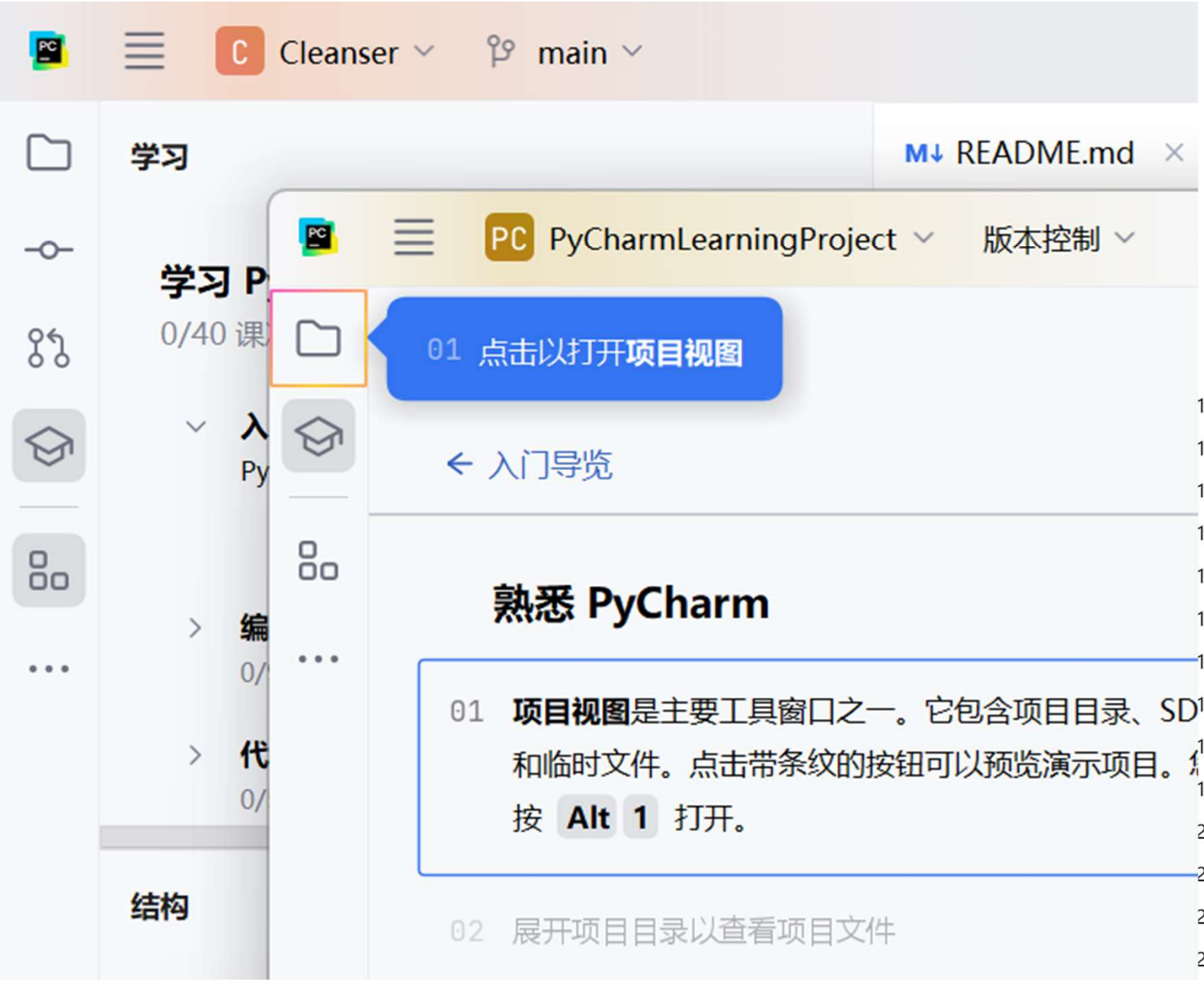

1. **基础语法** - 变量不用声明类型，直接赋值就行。
2. **运算符** - 常见的数学运算和逻辑判断方式。
3. **条件语句** - 根据条件选择执行不同的代码块。
4. **循环** - 让代码重复执行，直到条件满足。
5. **函数** - 定义可重复使用的代码块，传参和返回结果。
6. **数据结构** - 列表、元组、集合和字典，用来存储和处理数据。
7. **列表推导式** - 用简短的一行代码生成新列表。
8. **异常处理** - 处理程序中的错误，让它不崩溃。
9. **类和对象** - 面向对象编程，定义对象及其行为。
10. **文件操作** - 读写文件的简单方法。
11. **模块和包** - 导入外部库和模块来扩展功能。
12. **内置函数** - Python 自带的常用函数，让开发更简单。
13. **面向对象进阶** - 类方法、静态方法，给类添加高级功能。
14. **装饰器** - 改变函数行为的高级技巧。
15. **生成器** - 按需生成数据，节省内存。
16. **多线程和多进程** - 同时执行多个任务，提升效率。
17. **常用库** - 常见的第三方库，用于科学计算、数据分析等。
18. **虚拟环境** - 创建独立的开发环境，避免库冲突。
19. **正则表达式** - 字符串匹配的强大工具。
20. **命令行参数** - 获取用户在命令行输入的数据。
21. **序列化与反序列化** - 保存和读取数据的格式化工具。
22. **时间与日期** - 获取当前时间或操作日期。
23. **类型提示** - 给代码加上类型注释，让它更清晰。

