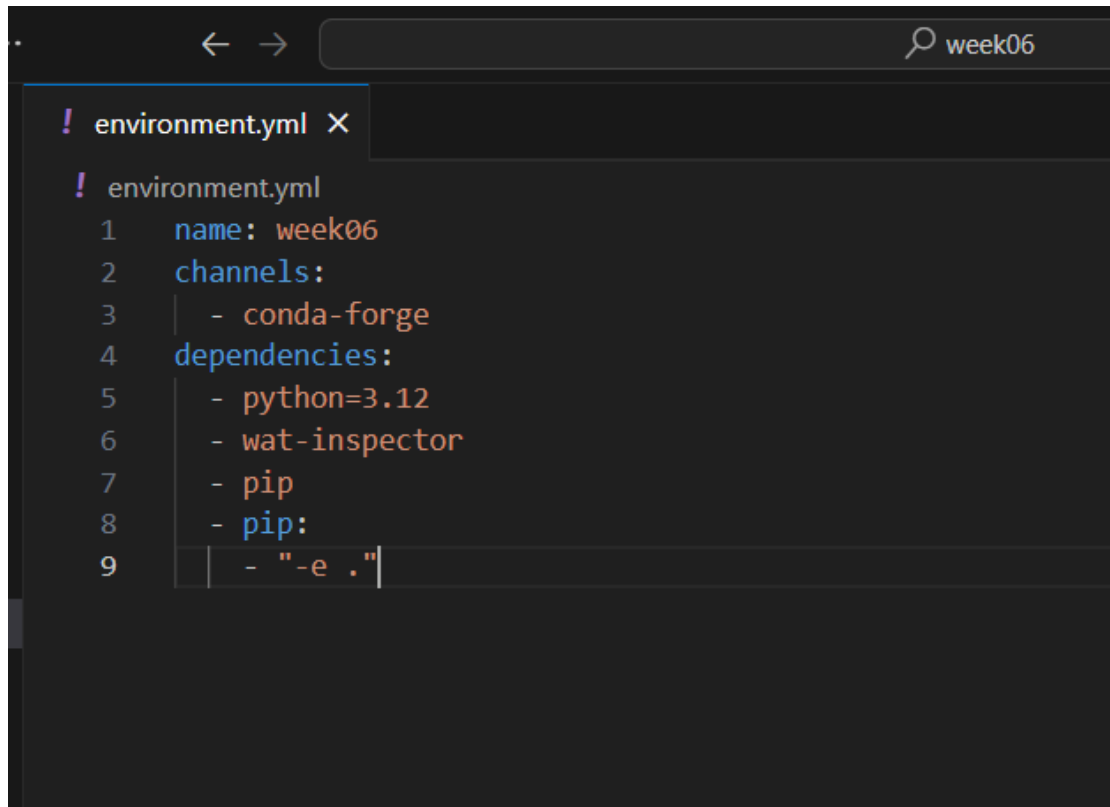


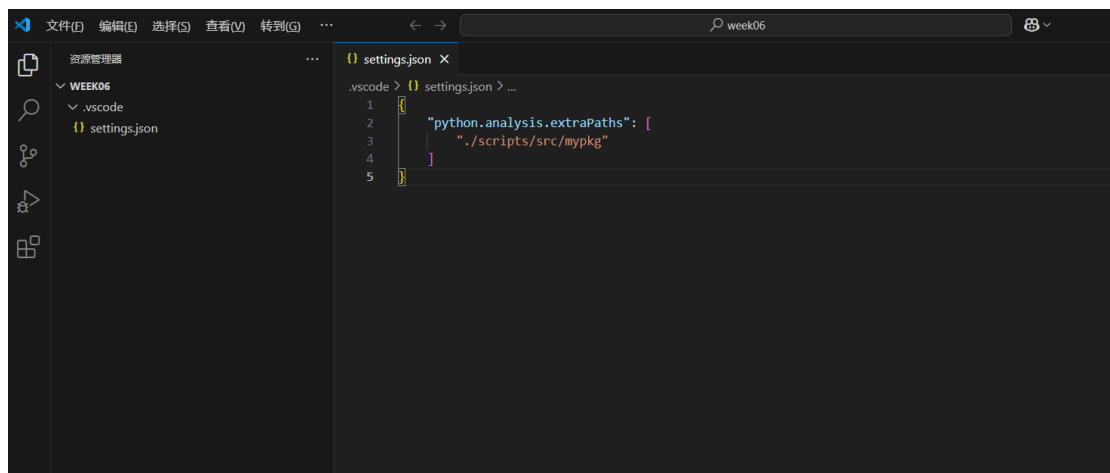
第六周学习笔记--Python 代码组织(初级)

