Python Cheat Sheet

WARNING - No descriptions or explainations. If you want to learn and need guidance, **CHECK OUR FULL PYTHON TUTORIAL HERE**.

Purpose: This is a cheat sheet. It is NOT meant to teach you Python. It is meant as a reference to help you remember syntax. It helps if you find yourself asking things like, "How do you check the length of an array?" or "What is that regex match character?". We don't cover everything, just the basics to get things done fast (no classes, eggs, threads, etc.).

```
p = [1, q, 4]
p[2] = x
a.append("new")
x = a.pop()
len(a)
```

```
d = {} # Empty dict
d = {'b’ : 1, 'a' : 2 } # dict
d['abc'] = 14.50
len(d)
del d[key_name] # remove key and value
d.remove(value) # first item with value
for key in map1:
```

```
# identity
in, not in, is,
is not, and, or, not

# value
==
```

```
if x < 0:
elif x == 0:
else:

while b < 10:
while True:
for w in words:
for i in range(len(a)): # array with indices
for i in range(5, 10): # 5-9
for k,d in enumerate(a):
```

```
import os
import shutil
shutil.copy(src, dst)
os.rename(src, dst)
os.unlink("/tmp/foo.txt")
os.listdir("/dir")
os.mkdir(path[, mode])
shutil.copy(src, dst) # copy content and
permission
os.rmdir(path) # remove dir, only empty dirs
os.removedirs(path) # remove empty dir,
recursive
```

```
f = open('workfile', 'w') # r or w
f.write('This is a test\n')
f.read() # entire file
f.readline() # line at a time
f.readlines() # entire file, as list
f.close()
```

https://low-orbit.net/python-cheat-sheet

```
def function1(one, two='Yes', *name2, **name3):
return value
```

```
import subprocess
subprocess.check_output( "uname -a", stderr=subprocess.STDOUT, shell=True )
```

```
p = re.compile(r”match this”) # r for raw
string
m = p.search('string goes here')
if m:
print 'Match found: ', m.group()
else:
print 'No match'

p.findall() # array of tuples, includes
groups

p = re.compile('(a(b)c)d') # nested groups
m.groups()

p = re.compile(r'\W+')
p.split('test test') # split, return list

p = re.compile( '(blue|white|red)')
x = p.sub( 'colour', 'blue socks and red
shoes')
```

```
\d decimal digit [0-9]
\s whitespace [ \t\n\r\f\v]
\w alphanumeric [a-zA-Z0-9_]
^ start
$ end

. any character except newline
* 0 or more
+ 1 or more
? 0 or 1
{0,1}

p = re.compile('ab*', re.I | re.M)
S Make . match any char, including newlines
I Do case-insensitive matches
M Multi-line matching, affecting ^ and $
```