



Learning Python

»»» Beginner Level

Python Run

► Student's Book



Level

1



Rana Dajani



PythonRun - Beginner Level



Published by **LKD Educational Resources 2022**

Amman - Jordan

Tel: +962 6 5374141

Fax: +962 6 5516404

P.O.Box: 851346

Email: info@lkd.com.jo

Website: www.lkd.com.jo

 **Author**

Rana Dajani

ISBN: 978-9923-781-06-7

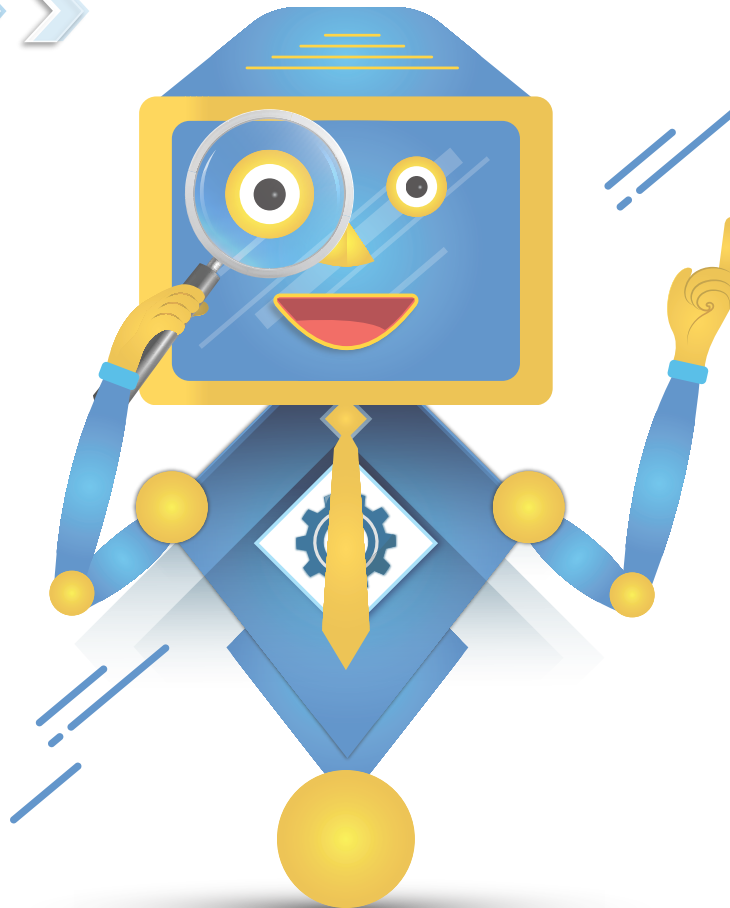


Learning Python Beginner Level Python Run



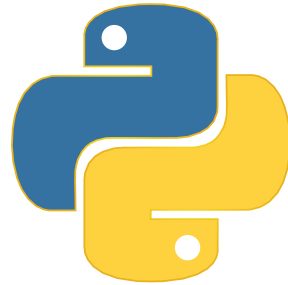
Level

1



A guide to learning Python programming language

Introduction

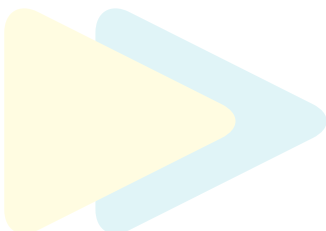


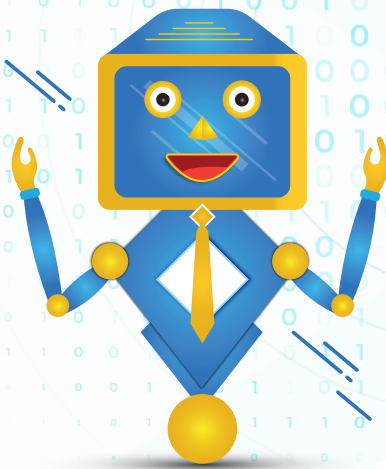
This book is the beginner level book in the *PythonRun* series. It is so exciting that you are learning to code from this book!

Coding is considered the new language in our current post - digital era. As our lives become more and more reliant on technology, there is an increasing demand for coding skills. Thrive to improve by learning this new literacy and you will find that programming allows you to unleash your creativity and apply your knowledge in your own personal field of passion.

Like anything you try for the first time, it is always best to start with the basics. As you go through this book, try each of the examples, so you can see how they work.

There are also programming problems at the end of most chapters for you to try, which will help improve your programming skills. Remember that the better you understand the basics, the easier it will be to understand more complicated ideas later on.





**START CODING
NOW!**



Table of Contents

Unit 1 >> Introduction to Programming

1.1	What is Programming.....	10
1.2	Python vs Other Languages.....	12
1.3	Basic Terms and Definitins.....	16
1.4	Python Compilers.....	18

Unit 2 >> Start Programming Here!

2.1	Print() Function.....	22
2.2	Comments and Documentation.....	24
2.3	Programming Flow Cycle.....	26
2.4	Encountering Errors.....	27
2.5	Keywords and Identifiers.....	30

Unit 3 >> Working with Data

3.1	Variables and Naming Rues.....	34
3.2	Datatypes.....	38
3.3	Strings.....	40
3.4	Input() Function.....	43
3.5	Type Casting /Type Conversion.....	46
	Quick Tests	48

Unit 4 >> Operations

4.1	Mathematical Operators.....	54
4.2	Order of Operations.....	55
4.3	Operations and Datatypes.....	57
4.4	Logical Operators.....	58
4.5	In Operator.....	61
	Quick Tests	62

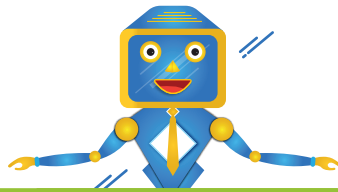


Table of Contents

Unit 5 >> Conditions

5.1 Conditions.....	68
5.2 If Statements.....	69
5.3 If-Else Statements.....	71
5.5 If and Elif Statements.....	73
5.6 Nesting Statements.....	77
Quick Tests	80

Unit 6 >> While On Loop

6.1 Ways to Loop.....	86
6.2 While Loop.....	87
6.3 Incrementing Condition.....	88
6.4 Break Statement.....	92
6.5 Continue Statement.....	94
Quick Tests	96

Unit 7 >> On and On for Loop









7.1 For Loop.....	102
7.2 String For Loop.....	103
7.3 Range() Function.....	106
7.4 Range For Loop.....	107
7.5 Nesting For Loops.....	110
Quick Tests	114

>> Project – Based Assessments



Unit 1

Introduction to Programming

-   What is Programming
-   Python vs Other Languages
-   Basic Terms and Definitions
-   Python Compilers

Introduction to Programming



1.1

What is Programming

Have you ever wondered how computers can make so much happen? How do they perform calculations, show movies and run amazing games? These are all examples of computer programs.



Programming is writing the commands called code for a computer to perform.

In this book, you will not only learn to write a code, but also how to communicate in a community that uses code as its language.



Why learn computer programming?

Programming fosters creativity, reasoning, and problem solving.

The programmer gets the opportunity to create something from nothing and use logic to turn programming constructs into a form that computers can run.



70% of coding jobs are in fields
outside of technology

Programming also develops **computational thinking skills**, which is a problem solving process that involves a number of skills. This includes: formulating problems in a way that enables us to use a computer and other tools to help solve them, as well as logically organizing and analyzing data.

1.2

Python vs Other Languages

Programming languages and computer coding have made our lives simpler. Whether it be automobiles, banks, home appliances, or hospitals, every aspect of our lives depends on codes. No wonder, coding is one of the core skills required by most well-paying jobs today.

There are dozens, even hundreds of programming languages that have different syntax (grammar), different vocabulary and different structures out there, with many similarities and many differences.





Why do so many languages exist?

Different languages are good for different things. You might want to have more control over the little details of how things are run, such as in complex video games or a complicated mathematical function. Or you might be more interested in a language that doesn't force you to think about all those details, if you are more interested in being able to design rapidly.

- ▶ Dozens of programming languages are evaluated for their relative popularity. **Python** ranks as the #5 most-popular language, after **Java**, **C**, **C++**, and **C#**. All four of these languages are low-level languages used a lot in developing highly complex programs, they are static, compiled languages.



Python is one of the top dynamic, scripting languages.

