





Module 1: Hello World

Introduction to the Mu Editor

Let's make our first program. Follow these steps.




- 1.1. Launch the Mu Editor program.
- 1.2. Click the "New" button. This will open an "untitled" tab.
- 1.3. Let's give the program a name. Do this:
 - Insert USB stick into the computer.
 - Click the "Save" button.
 - Browse to the USB stick.
 - Enter `hello` and click "Save".
 - The tab will now be called "hello.py"
 - NOTE: Although you just typed "hello", notice that ".py" was automatically added at the end by the MU Editor. That's called a filetype, and means "this is a Python program."
- 1.4. Enter the following program:



- 1.5. Click the “Run” button.
- 1.6.  **QUESTION:** What happened when you clicked the “Run” button?
- 1.7. If you run into problems:
 - Click the “Stop” button.
 - Edit the program to make your changes.
 - Click the “Run” button to run with your fixes.
- 1.8.  **WOW:** Good going! You just created your first Python program.
- 1.9.  **NOTE:** As you work, the Mu Editor updates the saved file. Look carefully at the message at the bottom of the previous example.
- 1.10.  **READ THIS:** <https://tinyurl.com/pyhi-files> – How to Create, Load, and Save Files in Mu

Interesting bits about this program

“Hello world” is a “string literal”

- 2.1. A string literal is text: The name comes from “string of characters.”
- 2.2. String literals can be enclosed in either ‘single quotes’ or “double quotes”. Our first program used single quotes.
- 2.3.  **DO THIS:** Click the “Stop” button. Then change the first double quote to a single quotes. What happens? Run the program and see what happens.
- 2.4.  **DO THIS:** Now change the second double quote to a single quote. What happens? Run the program and see what happens.
- 2.5.  **TAKEAWAY:** Python doesn’t care whether you use single quotes or double quotes. For messages and the like, double quotes are preferred *by convention*.
 - We should talk a little about what “by convention” means, and why good programmers try to follow common practices.

print() is a function

- 3.1. A function does some work. In the case of *print()* it prints stuff to the console.
- 3.2. A function can take arguments. Programmers say that arguments are “passed into” a function. An argument is information that the function works on. In our example program, we are passing the *print()* function a single argument: the string literal “Hello world”.

Exercises with print()

This section has many small programs for you to try, one at a time.

For each one: enter the program, run the program, and observe what the program does.

5.1. Enter and run this program.

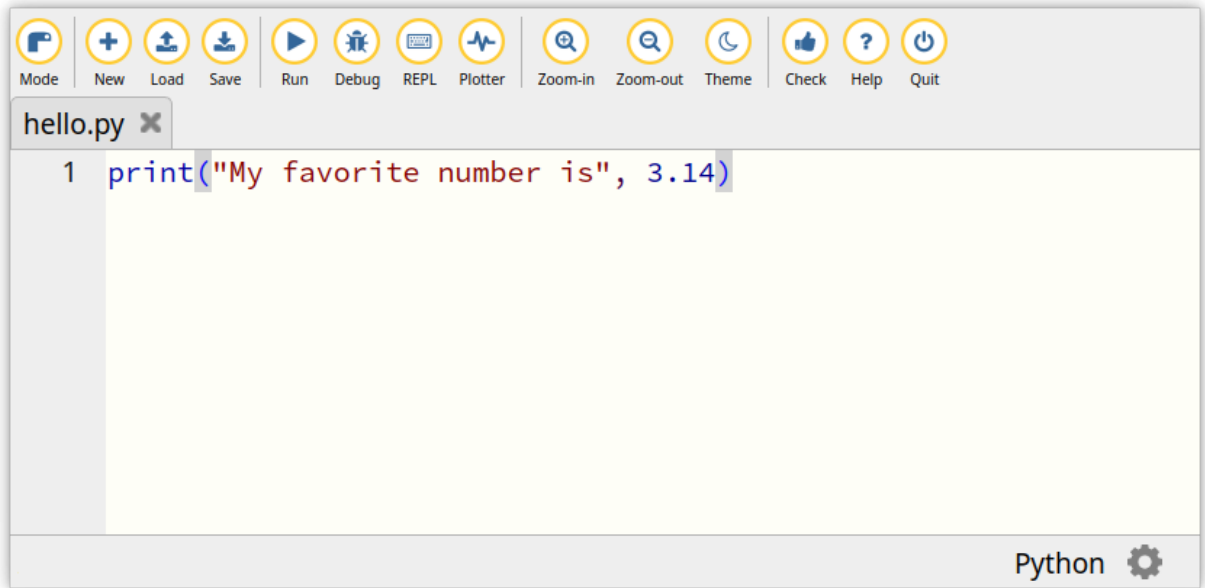
!! IMPORTANT: Don't type the line starting with "#"... it is just showing what's changed from the previous program.



