Cmpe 150 Lab 9: Strings

Previous Weeks

- We learned about variables, math and logical operators, functions, for loops, conditionals, and while loops.

- Today, we will focus on strings.

Strings

- They correspond to text, which is simply a list of characters.
- We can define a string using "Example String" or x = 'Another string'
- We all know these. Are we done?

Multiline Strings

- Use double """ like """Something"""

  """İlk satır

  Satır 2

  The last line"""

Special/Escape Characters

- '\n' -> New line
- x = 'This is a string\nconsisting of two lines.'
- '\t' -> Tab
- x = 'This is a string\twith a lot of space.' and
- x = 'This is a string\t\t\twith more space.'
- How to really say \ then? Does '\' work?
- '\\'
- Also, we will learn a number of string functions.

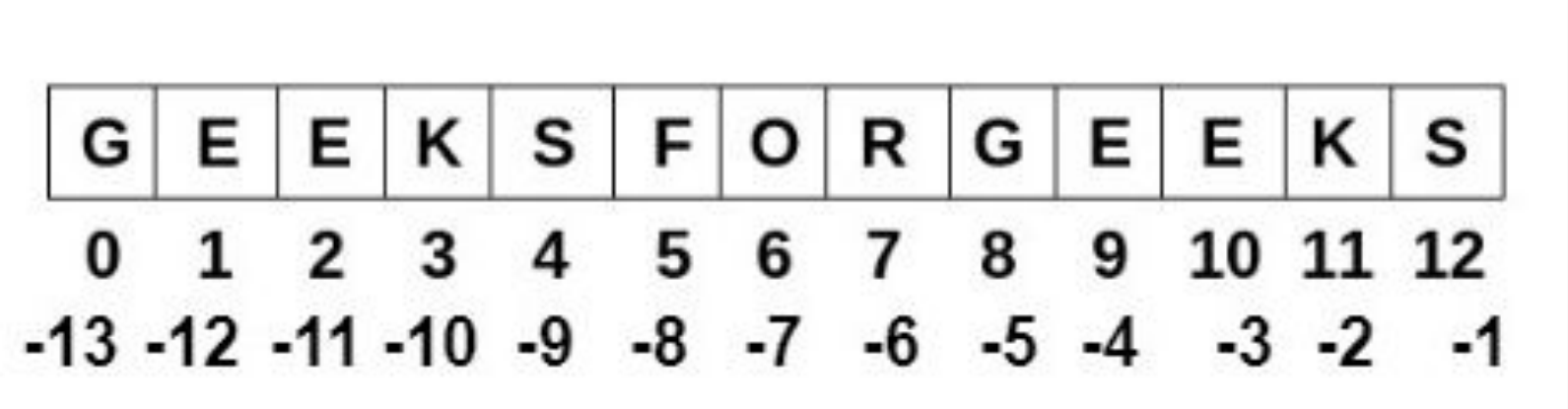
Some Functions of Strings

- x = "The quick brown fox jumps over the lazy dog"
- x.lower() -> "the quick brown fox jumps over the lazy dog"
- x.upper() -> "THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG"

Accessing Characters

- Get a certain character
- x = "The quick brown fox jumps over the lazy dog"
- x[0] -> "T"
- x[7] -> "c"
- x[-1] -> "g"
- x[-3] -> "d"

Indexing Starts from 0



len and in

- len(x) gives the length of the string
- print(len("This is a string with length 31")) or
- len_of_str = len("This is a string with length 31")

- To check if a string is a substring in another one, use in.
- "str" in "This is a string with length 31" -> True
- "str " in "This is a string with length 31" -> False

Iterating over a String Using a Loop

```
for i in range(len(x)):
        print(x[i])
```

or

```
for character in x:
        print(character)
```

String Slicing

- Get a substring (a portion)
- x = "The quick brown fox jumps over the lazy dog"
- x[4:7] -> qui
- y = x[4:-3] -> quick brown fox jumps over the lazy
- "The quick brown fox jumps over the lazy dog"[4:] ->
- quick brown fox jumps over the lazy dog
- x[:-2] -> The quick brown fox jumps over the lazy d

Reversing a String and Comparing Strings

- x = "The quick brown fox jumps over the lazy"
- x_rev = x[::-1]
- print(x_rev) -> "yzal eht revo spmuj xof nworb kciuq ehT"

- == works for string equality as well
- print("The quick brown fox jumps" == "The quick brown fox jumps ")

Comparing Strings (Cont.)

- print("car" < "dog")
- print("car" < "Dog")
- print("car" < "321sda")

- The ASCII table is the reference to compare characters https://www.asciitable.com/

Concatenation and Multiplication (Repeat)

- x = "First part"
- y = "Second part"
- z = x + y -> "First partSecond part"

- x = "First part" * 6
- y = 14 * "Second part"
- z = x + y -> See what happens

strip and replace

- x = " a string with spaces at the beginning and at the end   .   "
- print(x.strip())
- "a string with spaces at the beginning and at the end   ."

- x = "this is my old_substr and it contains the word old_substr"
- y = x.replace("old_substr", "new_substr")
- this is my new_substr and it contains the word new_substr

find

- x = "The quick brown fox jumps over the lazy dog"
- index = x.find("brown") -> 10 (starting from 0)

- If not found in the original string, it returns -1
- print("brown".find("fox"))

split

- Divides the string into smaller pieces using white space, tabs and new lines.
- x = "The quick brown fox jumps over the lazy dog"
- words = x.split()

- IT RETURNS A LIST
- print(words)
- Again we can access them using indexing like words[0] or words[-3]

split (Cont.)

- We can provide a **different** string to divide the original one.
- x = "this little black dress isn’t expensive."
- tokens = x.split("t") or tokens = x.split(" ")

- print(tokens)

join

- Connects a list of strings by using the given string

- x = "The quick brown fox jumps over the lazy dog"
- words = x.split()
- tab_separated_str = "\t".join(words)
- print(tab_separated_str)

And Others

- We also have other functions, yet these ones ought to be sufficient for now.

- If you want to learn more, check https://www.w3schools.com/python/python_ref_string.asp

Thanks

Any questions?

References

1. https://www.w3schools.com/python/python_strings.asp
2. https://www.geeksforgeeks.org/python-string/
3. https://www.w3schools.com/python/python_ref_string.asp