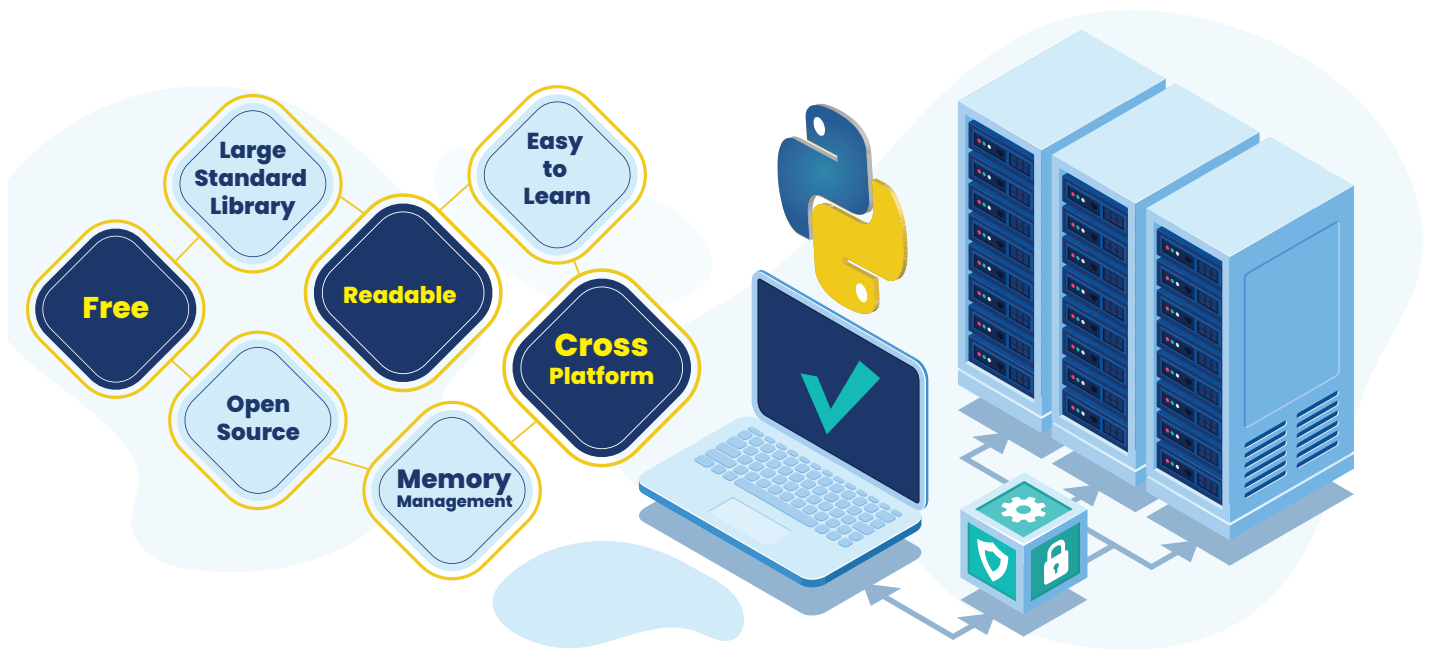
➤ Python is an interpreted, high-level, general-purpose programming language. It was first released in 1991, and it was designed to be highly extensible.

➤ Python is used in many applications such as:



➤ Python is very often used as a first programming language because it requires the least overhead and makes many of the concepts clear.

Features of Python programming language:

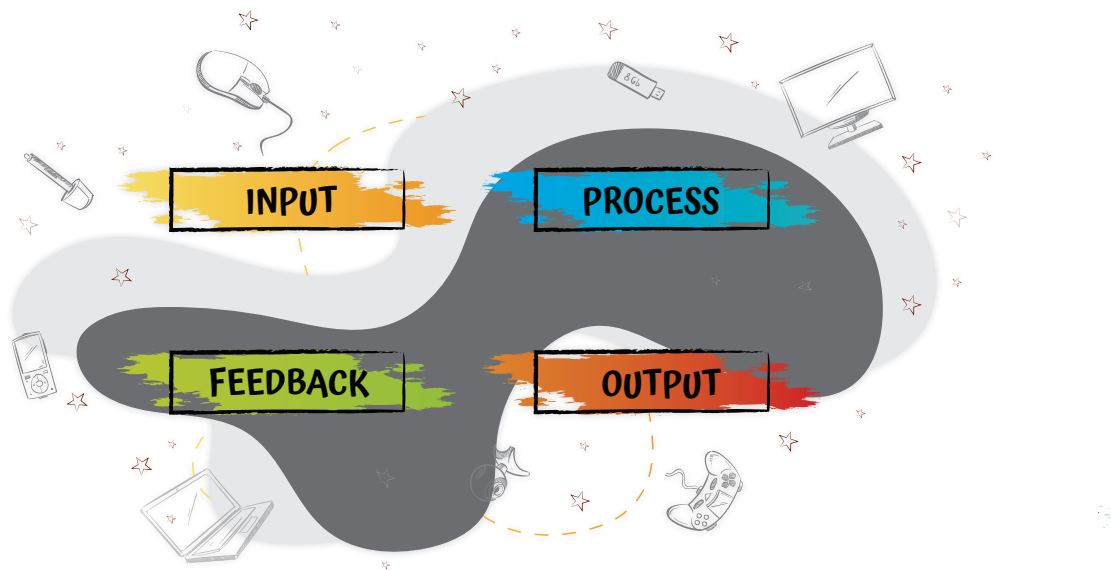


► After learning Python, it is common to go on to learning Java, C, or another languages. The principles of programming are pretty general; it is a lot easier to learn your second programming language than your first.

