Python Course Glossary

IDE

An IDE (Integrated Development Environment) is a software tool that allows you to write, edit, and debug your code all in one place. For example, in Lesson 2, we learned how to use chatbots to help write code, but an IDE can also enhance productivity by providing features like syntax highlighting and error detection.

Introduced in: Lesson_2.txt

Jupyter Notebook

A Jupyter Notebook is an interactive tool used to run and test Python code. In Lesson 4, students were shown how to execute code using Shift + Enter, starting with a simple "print('Hello, World!')" example. It allows for real-time code testing, experimenting, and learning by doing.

Introduced in: Lesson_4.txt

Shift Enter

Shift Enter in Jupyter Notebooks runs the current cell of code. In Lesson 3, you learned to press Shift and Enter together to execute the code and see the output immediately, like printing "Hello, world!".

Introduced in: Lesson_3.txt

bug

A "bug" is a mistake or error in your code that causes it to function incorrectly. For example, in Lesson 4, a typo in the print statement was a bug that caused an error message. Bugs are common and learning how to find and fix them is an important part of programming.

Introduced in: Lesson_4.txt

coding environment

A coding environment is a space where you write, run, and test your code. In Lesson 3, we learned to use a Jupyter Notebook, with a code area in the center and tools like chat support and video tutorials to help us.

Introduced in: Lesson_3.txt

comment

A comment is text in your code that Python ignores. It's used to leave notes or explanations for anyone reading the code. In Lesson 4, comments start with a hash `#` and do not affect the program's execution, like `# This is a comment`.

Introduced in: Lesson_4.txt

curly braces

Curly braces `{}` in Python are used inside f-strings to embed expressions. For example, in Lesson 6, you saw how to display calculations within text: `f"The temperature is {75 - 32 * 5/9:.2f}°C"`, which inserts the calculated value directly into the string.

Introduced in: Lesson_6.txt

def

'def' is used to define a function in Python. Functions let you group code into a reusable block, making your code easier to manage. For example, in the lesson 'Course Introduction,' creating a function would help automate repetitive tasks, enhancing productivity.

Introduced in: Course_Introduction.txt

error message

An error message is a notification from the computer that something went wrong with the code. In Lesson 4, we saw an example of a buggy line of code that produced an error message because of mismatched quotation marks. Error messages help debug and fix issues in your program.

Introduced in: Lesson_4.txt

f string

An f string in Python, introduced in Lesson 6, allows you to embed calculations and other values directly within a string using curly braces `{}`. For example, `f"The temperature is {75/1.8-32:.2f} degrees Celsius"` will display the calculated temperature formatted neatly.

Introduced in: Lesson_6.txt

float

A float in Python is a type of number that includes a decimal point. For example, 2.99 and 3.14 are floats. In 'Lesson 5', it was highlighted that floats can represent numbers like 3.14 or 2.99, unlike integers which have no decimal part.

Introduced in: Lesson_5.txt

function

A function in programming is a reusable block of code designed to perform a specific task. Functions help organize your code, making it easier to read, understand, and maintain. For instance, a function can easily handle repetitive tasks without rewriting the same code repeatedly.

Introduced in: Course_Introduction.txt

if

The 'if' statement in Python lets you make decisions in your code. It runs certain code only if a specified condition is true. For example, "if x > 10: print('x is greater than 10')" runs the print statement only if x is greater than 10.

Introduced in: Course_Introduction.txt

integer

An integer in Python is a whole number without a decimal part. Examples include 42, -9, and 100. They are used frequently for counting and basic arithmetic. In Lesson 5, integers were contrasted with floats, which are numbers that include decimal points, like 3.14.

Introduced in: Lesson_5.txt

len()

In 'Lesson 2', the `len()` function in Python is used to find the number of items in an object, like the number of characters in a string. For example, `len("Hello")` returns 5.

Introduced in: Lesson_2.txt

list

In Lesson 2, a 'list' is a way to store multiple items in one variable in Python. For example, `my_list = [1, 2, 3, 4]` stores numbers 1 through 4. Lists can hold any data type and allow you to add, remove, or modify items easily.

Introduced in: Lesson_2.txt

loop

A loop lets you repeat a set of instructions in your code, saving you time and effort. For example, you can use a loop to print numbers 1 to 10 without writing print statements ten times. It's a handy tool for repetitive tasks.

Introduced in: Course_Introduction.txt

paste

'Paste' means inserting copied code into the coding area. In 'Lesson 3,' you learned to copy code from the chatbot and paste it into the Jupyter Notebook using Command V or Control V, depending on your operating system.

Introduced in: Lesson_3.txt

print()

In Lesson 5, `print()` is introduced as a command in Python to display data, like strings or numbers, on the screen. For example, `print("Hello, World!")` prints the text "Hello, World!".

Introduced in: Lesson_5.txt

run code

Running code means executing written code to see its results. In Lesson 3, learners use a Jupyter Notebook and run code by pressing Shift + Enter after typing or pasting code to execute it, like running the command `print("Hello, World!")`.

Introduced in: Lesson_3.txt

string

In Python, a string is a sequence of characters enclosed in double quotation marks. For example, "Hello world" is a string. Strings can contain letters, numbers, symbols, and spaces. Lesson 5 explains how to create and print strings, and introduces multi-line strings using triple quotation marks.

Introduced in: Lesson_5.txt

syntax

Syntax in programming is the set of rules that defines the structure of code. In Lesson 2, it was highlighted that Python has a simple syntax, making it beginner-friendly. For example, you write `print("Hello, World!")` to display a message.

Introduced in: Lesson_2.txt

type()

In Python, the `type()` function checks the data type of a given value. For example, `type("Hello world")` returns `str` for a string, while `type(100)` returns `int` for an integer. This is introduced in Lesson 5.

Introduced in: Lesson_5.txt

variable

A variable in Python is like a container that holds a specific value or piece of data, which can be used or changed as needed in your code. For example, `age = 25` sets a variable named `age` with the value `25`. This topic was introduced in 'Course Introduction'.

Introduced in: Course_Introduction.txt