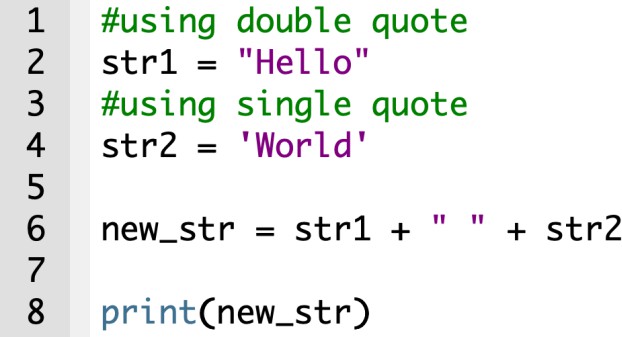
Lesson 2b worksheet

1. String in Python

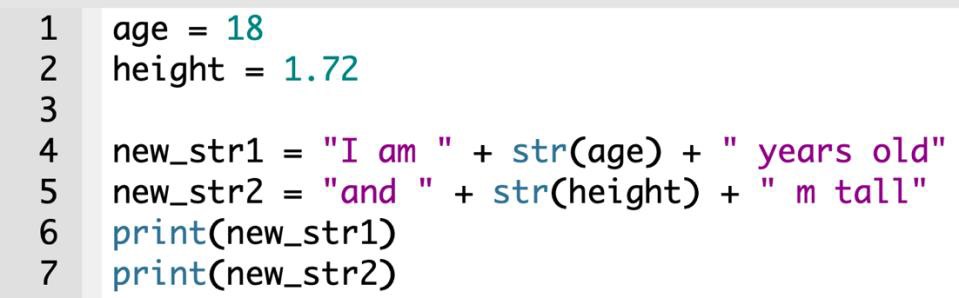
* Can be single or double quote.
* A close-up of a quote

  AI-generated content may be incorrect.Can only use either single or double quote.

1. String concatenate.

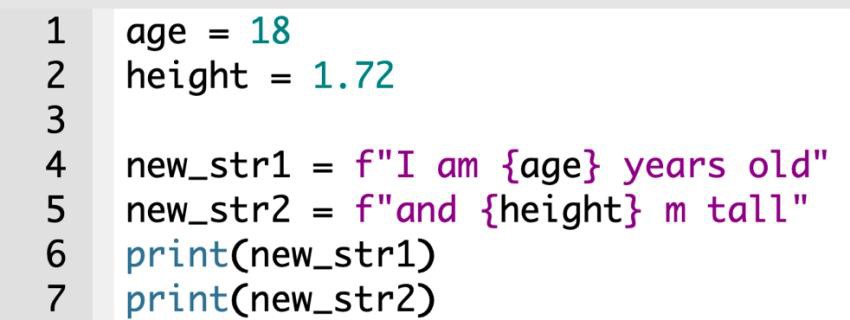
• Use “+” to concatenate (combine) two Strings.

1. String concatenates with int or double

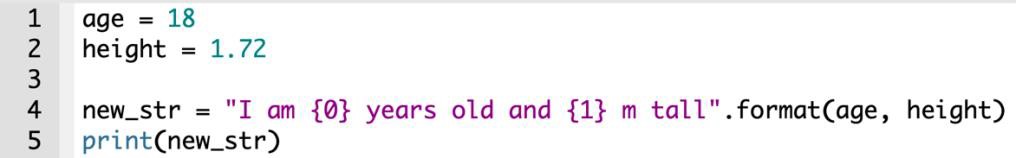
* Need to convert int or double to string (**casting**) for string concatenation.

