Scientific Programming With Python

Collections: Strings and Regular Expressions

UNIVERSITY of HOUSTON
DIVISION OF RESEARCH
HEWLETT PACKARD ENTERPRISE DATA SCIENCE INSTITUTE

Collection Data Types

- ints, floats, bool, complex …. are all scalar types
  - Store only one value
- Collection objects can hold more than one value
- Two kinds of collections, based on how the values are accessed
  - Sequence: access by positional index
    - (str)ing, list, tuple
  - Mapped: access by key
    - (dict)ionary

Collections and Strings

A string is a *collection* data type – those are composed of smaller pieces

- **as are `lists, tuples, dictionaries`**
- **`int, float, bool` are primitive data types**

A string is a sequential *collection of characters*

- 'Hello World!' or `"Hello World!"`
- `Or an empty string ''`

String operations

Addition and multiplications have different meanings:

```
lastname= 'Doe'
firstname = 'John’
fullname = firstname + lastname    → 'JohnDoe'
silly = 3*lastname                 → 'DoeDoeDoe'

firstname-1 or '34'+2         -> are illegal/not aqllowed
```

String Indexing

Index of an item is a position of the item in a string

```
s = 'Python'
```

`s[0] == 'P', s[1]=='y' … s[5] == 'n'`

Interestingly, a negative index is used to specify a position with respect to the "end"

The last item has index -1,

The second to last item has index -2,…

```
s[-1]  == 'n'
s[-3]  == 'h'
```

String Methods

- Strings are objects with attributes and methods.
- `ss = 'PythonGood'`
-
- `ss.upper()` → `PYTHONGOOD`
- `ss.lower()` → `pythongood`
- `ss.count('o')` → `3`
- `ss.find('o')` → `4`
- `ss.rfind('o')` → `8`

String Methods

Methods that return bool: True or False

Method	Description
isalnum()	Returns True if all characters in the string are alphanumeric
isalpha()	Returns True if all characters in the string are in the alphabet
isdecimal()	Returns True if all characters in the string are decimals
isdigit()	Returns True if all characters in the string are digits
isidentifier()	Returns True if the string is an identifier
islower()	Returns True if all characters in the string are lower case
isnumeric()	Returns True if all characters in the string are numeric
isprintable()	Returns True if all characters in the string are printable
isspace()	Returns True if all characters in the string are whitespaces
istitle()	Returns True if the string follows the rules of a title
isupper()	Returns True if all characters in the string are upper case

Source https://www.w3schools.com/

String Methods

Methods that return bool -> True or False

Method	Description
endswith()	Returns True if the string ends with the specified value
startswith()	Returns True if the string starts with the specified value

Source https://www.w3schools.com/

String Methods

Methods that return a modified view of the string

Method	Description
capitalize()	Converts the first character to upper case
casefold()	Converts string into lower case
lower()	Converts string into lower case
upper()	Converts a string into upper case
title()	Converts the first character of each word to upper case
swapcase()	Swaps cases, lower case becomes upper case and vice versa
translate()	Returns a translated string
rjust()	Returns a right justified version of the string
ljust()	Returns a left justified version of the string
zfill()	Fills the string with a specified number of 0 values at the beginning

Source https://www.w3schools.com/

String Functions and Operators

- `ss = ` 'PythonGood'
- Length function `len(ss)` → `10`
- String slices `([n:m]-- substring from n to m-1)`
- `ss[0:6]` → `'Python'`
- `ss[6:10]` → `'Good'`
- in and not in (if one string is a substring of other)
- `'n' in 'Python'` → `True`
- `'n' not in 'Python'` → `False`
- `'' in 'Python'` → `True`
- `'Python' in 'Python'` → `True`

String Comparison

```
ss = 'PythonGood'
     ss == 'PythonGood'          True
     ss == 'pythongood'          False
     'Python'< 'Java'   ?
     'Python' < 'Scala'?
     'Python' < 'python' ?
     (lexicographic)
ord() and chr() functions
>>> ord('a')
 97
>>> chr(97)
 'a'
```

Strings are Immutable

- Elements of strings cannot be modified
- `ss =` 'PythonGood'
- `ss[0] =` 'p' → `error`

- However
- `newss =` 'p' `+ ss[1:10]` → 'pythonGood'

Strings Constants

- provided by `string` module

- `string.ascii_lowercase`
- `string.ascii_uppercase`
- `string.digits`
- `string.punctuations`

Strings and regular expression

UNIVERSITY of HOUSTON
DIVISION OF RESEARCH
HEWLETT PACKARD ENTERPRISE DATA SCIENCE INSTITUTE

Matching simple patterns with string

String class has methods for counting and finding position of simple patterns

```
S1 = "I  do not see a  pattern here, I repeat I do not see the pattern"

print(S1.count('pattern'))

        2

print(S1.find('pattern'))

        17

print(S1.rfind('pattern'))

        57
```

