

Introduction

Distributed Network and Systems Laboratory,
Chonnam National University
Shivani Kolekar



Welcome !

Who Am I

✓ Hello Everyone. I am Shivani Kolekar.

(2017~2020)

✓ I studied computer science as Engineering major from Shivaji University, Maharashtra.

(2021~present)

✓ Currently I am a 3rd Year MS-Ph.D Student at

- Distributed Networks and Systems Laboratory,
- Department of AI Convergence,
- Chonnam National University, South Korea.

Email – shivani.kolekar@gmail.com



Course Goals

- What is Python and why its important
- Understanding basics of Python programming
- Learn to design and implement computer programs to solve problems

Some Thoughts about Programming

- ✓ “The only way to learn a new programming language is by writing programs in it.”
–D. Ritchie (developed C Programming language)
- ✓ “Computers are good at following instructions, but not at reading your mind.”
–D. Knuth (developed TeX typesetting language)

What is Python ?

- ✓ Python is a high-level programming language developed by Guido van Rossum in the Netherlands in the late 1980s. It was released in 1991.
- ✓ Python has twice received recognition as the language with the largest growth in popularity for the year (2007, 2010).
- ✓ Python is a programming language that lets you work quickly and integrate systems more efficiently.

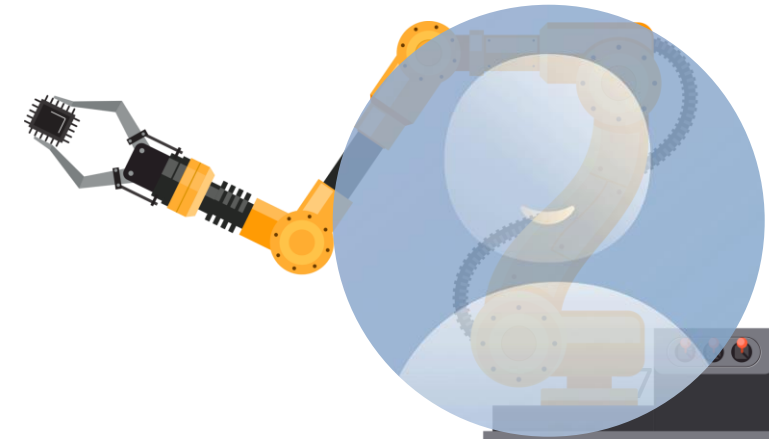
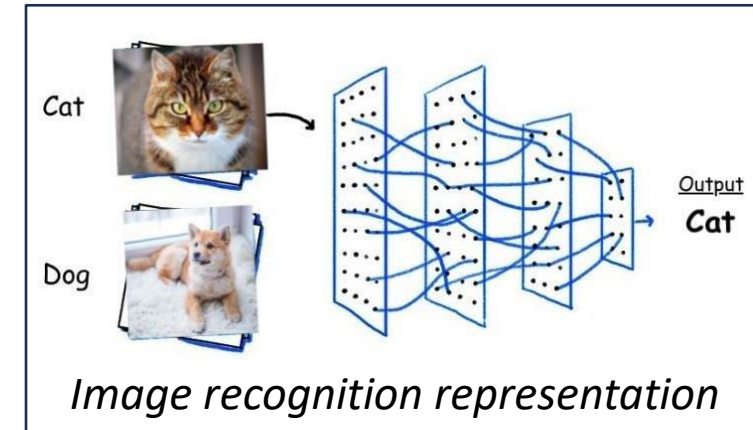
Python and AI

✓ What is AI?:

- ✓ AI, or Artificial Intelligence, is like teaching a computer to think, learn, and make decisions similar to a human.
- ✓ Examples: Siri on iPhones, recommendation systems on Netflix, or playing games like Chess against a computer.

Why Python for AI?:

- ✓ Python is a popular programming language that's easy to learn and has powerful capabilities for AI.
- ✓ It has a lot of libraries (collections of pre-written code) that make building AI projects easier and faster.
- ✓ Ex. TensorFlow for creating neural networks, NLTK for processing human language etc.



