Python

Python Module 1 Lesson 1

Name: Date:

Module 1 Lesson 1 Teacher's Copy

Learning Outcomes:

1. Variables
    a. Variable and Naming Convention
    b. Integers
    c. Strings

Explanation Points:

- Highlight how the type() function works, as well as the output value
    - Output will look like: <class 'str'>
- Explain what variables are
    - Use simple concepts, like serial numbers/bar codes
- Explain how to run python programmes
    - F5 or the Play button
- Programme Flow
    - Python reads code from top down
    - Get student to read each line and understand what it means
    - Get student to understand how the logic flows
- Importance of Capital Letters
    - Terms such as "True", "False" are case sensitive
- Overriding Variables
    - Understand when variables are overwritten
- Syntax and Number Errors
    - Explain what errors are, why they occur and highlight what the errors look like in the terminal

Breakdown of Lesson Plan:

Introduction to Python + What are variables	15 min
Lesson 1.1 (Variables)	15 min
Lesson 1.2 (Naming Variables)	10 min
Lesson 1.3 (Naming Variables - Keywords)	10 min
Lesson 1.4 (Integers)	5 min
Lesson 1.5 (String)	5 min
Lesson 1.6 (Floats)	5 min
Lesson 1.7 (Datatypes and When to Use Them)	10 min
Lesson 1 Quiz	15 min

Name: Date:

Introduction

Welcome to The Logic Coders Python Masterclass. The Python Masterclass aims teach you how to write programs in the Python Language. Throughout each Module, you will further your understanding of how and why we write programs, by solving simple Mathematical Questions. Your teachers are here to guide you along this journey, and will share your successes, and explain any problems you face.

The Python Masterclass is made of 8 Modules, and by the time you have completed the 8th Module, you would have already become a Master Programmer. Each Module is made of 12 Lessons, with a Test on Lessons 6 and 12. In order to progress onwards, you must be able to meet the graduation requirements for the 2 Tests.

In Module 1, you will learn the basics of Python Programming. The topics covered are as follows:

Lesson Number	Topics Covered
1	Preparing Your Workspace Variables Integers and Strings
2	String Concatenation Arithmetic Operators
3	Arithmetic Operators
4	String Slicing
5	Functions
6	Mid-Module Test
7	Global and Local Variables
8	Return
9	Intermediate Functions Composite Functions
10	
11	Final Test Preparation
12	Final Test

The topics covered in Module 1 helps you to build a basic understanding of how to write programs in Python. By the end of the Module, you should be able to write a simple function to carry out varying Arithmetic Operations.

Let us begin with the 1st Lesson.

Name: Date:

Lesson 1

Learning Objective

Lesson Number	Topics Covered
Introduction	Preparing Your Workspace
1.1	Variables
1.2	Naming Variables
1.3	Keywords
1.4	Integers
1.5	String
1.6	Floats
1.7	Datatypes and When to use them
Quiz	End of Lesson Quiz

In this lesson, we will be experimenting with Variables, the naming conventions required when using Variables, as well as the 3 main datatypes, Integers, Strings and Floats.

Name: Date:

Lesson 1.1

When we program, we often have multiple data and values that we need to use. While it is possible, it is very difficult for us to memorize and keep track of all these values. Hence, we can utilize Variables to help us store these data.

But what exactly is a Variable, and how do we use it?

In Programming, a variable is a named location used to store data in the memory. We can name these variables as we wish, as long as we follow certain rules. These rules will be explored in detail in Lesson 1.2

To assign a value to a variable, use the "=" sign. The "=" sign is called an assignment operator.

1	number = 10

It is helpful to think of variables as a container that holds data which can be changed at any time. This is extremely useful, as we can save such values and data in variables, which we can subsequently use for various actions.

The print() function will execute the variable and prints the specified message to the screen. We use the print() function in the format:

print(thing to print)

Note that the print() function is in small letters, and what we want to print needs to be in the brackets.

Let us try the example below:

1	number = 10
2	number = 20
3	print(number)

Let us run the code and see what we get.