1. String f’format

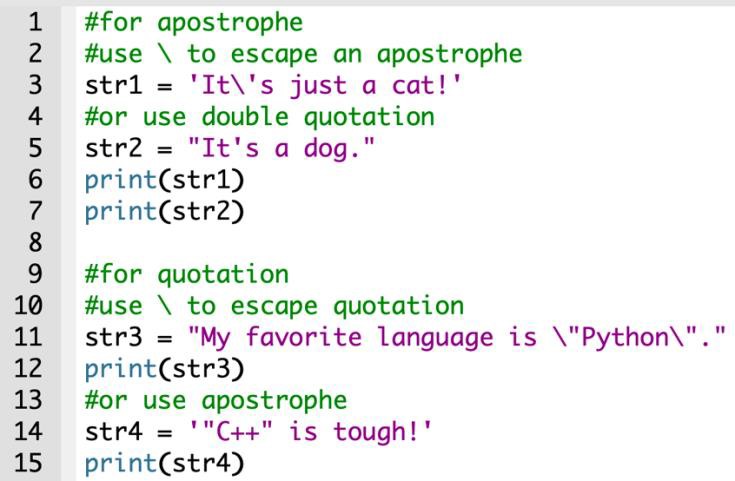
• A better way to form a string with variables.

• No need for casting

1. String format function

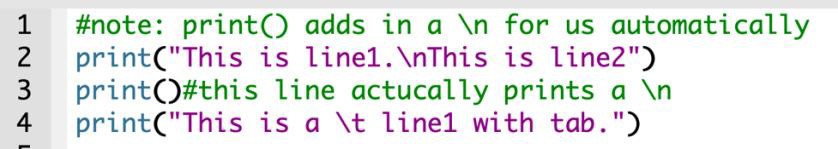
* Another way to format string.
* Note {0} and {1} are indexed place holder.
* {0} retrieve the value from the first item in format() function.
* {1} retrieve the value from the second item in format() function.
* Concept is similar to the index in list.

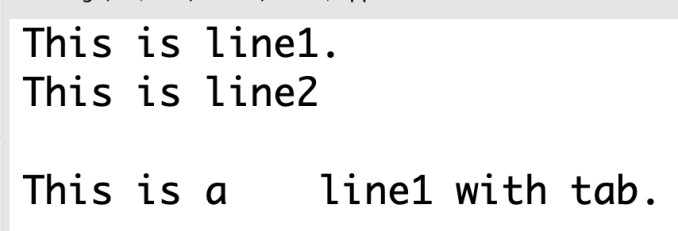
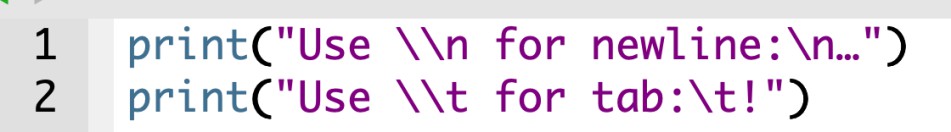
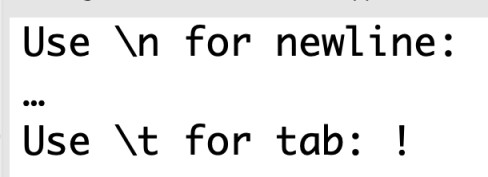
1. Escaping characters

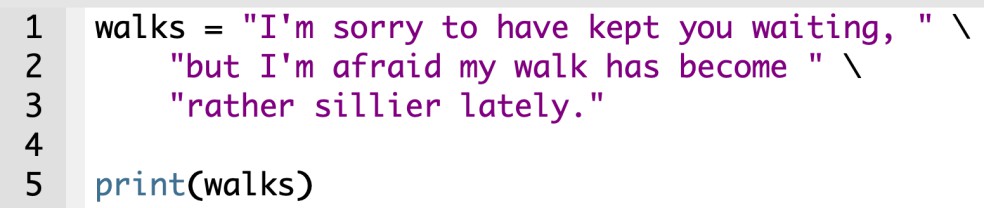
* If we need our string to contain a quotation mark or an apostrophe, “escape” it with  a “\”.
* Or we can use either quotation mark or an apostrophe to “escape” each other.

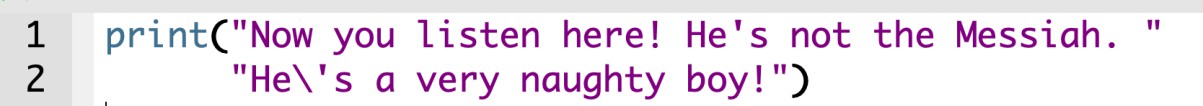
1. Special characters

* A computer store information in a binary (1 or 0) formats in a continuously form.
* To display our text in a certain format (e.g. next line or tab) we need to put a special  character in-between text to “inform” the computer that we want to move to next  line or a tab.