Python powering AI

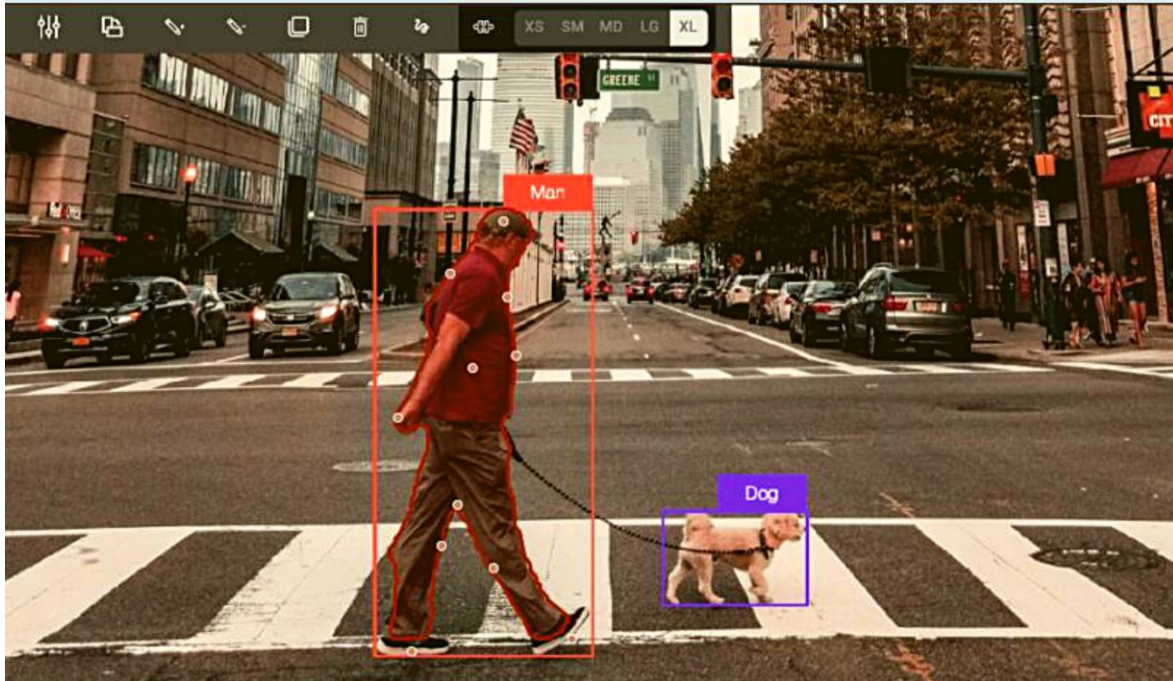
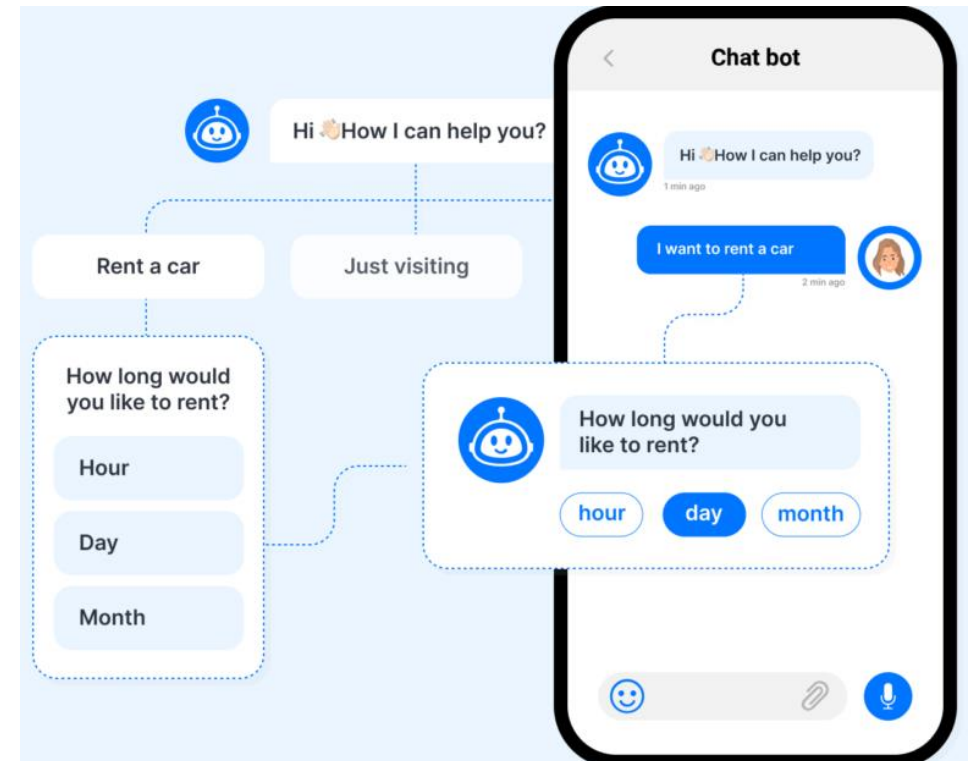


