

第六周学习报告

第 6 周 Python 代码组织 (初级)

1. Fork [第 06 周打卡](https://gitcode.com/cueb-fintech/week06)

仓库至你的名下，然后将你名下的这个仓库 Clone 到你的本地计算机

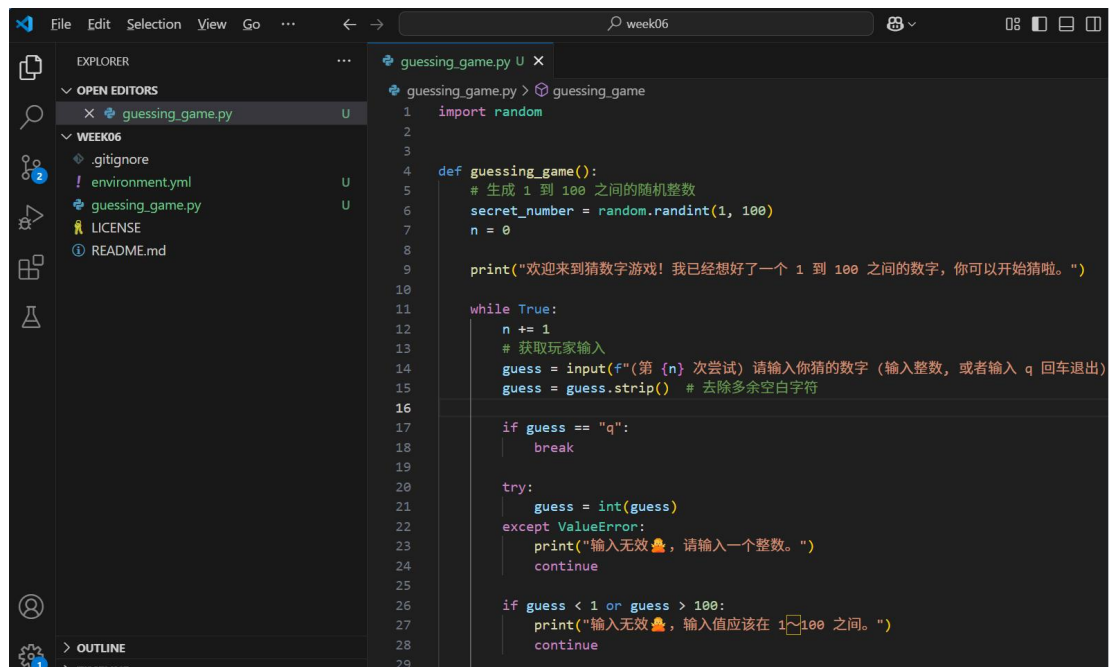
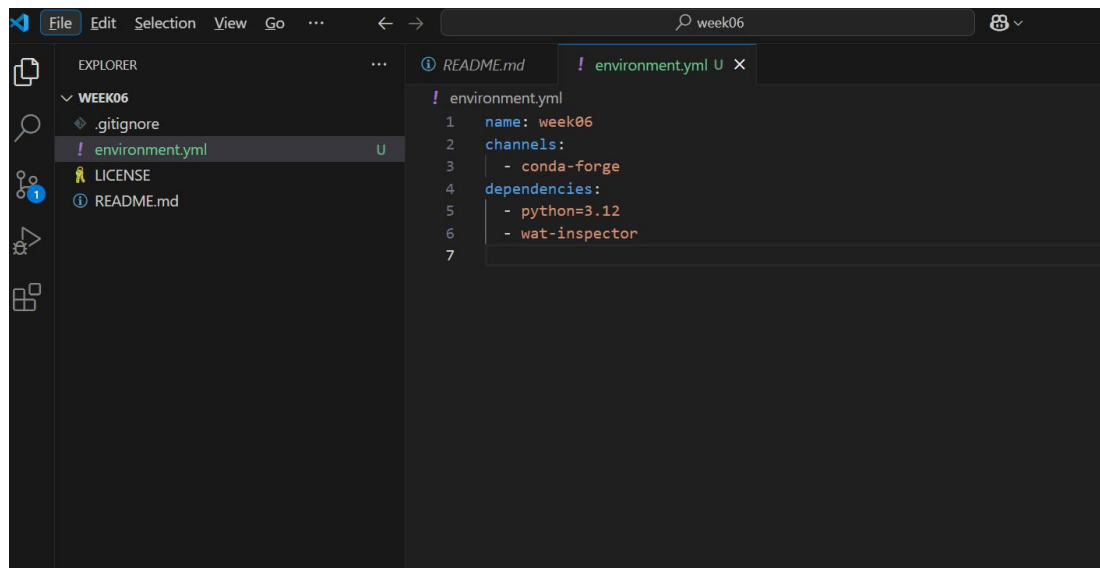
```
MINGW64:/c/Users/86153/rep  X + v
drwxr-xr-x 1 86153 197609 0 3月 9 11:57 week01/
drwxr-xr-x 1 86153 197609 0 3月 9 12:00 week01_3492/
drwxr-xr-x 1 86153 197609 0 3月 16 13:28 week02/
drwxr-xr-x 1 86153 197609 0 3月 20 20:57 week03/
drwxr-xr-x 1 86153 197609 0 3月 28 14:06 week04/
drwxr-xr-x 1 86153 197609 0 4月 6 21:36 week05/

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo
$ git clone git@gitcode.com:gossamer/week06.git
Cloning into 'week06'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5 (delta 0), reused 5 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (5/5), 8.45 KiB | 1.69 MiB/s, done.

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo
$ ll
total 25
drwxr-xr-x 1 86153 197609 0 3月 20 20:34 prj1/
-rw-r--r-- 1 86153 197609 221 3月 15 23:07 tkconch-script.py
drwxr-xr-x 1 86153 197609 0 3月 9 11:57 week01/
drwxr-xr-x 1 86153 197609 0 3月 9 12:00 week01_3492/
drwxr-xr-x 1 86153 197609 0 3月 16 13:28 week02/
drwxr-xr-x 1 86153 197609 0 3月 20 20:57 week03/
drwxr-xr-x 1 86153 197609 0 3月 28 14:06 week04/
drwxr-xr-x 1 86153 197609 0 4月 6 21:36 week05/
drwxr-xr-x 1 86153 197609 0 4月 17 14:10 week06/

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo
```