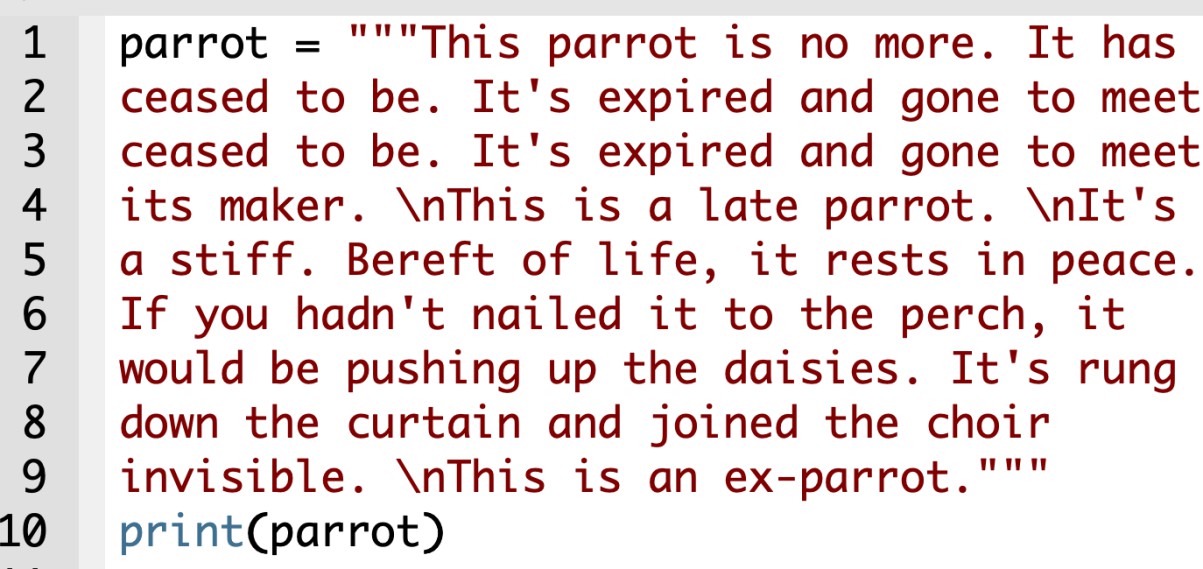


* Test out the code and you should get the text display before.
  + What if we want to print “\n” or “\n” ? We will need to “escape” the “escape” by  placing a “\” before a “\”.
  + Test out the code and you should get the text display:

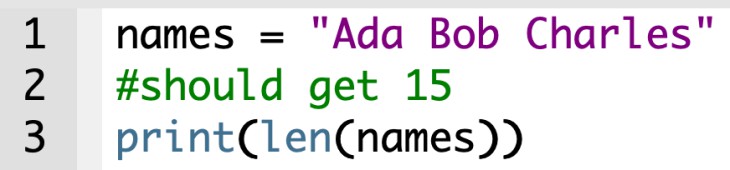
1. Long string
   * Python style suggests limiting each line of code to 79 characters
   * This lets you have multiple windows open at the same time with every line fully  visible.

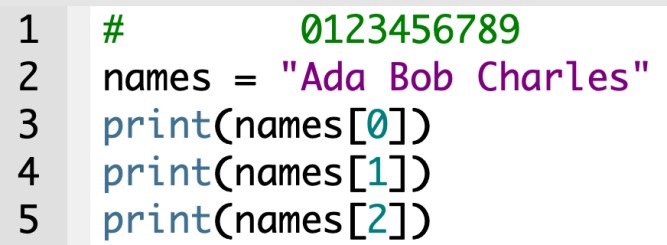
* If you're inside brackets, no need for \