Image recognition



Game AI

Chatbot Applications



Significant Applications of Python in real life

- ✓ Google (advertising services, search engine, various google products etc.)
- ✓ Youtube (management of user activities and data, comments display etc.)
- ✓ Instagram (Django python framework)
- ✓ Spotify (data analysis and backend services)
- ✓ Netflix (server-side data analysis, security, automation, risk management etc.)
- ✓ Reddit (web development)
- ✓ NASA (data analysis, scientific computing, mission critical projects etc.)
- ✓ BitTorrent (large-scale peer-to-peer file sharing systems)



Compilers

- Because the CPU understands only machine language instructions, programs that are written in a high-level language must be translated into machine language. Depending on the language in which a program has been written, the programmer will use either a compiler or an interpreter to make the translation.

- Compiler :**

Translate to high level lang program

- 1 The compiler is used to translate the high-level language program to a machine language program.

High-level language program

```
print ("Hello  
Earthling")  
  
and so forth...
```



Compiler



Machine language program

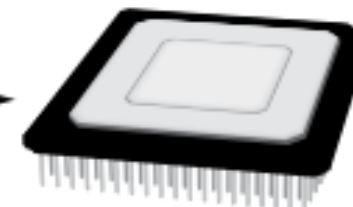
```
10100001  
10111000  
10011110  
  
and so forth...
```

Execute translated program

- 2 The machine language program can be executed at any time, without using the compiler.

Machine language program

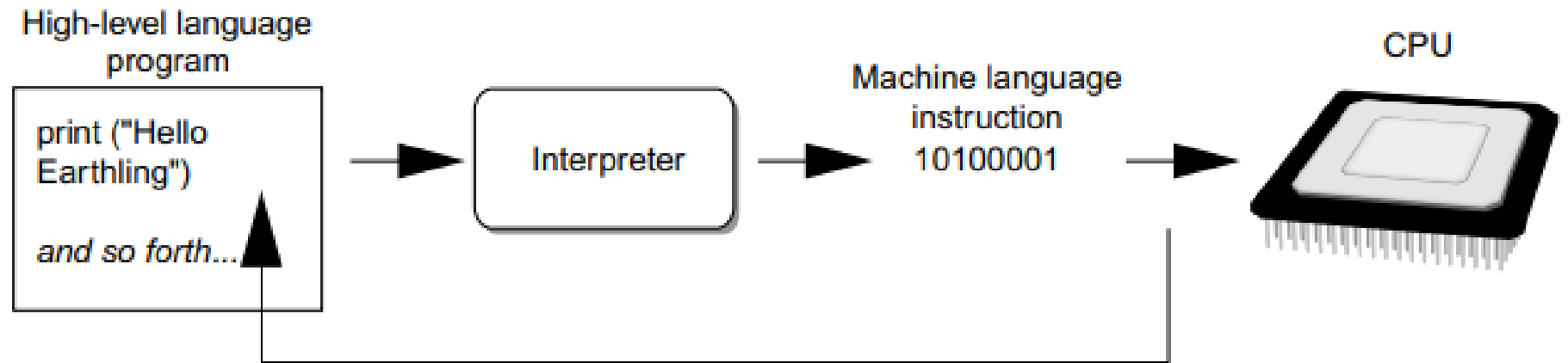
```
10100001  
10111000  
10011110  
  
and so forth...
```



Interpreter

- The Python language uses an **interpreter**, which 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. This process repeats for every instruction in the program.

