File Processing

* format strings
* how to demonstrate multiline strings
* how to read & write & append the text files in python

**String formatting**

Template string 模板字符串

Format specifier: The part between {} (where the value is to be inserted and how it should look)

Form of description: <index>: <format-specifier>

(1). Index: tells which parameter to insert into the slot(狭槽), it starts from 0

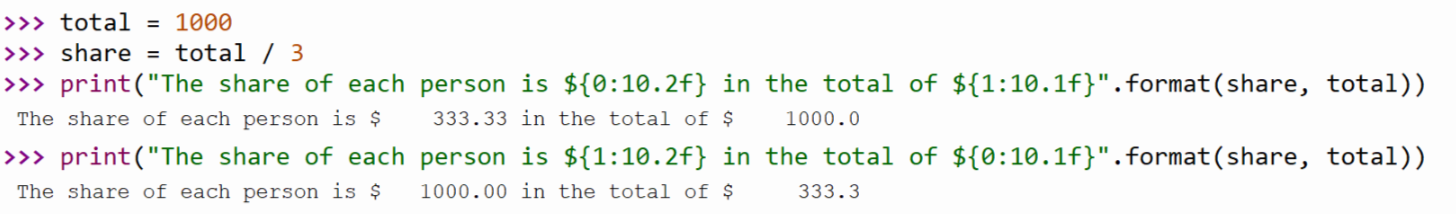
(2). format-specifier: <width>. <precision> <type>

1. <width>: tells us how many spaces to use to display the value,

0 means use as much spaces as necessary

1. <precision>: the number of decimal places

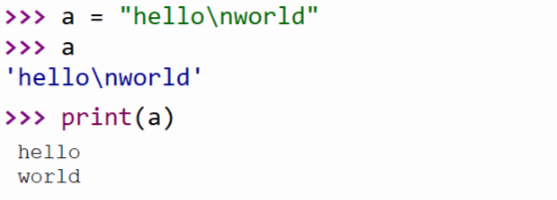
**PS: string formatting just change the format, the real value does not change.**

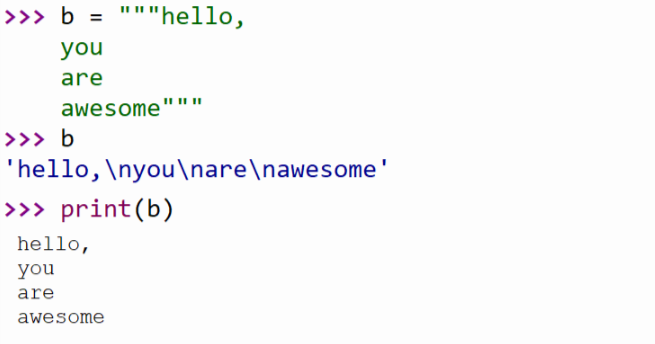


**Multiline Strings**

Three ways to mark new lines:

1. “\n”
2. “”” something inside “”” also available
3. “\t” is the tab character





PS: when you just output a or b, you would get what they really are. See the blue font.

**File Processing**

PS: files can contain any data type, but the easiest to work with are text.

\n: standard newline character -> to mark line breaks

1. First you need to open the file

Create a new variable to link the file.

For instance, f = open (“file type”, “operation”)

1. Three aspects of file processing

Write () -> “w”

* Whatever you write into the file, you would cover the original content in the file. That’s why you need append.

Read () -> “r”

* <file>. read () -> return the remaining contents of the files as a single string (watch out for final \n)
* <file>. readline () -> return the next line of the file. this is all text up to and including the nextline character.
* <file>. readlines () -> return a list of the remaining lines in the file. each list item is a single line including the nextline character.

Append () -> “a”

1. Finally, you must close the file when you have done the operation.

For instance, f. close ()