一、安装 Fork 第 06 周打卡仓库至自己的名下，然后将名下的这个仓库 Clone 到本地计算机

二、安装用 VS Code 打开项目目录，新建 environment.yml 文件，指定安装 Python3.12，然后运行 condaenvcreate 命令创建 Conda 环境



```
! environment.yml X
! environment.yml
1  name: week06
2  channels:
3    - conda-forge
4  dependencies:
5    - python=3.12
6    - wat-inspector
7    - pip
8    - pip:
9      - "-e ."
```



```
settings.json X
.vscode > settings.json > ...
1  {}
2  "python.analysis.extraPaths": [
3    "./scripts/src/mypkg"
4  ]
5  }
```

三、创建一个 guessing game.py 文件，复制粘贴以下代码，运用 pdb 调试器理解其运行流程

```
← → week06
! environment.yml  guessing_game.py X
scripts > src > mypkg > guessing_game.py > ...
4  def guessing_game():
21
22     try:
23         guess = int(guess)
24     except ValueError:
25         print("输入无效 🚫, 请输入一个整数。")
26         continue
27
28     if guess < 1 or guess > 100:
29         print("输入无效 🚫, 输入值应该在 1~100 之间。")
30         continue
31
32     if guess == secret_number:
33         print("恭喜你 🎉, 猜对了!")
34         break
35
36     if guess < secret_number:
37         print("猜的数字太小了, 再试试 🔍。")
38         continue
39
40     if guess > secret_number:
41         print("猜的数字太大了, 再试试 🔍。")
42         continue
43
44     raise NotImplementedError
45
46     print("游戏结束, 再见 👋。")
47
48
49 if __name__ == "__main__":
50     guessing_game()
51
```

四、创建一个 fowcontrols.py 文件, 运行以下 Python 流程控制语句

```
← → week06
! environment.yml flow_controls.py ×
flow_controls.py > ...
179     print(result)
180 except ZeroDivisionError:
181     print("错误: 除数不能为零。")
182
183 # try...except...else...finally语句
184 try:
185     num1 = int(input("请输入第一个整数: "))
186     num2 = int(input("请输入第二个整数: "))
187     result = num1 / num2
188 except ValueError:
189     print("输入错误: 请输入有效的整数。")
190 except ZeroDivisionError:
191     print("错误: 除数不能为零。")
192 else:
193     print(f"除法运算结果是: {result}")
194 finally:
195     print("无论是否发生异常, 此代码块都会执行。")
196
197
198 # raise语句
199 def divide_numbers(a, b):
200     if b == 0:
201         raise ZeroDivisionError("除数不能为零。")
202     return a / b
203
204
205 try:
206     result = divide_numbers(15, 0)
207     print(result)
208 except ZeroDivisionError as e:
209     print(f"捕获到异常: {e}")
210
```

五、创建 mylib.py 模块(module)和 myjob.py 脚本(script)

