntax***, which is a set of rules that must be strictly followed when writing a program.

The ***syntax*** rules dictate how keywords, operators, and various punctuation characters must be used in a program

The individual instructions that you use to write a program in a high-level programming language are called ***statements***.

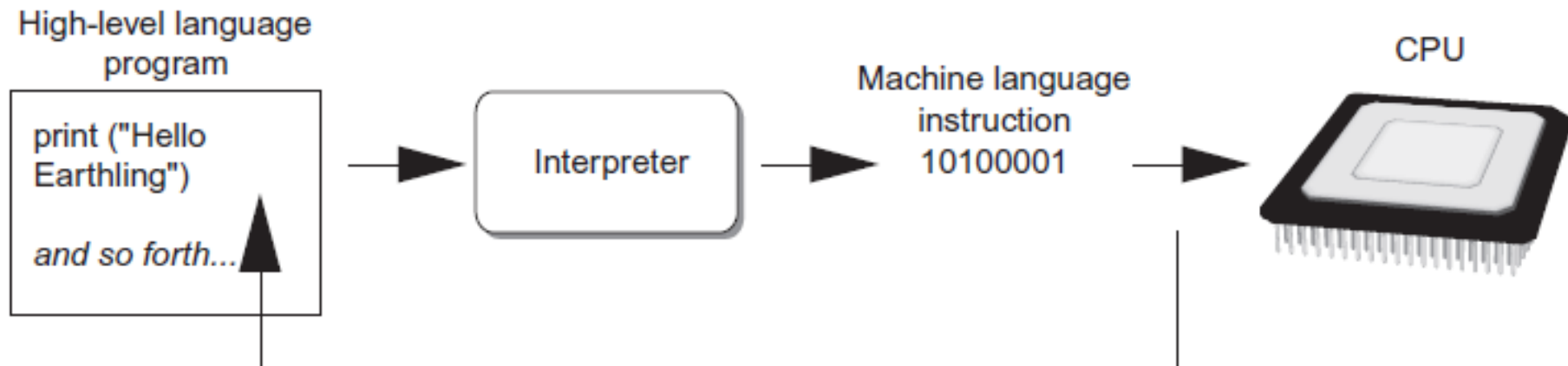
Compilers and Interpreters

A **compiler** is a program that translates a high-level language program into a separate machine language program. The machine language program can then be executed any time it is needed.



Compilers and Interpreters

A ***interpreter*** is a program that both translates and executes the instructions in a high-level language program. As the interpreter reads each individual instruction in the program, it converts it to machine language instructions then immediately executes them



The interpreter translates each high-level instruction to its equivalent machine language instructions then immediately executes them.

This process is repeated for each high-level instruction.

Source Code

The statements that a programmer writes in a high-level language are called ***source code***, or simply ***code***

Typically, the programmer types a program's code into a text editor then saves the code in a file on the computer's disk.

Next, the programmer uses a compiler to translate the code into a machine language program, or an interpreter to translate and execute the code

Module Roadmap

- 1 Introduction
- 2 Hardware and Software
- 3 How Computers Store Data
- 4 How Programs Works
- 5 Using Python

Before Start!

Learn programming
(Software development)
fundamentals

Python is just a **tool**, you should use it for **your purpose**

train your brain to think as a
problem solver

Learn how to use python first,
not memorize how python works

What is python?

Clean and simple syntax,
Follows a “batteries included”
approach, Offers great
documentation

Python is a **powerful**, **easy-to-use**, **object-oriented** programming language

It's performant,
it runs on all operating systems,
It's extremely versatile

You can use classes, inheritance
and allows you to work with
complex data structures

Python's Versatility

Programs & Script

Use it like a tool
(REPL)

Utility scripts or complex
programs

Create complete desktop
applications(with ui)

Web Development

As a server side language

Build full stack app or
APIs

Use frameworks

Ali Samanipour
[linkedin.com/in/Samanipour](https://www.linkedin.com/in/Samanipour)