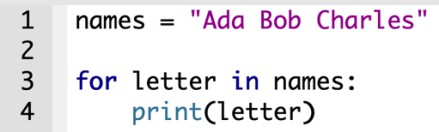
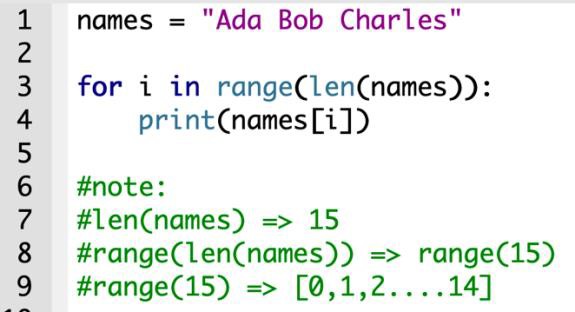
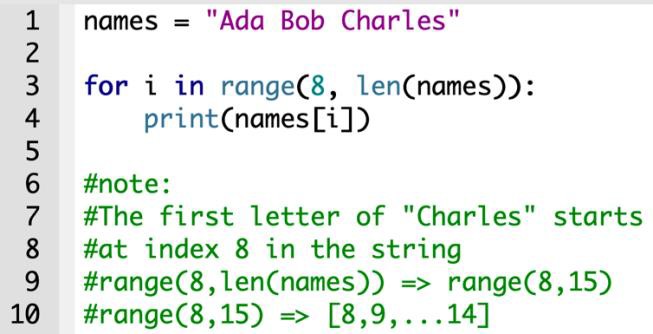
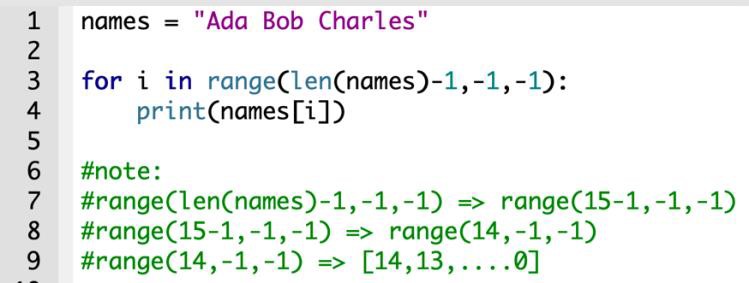
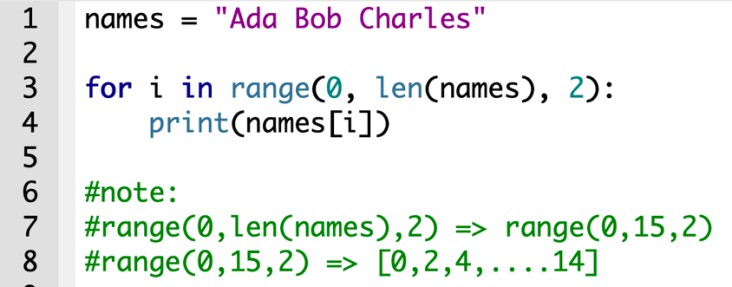
1. Very long string
   * + Use triple quotes to create a very long string, wrapping across multiple lines.
     + There is no need to escape the apostrophise within triple quotes – much easier to  maintain.



1. Length of String

* To know how many characters in a string, use the len() function.
  + - Note a blank is considered as a character as well.

1. String indexing
   * Each character in the string has an index.
   * Index starts at 0 and ends at length of string – 1.
2. Accessing individual characters using for loop
   * Think of a string as a List of Characters

* Use a standard for loop
* Use range() and len()
* Start printing from a particular index
* Print backward
* Print only even index (skip odd index)

1. Negative index

* Python supports negative index for string and list.

A screenshot of a computer

AI-generated content may be incorrect.

* + - The last char index is -1, the second last char is -2 and so forth.

1. Printing strings - separators
   * + When printing strings we have more flexibility than concatenating
     + If we want a character printed between each variable, we use sep=
     + Default separator is ' '

A screenshot of a computer

AI-generated content may be incorrect.