In your output, you should see the number 20. But why is it that the answer is 20, and not 10? Let us track the value of *number*.

Variable	Value
number	10 -> 20

Initially, the value of *number* was 10. Later it's changed to 20. Hence, the output will show 20. Notice that each row represents one unique variable. In some of the examples, we may have multiple variables to track.

With this knowledge, let us try some examples!

Python

Python Module 1 Lesson 1

Name: Date:

Lesson 1.1

Type the following and fill in the output after pressing F5

**The numbers represent the lines.*

Task 1:

1	number = 10
2	print(number)

Output

1	

Variables

Name	Value
number	

Task 2:

1	number = 10
2	number = 15
3	print(number)

Output

1	

Variables

Name	Value
number	

Task 3:

1	cat = 5
2	print(cat)
3	dog = 4
4	print(dog)

Output

1	
2	

Variables

Name	Value
cat	
dog	

Task 4:

1	teacherA = 3
2	print(teacherA)
3	principal = 1
4	print(principal)
5	teacherB = 3
6	print(teacherB)

Output

1	
2	
3	

Variables

Name	Value
teacherA	
principal	
teacherB	

Task 5:

1	pokemon = 5
2	babyshark = 20
3	print(babyshark)
4	ninja = 3
5	print(ninja)

Output

1	
2	

Variables

Name	Value

In Task 5, we have unused variables. Can you spit which variable was not used? Ideally, in your code, your variables should be used in calculation or in a function. Unused variables should not be generated.

Name: Date:

Lesson 1.1

We now have a good understanding of how to write variables. Let us explore why we need variables by doing the examples below!

Task 6A:

Martyn is planning a party! He plans the amount of food his friends would eat, and tabulated the results in the table below:

Friend	Food Price ($)
Jasmine	5
Alfred	12
James	8
Anisha	9
Kane	8

Try assigning each friend to a variable, and print the Food Price for each friend!

1	Jasmine =
2	Alfred =
3	James =
4	Anisha =
5	Kane =
6	print(Jasmine)
7	print(Alfred)
8	print(James)
9	print(Anisha)
10	print(Kane)

Output:

1	
2	
3	
4	
5	

Task 6B:

Continuing from Task 6, Martyn realizes he recorded the Food Price for Jasmine wrongly. He meant to write 7, and accidentally wrote 5. Try adding a line to reassign the food price for Jasmine. Where would you add this line? Why?

Name: Date:

Lesson 1.2

While using variables, there are some rules that must be followed. The main rules and naming conventions are listed below:

Rules and naming convention for variables:

1. **Do not use symbols (! @ # $ % ^ & *)**
2. **Do not start with a digit**
3. **Multiple words can be separated with the use of _ (underscore)**
4. **Create a name that makes sense**
5. **Python is case sensitive**

While programming, we need to recognize that our codes will be seen by others, so it is important to choose variable names that makes sense.

Type the following and fill in the output after pressing F5 if the name is allowed, else leave blank
Fill in the column Yes or No to the question "Is the name allowed"
**The numbers represent the lines.*

Task 1:

1	@youandhim = 10
2	print(@youandhim)

Output

1	

Is the name allowed?

Task 2:

1	4everfriends = 5
2	print(4everfriends)

Output

1	

Task 3:

1	John = 8
2	print(John)

Output

1	

Task 4:

1	powerrangers= 5
2	print(powerrangers)

Output

1	

Task 5:

1	_^*me&you= 5
2	print(_^*me&you)

Output

1	

Name: Date:

Lesson 1.2

Task 6:

1	Ninja_steel = 100
2	Print(Ninja_steel)

Output

1	

Is the name allowed?

Task 7:

1	@!sick = 11
2	print(@!sick)

Output

1	

Task 8:

1	enemies = 15
2	print(enemies)

Output

1	

Task 9:

1	_&^yes = 59
2	print(_&^yes)

Output

1	

Task 10:

1	cold_drinks = 400
2	Print(cold_drinks)

Output

1	

Python

Python Module 1 Lesson 1

Name: Date:

Lesson 1.3

In Python, there are some keywords that are used to create structure in our programs. These keywords are shown in the table below. We cannot use a keyword as a variable name.