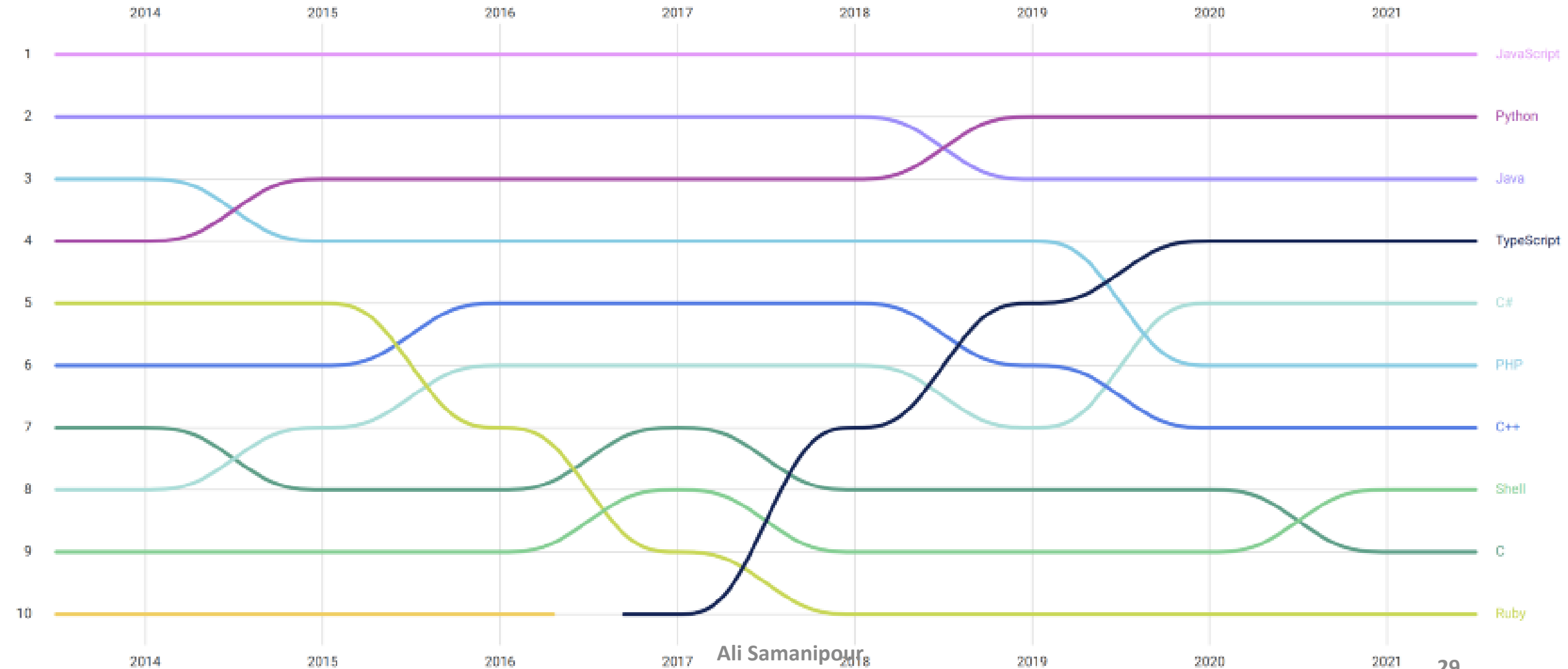
Data Science & AI

Rich set of third-party
packages

Data gathering and
processing

Data analysis, machine
learning,...

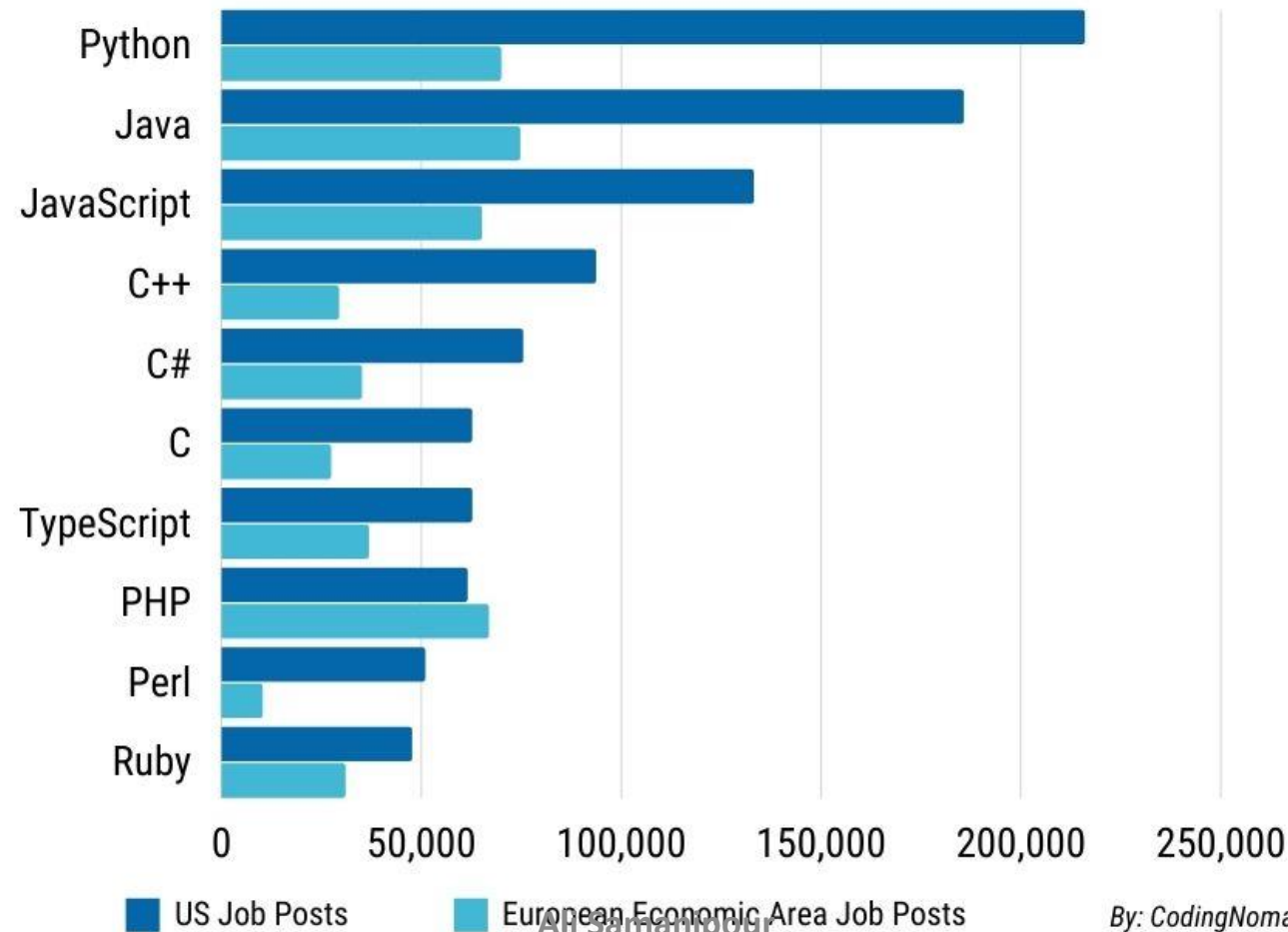
Python's Past, Present & future!?



Python's Past, Present & future!?!...