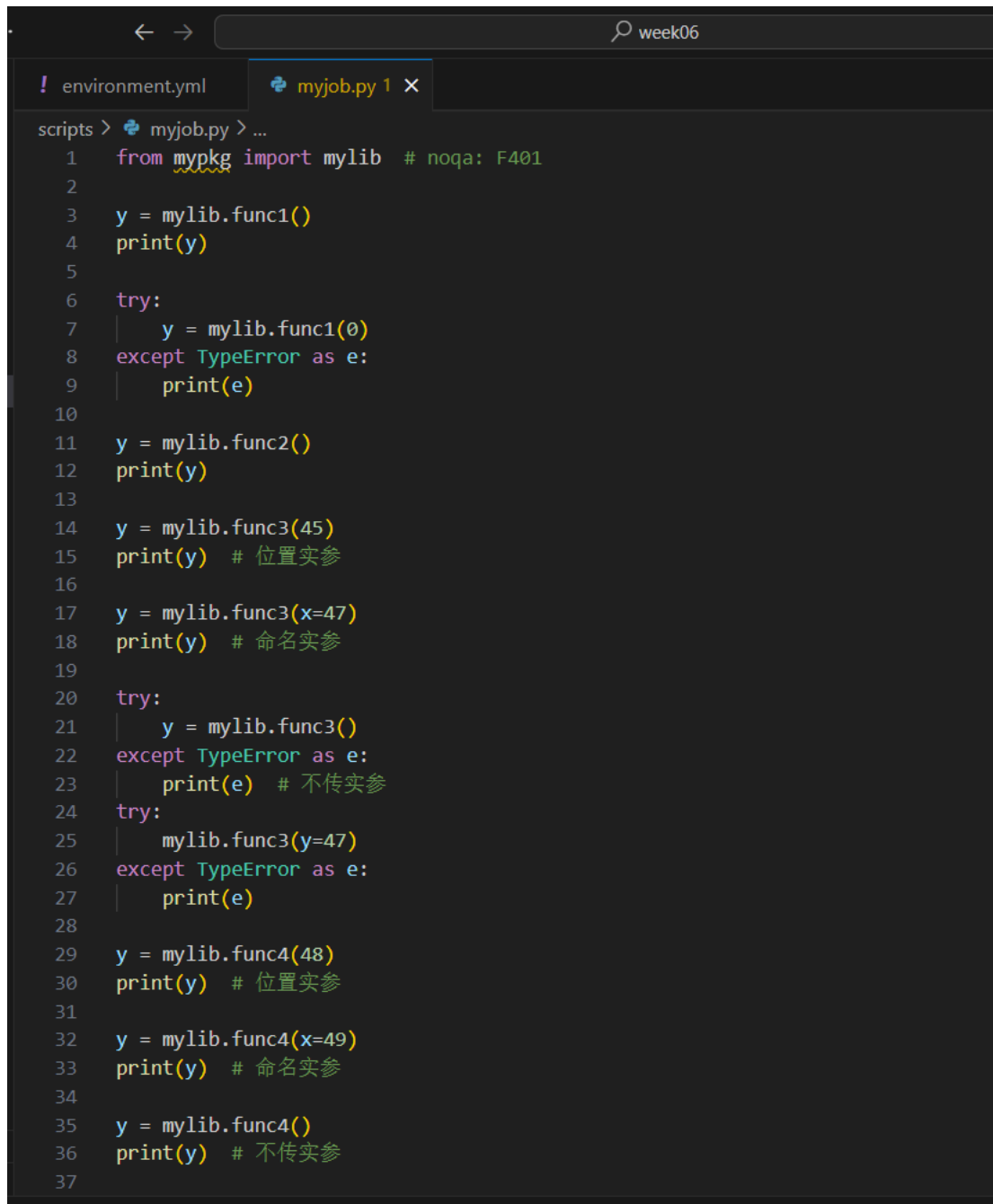
```
! environment.yml mylib.py x
scripts > src > mypkg > mylib.py > ...
1 def func1():
2     x = 60 # 修改 x 的值为 60
3     y = x*0.8 - 82
4     print(y)
5
6
7 def func2():
8     x = 120 # 修改 x 的值为 120
9     y = x*0.8 - 82
10    print(y)
11    return y
12
13
14 def func3(x):
15     y = x*0.8 - 82
16     return y
17
18
19 def func4(x=800): # 修改默认参数值为 800
20     y = x*0.8 - 82
21     return y
22
23
24 # 定义一个函数，包含位置形参和命名形参
25 def calculate_area(length, width=15): # 修改 width 默认值为 15
26     return length * width
27
28
29 # 调用函数，使用位置实参和命名实参
30 area1 = calculate_area(6) # 使用位置实参传递 length 为 6, width 使用默认值
31 area2 = calculate_area(length=8, width=4) # 使用命名实参传递参数
32
33
34 def func6(a, /, b, operation="add"):
35     if operation == "add":
36         return a + b
37     elif operation == "subtract":
38         return a - b
39     else:
40         return None
41
42
43 def func7(a, /, b, *, operation="add"):
44     if operation == "add":
45         return a + b
46     elif operation == "subtract":
47         return a - b
48     else:
49         return None
50
51
52 def func8(*args):
53     total = 0
54     for num in args:
55         total = total + num
56     return total
57
58
59 def func9(**kwargs):
60     for key, value in kwargs.items():
61         print(f"{key}: {value}")
62
63
64 def func10(positional_arg1, positional_arg2, named_arg=15): # 修改默认参数值为 15
65     """
66     该函数接受两个位置形参和一个带有默认值的命名形参。
67     函数会打印出这些参数的值，并返回它们的和。
68     """
```

```
! environment.yml mylib.py x
scripts > src > mypkg > mylib.py > ...
33
34 def func6(a, /, b, operation="add"):
35     if operation == "add":
36         return a + b
37     elif operation == "subtract":
38         return a - b
39     else:
40         return None
41
42
43 def func7(a, /, b, *, operation="add"):
44     if operation == "add":
45         return a + b
46     elif operation == "subtract":
47         return a - b
48     else:
49         return None
50
51
52 def func8(*args):
53     total = 0
54     for num in args:
55         total = total + num
56     return total
57
58
59 def func9(**kwargs):
60     for key, value in kwargs.items():
61         print(f"{key}: {value}")
62
63
64 def func10(positional_arg1, positional_arg2, named_arg=15): # 修改默认参数值为 15
65     """
66     该函数接受两个位置形参和一个带有默认值的命名形参。
67     函数会打印出这些参数的值，并返回它们的和。
68     """
```

```

63
64 def func10(positional_arg1, positional_arg2, named_arg=15): # 修改默认参数值为 15
65     """
66     该函数接受两个位置形参和一个带有默认值的命名形参。
67     函数会打印出这些参数的值，并返回它们的和。
68     """
69     print(f"第一个位置形参的值: {positional_arg1}")
70     print(f"第二个位置形参的值: {positional_arg2}")
71     print(f"命名形参的值: {named_arg}")
72     return positional_arg1 + positional_arg2 + named_arg
73
74
75 tuple_args = (3, 4) # 修改元组参数
76 result = func10(*tuple_args)
77 print(f"三个参数的总和: {result}")
78
79
80 def func12(
81     positional_arg1: str,
82     positional_arg2: int,
83     named_arg: str = "new_default", # 修改默认参数值
84 ) -> None:
85     "多个参数的例子"
86     print(f"第一个位置形参的值: { (parameter) positional_arg2: int")
87     print(f"第二个位置形参的值: {positional_arg2}")
88     print(f"命名形参的值: {named_arg}")
89     return positional_arg1 + str(positional_arg2) + named_arg
90