💡 **TAKEAWAY:** The `print()` function accepts multiple arguments. It prints them all to the console.

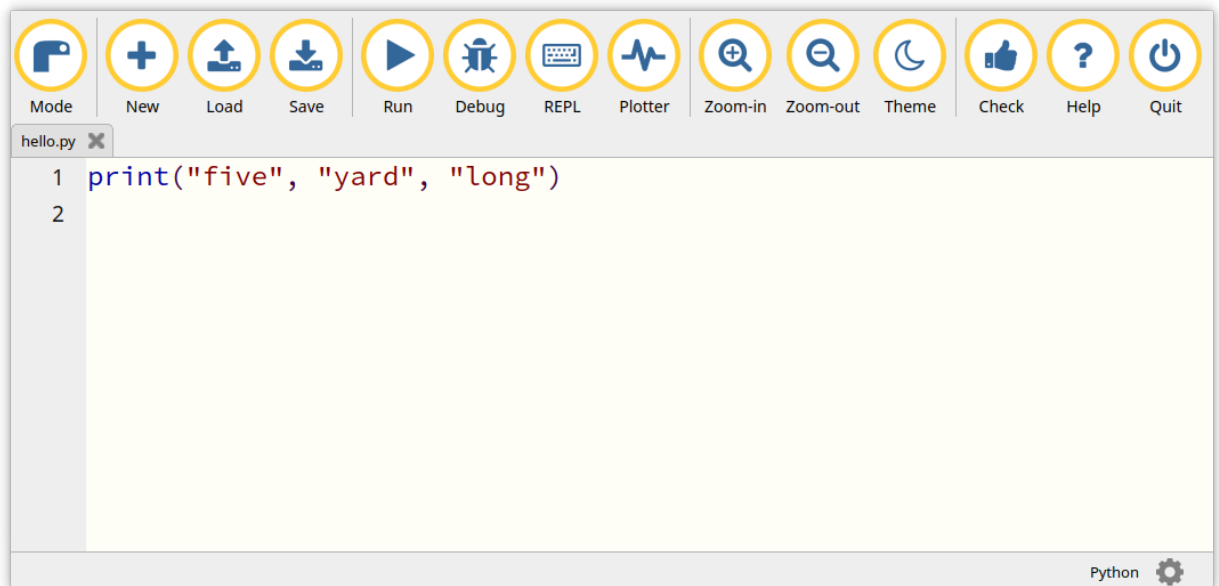
5.2. Enter and run this program:

!! PRO TIP: If you want to start from scratch, you can clear the editor window easily by typing CTRL/A (for “select all”) then BACKSPACE.




💡 NOTE: Numbers without quotes are not strings – they are numbers! The `print()` function accepts them too. (We’ll be dealing more with numbers shortly.)