Most in-demand programming languages of 2022

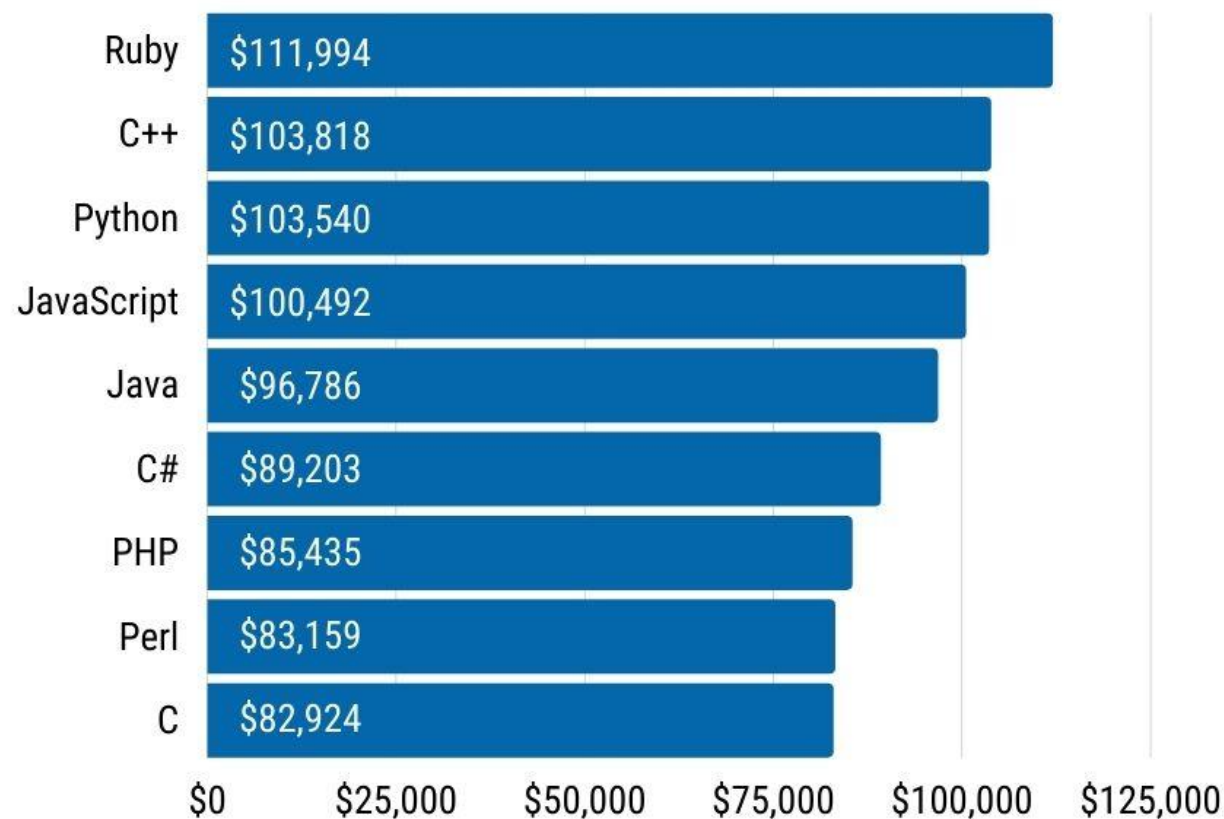
Based on LinkedIn job postings in the USA & Europe



Python's Past, Present & future!?!...

Average US Developer Salaries 2022

Based on averages from Indeed & Glassdoor






















By: CodingNomads

Python's Past, Present & future!?!...

Which Programming Language to Learn

Based on Your Career Goals

Front-end web development	Back-end web development	Mobile development
 JavaScript	 JavaScript	 Swift
 Elm	 Scala	 Java
 TypeScript	 Python	 Objective C
	 Go	 JavaScript
	 Ruby	
Game development	Desktop applications	Systems programming
 Unity	 Scala	 Go
 TypeScript	 Go	 Rust
	 Python	

Ali Samanipour

[linkedin.com/in/Samanipour](https://www.linkedin.com/in/Samanipour)

Lets dirty our hand!!!

```
PS C:\Users\ali> python
Python 3.10.1 (tags/v3.10.1:2cd268a, Dec 6 2021, 19:10:37) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 128567*6687
859727529
>>> input("please enter your name: ")
please enter your name: Ali
'Ali'
>>> name = input("please enter your name: ")
please enter your name: Ali Samanipour
>>> name
'Ali Samanipour'
>>> file = open('myName.txt',mode='w')
>>> file.write(name)
14
>>> file.close()
>>> file = open('myName.txt', mode='r')
>>> file.read()
'Ali Samanipour'
>>>
```

Appendix: Installing python, Step1

Go to python website:

