Common Python Libraries: Modules, Functions & Examples

This document contains a comprehensive list of commonly used Python libraries with their functions and examples.

Library: math

Function: ceil(x)

```
import math

print(math.ceil(4.3))  # Round up to 5
```

Function: sqrt(x)

```
import math

print(math.sqrt(16))  # Square root of 16
```

Function: factorial(x)

```
import math

print(math.factorial(5))  # 5! = 120
```

Library: datetime

Function: today()

```
from datetime import datetime

print(datetime.today())
```

Function: now()

```
from datetime import datetime

print(datetime.now())
```

Function: date(year, month, day)

```
from datetime import date

print(date(2025, 3, 20))
```

Library: random

Function: randint(a, b)

```
import random

print(random.randint(1, 10))  # Random integer between 1 and 10
```

Function: choice(seq)

```
import random

print(random.choice(['apple', 'banana', 'cherry']))
```

Function: shuffle(x)

```
import random

lst = [1, 2, 3, 4]

random.shuffle(lst)

print(lst)
```

Library: os

Function: getcwd()

```
import os

print(os.getcwd())  # Current working directory
```

Function: listdir(path)

```
import os

print(os.listdir('.'))  # List files in current directory
```

Function: remove(path)

```
import os
```

```
# os.remove('file.txt')  # Remove file (uncomment to use)
```

Library: sys

Function: version

```
import sys
```

```
print(sys.version)  # Python version
```

Function: platform

```
import sys
```

```
print(sys.platform)  # OS platform
```

Function: argv

```
import sys
```

```
print(sys.argv)  # Command line arguments
```

Library: json

Function: dumps(obj)

```
import json
```

```
print(json.dumps({'name': 'Alice', 'age': 25}))
```

Function: loads(s)

```
import json
```

```
print(json.loads('{"name": "Alice", "age": 25}'))
```

Function: dump(obj, file)

```
import json
```

```
# json.dump({'name': 'Bob'}, open('data.json', 'w'))
```

Library: re

Function: match(pattern, string)

```
import re

print(re.match(r'\d+', '123abc'))
```

Function: search(pattern, string)

```
import re

print(re.search(r'abc', '123abc456'))
```

Function: findall(pattern, string)

```
import re

print(re.findall(r'\d+', '12, 34, 56'))
```

Library: time

Function: time()

```
import time

print(time.time())  # Current time in seconds since epoch
```

Function: sleep(seconds)

```
import time

time.sleep(2)  # Sleep for 2 seconds
```

Function: ctime(seconds)

```
import time

print(time.ctime())  # Current local time
```

Library: statistics

Function: mean(data)

```
import statistics

print(statistics.mean([1, 2, 3, 4]))
```

Function: median(data)

```
import statistics

print(statistics.median([1, 3, 5, 7]))
```

Function: stdev(data)

```
import statistics

print(statistics.stdev([1, 2, 3, 4, 5]))
```

Library: functools

Function: lru_cache

```
from functools import lru_cache

@lru_cache

def fib(n): return n if n <= 1 else fib(n-1) + fib(n-2)

print(fib(10))
```

Function: reduce(func, seq)

```
from functools import reduce

print(reduce(lambda x, y: x + y, [1, 2, 3, 4]))
```

Function: partial(func, *args)

```
from functools import partial

def power(base, exp): return base ** exp

square = partial(power, exp=2)

print(square(5))
```

Library: itertools

Function: count(start, step)

```
from itertools import count
```

```
for i in count(10, 2):

    print(i)

    if i > 20: break
```

Function: cycle(iterable)

```
from itertools import cycle

for i in cycle('AB'):

    print(i)

    break
```

Function: permutations(iterable)

```
from itertools import permutations

print(list(permutations('AB', 2)))
```