Working with strings

Introduction

Strings are an example of a data type. There are several different data types in python, so far we have only worked with strings (`str`) and integers (`int`).

String data is always treated as text by the program, even if it contains numeric characters. You cannot do mathematical calculations (addition, subtraction, etc.) with strings. They have to be converted (cast) to a numeric data type (`int` , `float`) first.

Printing strings

As you have already seen, strings can be printed using the `print()` fucntion:

```
print("Hello world!")
print('Hello Python!')
```

Note that strings must be surrounded by quotes - either single (apostrophe key) or double (quotation mark key). Either will work for simple text. You will want to choose judiciously when including apostrophes for quotes in your text, though.

Let's say you want to print the statement `Let's get started!` in your program. If you use single quotes, Python will get confused as to where the string ends:

```
print('Let's get started')
```

Python will consider the string to be defined by the first 2 quotes (`Let`) and it will try to make the rest of the phrase into a variable. It isn't a valid variable name, and you'll get an error.

To print this phrase, use double quotes around the entire string:

```
print("Let's get started")
```

Similarly, if you wanted to include quotation marks in your string, you would use single quotes at the start and end.

```
print('She said, "I like Python!"')
```

What if you want to include both an apostrophe and quotes in your string? You can use something called escape characters - more on that later. (See the section below on escape characters!)

Concatenation

Strings can be stuck together (concatenated) using the `+` operator. For example, `'cat' + 'dog'` will create the string `'catdog'` and `'2' + '3'` will give `'23'` . Note that this is a string and should not be confused with the number `23` .

Escape Characters

You can do some logical comparisons of strings (equal to, not equal to). Python has lots of string functions built in - this means that they have already been coded and included in python and we can use them by calling them (typing their name) and setting

their parameters like this:

```
string_function_name(parameter)
```

Some string functions take the string as their only parameter, others need further parameters, which we will look at in the examples.
All of these functions return a value. We will start by storing these values in variables, but will progress on to skipping that step and using the functions directly in output, selection etc.
The advantages of using built in functions are that it saves us time, and that these functions have already been pre-tested so we know that they will work.
There are lots of different string functions, we are just going to use some of the most common ones.

1 - Len - returns the number of characters in a string.

TEACHER NOTES
The len() function returns the number of characters in a string. It takes the string as its parameter.
It's one of the simplest built in functions so we will use it to introduce the skill.
Space is counted as a character, as is any punctuation.

This set of tasks will also introduce the concept of coding more efficiently by including the function in the print or selection condition. This can be applied to all of the subsequent functions (examples 3b & 4b), higher ability students will probably prefer this. If students struggle to conceptualise this then encourage them to call the function and return the result into a variable (examples 3a and 3b).
Examples

Example 1 - The function on its own
len("Banana")
Returns 6.

Example 2 - Storing the returned value in a variable

```
# Returns the number 6 into the variable wordLength
wordLength = len("Banana")
# Outputs the data in the wordLength variable
print(wordLength)
```

Example 3a- Getting input and finding out its length

```
# Gets input and stores it in the word variable
word = input("Enter a word")
# Returns the length of the string in the word variable into the variable wordLength
wordLength = len(word)
```

Outputs the data in the wordLength variable

print(wordLength)

Example 3b - A more efficient way of coding example 3a

Gets input and stores it in the word variable

word = input(“Enter a word”)

Uses the len() function as part of the print() function. Removes the need for the # wordLength variable.

print(len(word))

Example 4a - len with selection

Gets input and stores it in the word variable

word = input(“Enter a word”)

Returns the length of the string in the word variable into the variable wordLength

wordLength = len(word)

Outputs a message if the string in wordLength has more than 50 characters in it.

If wordLength > 50:
print(“Wow, what a long word!”)

Example 4b - a more efficient way of coding

Gets input and stores it in the word variable

word = input(“Enter a word”)

Uses the len() function as part of the condition. Outputs a message if the string # in wordLength has more than 50 characters in it.

If len(word) > 50:
print(“Wow, what a long word!”)

Tasks
Len - Predict and Run

Task - https://repl.it/@MrAColley/11-len-1
Example solution - https://repl.it/@MrAColley/11-len-1-solution

Task

Add comments to each line of the code to predict what it will do. Run it to see if your predictions were correct.

wordLength = len("Hello World!")

print(wordLength)

word = "I love strings!"

wordLength = len(word)

print(wordLength)

Len - Investigate & Modify

This task requires students to finish incomplete code.

Task - https://repl.it/@MrAColley/12-Len-2
Example solution - https://repl.it/@MrAColley/12-Len-2-solution

Task

Complete the code below using the comments to tell you what to do.abs

Get input from the user and store it in a variable called word.

input("Enter a word")

Get the length of the word variable and store it in the wordLength variable

wordLength =

Output the wordLength variable

print

Len With Selection

Task - https://repl.it/@MrAColley/13-len-with-selection
Example solution - https://repl.it/@MrAColley/13-len-with-selection-solution

Task

Add comments to explain what the code below should do.
#Run it to see if your predictions were correct
word = input(“Enter a word")
wordLength = len(word)
if wordLength > 50:
print(“Wow! That’s a long word!”)

Task

Copy the code from above. Modify it so that it checks to see if the length of the word is less than 4 and outputs a suitable message if it is.