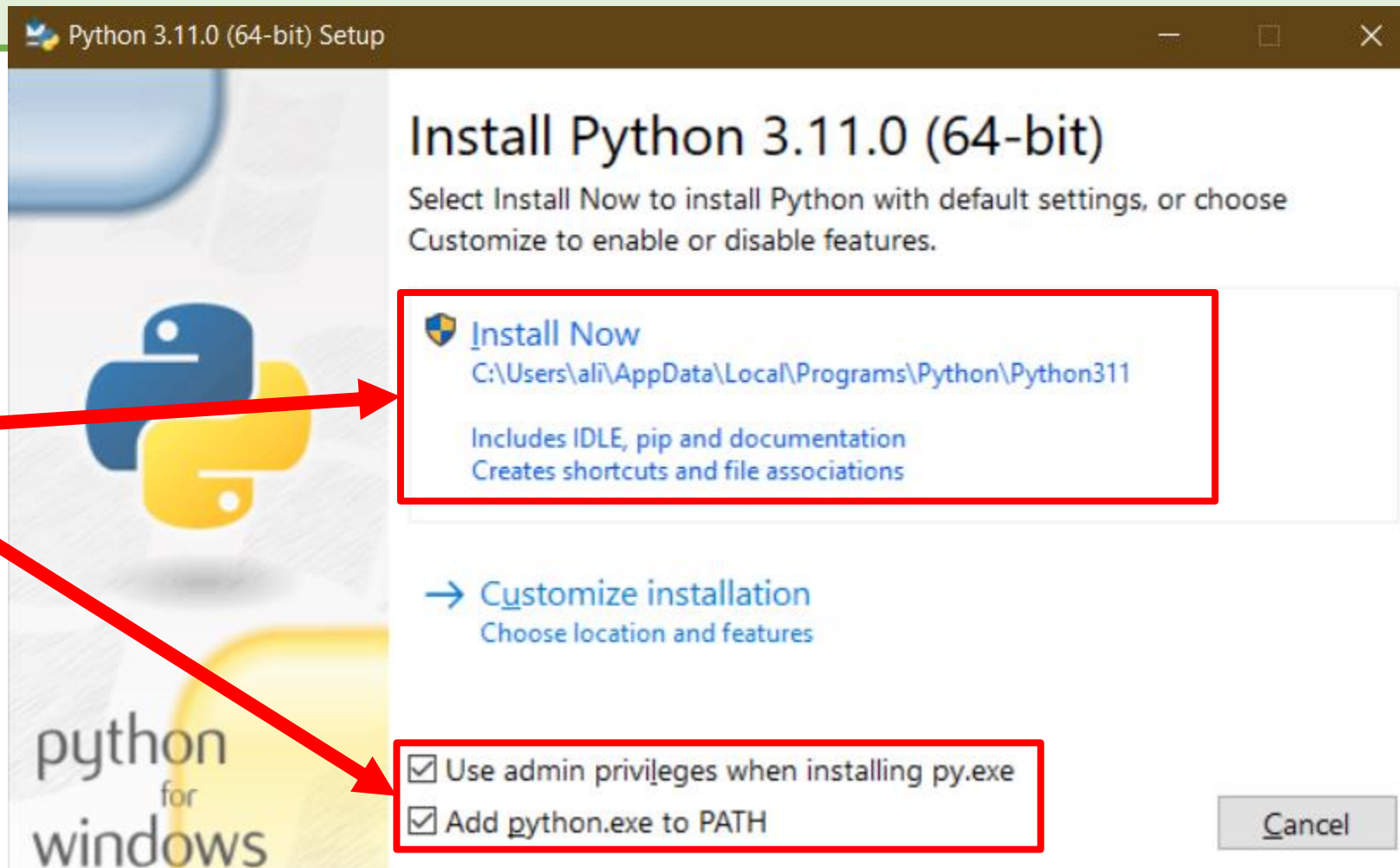
<https://www.python.org/downloads/>

And download installer



Appendix: Installing python, Step2

Open Installer and you have downloaded in step 1 by double clicking on it
an fallow installation steps like below

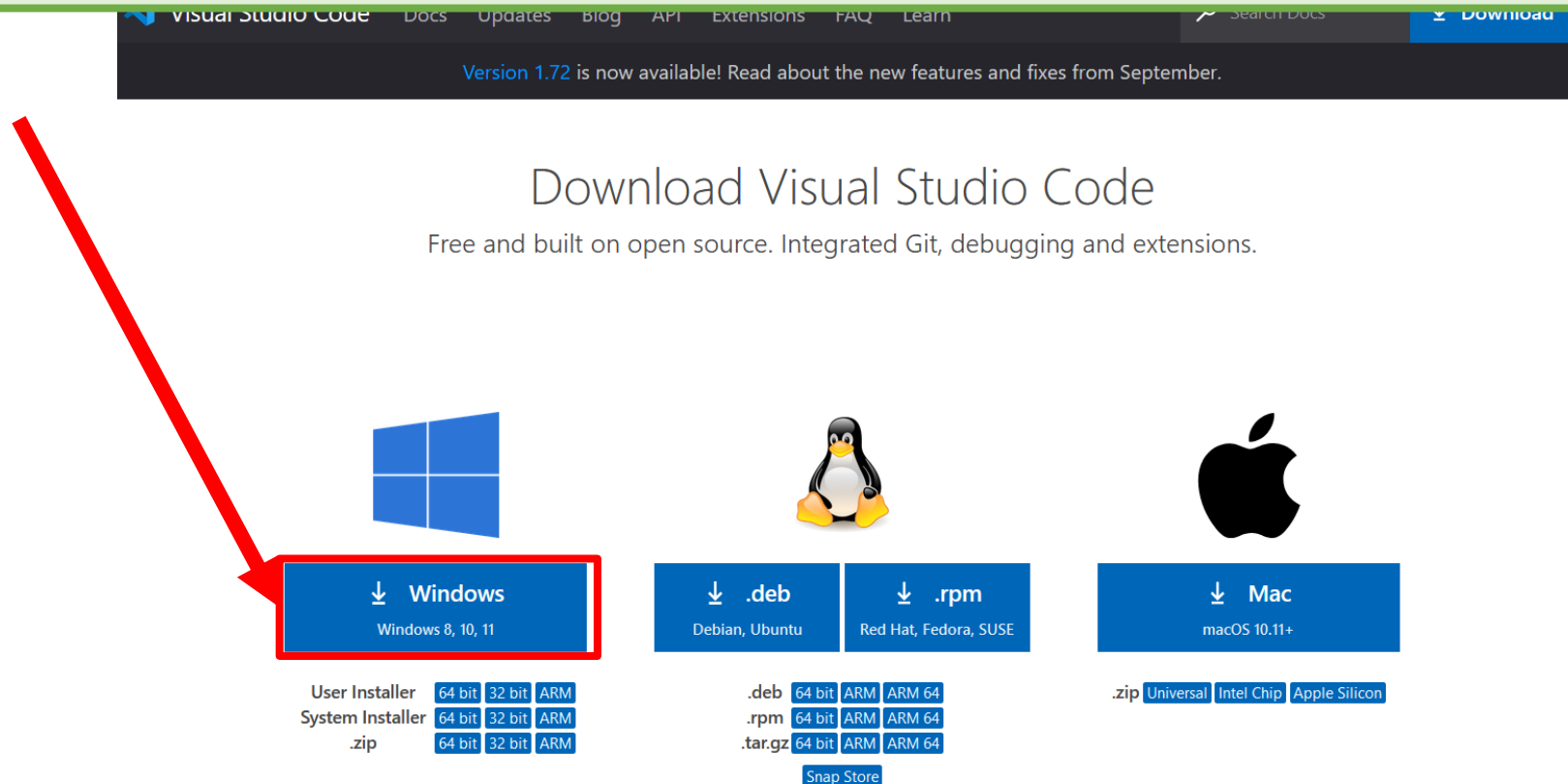


Appendix: Installing VSCode, Step1

Go to Vscode website:

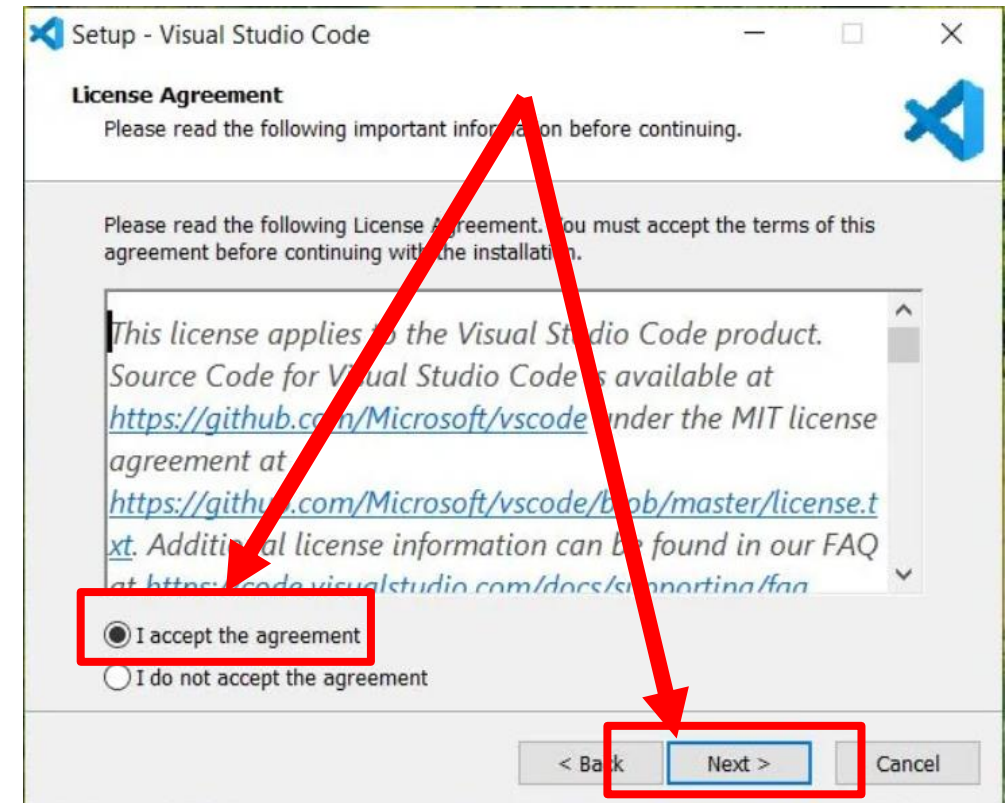
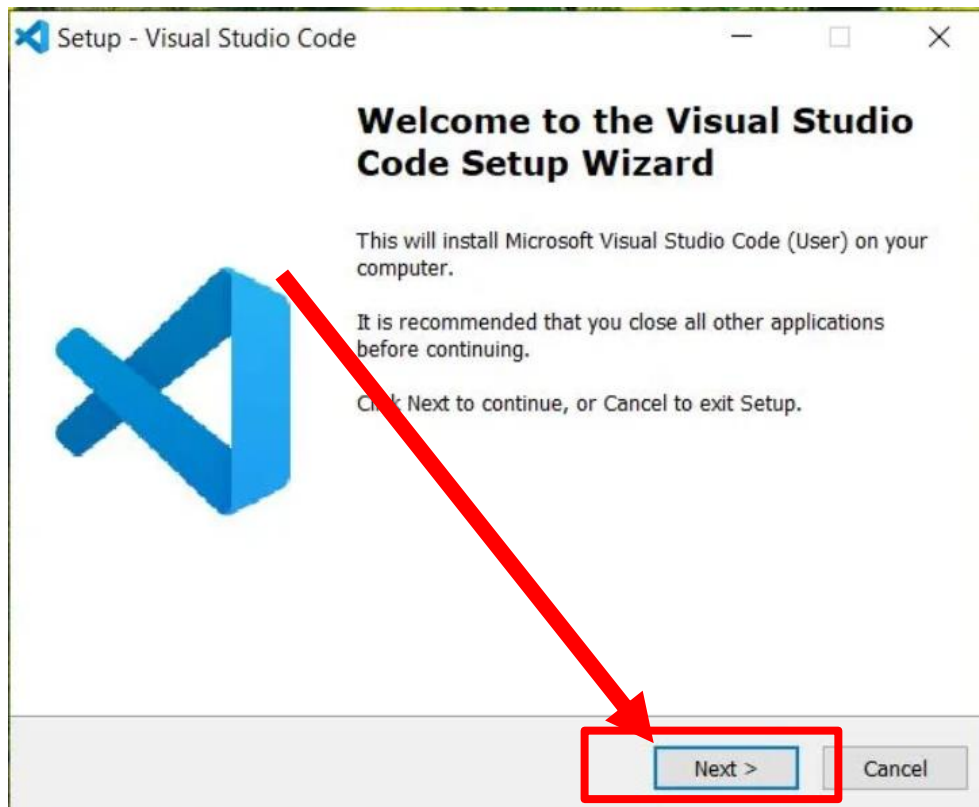
<https://code.visualstudio.com/download>