```



```
scripts > myjob.py > ...
1  from mypkg import mylib # noqa: F401
2
3  y = mylib.func1()
4  print(y)
5
6  try:
7      y = mylib.func1(0)
8  except TypeError as e:
9      print(e)
10
11 y = mylib.func2()
12 print(y)
13
14 y = mylib.func3(45)
15 print(y) # 位置实参
16
17 y = mylib.func3(x=47)
18 print(y) # 命名实参
19
20 try:
21     y = mylib.func3()
22 except TypeError as e:
23     print(e) # 不传实参
24 try:
25     mylib.func3(y=47)
26 except TypeError as e:
27     print(e)
28
29 y = mylib.func4(48)
30 print(y) # 位置实参
31
32 y = mylib.func4(x=49)
33 print(y) # 命名实参
34
35 y = mylib.func4()
36 print(y) # 不传实参
37
```

六、mylib 模块转变为软件包(package)安装进当前的 Conda 环境来使用

6.1 把 myjob.py 脚本移动至 scripts/myjob.py, 再次尝试运行, 会发现 import mylib 失败, 这是由于 mylib 并没有打包成软件包(package)安装

← → week06

! environment.yml myjob.py 1 pyproject.toml ×

⚙️ pyproject.toml

```
1 [project]
2 name = "mypackage"
3 version = "2025.4.14"
4 dependencies = [
5     "openpyxl",
6 ]
7 authors = [
8     {name = "Ting", email = "2141619615@qq.com"},
9 ]
10 description = "测试一下"
11
12 [project.optional-dependencies]
13 dev = [
14     "pytest",
15 ]
16
17 [build-system]
18 requires = ["hatchling"]
19 build-backend = "hatchling.build"
20
21 [tool.hatch.build.targets.wheel]
22 packages = [
23     "src/mypkg",
24 ]
```