Strings

Strings in python are surrounded by either single quotation marks, or double quotation marks.

'hello' is the same as "hello".

You can display a string literal with the print() function:

Example

```
print("Hello")

print('Hello')
```

Assign String to a Variable

Assigning a string to a variable is done with the variable name followed by an equal sign and the string:

Example

```
a = "Hello"

print(a)
```

Multiline Strings

You can assign a multiline string to a variable by using three quotes:

Example

You can use three double quotes:

```
a = """Lorem ipsum dolor sit amet,

consectetur adipiscing elit,

sed do eiusmod tempor incididunt

ut labore et dolore magna aliqua."""

print(a)
```

Or three single quotes:

Example

```
a = '''Lorem ipsum dolor sit amet,

consectetur adipiscing elit,

sed do eiusmod tempor incididunt

ut labore et dolore magna aliqua.'''

print(a)
```

Strings are Arrays

Like many other popular programming languages, strings in Python are arrays of bytes representing unicode characters.

However, Python does not have a character data type, a single character is simply a string with a length of 1.

Example

```
a = "Hello, World!"

print(a[1])
```

Looping Through a String

Since strings are arrays, we can loop through the characters in a string, with a for loop.

Example

```
for x in "banana":

  print(x)
```

String Length

To get the length of a string, use the len() function.

Example

The len() function returns the length of a string:

```
a = "Hello, World!"
```

```
print(len(a))
```

Check String

To check if a certain phrase or character is present in a string, we can use the keyword in.

Example

```
txt = "The best things in life are free!"

print("free" in txt)
```

Use it in an if statement:

Example

```
txt = "The best things in life are free!"

if "free" in txt:

  print("Yes, 'free' is present.")
```

Check if NOT

To check if a certain phrase or character is NOT present in a string, we can use the keyword not in.

Example

```
txt = "The best things in life are free!"

print("expensive" not in txt)
```

Use it in an if statement:

Example

```
txt = "The best things in life are free!"

if "expensive" not in txt:

  print("No, 'expensive' is NOT present.")
```

Slicing

You can return a range of characters by using the slice syntax.

Specify the start index and the end index, separated by a colon, to return a part of the string.

Example

```
b = "Hello, World!"

print(b[2:5])
```

Slice From the Start

By leaving out the start index, the range will start at the first character:

Example

```
b = "Hello, World!"

print(b[:5])
```

Slice To the End

By leaving out the end index, the range will go to the end:

Example

```
b = "Hello, World!"

print(b[2:])
```

Negative Indexing

Use negative indexes to start the slice from the end of the string:

Example

From: "o" in "World!" (position -5)

To, but not included: "d" in "World!" (position -2):

```
b = "Hello, World!"

print(b[-5:-2])
```

Upper Case

Example

The upper() method returns the string in upper case:

```
a = "Hello, World!"

print(a.upper())
```

Lower Case

Example

```
a = "Hello, World!"

print(a.lower())
```

Remove Whitespace

Whitespace is the space before and/or after the actual text, and very often you want to remove this space.

Example

The strip() method removes any whitespace from the beginning or the end:

```
a = " Hello, World! "

print(a.strip())
```

Replace String

Example

The replace() method replaces a string with another string:

```
a = "Hello, World!"

print(a.replace("H", "J"))
```

Split String

The split() method returns a list where the text between the specified separator becomes the list items.

Example

The split() method splits the string into substrings if it finds instances of the separator:

```
a = "Hello, World!"

print(a.split(","))
```

String Concatenation

To concatenate, or combine, two strings you can use the + operator.

Example

Merge variable a with variable b into variable c:

```
a = "Hello"

b = "World"

c = a + b

print(c)
```

Example

```
        a = "Hello"

        b = "World"

        c = a + " " + b

        print(c)
```

String Format

As we learned in the Python Variables chapter, we cannot combine strings and numbers like this:

Example

```
age = 36

txt = "My name is John, I am " + age

print(txt)
```

But we can combine strings and numbers by using the format() method!

The format() method takes the passed arguments, formats them, and places them in the string where the placeholders {} are:

Example

Use the format() method to insert numbers into strings:

```
age = 36

txt = "My name is John, and I am {}"

print(txt.format(age))
```

The format() method takes unlimited number of arguments, and are placed into the respective placeholders:

Example

```
        quantity = 3

        itemno = 567

        price = 49.95

        myorder = "I want {} pieces of item {} for {} dollars."

        print(myorder.format(quantity, itemno, price))
```

You can use index numbers {0} to be sure the arguments are placed in the correct placeholders:

Example

```
quantity = 3

itemno = 567

price = 49.95

myorder = "I want to pay {2} dollars for {0} pieces of item {1}."

print(myorder.format(quantity, itemno, price))
```

Escape Character

To insert characters that are illegal in a string, use an escape character.

An escape character is a backslash \ followed by the character you want to insert.

An example of an illegal character is a double quote inside a string that is surrounded by double quotes:

Example

```
txt = "We are the so-called "Vikings" from the north."

print(txt)
```

To fix this problem, use the escape character \":

Example

```
txt = "We are the so-called \"Vikings\" from the north."

print(txt)
```