And download installer based on your OS



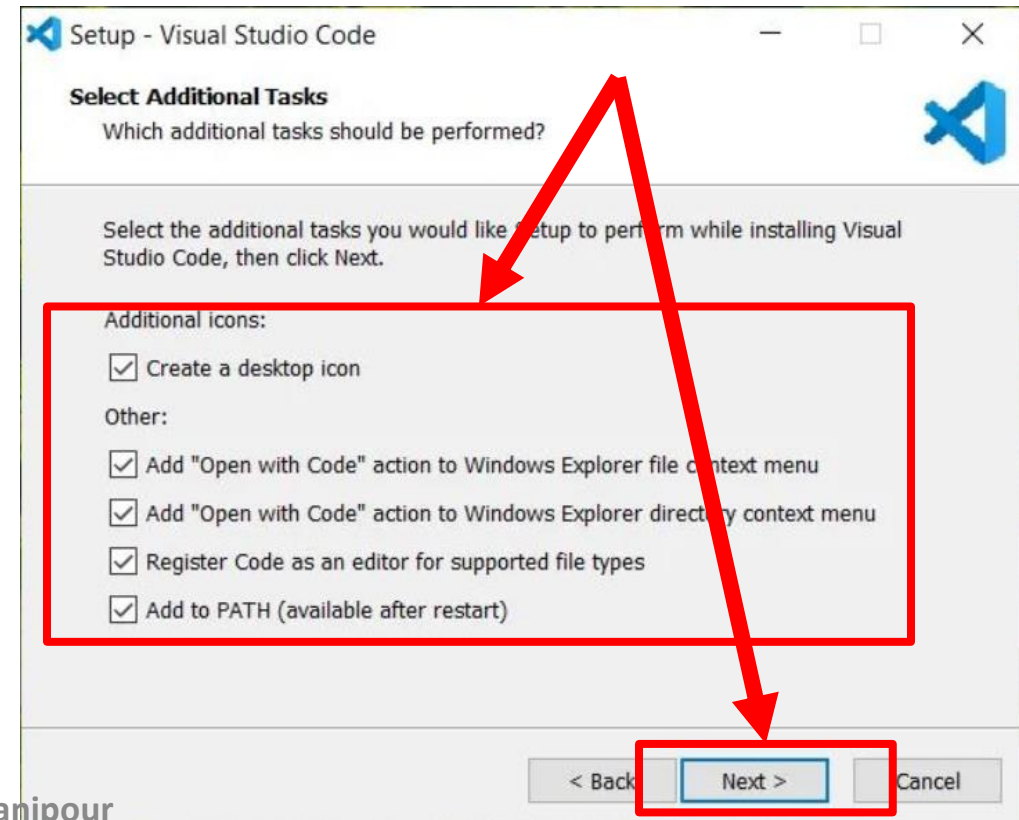
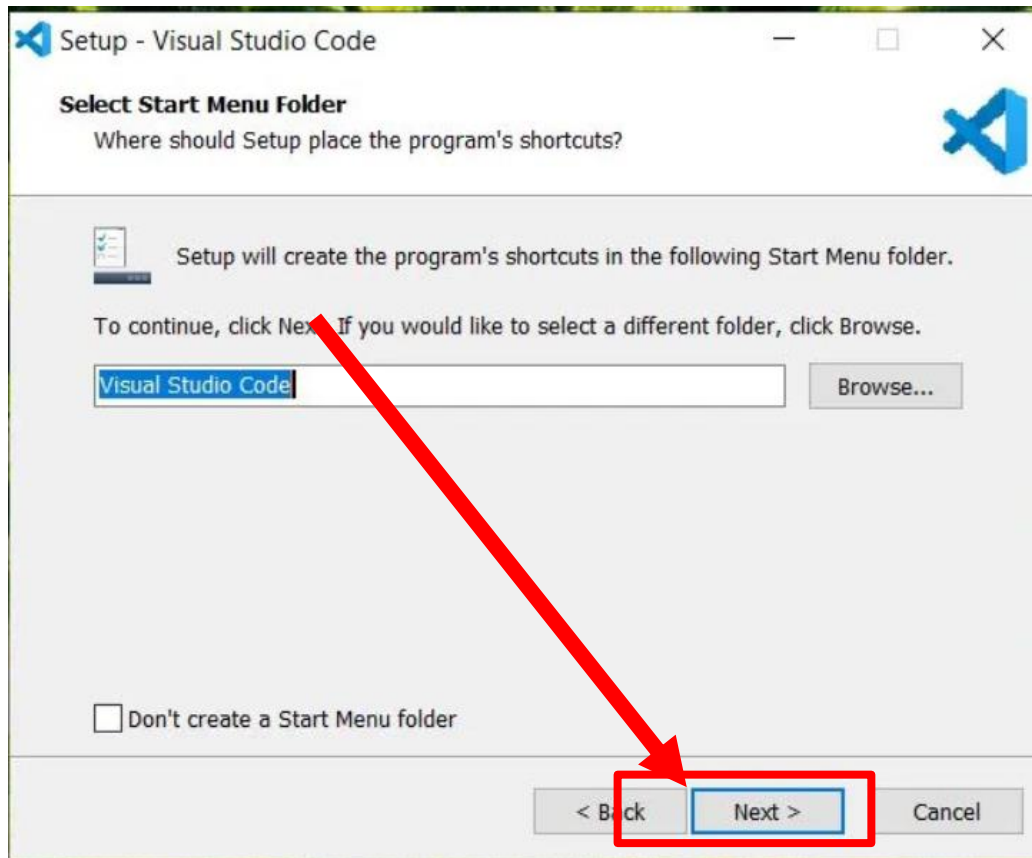
Appendix: Installing VSCode, Step2

Open Installer and you have downloaded in step 1 by double clicking on it and follow installation steps like below



