

Module 2: Data types, variables, basic input-output operations, basic operators
--

2.5 Comments	2
2.5.1 Comments – why, when, and how?	2
2.5.2 Marking fragments of code	2

2.5 Comments

2.5.1 Comments – why, when, and how?

You may want to put in a few words addressed not to Python but to humans, usually to explain to other readers of the code how the tricks used in the code work, or the meanings of the variables, and eventually, in order to keep stored information on who the author is and when the program was written.

A remark inserted into the program, which is **omitted at runtime**, is called a **comment**.

How do you leave this kind of comment in the source code? It has to be done in a way that won't force Python to interpret it as part of the code.

Whenever Python encounters a comment in your program, the comment is completely transparent to it – from Python's point of view, this is only one space (regardless of how long the real comment is).

In Python, a comment is a piece of text that begins with a # (hash) sign and extends to the end of the line.

If you want a comment that spans several lines, you have to put a hash in front of them all. Just like here:

```
1 # This program evaluates the hypotenuse c.
2 # a and b are the lengths of the legs.
3 a = 3.0
4 b = 4.0
5 c = (a ** 2 + b ** 2) ** 0.5 # We use ** instead of a square root.
6 print("c =", c)
7
```

Good, responsible developers **describe each important piece of code**, for example, by explaining the role of the variables. Although it must be stated that the best way of commenting variables is to name them in an unambiguous manner.

For example, if a particular variable is designed to store an area of some unique square, the name `square_area` will obviously be better than `aunt_jane`.

We say that the first name is **self-commenting**.

2.5.2 Marking fragments of code

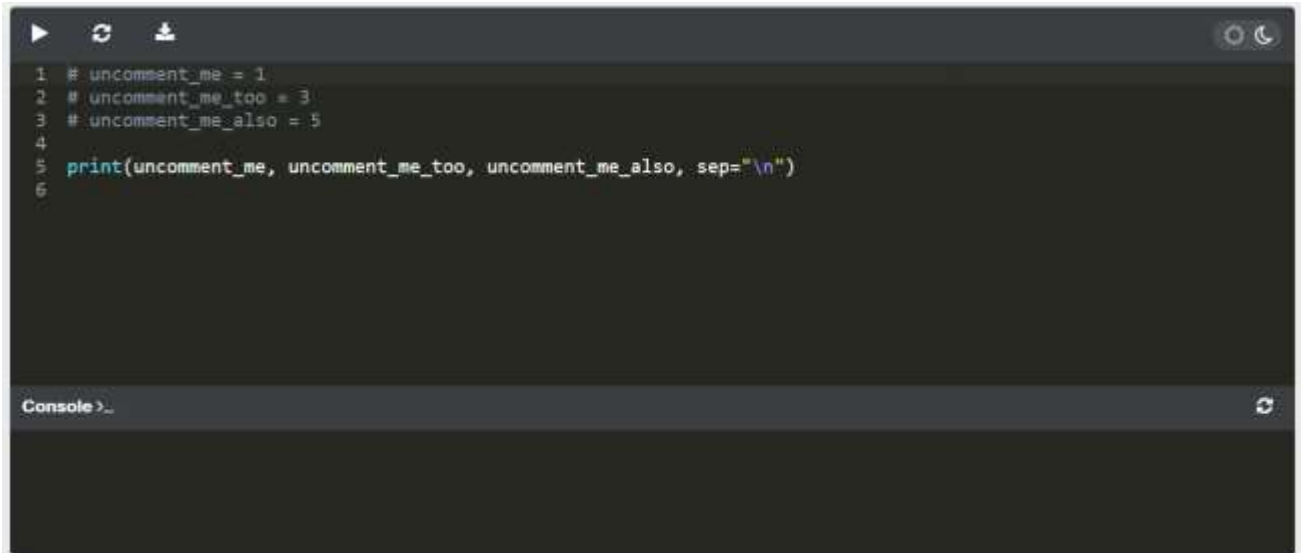
Comments may be useful in another respect – you can use them to **mark a piece of code that currently isn't needed** for whatever reason. Look at the example below, if you **uncomment** the highlighted line, this will affect the output of the code:

```
1 # This is a test program.
2 x = 1
3 y = 2
4 # y = y + x
5 print(x + y)
6
```

This is often done during the testing of a program, in order to isolate the place where an error might be hidden.

Tip

If you'd like to quickly comment or uncomment multiple lines of code, select the line(s) you wish to modify and use the following keyboard shortcut: **CTRL + /** (Windows) or **CMD + /** (Mac OS). It's a very useful trick, isn't it? Now experiment with the code in the editor.



```
1 # uncomment_me = 1
2 # uncomment_me_too = 3
3 # uncomment_me_also = 5
4
5 print(uncomment_me, uncomment_me_too, uncomment_me_also, sep="\n")
6
```

Console >_