5.3. Enter and run this program:



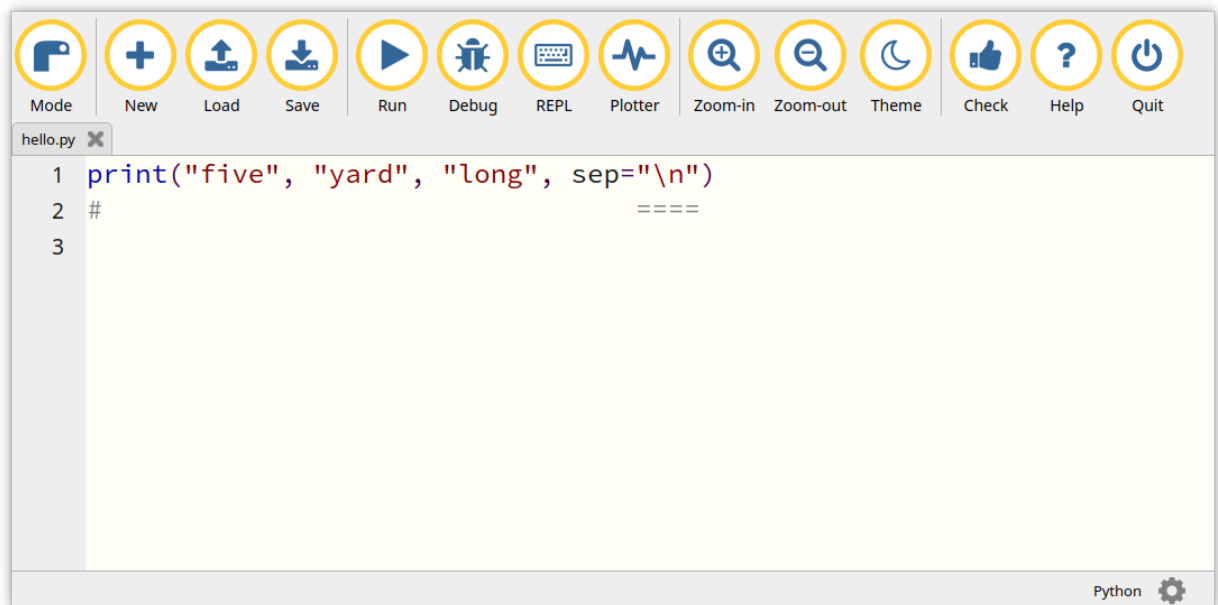
5.4. Make the highlighted change to the previous program and run it:




 **NOTE:** The `sep=` argument is known as a *keyword argument*. The `sep` keyword tells `print()` what to use to separate the printed arguments.

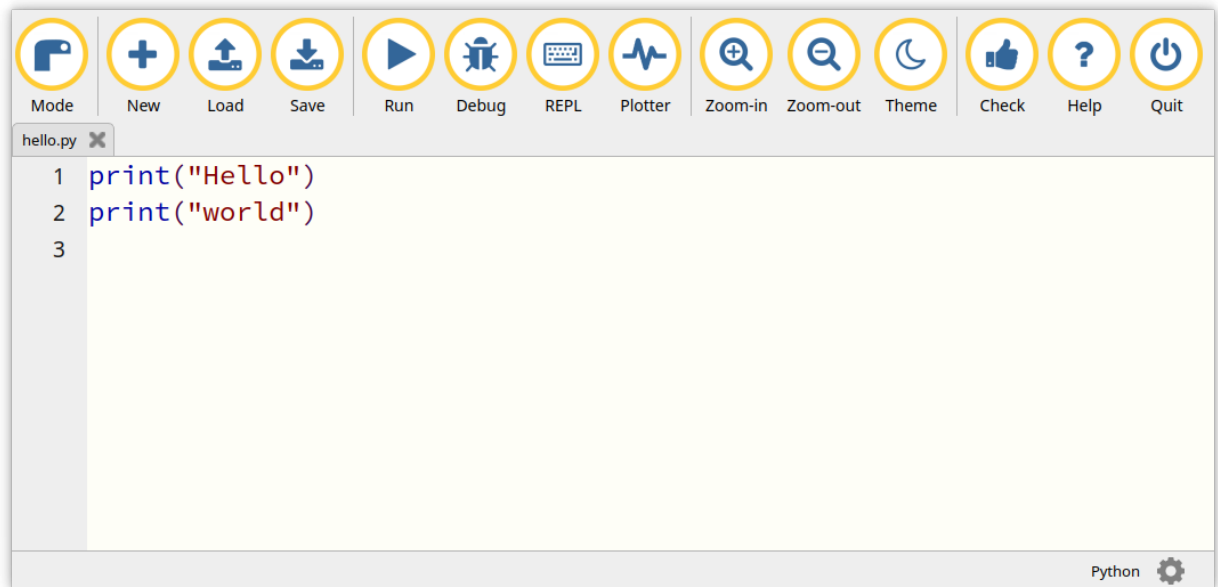
 **QUESTION:** What is the default when you don't give a `sep` argument?