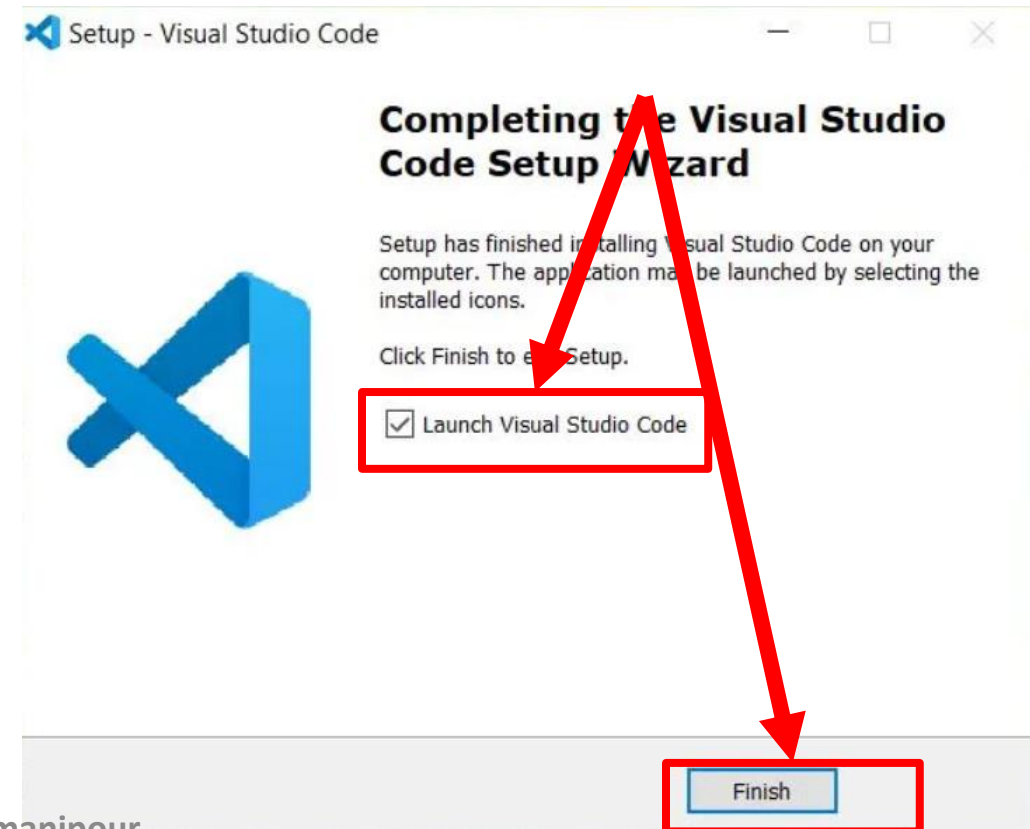
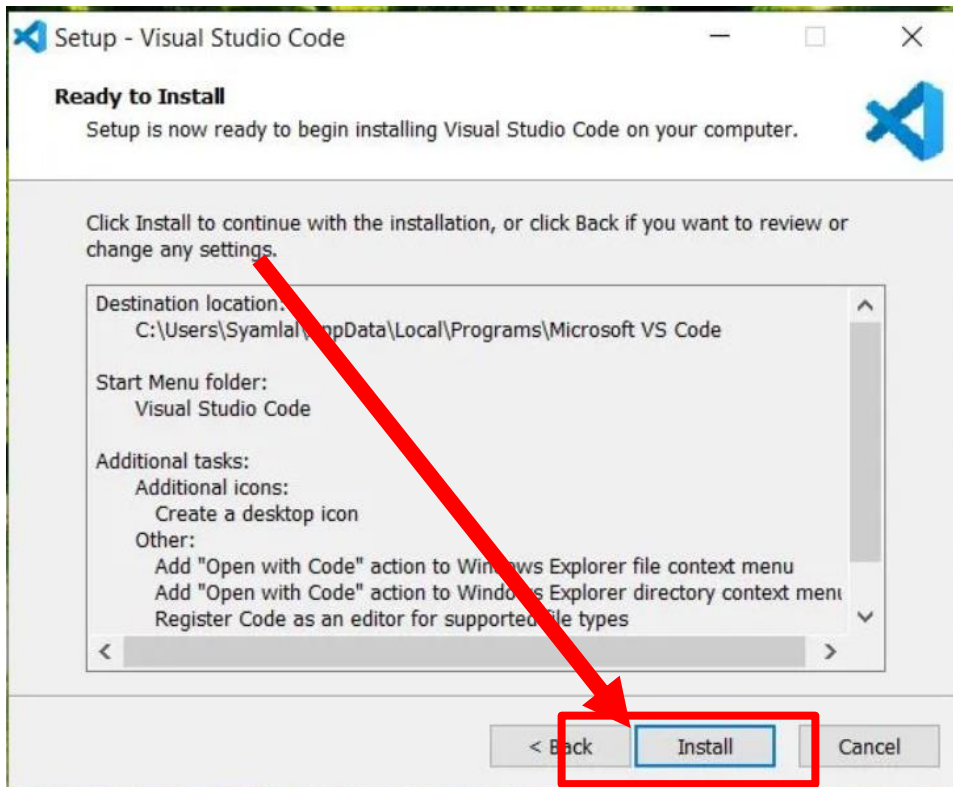
Appendix: Installing VSCode, Step2 ...