Interpreter:



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.

✓ You can use the interpreter in two modes: interactive mode and script mode.

In **interactive mode**, the interpreter waits for you to type Python statements on the keyboard. Once you type a statement, the interpreter executes it and then waits for you to type another statement.

In **script mode**, the interpreter reads the contents of a file that contains Python statements. Such a file is known as a Python program or a Python script. The interpreter executes each statement in the Python program as it reads it

To install Python on your personal computer / laptop, you can download it for free at:

www.python.org/downloads

- ✓ There are two major versions: Python 2 and Python 3. Python 3 is newer and *is not backward compatible with Python 2*. Make sure you're running Python 3.8.
- ✓ It's available for Windows, Mac OS, Linux.
- ✓ If you have a Mac, it *may* already be pre-installed.
- ✓ It should already be available on most computers on campus. It comes with an editor and user interface called IDLE.
- ✓ I strongly recommend downloading and installing the PyCharm, Educational version, IDE.

What is Pycharm

- PyCharm is a software program, known as an Integrated Development Environment (IDE), that's specifically designed for writing, editing, and organizing Python code.
- Think of it as a specialized tool that helps you write and manage your Python projects more efficiently.

Why use it?

- ✓ Easy to Write Code
- ✓ Debugging Tools
- ✓ Run and Test Your Code Easily
- ✓ Manage Projects

Install Pycharm

Assuming you use windows,
Install PyCharm (cross platform IDE (integrated development env.))

Link:

<https://www.jetbrains.com/pycharm/download/?section=windows>

Install Community edition

IDLE Installation : PyCharm

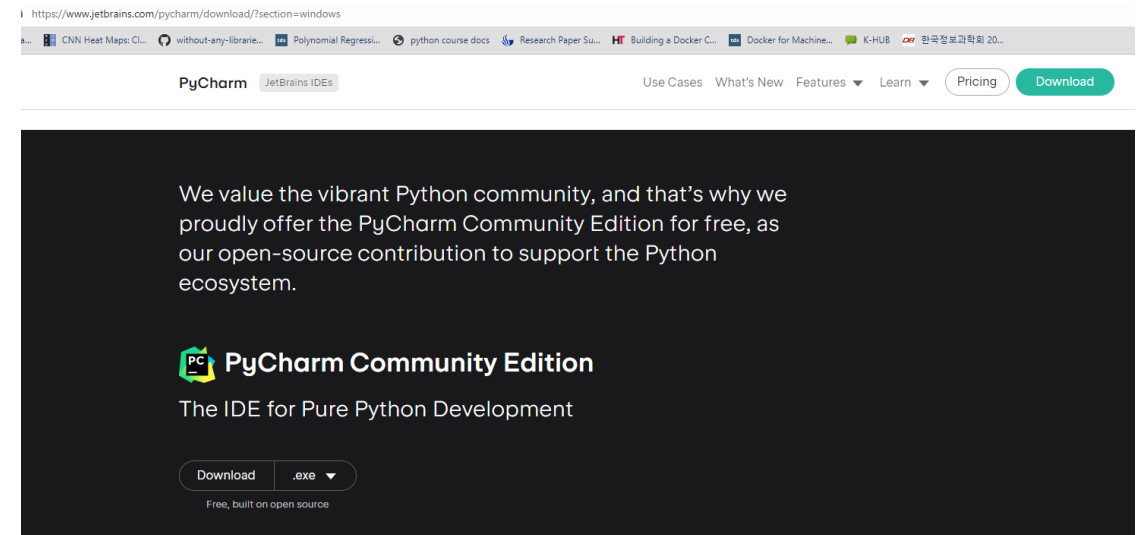
Installation steps:

1) <https://www.jetbrains.com/pycharm/download/?section=windows>

Download Pycharm community edition from this link

2) Installation guide can also be referred at :