Matching simple patterns with string

String pattern matches are limited to simple patterns

Alternative is to use Regular Expressions

Provided by “re” library

import re

Regular Expression

A regular expression (or RE) specifies a set of strings that matches it. Useful for creating search patterns and finding/counting matches

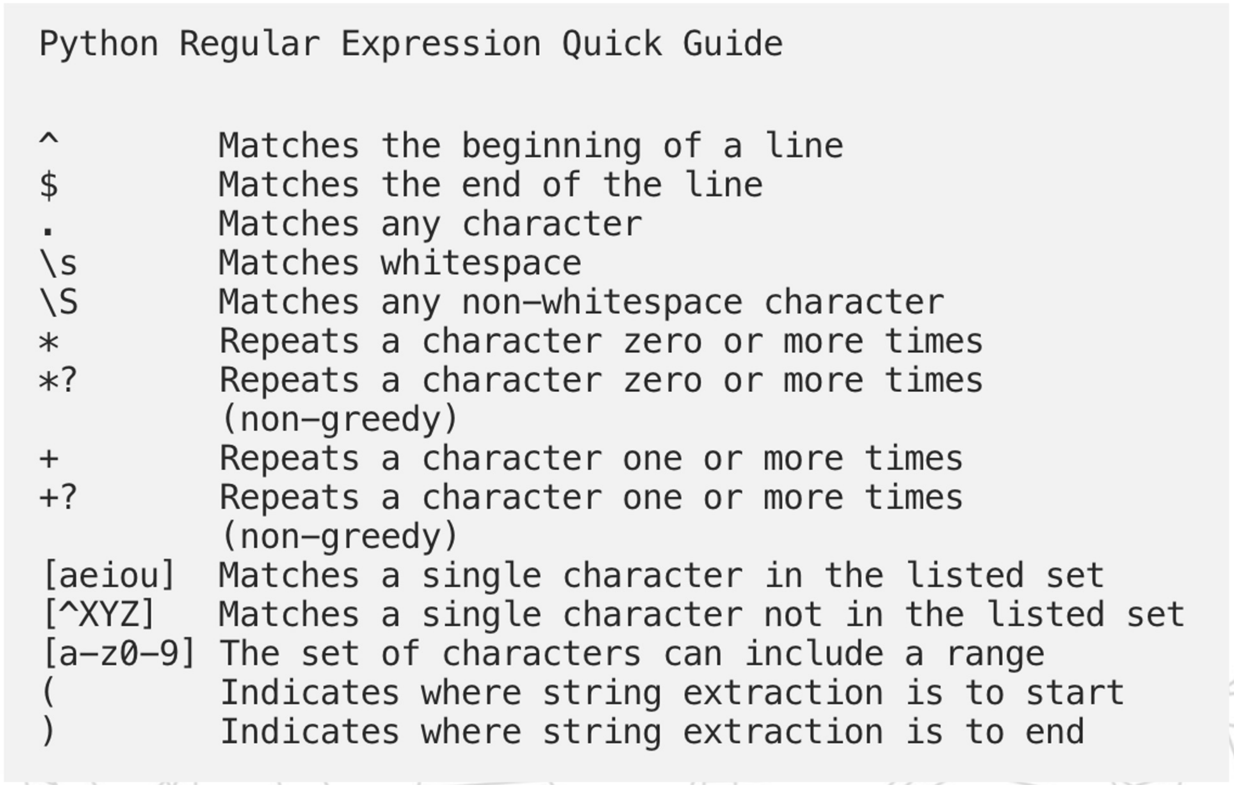
The functions in “re” module lets you:

- check if a particular string matches a given a specific string pattern i.e. regular expression

UNIVERSITY of HOUSTON
DIVISION OF RESEARCH
HEWLETT PACKARD ENTERPRISE DATA SCIENCE INSTITUTE

Regular Expression, findall method

```
import re

pattern='G...T.'

DNA_SAMPLE=”ATATATGGTGGTGGAAAAGATCAACAATTAGGAAGATCTTATAGAGAAGTTATGAATACTAAATAC
AATAATAAGAAGAGCGCATTATTCTGAAAATTTTAAATTTAAAGATAGCAA”

search_result = re.findall (pattern, DNA_SAMPLE)

print (search_result)

['GTGGTG', 'GATCTT', 'GAAGTT', 'GCATTA']
```



```
Python Regular Expression Quick Guide

^         Matches the beginning of a line
$         Matches the end of the line
.         Matches any character
\s        Matches whitespace
\S        Matches any non-whitespace character
*         Repeats a character zero or more times
*?        Repeats a character zero or more times
          (non-greedy)
+         Repeats a character one or more times
+?        Repeats a character one or more times
          (non-greedy)
[aeiou]   Matches a single character in the listed set
[^XYZ]    Matches a single character not in the listed set
[a-z0-9]  The set of characters can include a range
(         Indicates where string extraction is to start
)         Indicates where string extraction is to end
```

