

Unicode

Code point higher than 127

"\u0915" or "क"

File Modes

	r	r+	w	w+	a	a+
read	*	*		*		*
write		*	*	*	*	*
create			*	*	*	*
truncate			*	*		
position at start	*	*	*	*		
position at end					*	*

Reading a File

Opens and closing a file

file = open(file_name, encoding = 'utf-8', 'r')

with open(file_name, "r") as file:

file.close()

Reading a file

Reads entire file as a string
__content=f-
ile.read()

Reads the first 10 characters
__content =
file.read(10)

Stores contents as a list of lines
__file.readlines()

Reading a File (cont)

Read next line as a string
__file.readline()

Iterating through lines

for line in lines:

strip to remove
__print(line.strip())
newline charc

Writing to a text file

with open(file_name,
'w') as file:

writing one line
__file.write("Hello,
World!\n")

writing list of lines
__file.writelines(list)

Appending to a text file

with open(file_name, 'a') as file:

__file.write("\nAppending a new line.")

Checking file existence

import os

os.path.exists(file_name)

Handling file exceptions

try:

__with open(file) as f:

__content = f.read()

__print(content)

except FileNotFoundError:

Reading a File (cont)

__print("File not found")

except Exception as e:

__print(f"An error occurred: {e}")

Telling/seeking in a file

with open(file)
as f:

Gives current position
__f.tell()

Move the cursor to the beginning
__f.seek(0)

csv Library

import csv

Reading CSV file

returns a list

Writing to CSV file

writer = csv.writer(file)

writer.writerows(list)

Loading JSONs

import json

Loading json file

json_data = json.loads(json_file)

Change json to string

string = json.dumps(data, indent=2)



By leenmajz

cheatography.com/leenmajz/

Published 15th March, 2024.

Last updated 15th March, 2024.

Page 1 of 3.

Sponsored by **ApolloPad.com**

Everyone has a novel in them. Finish

Yours!

<https://apollopadd.com>

Random library

Returns a random float in the range[0,0,1)

Return a random float in the range[a,b] `random.uniform(a,b)`

Returns an integer in the range [0,b)

Returns an integer in the range [a,b) skipping c steps

Returns an integer in the range [a,b] `random.randint(a,b)`

Randomly change position of elements `print (random.shuffle-(luck))`

Random Choices

Uniformly randomly picks one item from a list `random.choice(list)`

`list[randrange(4)]`

Selects k items without repetition `random.sample(list, k=2)`

Selects k items with repetition `random.choices(list, k=2)`

Random Choices (cont)

Selects k items without uniformity may repeat `random.choices(list, weights=[10,-150,20], k =2)`

These methods also work on strings

Raw Strings

Raw Strings

r-strings `r"A raw string"`

only a single backslash not valid

odd number of ending backslash not valid

Regular Expressions- Patterns

Regular- Expression Patterns

^ Matches beginning of line.
\$ Matches end of line.
. Matches any single char except newline.
[...] Matches any single char in brackets.
[^...] Matches any single char not in brackets.
\w Matches word characters.
\W Matches nonword characters.
\s Matches whitespace.
\S Matches nonwhitespace.
\d Matches digits.
\D Matches nondigits.
\A Matches beginning of string.
\Z Matches end of string.
\z Matches end of string.
\G Matches point where last match finished.
x|y Matches either x or y.
[0-9] Match any digit; same as [0123456789]
[a-z] Match any lowercase ASCII letter
[A-Z] Match any uppercase ASCII letter
[a-zA-Z0-9] Match any of the above
[^aeiou] Match any other than a lowercase vowel
[^0-9] Match anything other than a digit.

Regular - Expression Patterns (continued)

*
+ Match its preceding element one or more times.

?
{n}
{n ,m} Match its preceding element from n to m times.

re Methods

Searches the string for a match and returns a Match object `re.match(-pattern, string)`

Searches for the first occurrence of the pattern anywhere in the string `re.search-(pattern, string)`

Finds all occurrences of the pattern in the string returns a list `re.findall(pattern,string)`

Returns an iterator yielding match objects for all matches. `re.finditer(pattern, string)`



By [leenmajz](https://cheatography.com/leenmajz/)
cheatography.com/leenmajz/

Published 15th March, 2024.
Last updated 15th March, 2024.
Page 2 of 3.

Sponsored by [ApolloPad.com](https://apollopadd.com)
Everyone has a novel in them. Finish Yours!
<https://apollopadd.com>

re Methods (cont)

Replaces matching substrings with new string for all occurrences or a specified number	<code>re.sub(pattern, replacement, string)</code>
Splits the string where there is a match and returns list of strings based on splits	<code>re.split(pattern, string)</code>

Get requests using requests

```
import requests
url = "https://www.wikipedia.org/"
r = requests.get(url)
text = r.text
```

Webscrapping

```
from bs4 import BeautifulSoup
# Parse HTML stored as a string
# 'html5lib' 'html.parser' or 'lxml'
soup = BeautifulSoup(html, 'html5lib')
# Returns formatted html
soup.prettify()
# Find the first instance of an HTML tag
soup.find(tag, attrs={"class": "__"})
# Find all instances of an HTML tag
soup.findAll(tag, attrs={"class": "__"})
```

Get requests using urllib

```
from urllib.request import urlopen, Request
url = "https://www.wikipedia.org/"
request = Request(url)
response = urlopen(request)
html = response.read()
response.close()
```

Higher Order Functions

```
# min/max
max(iterable[, default=obj, key=func])
min(iterable[, default=obj, key=func])
a = min([12, "apple", 223, "A", "B"], key=lambda c: len(str(c)))
# Output: "A" (minimum value in the list based on len(str) of the list object)
students = [{"name": "Santosh", "age": 25}, {"name": "Watson", "age": 35}, {"name": "Karlson", "age": 21}, {"name": "Kenzo", "age": 15}]
youngest = min(students, key=lambda x: x["age"]) # Output: {"name": "Kenzo", "age": 15}
oldest = max(students, key=lambda x: x["age"]) # Output: {"name": "Watson", "age": 35}
#sorted
sorted(iterable, key=func, reverse=False)
L = ["apple", "banana", "dog", "aeroplane"]
```

Higher Order Functions (cont)

```
> print(sorted(L, key=len)) # Sort based on length of string
#map
map(function, iterable, ...)
applies a function to the iterable and returns a mapped object
Use print(list(map)) to print the value!
# A function to return the square of n
def addition(n):
    return n**2
# Some iterable
list = [1,2,3,4]
#map the function with the iterable and apply list to the map object
print(list(map(addition, list))) # Output: [1,4,9,16]
# Filter
map(function, iterable, ...)
The filter runs through each element of iterable and applies function to it.
It filters out list elements for which function doesn't give a True value
seq = [0, 1, 2, 3, 5, 8, 13]
# result filters out non odd numbers
result = filter(lambda x: x % 2 != 0, seq)
print(list(result)) # Output: [1,3,5,13]
```