<https://www.jetbrains.com/help/pycharm/installation-guide.html>



Creating a virtual environment for our project

- 1) After installing PyCharm, create a directory where you will save the python basics documents (scripts, environment file etc.)
- 2) Open command Prompt on windows and create a virtual environment in the above created directory
Follow the steps below:

```
D:\>cd PYTHON_BASICS/python_env  
  
D:\PYTHON_BASICS\python_env>python -m venv myenv  
  
D:\PYTHON_BASICS\python_env>myenv\Scripts\activate  
  
(myenv) D:\PYTHON_BASICS\python_env>
```

Creating a virtual environment for our project

3) From PyCharm, in settings, Select the 'python.exe' file from the folder

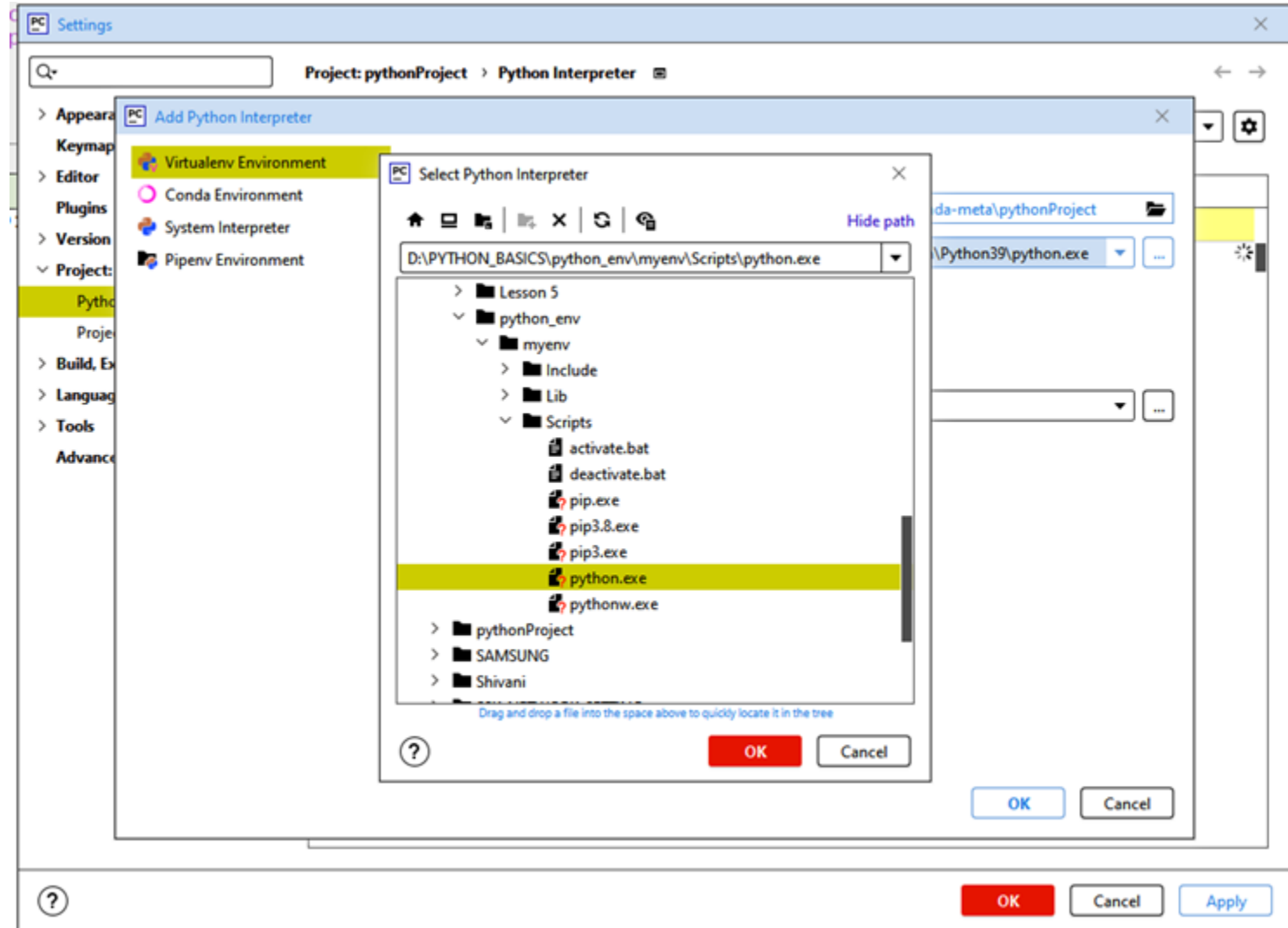
4) It will create a virtual environment which can be observed at the right corner of IDE.

