



# Table of contents

<b>1</b>		<b>3</b>
1.1	. . . . .	3
1.2	. . . . .	3
1.3	. . . . .	4
1.4	. . . . .	4
1.5	. . . . .	5
1.6	. . . . .	6
1.7	. . . . .	6
1.8	. . . . .	7
1.9	. . . . .	7
1.10	. . . . .	8
1.11	: . . . . .	8
1.12	: . . . . .	9
1.13	: . . . . .	10
1.14	. . . . .	11
1.15	. . . . .	11
1.16	. . . . .	11
1.16.1	. . . . .	11
1.17	. . . . .	11

# 1

## 1.1

- Python
- 
- .py
- 
- 

```
# math_utils.py
def add(a, b):
    return a + b

def multiply(a, b):
    return a * b

PI = 3.14159
```

## 1.2

:

- :
- :
- :
- :
- :

:

```
my_project/
  main.py
  utils/
    math_utils.py
```

```
    string_utils.py
    file_utils.py
models/
    user.py
    product.py
```

## 1.3

:

```
import math

result = math.sqrt(16) # 4.0
print(math.pi)         # 3.141592653589793
```

:

```
from math import sqrt, pi

result = sqrt(16) # math.sqrt
print(pi)         # math.pi
```

:

```
import math as m
import numpy as np #

result = m.sqrt(16)
array = np.array([1, 2, 3])
```

## 1.4

:

```
from math import *

result = sqrt(16) #
#
```

:

```
from math import sqrt as square_root
from math import pi as PI_VALUE

result = square_root(16)
print(PI_VALUE)
```

## 1.5

**calculator.py:**

```
"""      """

def add(a, b):
    """      """
    return a + b

def subtract(a, b):
    """      """
    return a - b

def divide(a, b):
    """      """
    if b == 0:
        raise ValueError("      ")
    return a / b

VERSION = "1.0.0"
```

**main.py:**

```
import calculator

result = calculator.add(10, 5)
print(f" : {result}")
print(f" : {calculator.VERSION}")
```

## 1.6

:

```
my_package/  
  __init__.py      #  
  math_ops.py  
  string_ops.py  
  subpackage/  
    __init__.py  
    advanced.py
```

**init.py:**

```
"""My Package - """  
  
from .math_ops import add, multiply  
from .string_ops import capitalize_words  
  
__version__ = "1.0.0"  
__all__ = ["add", "multiply", "capitalize_words"]
```

:

```
from my_package import add, capitalize_words  
  
result = add(5, 3)  
text = capitalize_words("hello world")
```

## 1.7

:

```
import os          #  
import sys         #  
import datetime    #  
import random      #  
import json        # JSON  
import re          #
```

```
import pathlib      # modern way

#
current_time = datetime.datetime.now()
random_number = random.randint(1, 100)
file_path = pathlib.Path("data.txt")
```

## 1.8

:

```
import requests     # HTTP
import pandas as pd  #
import numpy as np   #
import matplotlib.pyplot as plt #

#
# uv add requests pandas numpy matplotlib
```

:

```
# Web API
response = requests.get("https://api.example.com/data")
data = response.json()

# pandas
df = pd.DataFrame(data)
print(df.head())
```

## 1.9

```
import sys

# Python
print("      :")
for path in sys.path:
    print(f" {path}")
```

```
#
sys.path.append("/custom/module/path")

#
import os
print(f"      : {os.getcwd()}")
```

## 1.10

## 1.11 :

file\_utils.py:

```
"""      """
import json
import csv
from pathlib import Path

def read_json(filename):
    """JSON      """
    with open(filename, 'r', encoding='utf-8') as f:
        return json.load(f)

def write_json(data, filename):
    """JSON      """
    with open(filename, 'w', encoding='utf-8') as f:
        json.dump(data, f, ensure_ascii=False, indent=2)

def read_csv(filename):
    """CSV      """
    with open(filename, 'r', encoding='utf-8') as f:
        return list(csv.DictReader(f))

def get_file_size(filename):
    """      """
    return Path(filename).stat().st_size
```



## 1.12 :

math\_advanced.py:

```
""" """
import math

def factorial(n):
    """ """
    if n < 0:
        raise ValueError(" ")
    if n <= 1:
        return 1
    return n * factorial(n - 1)

def fibonacci(n):
    """ n """
    if n <= 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fibonacci(n-1) + fibonacci(n-2)

def is_prime(n):
    """ """
    if n < 2:
        return False
    for i in range(2, int(math.sqrt(n)) + 1):
        if n % i == 0:
            return False
    return True

def prime_factors(n):
    """ """
    factors = []
    d = 2
    while d * d <= n:
        while n % d == 0:
            factors.append(d)
            n //= d
        d += 1
```

```
if n > 1:
    factors.append(n)
return factors
```

## 1.13 :

```
# main.py
from file_utils import read_json, write_json
from math_advanced import factorial, is_prime, prime_factors

def main():
    #
    data = {
        "numbers": [5, 7, 12, 17, 20],
        "operation": "analysis"
    }

    #
    write_json(data, "input.json")

    #
    loaded_data = read_json("input.json")

    #
    results = {}
    for num in loaded_data["numbers"]:
        results[num] = {
            "factorial": factorial(num) if num <= 10 else "Too large",
            "is_prime": is_prime(num),
            "prime_factors": prime_factors(num) if num > 1 else []
        }

    #
    write_json(results, "results.json")

    print("    results.json    ")

if __name__ == "__main__":
    main()
```

## 1.14

1. -
2. - docstring
3. `name == "main"` -
4. -
5. -

## 1.15

- ( )
- ( )
- (Python )
- (getter/setter )

## 1.16

### 1.16.1

---

: - [Python.org](#) - - [PyPI](#) - Python Package Index - [Real Python](#) -

---

## 1.17

:

:

|

: