

Raspberry Pi for Beginners

LESSON 5

MAKERHOUSE

EMPOWERING MAKERS

Table of Contents

[1 Introduction 3](#_Toc13206020)

[1.1 Getting Started 3](#_Toc13206021)

[2 Python Basics 4](#_Toc13206022)

[2.1 Variables, and Values 4](#_Toc13206023)

[2.1.1 Types 4](#_Toc13206024)

[2.1.2 Numbers Operation 5](#_Toc13206025)

[2.1.3 String Operation 6](#_Toc13206026)

[2.1.4 Boolean 6](#_Toc13206027)

[2.1.5 Test Your Knowledge 6](#_Toc13206028)

[3 Storing Value in Structure 7](#_Toc13206029)

[3.1 Sequences 7](#_Toc13206030)

[3.1.1 Operation 7](#_Toc13206031)

[3.2 Dictionaries 8](#_Toc13206032)

[4 HEADING 1 8](#_Toc13206033)

[4.1 Heading 2 8](#_Toc13206034)

[4.2 Heading 2 8](#_Toc13206035)

[5 HEADING 1 9](#_Toc13206036)

[5.1 Heading 2 9](#_Toc13206037)

[5.1.1 Heading 3 9](#_Toc13206038)

[5.1.2 Heading 3 9](#_Toc13206039)

[5.2 Heading 2 9](#_Toc13206040)

[6 HEADING 1 9](#_Toc13206041)

[6.1 Heading 2 9](#_Toc13206042)

[6.2 Heading 2 9](#_Toc13206043)

[7 HEADING 1 10](#_Toc13206044)

[7.1 Heading 2 10](#_Toc13206045)

[7.1.1 Heading 3 10](#_Toc13206046)

[7.1.2 Heading 3 10](#_Toc13206047)

[7.2 Heading 2 10](#_Toc13206048)

[8 HEADING 1 10](#_Toc13206049)

[8.1 Heading 2 10](#_Toc13206050)

[8.2 Heading 2 10](#_Toc13206051)

[9 Section 11](#_Toc13206052)

[9.1 Heading 11](#_Toc13206053)

[9.1.1 asdfjlksfjd alkf ja;skdfj a;lkdj 11](#_Toc13206054)

[10 fsd sdf sfgd sd 11](#_Toc13206055)

Getting Started with Python Basics

# Introduction

We have been introduced with Python as early in Lesson 3 -- Python programming with Turtle. Noticed that, Python is easy to learn and use – modest, little time and effort to start, plus the syntax is readable – great for first programming language to learn.

***Did you agree on this after first experienced -- controlling turtle to do whatever you want in lesson 3 -- with Python?***

Moreover, Python is broadly adopted and supported, where we can see it very popular and widely used – building professional quality software – standalone or web services.

A close up of a logo

Description automatically generated

Figure 1: Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace.

And for today the lesson will be different from Lesson 3. If in Lesson 3 we directly use the Turtle module in Python to do drawings based on arts, computer sciences and maths but on today lesson, we will be focused on Basics of Python language – which we didn’t detailed explained in Lesson 3.

***Are you ready? Let’s go!***

## Getting Started

***Still remember Thonny and Terminal?***

Even though Python don’t specifically require special integrated development environment (IDE), we will be using Thonny since it is pre-installed in our Raspbian OS. We will continue what we left in Lesson 3. Additionally, Terminal (ltxterminal) will be used too – to executes the Python scripts etc.

***With Great Power, Comes Great Responsibility***!

# Python Basics

This lesson will be emphasized on the basics only in which advanced python will be covering in later lesson – each step you take reveals new horizon, you have taken the first step today!

## Variables, and Values

**Variables** is very useful in Python (every programming language actually). Its store data which in turn will be used later and elsewhere.

***Python executes instruction in orderly – variables need to be declared before can be used!***

Your variable can have almost any name, however, shall start with letter or underscore. Importantly, it cannot have same name as Python reserved keywords.

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

**Table 1: Reserved words in Python -- cannot use a keyword as variable name, function name or any other identifier.**

Here is how we write (declaring) the variables with specific types of values:

1. variable\_name = 12
2. \_variable\_name = “twelve”
3. variableName = 4.5

***Recommendation: use lowercase and separated with underscore – easy to maintain plus help others to understand your code well.***

### Types

Python variables can hold any types of values – **int** (integer)**, float** (floating numbers)**, str** (string)**, bool** (Boolean: True/False). There is – **type(**variable**)** function – available to check it. Here if use **type** function as in previous variables name declaration:

1. <type ‘int’>
2. <type ‘str’>
3. <type ‘float’>

### Numbers Operation

Two main operation exists for numbers – **integer (int)** and **float** **(float)**-- are **comparison** and **numerical**. Both are tabulated as

|  |  |  |  |
| --- | --- | --- | --- |
| No | Operation | Description | Example |
| 1 | < | Less than | 3<2 → False |
| 2 | > | Greater than | 3>2 → True |
| 3 | == | Equal | 3==3 → True |
| 4 | <= | Less than or equal | 3<=3 → True |
| 5 | >= | Greater than or equal | 3>=4 → False |
| 6 | != | Not Equal | 3!=4 → True |