2. 用 VS Code 打开项目目录，新建一个 `environment.yml` 文件，指定安装 Python 3.12，然后运行 `conda env create` 命令创建 Conda 环境



3. 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：

```
python
```

```
import random
```

```
def guessing_game():  
    # 生成 1 到 100 之间的随机整数  
    secret_number = random.randint(1, 100)  
    n = 0  
  
    print("欢迎来到猜数字游戏！我已经想好了一个 1 到  
100 之间的数字，你可以开始猜啦。")  
  
    while True:  
        n += 1  
        # 获取玩家输入  
        guess = input(f"第 {n} 次尝试) 请输入你猜的数  
字 (输入整数, 或者输入 q 回车退出): ")  
        guess = guess.strip() # 去除多余空白字符  
  
        if guess == "q":  
            break  
  
        try:  
            guess = int(guess)  
        except ValueError:
```

```
print("输入无效 ， 请输入一个整数。")
```

```
continue
```

```
if guess < 1 or guess > 100:
```

```
    print("输入无效 ， 输入值应该在 1~100 之  
间。")
```

```
    continue
```

```
if guess == secret_number:
```

```
    print("恭喜你 ， 猜对了！ ")
```

```
    break
```

```
if guess < secret_number:
```

```
    print("猜的数字太小了， 再试试 。")
```

```
    continue
```

```
if guess > secret_number:
```

```
    print("猜的数字太大了， 再试试 。")
```

```
    continue
```