1. Printing strings – ends

* If we want a character printed at the end of each line, we use end=
* Default separator is '\n' (i.e. next line)
* Handy for keeping lines together when printing in loops

A screenshot of a computer code

AI-generated content may be incorrect.

1. Check is a char is in a string

* Use “in”
* True if an item of s is equal to x, else False

A screenshot of a computer code

AI-generated content may be incorrect.

1. Letter count in string
   * + Use string count() to get the total number of occurrences of a letter

A screenshot of a computer code

AI-generated content may be incorrect.

1. String duplicate
   * Multiply a string with an int n.
   * This adds the string back to itself n times

A close up of a smiley

AI-generated content may be incorrect.

1. Unicode
   * Computer only store binary number.
   * Each char/letter in computer is presented by a number.
   * Unicode is the universal standard for all letters presentation.
   * https://en.wikipedia.org/wiki/List\_of\_Unicode\_characters • Note the Unicode value of A is 0041 (hex number)

A screenshot of a table

AI-generated content may be incorrect.

* + - Try printing the Unicode 0041 and you will get a letter A.

A close-up of a logo

AI-generated content may be incorrect.

* + - The lower case letter has its own set of value.

A screenshot of a table

AI-generated content may be incorrect.

* + - So upper-case and lower-case letters are different letters in the computer world.

1. Using min() and max() with string

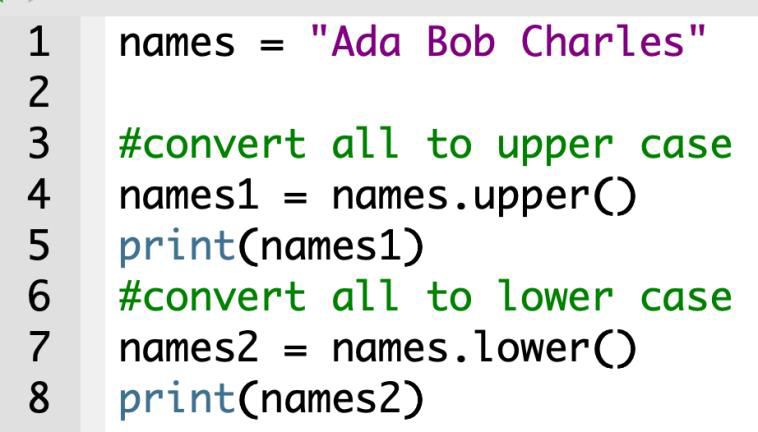
* min() look for the lowest unicode value in a string
* max() look for the highest unicode value in a string

A screenshot of a computer code

AI-generated content may be incorrect.

1. Convert string to upper or lower case.

* upper() returns a copy of string with all letters converted to uppercase
* lower() returns a copy of string with all letters converted to lowercase



1. Check if a string starts or ends with a sub string.

* s.startswith(pre) Returns True if s starts with pre
* s.endswith(post) Returns True if s ends with post

A screen shot of a computer code

AI-generated content may be incorrect.

1. Remove white spaces

* Use strip() function to remove leading and trailing whitespace

A text on a white background

AI-generated content may be incorrect.

1. Check if the string has only numeric chars

* Use isnumeric() to check if the string has only numeric chars

A screenshot of a computer code

AI-generated content may be incorrect.

1. Slicing

* Slice strings to access parts of them.

A screenshot of a computer

AI-generated content may be incorrect.

1. Slicing with steps

A screenshot of a computer code

AI-generated content may be incorrect.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | d | a |  | B | o | b |  | C | h | a | r | l | e | s |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |

1. Negative slicing

* Slicing with negative index
* The last letter index is -1, second last is -2 and so on

A screenshot of a computer code

AI-generated content may be incorrect.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P | y | t | h | o | n |
| -6 | -5 | -4 | -3 | -2 | -1 |

1. Negative step slicing (string reverse)

A screenshot of a computer program

AI-generated content may be incorrect.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P | y | t | h | o | n |
| 0 | 1 | 2 | 3 | 4 | 5 |

1. String to list conversion

* A common and regular task is to split a strings into pieces, based on a delimiter.
* The split method returns a list of strings.