Which looks something like this :



Windows.

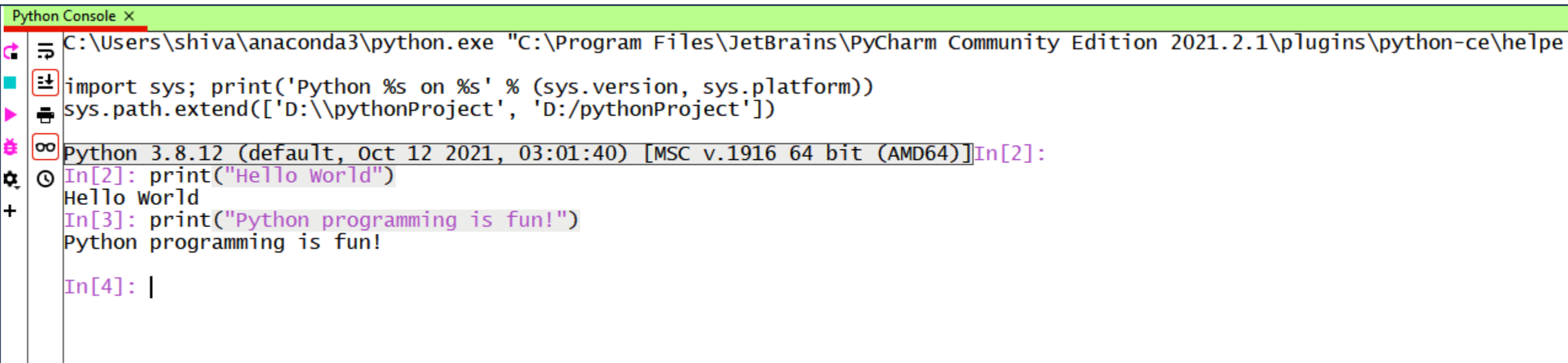
Event Log
Python 3.8 (myenv)



A simple Python Program: Interactive Mode

This illustrates using Python in interactive mode from the command line.

Pycharm → 'Python console' mode



The screenshot shows the PyCharm Python Console window. The title bar is 'Python Console X'. The console output shows the execution of a Python script. The first two lines of the script are `import sys; print('Python %s on %s' % (sys.version, sys.platform))` and `sys.path.extend(['D:\\pythonProject', 'D:/pythonProject'])`. The output of the first line is `Python 3.8.12 (default, Oct 12 2021, 03:01:40) [MSC v.1916 64 bit (AMD64)]`. The second line of the script is `In[2]: print("Hello world")`, and the output is `Hello world`. The third line of the script is `In[3]: print("Python programming is fun!")`, and the output is `Python programming is fun!`. The fourth line of the script is `In[4]: |`, indicating that the user is currently in the interactive mode.

```
Python Console X
C:\Users\shiva\anaconda3\python.exe "C:\Program Files\JetBrains\PyCharm Community Edition 2021.2.1\plugins\python-ce\help
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.extend(['D:\\pythonProject', 'D:/pythonProject'])

Python 3.8.12 (default, Oct 12 2021, 03:01:40) [MSC v.1916 64 bit (AMD64)]In[2]:
In[2]: print("Hello world")
Hello world
In[3]: print("Python programming is fun!")
Python programming is fun!

In[4]: |
```