Len - Make

Task instructions - https://repl.it/@MrAColley/14-Len-Independent-Task
Example solution - https://repl.it/@MrAColley/14-Len-Independent-Task-solution

Write a program that:

Asks the user to input an 8 letter word.
Stores the input in a variable.
Calculates the length of the word input.
If the word is more than 8 characters, output ‘Too long’
If the word is less than 8 characters, output ‘Too short’
If the word is 8 characters, output ‘Perfect, thank you!’

2 - GetChar - Slice - Returns one character from a string.
Teacher Notes

Get Char returns one character from a string. Python uses slices to do this.
Slices treat strings like lists. Each character is given an index number starting with 0.
The get char slice uses one index, the index of the character returned.
The slice index comes in brackets after the string or the variable containing the string.

Examples

Outputs the characters ‘p’

print(“Computing”[3])

Gets input and stores it in the word variable

word = “dalmatian”

#returns the character ‘t’ (character at index 5 of the word variable) into the letter variable
letter = word[5]

Tasks

GetChar - Predict & Run
Task instructions - https://repl.it/@MrAColley/15-GetChar-Predict-and-Run
Example solution - https://repl.it/@MrAColley/15-GetChar-Predict-and-Run-Example-Solution

GetChar - Investigate & Modify
Task instructions - https://repl.it/@MrAColley/2106-GetChar-Investigate-Modify
Example solution - https://repl.it/@MrAColley/2106-GetChar-Investigate-Modify-Solution

GetChar - Make

Students will need to convert the number input to int using int(input(“Enter a number”) and convert the int back to a string when outputting using print("The letter at position " + str(num1)

Example solution - https://repl.it/@MrAColley/2106-GetChar-Investigate-Modify-Solution

Write a program that:

Asks the user to input a word and stores it in a suitably named variable
Asks the user to input a number and stores it in a suitably named variable.
If the number entered is larger than the length of the word input then output an error message.
Else output the character from the word at the position input as part of a sentence.

Eg for inputs ‘Jimi’ and ‘2’ the program outputs ‘The letter m is at position 2 in your name’.
For inputs ‘Jimi’ and ‘6’ the program outputs ‘The number you entered is too large’.

3 - Slice/Substring - Returns a part of a string
Teacher Notes

A substring returns a sequence of characters from a string. Python uses slices to do this.
Slices treat strings like lists. Each character is given an index number starting with 0.
The slice uses two indexes, the start character and the end character (NOTE the end character is NOT INCLUDED in the slice).
The slice indexes come in brackets after the string or the variable containing the string.

Examples:

Outputs the characters ‘ompu’

print(“Computing”[1:5]

Gets input and stores it in the word variable

word = “dalmatian”

#returns the characters ‘ma’ (characters at indexes 3 and 4)
subString = word[3:5]

Tasks

Substring - Predict & Run
Task instructions - https://repl.it/@MrAColley/15-Substrings-1
Example solution -https://repl.it/@MrAColley/17-Substrings-Predict-Run-Example-Solution

Substring - Investigate & Modify
Task instructions - https://repl.it/@MrAColley/18-Substrings-Investigate-Modify
Example solution - https://repl.it/@MrAColley/18-Substrings-Investigate-Modify-Example-Solution

Substring - Make

Example solution - https://repl.it/@MrAColley/2110-Substrings-Make-Solution

Write a program that:

Asks the user to input a phrase and stores it in a suitably named variable
Asks the user to input a number between 0 and the length of the phrase and stores it in a suitably named variable.
Asks the user to input a second number between the first number and the length of the phrase and stores it in a suitably named variable.
Gets and outputs the substring of characters between the two numbers entered

Eg for inputs 'I love Computing' and numbers '3' and '7' the output would be 'ove'

4 - Change Case - converts lower case to CAPS and vice versa
Teacher Notes
These technique use a methods rather than functions, which is why the parameter doesn't go in the brackets after the name. However they're so useful that I've included them here anyway. This is how to get the computer to ignore case, which means that the user can input any combination of caps/non caps. Extremely useful when getting input where case doesn't matter, such as for usernames.

Whatever the string is, it will be converted to upper case when string.upper() is used.
Whatever the string is, it will be converted to lower case when string.lower() is used.
The original string can be in any combination of upper/lower case, it doesn't have to be all caps or all lowercase.
Just like with other functions we've used, this technique can be combined with conditions used in selection and iteration.

The tasks will start with us converting and assigning to variables, before using these variables in conditions. Better students will be able to get rid of the variable by using the method in line with the condition.
Examples:

Converts 'hello world!' to uppercase.

"hello world".upper()

Outputs 'HELLO WORLD!'

print("hello world".upper())

Converts contents of the 'word' variable to uppercase.

word.upper()

Outputs the data in the ‘word’ variable in uppercase.

print(word.upper())

Converts ‘HELLO WORLD!’ to lowercase.

“HELLO WORLD”.upper()

Using with a condition.

Gets user input as answer to a question.

guess =

Tasks

Change case - Predict & Run
Task instructions - https://repl.it/@MrAColley/2111-Change-Case-Predict-RUN
Example solution - https://repl.it/@MrAColley/2111-Change-Case-Predict-RUN-Solution

Change case - Investigate & Modify
Task instructions - https://repl.it/@MrAColley/2112-Cahnge-Case-Investigate-Modify
Example solution - https://repl.it/@MrAColley/2112-Change-Case-Investigate-Modify-Solution

Change case - Make

Be careful that students do not ignore the case for the passwords. Passwords are always case sensitive.

Task instructions - https://repl.it/@MrAColley/2113-Change-Case-Make
Example solution - https://repl.it/@MrAColley/2113-Change-Case-Make-Solution