Open Installer and you have downloaded in step 1 by double clicking on it
an fallow installation steps like below



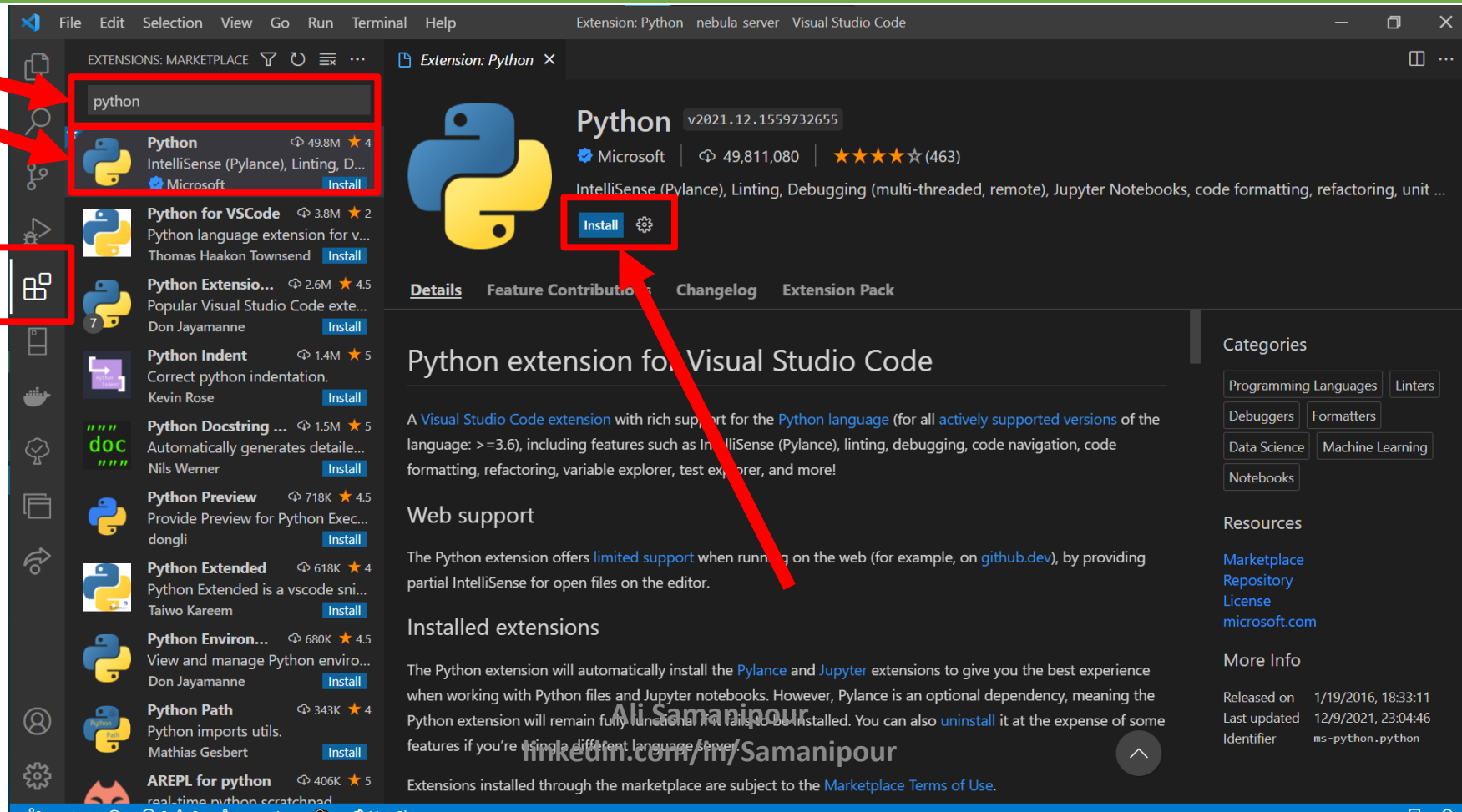
Appendix: Installing VSCode, Step2 ...

Open Installer and you have downloaded in step 1 by double clicking on it and follow installation steps like below



Appendix: Installing python extension on VSCode

Open VSCode go to extensions, search Python and install it



Appendix: Python history

- ❑ Python was created by **Guido Van Rossum**, who was a Dutch person from Netherlands in the Year 1991.
- ❑ Guido Van Rossum created it when he was working in National Research Institute of Mathematics and Computer Science (NRIM&CS) in Netherlands.
- ❑ He thought to create a scripting language as a “Hobby” in Christmas break in 1980. He studied all the languages like ABC (All Basic Code), C, C++, Modula-3, Smalltalk, Algol-68 and Unix Shell and collected best features.
- ❑ He started implementing it from 1989 and finished and released first working version of Python in 1991.
- ❑ He named it as “Python”, being a big fan of “Monty Python’s Flying Circus” comedy show broadcasted in BBC from 1969 to 1974.

Appendix: Python Versions

□ Python 1.0 (1994)

- Python 1.5(1997)
- Python 1.6 (2000)

□ Python 2.0 (2000)

- Python 2.1 (2001)
- Python 2.2 (2001)
- Python 2.3 (2003)
- Python 2.4 (2004)
- Python 2.5 (2006)
- Python 2.6 (2008)
- Python 2.7 (2010) *-Currently we are using this version*

□ Python 3.0 (2008)

- Python 3.1(2009)
- Python 3.2(2011)
- Python 3.3 (2012)
- Python 3.4(2014)

□ **Note:** Python 3.0 was designed to rectify the problems in the older versions. It has

lot many new features when compared to python. Ali Samanipour
[linkedin.com/in/Samanipour](https://www.linkedin.com/in/Samanipour)

Accessing Resources (Codes, Slides, etc.)



github.com/Samanipour



t.me/SamaniGroup

References

- **Automate The Boring Stuff With Python, 2nd Edition: Practical Programming For Total Beginners**
- **Python Crash Course, 2nd Edition: A Hands-On, Project-Based Introduction To Programming**
- **Supercharged Python: Take Your Code to the Next Level**
- **Python Tricks: A Buffet of Awesome Python Features**
- **Learn Python the Hard Way: A Very Simple Introduction to the Terrifyingly Beautiful World of Computers and Code**
- **Starting Out with Python, Tony Gaddis, Global Edition**