Keywords are case sensitive

All the keywords except True, False and None are in lowercase and they must be written as it is.

`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`	

Type the following and fill in the output after pressing F5 if the name is allowed, else leave blank
Fill in the column Yes or No to the question "Is the name allowed"
**The numbers represent the lines.*

Task 1:

1	true = 10
2	print(true)

Output

1	

Is the name allowed?

Task 2:

1	elif = 5
2	print(elif)

Output

1	

Task 3:

1	jane=10
2	print()

Output

1	

Task 4:

1	if=900
2	print(if)

Output

1	

Task 5:

1	x=28
2	print(x)

Output

1	

Name: Date:

Lesson 1.3

Task 6:

1	and = 10
2	print(and)

Output

1	

Is the name allowed?

Task 7:

1	else = 5
2	print(elif)

Output

1	

Task 8:

1	ben=100
2	print(ben)

Output

1	

Task 9:

1	y=9
2	print(y)

Output

1	

Task 10:

1	j=289
2	print(j)

Output

1	

Task 11:

Assign 10 to False and 8 points to True. Print False's points followed by True's points.

Lesson 1.4

Name: Date:

Datatypes defines the kind of value a variable will take. This allows the compiler to recognize what the Programmer wants to do with the value. Datatypes are used to decide the kind of data in a variable. (like classifying animals according to their species)

There are a few kinds of datatypes in Python. For this lesson, we will be looking at the first type, called Integers.

In Mathematics, Integers refers to whole numbers. They represent the common numbers we use, like 1, 2, 3, etc. In Python, integers can be shortened to the form 'int'.

Integers → Defined as 'int'.

But how do we check if a variable is indeed an integer? Well, we can use the type() function to know which class a variable or a value belongs to.

To use the type() function, we need to arrange it in this order:

type(variable)

Let us explore a bit more about Integers.

Type the following and fill in the output after pressing F5
**The numbers represent the lines.*

Definition of Integer ('int') :

Task 1:

1	count=5
2	print(count)

Output

1	

Task 2:

1	weight=35
2	print(weight)

Output

1	

Task 3:

1	count=5
2	print(type(count))

Output

1	

Task 4:

1	weight=35
2	print(type(weight))

Output

1	

Lesson 1.5

Name: Date:

While storing numbers may be useful, it is difficult to communicate with the Users purely through numbers. Hence, let us learn the next datatype, known as a string.

Strings are useful, as they allow us to form sentences that people can read. It is the shot form of 'String of Characters'. In Python, string can be shortened to 'str'.

Strings → Defined as 'str'.

While programming, it is easy to recognize strings, as all strings have a single quote (' ') or double quotes (" "). Notice that in some environments, the strings change a different color.

Note: Strings are different from variable naming

Type the following and fill in the output after pressing F5
**The numbers represent the lines.*

Task 1: Output

1	myname="jane"
2	print(myname)

1	

Task 2: Output

1	enemy= "is fat"
2	print(enemy)

1	

Task 3: Output

1	Date = "3/4/2020"
2	print(type(Date))

1	

Task 4: Output

1	On_hit = 'Boing!'
2	print(type(On_hit))

1	

Task 5: Output

1	Ping = "Pong"
2	print(Ping)
3	print(type(Ping))

1	
2	

Lesson 1.6

Name: Date:

The last datatype we shall explore in this lesson is known as a float. While programming, it is unavoidable that decimals will be used to increase accuracy. These numbers with decimals are known as floats in Python.

Floats are useful to increase the accuracy of our calculations. Unlike strings and integers, floats does not have a shortened form (no, not even to 'flt'

At times, it is difficult to recognize integers from floats. Therefore, it is important to remember that floats always have a decimal point.