1. 基础语法

- **变量赋值**

  `x = 10`

  Python 是动态类型，无需声明类型。

- **数据类型**

  - 整数（`int`）：`x = 10`
  - 浮点数（`float`）：`x = 10.5`
  - 字符串（`str`）：`x = "Hello"`
  - 布尔型（`bool`）：`x = True`
  - 空值（`None`）：`x = None`

2. 运算符

- **算术运算符**: `+`, `-`, `*`, `/`, `//` (整除), `%` (取余), `**` (幂)
- **比较运算符**: `==`, `!=`, `>`, `<`, `>=`, `<=`
- **逻辑运算符**: `and`, `or`, `not`

3. 条件语句

python Copy code

```python
if condition:
    # code block
elif condition:
    # code block
else:
    # code block
```

4. 循环

- **while 循环**:

```python
while condition:
    # code block
```

- **for 循环**:

```python
for i in range(5):
    # 0 to 4
```

5. 函数

python Copy code

```python
def function_name(parameters):
    # code block
    return value
```

- **Lambda 表达式**: `lambda x, y: x + y`

6. 数据结构

- **列表 (List)**:
  可变、可重复

  python Copy code
  ```python
  lst = [1, 2, 3]
  lst.append(4)
  ```

- **元组 (Tuple)**:
  不可变

  python Copy code
  ```python
  tpl = (1, 2, 3)
  ```

- **集合 (Set)**:
  无序、唯一

  python Copy code
  ```python
  s = {1, 2, 3}
  ```

- **字典 (Dictionary)**:
  键值对存储

  python Copy code
  ```python
  d = {"key": "value"}
  d["new_key"] = "new_value"
  ```

7. 列表推导式

python Copy code

```python
lst = [x**2 for x in range(5)]
```

8. 异常处理

python Copy code

```python
try:
    # code block
except Exception as e:
    # exception handling
finally:
    # always executed
```

9. 类和对象

python Copy code

```python
class MyClass:
    def __init__(self, value):
        self.value = value

    def method(self):
        return self.value
```

- **继承**:

python Copy code

```python
class SubClass(MyClass):
    pass
```

10. 文件操作

python Copy code

```python
with open("file.txt", "r") as file:
    content = file.read()
```

11. 模块和包

- **导入模块**:

```python
import math
from math import sqrt
```

12. 内置函数

- 常用函数: `len()`, `max()`, `min()`, `sum()`, `abs()`, `sorted()`, `zip()`

17. 常用库

- **NumPy**: 数组与矩阵操作
- **Pandas**: 数据处理与分析
- **Matplotlib**: 数据可视化
- **TensorFlow/PyTorch**: 机器学习框架

13. 面向对象进阶

- **类方法和静态方法**:

python Copy code

```python
class MyClass:
    @classmethod
    def class_method(cls):
        pass

    @staticmethod
    def static_method():
        pass
```

14. 装饰器

python Copy code

```python
def decorator(func):
    def wrapper(*args, **kwargs):
        # code before
        result = func(*args, **kwargs)
        # code after
        return result
    return wrapper
```

15. 生成器

python Copy code

```python
def my_generator():
    yield 1
    yield 2
```

16. 多线程和多进程

- 多线程:

```python
import threading
t = threading.Thread(target=function)
t.start()
```

- 多进程:

```python
import multiprocessing
p = multiprocessing.Process(target=function)
p.start()
```

18. 虚拟环境

bash Copy code

```bash
python -m venv env
source env/bin/activate
```

19. 正则表达式

python Copy code

```python
import re
pattern = r'\d+'
match = re.match(pattern, '123')
```

20. 命令行参数

python Copy code

```python
import sys
print(sys.argv)
```

21. 序列化与反序列化

- **JSON**:

python Copy code

```python
import json
json_str = json.dumps(data)
data = json.loads(json_str)
```

- **Pickle**:

python Copy code

```python
import pickle
with open("file.pkl", "wb") as file:
    pickle.dump(data, file)
```

22. 时间与日期

python Copy code

```python
import datetime
now = datetime.datetime.now()
```

23. 类型提示

python Copy code

```python
def add(a: int, b: int) -> int:
    return a + b
```