5.5. Make the highlighted change to the previous program and run it:



 **NOTE:** The special character sequence “`\n`” (pronounced “backslash en”) means a newline character.

5.6. Run this program:



!! WOW: You just created a two line program. You deserve a round of applause.

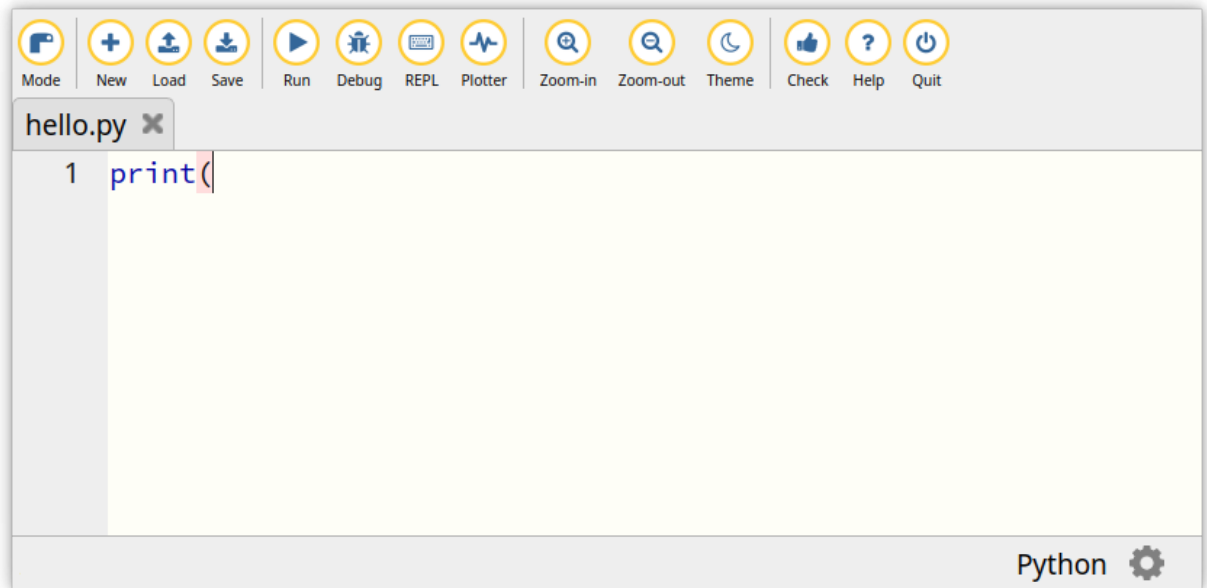
5.7. Make the following change to the previous program and run it:



? QUESTION: What does the *end* keyword argument do?

? QUESTION: What is the default when there is no *end* keyword argument?

5.8. Clear the edit window and type “print(“. Stop at the “(“! Do not click “Run”!



❗ **QUESTION:** What happens?