Type the following and fill in the output after pressing F5
**The numbers represent the lines.*

Task 1:

1	Time = 5.3
2	print(Time)

Output

1	

Task 2:

1	Height = 1.75
2	print(Height)

Output

1	

Task 3:

1	Val = 5.0
2	print(type(Val))

Output

1	

Task 4:

1	Weight = 52.5
2	print(type(Weight))

Output

1	

Task 5:

1	Speed = 25.4
2	print(Speed)
3	print(type(Speed))

Output

1	
2	

Lesson 1.7

Name: Date:

We have learnt the 3 main datatypes, namely Integers, Strings and Floats. While knowing these 3 datatypes are extremely useful, it is more important to know when to use these datatypes, and when we should not. Let us work on this guided example:

James wants to calculate how much he spent in a week. He knows he spends between $1.20 to $5.00 a day. What datatype should he use to store these values? Why?

Let us observe this problem. We know James wants to track his spending. In the example, his spending ranges between $1.20 to $5.00. In this case, it would be easy to track his spending using floats, as 1.20 and 5.00 are clearly floats.

At the same time, you can also argue that we can use integers instead, where we calculate his $1.20 as 120 cents, and $5.00 as 500 cents. This also makes sense.

Let us observe some other problems, and evaluate which datatype is suitable in each case.

Read each question carefully, and decide which datatype is applicable.

Task 1A:

Jamie wants to write a program that prints the current date in the format "DD/MM/YY" (e.g. "07/12/20"). Supposing she does not use any other programs, which datatype should she use to store these values? Why?

Further Thinking:

Why would strings be more useful to print these data, rather than integers or floats?

Task 1B:

Try writing a program to print the current date!

1	

1	Date =
2	
3	print(

Output:

Lesson 1.7

Name: Date:

Task 2A:

Kane works in a warehouse, and tracks the amount of basketballs being stored inside. He wants to utilize Python to help him. What datatype should he use to store this information? Why?

Task 2B:

Try writing a program to store the number of basketballs in the Warehouse! There are 20 basketballs in total.

1	BBalls =
2	
3	print(BBalls)

Output:

1	

Task 3A:

Amy wants to create a simple program to welcome visitors to her website. She specifically wants to print:

“Welcome to Amy’s Website!”

What datatype should she use to print the above statement? Why?

Task 3B:

Try writing a program to create such a text!

1	Text=
2	
3	print(

Output:

1	

Name: Date:

End of Lesson 1 Quiz

Fill in the blanks and check your answer by typing the code into the computer

Question 1

1	x=5
2	y=2
3	x=3
4	print(x)

Answer

1	

Question 2

1	x=55
2	y=22
3	x=33
4	print(x)
5	print(y)

Answer

1	
2	

Question 3

1	if=100
2	print(type(if))

Answer

1	

Question 4

1	i=59
2	print(type(i))

Answer

1	

Question 5

1	jane= “happy”
2	jack = “2/2/19”
3	print(jane)
4	print(jack)

Answer

1	
2	

Name: Date:

End of Lesson 1 Quiz

Question 6

Name a variable John with a value of 10 and a variable Joker with a value of “10”.

Print John and Joker and tell me their type

1	
2	
3	
4	
5	
6	

Answer

1	
2	
3	
4	

Question 7

Fill in the column with “Allowed” or “Not Allowed”

		Is it allowed?
1	Jake	
2	class	
3	elif	
4	Break	
5	John	
6	Numberoftimes	
7	except	
8	howmuch	
9	return	
10	returnme	
11	correctme	
12	difference	
13	dylan	

Question 8:

Circle all the Integers!

“5”	7	9.2	8.4
6	9.0	“10”	15
21	29.3	5	‘7.0’
11.4	’13.3’	6.7	20

End of Lesson 1 Quiz

Name: Date:

Question 9A

James wants to track the number of snails in his garden. He knows that there are 20 snails currently in his garden. What datatype should he use to store this value? Why?

Question 9B

Try writing a program such that the output is as shown!

1	
2	
3	
4	

Output:

1	20
2	<class 'int'>

HINT: Remember the type() function will show the datatype of the variable.