1.3

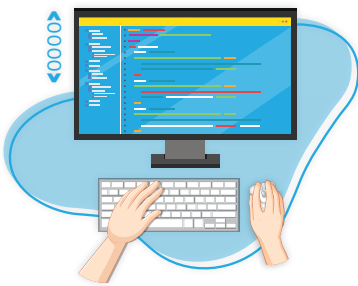
Basic terms & Definitions

In order to talk about programming, there are some basic terms we need to know.



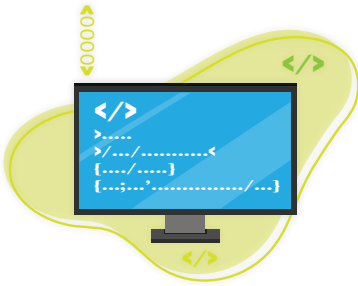
There are four basic elements to the systems' model:

- ▶ **Inputs** represent the basic materials or resources that will be transformed to the output. The output of some portion of code which we write, may become the input into some other portion of code.
- ▶ **Process** represents the operations that occur to transform the inputs to the desired outputs.
- ▶ **Output** represents the outcome returned by the computer. Output does not necessarily have to show on screen, it could also be sound for example.
- ▶ **Feedback** is the element of control. If the desired output is not achieved, the process and/or the inputs must be adjusted to achieve the desired result.



Data Input

Is data that is fed into a program for it to operate on. Our code is responsible for translating user input, using keyboard, into that observable output. But we can have lots of other different kinds of input (e.g. files and hyperlinks).



Code

Is a command given to a computer in order to perform a task. A line of code is generally a single command, which is the smallest unit we are interested in dealing with at this stage.



Compile Program

A program is like a house, made up of individual bricks called code. A collection of lines of code serve one or more overall functions, making a program.

Compiling is to translate human-readable computer code into instructions computers can execute. In the programming flow, this functions as a check on the code the user has written to make sure it makes sense to the computer.



Console

Is an output medium for a program to show exclusively text-based output.



Graphical User Interface (GUI)

Is an output medium that uses more than just text, like forms, buttons, tabs, and more. Most programs are graphical user interfaces.

>>> As we learn to program, we will start with console-based programs and work up to graphical programs <<<

66

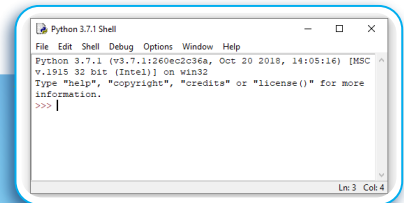
1.4

Python Compilers

Using a proper editor to write code can help a lot while programming or while working on actual projects.



Python IDLE



Install Python, and set up a Python shell program **IDLE** (Integrated Development and Learning Environment): This is an interactive interpreter that executes the code line by line.

<https://realpython.com/python-idle/>



Mu editor



Mu editor is a very simple and beginner friendly editor for anyone starting with Python.

<https://codewith.mu/>



Alternative option: Online Python compiler (to code online in your browser)

<https://www.programiz.com/python-programming/online-compiler/>



The Python shell is a basic Read-Eval-Print Loop (REPL). It reads a Python statement, evaluates the result of that statement, prints the result on the screen. and then, it loops back to read the next statement.

The three greater-than signs (`>>>`) are called the prompt. This is where we enter code lines.

Every programmer needs to be able to edit and save text files. Python programs are files with the **.py** extension that contain lines of Python code. Python compilers gives you the ability to create and edit these files with ease.

To save a new program in Python IDLE

- Open IDLE and choose File>New Window. An empty window will appear, with “Untitled*” in the menu bar.
- Choose File>Save. When prompted for a file name, enter ‘name_of_your_project’.py, and save the file to your desktop.
- Choose Run>Run Module (F5).

To save a new program in Mu editor

- Write any code you want to execute.
- Save it with a proper filename.
- Click on the Run button to execute the code.

When naming your file for saving it you should abide by the variables naming rules (explained on page 35).