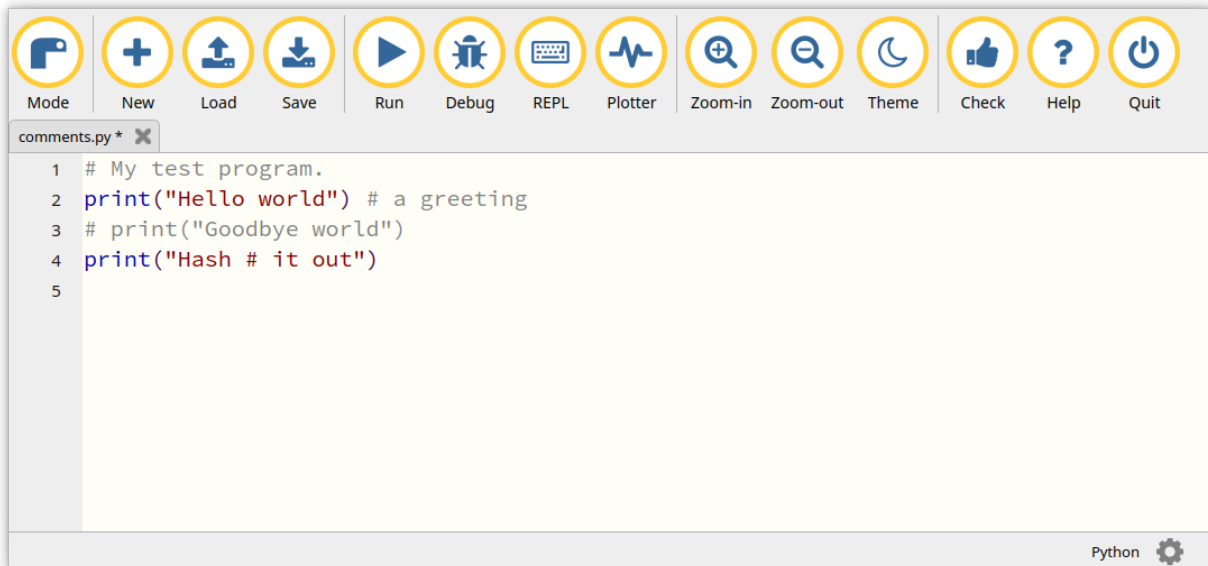
💡 **NOTE:** The Mu Editor has built-in help for most (but not all!!!) functions.

Comments

Comments are used to put notes into programs that Python ignores.

Comments start with a pound sign (“#”) and run to the end of the line.

Enter the following program into the Mu Editor:



Run it. What happens?

What are reasons we may want to use comments in a program?

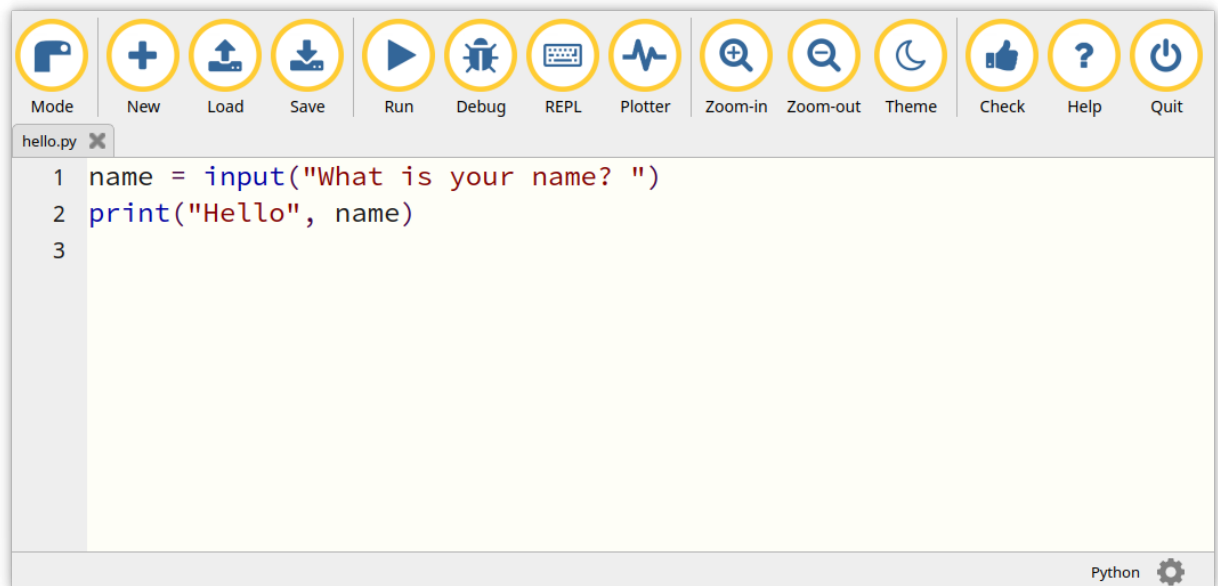
1.

