Strings

Carlos Tavares

November 1, 2024

Strings

Strings i

We have been dealing with the notion of text since the beggining of the course:

» x = "something"

» x = input ("Write something")

» if x == "anything":

    ....

Text are formally known as strings, or chain of characters and there is a wide range of specific functions to deal with strings.

Strings ii

What is a **string**?

It is a collection of characters.

What is a **character?**

A "letter" that can be used to compose text in python, for instance: a, A, z, 0, 9, +, -, %.

A more extensive list: all characters [a-z], [A-Z], $\{*, -, +, =, \&, ?, !, , (,), [,], ...\}$

Strings iii

How many characters there are?

All standard characters are contained in the standard ascii table, which possesses 255 characters:

https://www.asciitable.com

However, it was necessary to extend this list of characters, hence the utf-8 and utf-16 formats were created.

A character is stored as a number of 8 bits (byte) in memory, when using ascii. When using UTF the character is represented by 1 to 5 bytes.

Strings iv

A multiline string (encoded with three quotes or apostrophes)

» x = """This is a multiline
String"""

Declaring an unicode string:

» u"A utf string"

Special characters

$\backslash n$, $\backslash t$, $\backslash b$

s = "This is the $\backslash n$ class of programming"

How to deal with strings in python i

Strings behave like tuples in python: they allow *random access* and are immutable

» s = "Aerospace class"

It is possibe to access by index: **s [index]**

Example:
» s [0]
'A'
» s [0] ='B'
...
TypeError: 'str' object does not support item assignment

How to deal with strings in python ii

It is possible to access strings using intervals **s [i:j]**

» s [4:9]
"space"

» s [1:]
?

» s [:3]
?

How to deal with strings in python iii

The operator in: substring is part of string.

» 'A' **in** s
True
» ' Aerospace' **in** s
True
» 'Water' **in** s
False

Strings are iterable:

```
for i in s:
        print (i)
```

Operations with strings i

It is possible to build strings out of many other objects through the constructor **str ()**

Allows the construction of strings from any **kind of number:**

```
» s = str (12.3)
» s
"12.3"
```

Allows the construction from a **tuple**:

```
» t = (12, "Exp", 14)
» s = str (t)
» s
"(12, "Exp", 14)"
```

Operations with strings ii

Allows the construction from a **list**:

```
» l = [1, 2, 3, 14]
» s = str (l)
» s
"[1, 2, 3, 14]"
```

The print function internally uses this constructor to print objects. The object can printed as long python knows how to build a string out from the object.

Operations with strings iii

Concatenating strings: the operator "+"

```
» s = ""
» s = s + "Experiencia 1"
» s = s + "Experiencia 2"
» print (s)
Experiencia 1;Experiencia 2
```

It is also possible to create copies of strings

```
» s *= 2
» s
'Experiencia 1;Experiencia 2Experiencia 1;Experiencia 2'
```

Exercises

Exercise 1: Make a program to remove from a string a specific set of characters given by another string.

Exercise 2: Make a program to duplicate the characters of a string, specified in another string.

Exercise 3: Consider a list of tuples with the following shape: (number_of_copies, interval_lower_end, interval_upper_end). Build a program to build copies of the substrings of a string, where the number of copies, and the intervals, are given by the list of tuples.

Important functions about strings i

Separate a string into substrings, by using a separator (funtion **split**):

string_variable.split (separator)

Example:

»s = "This isn’s a phrase that makes sense, just an example of a sentence"
» result = s.split (" ")
» result

Important functions about strings ii

Create a string from substrings, using a string as separator (function **join**):

string_variable.join (list_of_strings)

Example:

```
» j = " "
» rejoined = j.join (result)
» rejoined
```

Important functions about strings iii

Find and replace methods

The **find** method:

string_variable.find (substr, begin, end)

Example:
»s = "Some string with an interesting word in the middle"
»index = s.find ("word")
» index
32

Important functions about strings iv

The **replace** method:

string_variable.find (old, new, [max_replacements])

Example:
» s = "Someday is going to be rainy"
» s.replace ("day", "where")
"Somewhere is going to be rainy"

Important functions about strings v

There are a bunch of relevant about direct character manipulation.

isalpha()	upper ()
isdigit()	swapcase ()
isspace()	lower ()
islower	

Reference:
https://python-reference.readthedocs.io/en/latest/docs/str/

Exercises

Exercise 1: Make a program to count:

- The number of space;
- The number of digits;
- The number of alpha numeric characters.

Exercise 2: Make a program to verify if a word is a palindrome.

Questions?