1. **Comparison Operation**

|  |  |  |  |
| --- | --- | --- | --- |
| No | Operation | Description | Example |
| 1 | + | Addition | 2 + 2 → 4 |
| 2 | - | Subtraction | 3 − 2 → 1 |
| 3 | \* | Multiplication | 2 \* 3 → 6 |
| 4 | / | Division | 10/2 → 5 |
| 5 | % | Modulus – get the remainders | 5%2 → 1 |
| 6 | \*\* | Power | 4 \*\* 2 → 16 |
| 7 | int(variable) | Change to integer type format | int(3.2) → 3 |
| 8 | float(variable) | Change to float type format | float(2) → 2.0 |

1. **Numerical Operation**

**Table 2: Numbers type of operation – comparison and numerical**

### String Operation

It easy to recognize a text – **string (str)** -- in Python which normally has a double quotes mark.

|  |  |  |  |
| --- | --- | --- | --- |
| No | Operation | Description | Example |
| 1 | string[x] | Get the xth character (starts from 0th) | "abcde"[1] → "b" |
| 2 | string[x:y] | Get all the characters from the xth to the yth | "abcde"[1:3] → "bc" |
| 3 | string[:y] | Get every character up until the yth | "abcde"[:3] → "abc" |
| 4 | string[x:] | Get every character from the xth to the end | "abcde"[3:] → "de" |
| 5 | len[string] | Return the length of the string | len(“abcde”) → 5 |
| 6 | string + string | Join two strings together | "abc" + "def" → "abcdef" |

**Table 2: String type of operation**

### Boolean

**True** and **False** -- both are the simplest types of data in Python. Identical like string but It always has the capital letter but with no quote’s marks. Loops are where it often been found and used.

### Test Your Knowledge

With the brief introduction of variables, values and type with their operation:

1. **int**("two")
2. **print**(**str**(3+3) + "3")
3. **type**(3=3)
4. "4" == 4
5. "Python"[4]
6. (3 > 2) **or** (2 > 3)
7. **not** “True”
8. 2345[2]
9. **str**((**not** **True**) **and** (**not** **False**))
10. 10 **%** 3

***\*\* using shell available in Thonny Python IDE***

# Storing Value in Structure

There are two ways for storing Python data – numbers, string, Boolean – which are:

1. Sequences
2. Dictionaries

## Sequences

Sequence is a method of storing Python data by an **indexing** and there are list and tuple. Here an example:

1. list = [1, 2, 3, 4, 5, 6, a, b, c, d, e]
2. tuple = (1, 2, 3, 4, 5, 6, a, b, c, d, e)

Clearly, list – square bracket, tuple – round bracket. Retrieving or updating data can be done by using index.

**Retrieving:**

1. list[2] 🡪 3 **same with** tuple[2] 🡪 3
2. list[0] 🡪 1 **same with** tuple[0] 🡪 1

**Updating:**

1. list[2] = 66 🡪 list = [1, 2, 66, 4, 5, 6, a, b, c, d, e]
2. list[0] = 77 🡪 list = [77, 2, 66, 4, 5, 6, a, b, c, d, e]

However, for tuple updating is different – can’t be done using indexing – need to overwrite all the tuple sequence lot. Or, we can simply understand by re-declaring the tuple sequences.

***\*\*String operation can be applied in list***

### Operation

***We will be using list = [1, 2, 3, 4]***

|  |  |  |  |
| --- | --- | --- | --- |
| No | Operation | Description | Example |
| 1 | list.append(item) | Add item to the end of the list | list.append(0) → [1, 2, 3, 4, **0**] |
| 2 | list.extend(item) | Join new list to the end of list | list.extend([1, 2]) → [1, 2, 3, 4, **1, 2**] |
| 3 | list.pop(item) | Return and remove the xth item | list.pop(3) → [1, 2, 3] |
| 4 | list.insert(x, item) | Insert item at the xth position | list.insert(0, 20) → [**20**, 1, 2, 3, 4] |
| 5 | list.sort() | Sort the list | list.sort(reverse=True) → [**4, 3, 2, 1**] |
| 6 | list.index(item) | Return the position of the first occurrence of item in list | list.index(4) → **3** |
| 7 | list.count(item) | Count how many times item appears in list | list.count(3) → **1** |
| 8 | list.remove(item) | Remove the irst occurrence of item in list | list.remove(4) → [1, 2, 3] |

**Table 3: List operation**

## Dictionaries

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

# HEADING 1

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

## Heading 2

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

## Heading 2

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

# HEADING 1

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

## Heading 2

### Heading 3

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

### Heading 3

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

## Heading 2

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

# HEADING 1

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

## Heading 2

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

## Heading 2

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

# HEADING 1

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

## Heading 2

### Heading 3

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

### Heading 3

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

## Heading 2

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

# HEADING 1

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

## Heading 2

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

## Heading 2

Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal Normal normal normal

# Section

Gfjlkdfjg lkj g;lksj ;lk jl;jk

## Heading

Alkjf ;aljkf ;lsjf ;lkj sa;klfj asdlfkj as;dklj

### asdfjlksfjd alkf ja;skdfj a;lkdj

Fsd sdfg sdcg

Dsf

G

Sdfg

# fsd sdf sfgd sd