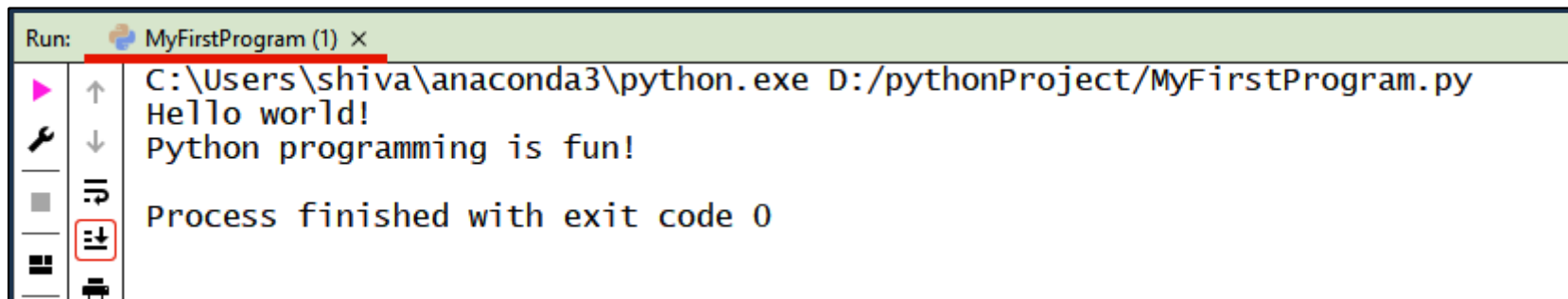
A simple Python Program: Script Mode

- Here's the “same” program as I'd be more likely to write it.
- Enter the following text using a text editor into a file called, say, `MyFirstProgram.py`.
- This is called script mode.

```
1 def main():
2
3     ''' Display the messages'''
4
5     print("Hello world!")
6     print("Python programming is fun!")
7
8     main()
```

- ✓ `def main()` is a function in Python. Think of a function as a mini-program inside your program that can do a specific task.
- ✓ `def` is a keyword that tells Python you are about to define a function.
- ✓ `main` is the name of this function. It's a common convention to use `main` as the entry point of a program.
- ✓ Everything indented under `def main():` is part of the main function.

- Execute the python script and check the result in Terminal as follows:



The screenshot shows a terminal window titled "Run: MyFirstProgram (1) x". The command executed is `C:\Users\shiva\anaconda3\python.exe D:/pythonProject/MyFirstProgram.py`. The output of the script is displayed on the next two lines: `Hello world!` and `Python programming is fun!`. The final line of the terminal output is `Process finished with exit code 0`. On the left side of the terminal, there is a vertical toolbar with icons for running, stopping, and other actions. The "Run" icon (a green play button) is highlighted with a red box.

Computer memory

- ✓ 1 bit -> a single 0 or 1 (binary)
- ✓ 1 byte = 8 bits
- ✓ A typical laptop or desktop circa 2023
- ✓ ... has 4 to 32 Gigabytes of RAM, also known as main memory.
- ✓ 1 Gigabyte -> 1 billion bytes
- ✓ The programs that are running store their instructions and data (typically) in the RAM
- ✓ ... have 100s of Gigabytes up to several Terabytes (trillions of bytes) in secondary storage
 - . Long term storage of data, files
- ✓ Typically spinning disks (Hard disk drives - HDD) or solid state drives (SSD).



Framework of a simple Python program

Define your program in file
filename.py:

```
def main ():  
    Python statement  
    Python statement  
    Python statement  
    ...  
    Python statement  
    Python statement  
    Python statement  
  
main ()
```

To run it:

```
> python filename.py
```

Defining a function called main.

These are the instructions that make up your program. *Indent all of them the same amount (usually 4 spaces).*

This says to execute the function main.

This submits your program in file_name.py to the Python interpreter.

Errors : Syntax error

- ✓ We will encounter three types of errors when developing our Python program.
1. **syntax errors:** these are ill-formed Python and caught by the interpreter prior to executing your code.

```
>>> 3 = x
      File "<stdin>", line 1
      SyntaxError: can't assign to
      literal
```

These are typically the easiest to find and fix.

Errors : Runtime error

- ✓ **runtime errors:** you try something illegal while your code is executing

```
>>> x = 0
>>> y = 3
>>> y / x
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero
```


- ✓ **logic errors:** your program runs but returns an incorrect result

```
def calculate_average(num1, num2, num3):  
    average = (num1 + num2 + num3) / 2 # Logic error here  
    return average  
  
# Testing the function  
result = calculate_average(10, 20, 30)  
print("The average is:", result)
```

This program is syntactically fine and runs without error. But it probably doesn't do what the programmer intended.

How would you fix it?

Logic errors are often the hardest errors to find and fix.