Regular Expression, findall method

```
import re

DNA_SAMPLE=”ATATATGGTGGTGGAAAAGATCAACAATTAGGAAGATCTTATAGAGAAGTTATGAATACTAA
ATACAATAATAAGAAGAGCGCATTATTCTGAAAATTTTAAATTTAAAGATAGCAA”

search_result = re.findall('^A..TA',DNA_SAMPLE)

print (search_result)
```

```
 ['ATATA']
```

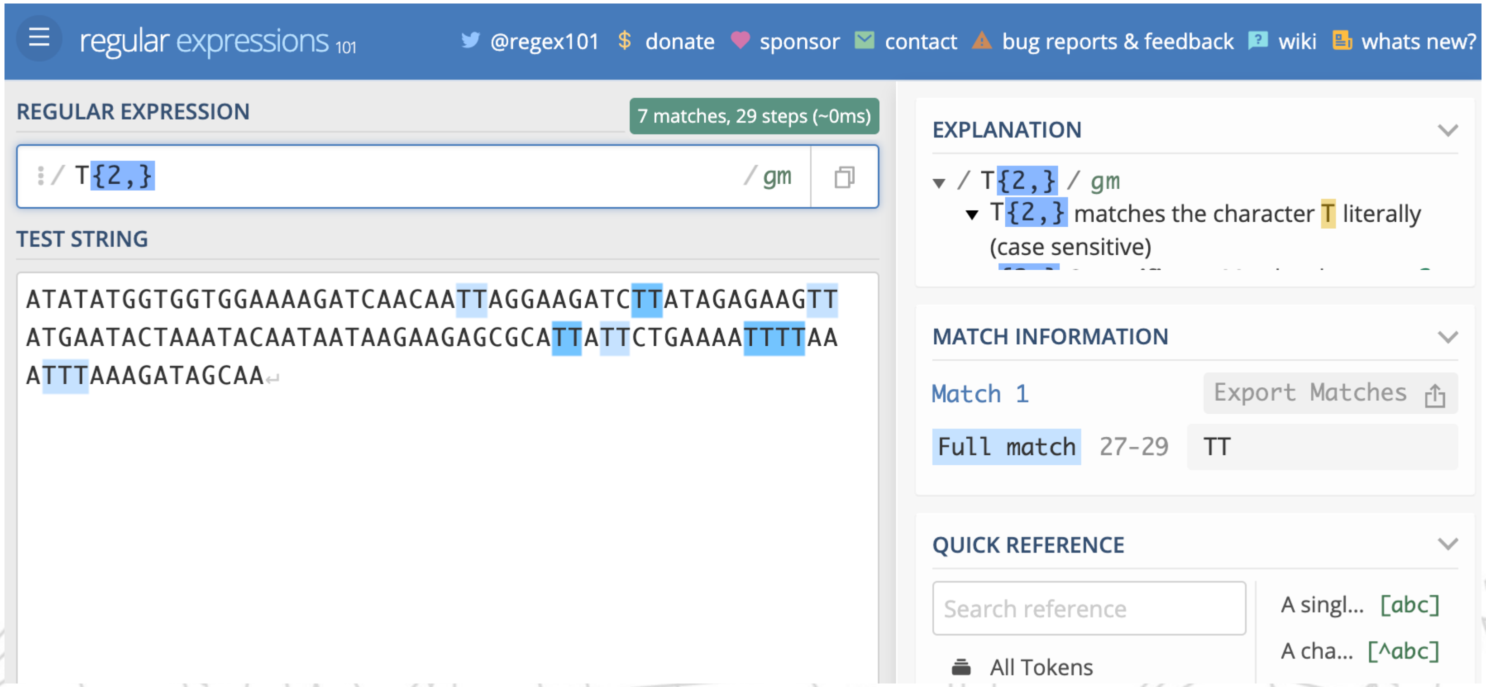
Free online tool(s) for generating and verifying regex

https://regex101.com/

https://www.regextester.com/

https://regexr.com/

Regex 101 https://regex101.com/



regular expressions 101
@regex101 donate sponsor contact bug reports & feedback wiki whats new?

REGULAR EXPRESSION

7 matches, 29 steps (~0ms)

```
/ T{2,} / gm
```

TEST STRING

```
ATATATGGTGGTGGAAAAGATCAACAATTAGGAAGATCTTATAGAGAAGTT
ATGAATACTAAATACAATAATAAGAAGAGCGCATTATTCTGAAAATTTTAA
ATTTAAAGATAGCAA
```

EXPLANATION

/ T{2,} / gm

T{2,} matches the character T literally (case sensitive)

MATCH INFORMATION

Match 1 Export Matches

Full match	27-29	TT

QUICK REFERENCE

Search reference

All Tokens

A singl... [abc]

A cha... [^abc]