Previous: Python Function

Next: Python Comment ›

Home » Introduction to Python » Your First Python Program

Your First Python Program

November 19, 2022

If you followed the Python tutorial from the start, you've learned a lot by now. We've covered key topics, like booleans and conditional programming, strings, and functions. What we haven't done yet, is create an actual program. So let's wrap this up by combining what we learned into a nice little program. We will create your first Python program together.

Table of Contents [hide]

Beginners Python Course (2023)

Modules, Packages, And Virtual Environments (2023)

Entering the code in the REPL

Let me share the program first. Please analyze it thoroughly before you continue reading. There's one function you don't know yet (input), but that I'll explain shortly:

```
def say_hi(name):
    if name == '':
        print("You didn't enter your name!")
    else:
        print("Hi there...")
        for letter in name:
            print(letter)
```

```
name = input("Your name: ")
say_hi(name)
```

Now it's time to try this program yourself. If you simply copy and paste the above code into the REPL, you'll notice that it won't work. What does work is:

1. First copy the function `say_hi(name)` and hit enter. Now the REPL knows this function.
2. After that, copy and paste the `name = input("Your name: ")` and hit enter again. It will ask for your name, so enter it.
3. Finally, you can copy and paste the `say_hi(name)` line and see what happens!

In the REPL, it should end up looking like the following:

```
>>> def say_hi(name):
...     if name == '':
...         print("You didn't enter your name!")
...     else:
...         print("Hi there...")
...         for letter in name:
...             print(letter)
...
>>> name = input("Your name: ")
 < enter your name at this point >
>>> say_hi(name)
```

If you still have problems, here's a version in my online Python interpreter, that you can run right from this page. Note that `input()` doesn't work in this system, so we just call `say_hi()` with a predefined string instead:

Your first Python program

Analyzing your first Python program

Let's go over the code step by step.

Asking for input with Python

I managed to cram in one more new thing, the built-in function `input()`. It does exactly what you expect it to do: ask for input and assign that input to a variable. If you give `input` a string as an argument, it will print it as a prefix. In this case, it will print 'Your name: ' and wait for you to enter your name.

The say_hi function with one argument

The `say_hi(name)` Python function that we defined, takes one argument, the name, and prints the name to the screen. The function does not return anything. It doesn't need to, since it does all the printing work itself.

An if .. else block

Our function only greets us if the name is not an empty string. Why's that?

When someone just hits enter when asked for input, the `input()` function returns an empty string. You can check for yourself in the REPL:

```
>>> input()
''
```

Just hit enter when asked for input, and you'll see that the call to input() results in an empty string. So to not make a fool of ourselves, we won't greet someone showing such rude behavior. We do this by checking if the input is equal to an empty string using an if-else block.

A for loop

Finally, with Python's for loop, we print each letter of the entered name on a new line, just because we can.

Assignment: adapt your first Python program

Since we defined a function, `say_hi(name)`, we can reuse this function. You can repeatedly ask for a name and repeatedly call say_hi. Here's a little assignment:

Question Answer

```
Create an infinite loop that keeps asking for names,
and that keeps greeting us using the entered name.
Hint 1: use the say_hi function from above.
```

```
Hint 2: revisit the section about loops if you need to.
```

And now what?

At this point, using the interactive Python shell starts to work against us. Chances are you've been fiddling a lot to get this first Python program working, mainly because of indentation issues. Luckily, we can also store our Python programs in files, as you'll learn in the next section: creating Python programs. But before we do so, we'll first dive into Python comments.

For now: congratulations. If you followed along, you should have a basic understanding of programming with Python. You created your first Python program! I recommend you to keep experimenting inside the REPL. You may need to re-read some or all of the sections. That's OK and perfectly normal.

The most important piece of advice I'd like to give at this point is that you don't learn programming by reading alone, just like you don't become a doctor just by reading. You'll have to get your hands dirty and practice.

If you feel ready, continue with the next chapter!

Get certified with our courses

Learn Python properly through small, easy-to-digest lessons, progress tracking, quizzes to test your knowledge, and practice sessions. Each course will earn you a downloadable course certificate.

Beginners Python Course (2023)

Modules, Packages, And Virtual Environments (2023)

NumPy Course: The Hands-on Introduction To NumPy (2023)

‹ Previous: Python Function

Next: Python Comment