2.

Exercise with Input()

Let's make our next program. Follow these steps.

- 5.1. Enter the following program into the Mu Editor:



- 5.2. Run it.
- 5.3. You will be asked a question on the console. Type your answer then press ENTER.
- 5.4. **?! QUESTION:** What does this program do?
- 5.5. Try “Stop” then “Run” several times. Try different answers and see what happens.

Interesting Bits About This Program

input() is a Function

1. This is the second time we've seen a function. We've already seen the *print()* function.
2. The *input()* function takes a single argument called a “prompt”. Notice that we ended the prompt with “? ” (question mark, space). That makes it nice because:
 - The question mark helps the user understand they are being asked a question.
 - The space separates when the user types from the prompt.

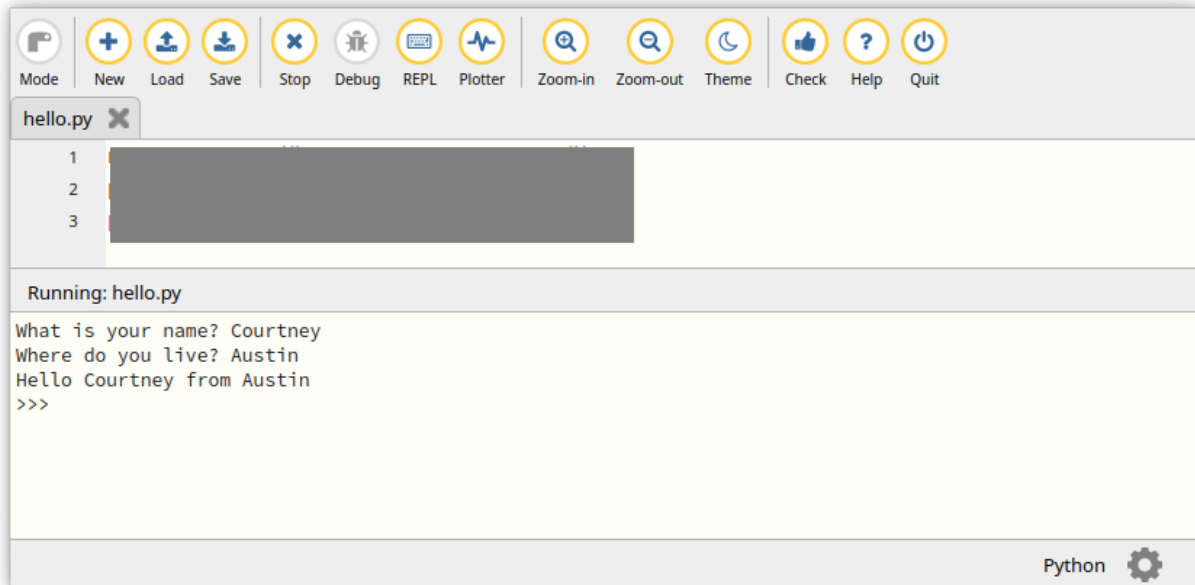
3. The `input()` function returns a value – whatever was typed at the console. The `print()` function did not return anything.
4. We're saving the `input()` result into a variable called `name`.


`name` is a Variable

1. In a computer program, a variable is used to hold a value.
2. Think of a variable as a box with a label on it:
 - The label is the “variable name”.
 - What's inside is the “variable value”.
3. A good variable name describes its contents. In this case, the variable holds a person's name, so we called it “name.”
4. Variable name rules:
 - Variable names can be made of uppercase letters, lowercase letters, digits, and the underscore (“_”) character.
 - Usually, variables should start with a lowercase letter.
 - Variable names are case sensitive. (“name” and “Name” are two different variables.)
5. 🖐️ **DO THIS:** Can you come up with a good name for this variable other than “name”? Can you think of a bad name for this variable?

Programming Challenge (hello.py)

Write a program that does the following when run:



 **HINT:** Start with the previous program that asked for your name, then make changes to handle where you live.