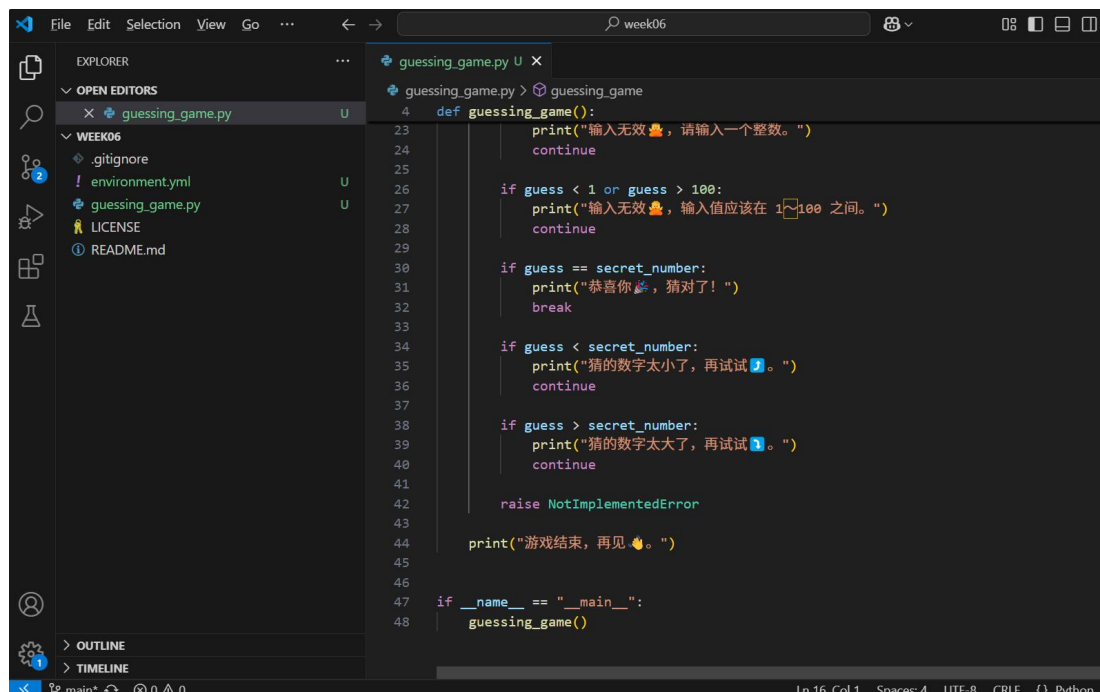
```
raise NotImplementedError
```

```
print("游戏结束，再见 。")
```

```
if __name__ == "__main__":
```

```
    guessing_game()
```

```
...
```



```
File Edit Selection View Go ... week06
EXPLORER
OPEN EDITORS
X guessing_game.py U
WEEK06
  .gitignore
  ! environment.yml U
  guessing_game.py U
  LICENSE
  README.md
OUTLINE
TIMELINE
Ln 16, Col 1, Spaces: 4, UTF-8, CRLF, Python
```

```
4 def guessing_game():
23     print("输入无效 🚫, 请输入一个整数。")
24     continue
25
26     if guess < 1 or guess > 100:
27         print("输入无效 🚫, 输入值应该在 1~100 之间。")
28         continue
29
30     if guess == secret_number:
31         print("恭喜你 🎉, 猜对了! ")
32         break
33
34     if guess < secret_number:
35         print("猜的数字太小了, 再试试 🤔。")
36         continue
37
38     if guess > secret_number:
39         print("猜的数字太大了, 再试试 🤔。")
40         continue
41
42     raise NotImplementedError
43
44     print("游戏结束，再见 🙋。")
45
46
47 if __name__ == "__main__":
48     guessing_game()
```

```
MINGW64: c:/Users/86153/rep  ×  +  ∨

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo
$ cd week06

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week06 (main)
$ conda active week06
usage: conda-script.py [-h] [-v] [--no-plugins] [-V] COMMAND ...
conda-script.py: error: argument COMMAND: invalid choice: 'active' (choose from
e', 'config', 'create', 'deactivate', 'env', 'export', 'info', 'init', 'install
'content-trust', 'convert', 'debug', 'develop', 'doctor', 'index', 'inspect',
eleton', 'token', 'server', 'pack', 'repo', 'remove', 'uninstall', 'rename', 'r

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week06 (main)
$ ls -l
total 29
-rw-r--r-- 1 86153 197609 93 4月 17 14:19 environment.yml
-rw-r--r-- 1 86153 197609 1380 4月 17 14:31 guessing_game.py
-rw-r--r-- 1 86153 197609 18805 4月 17 14:10 LICENSE
-rw-r--r-- 1 86153 197609 2239 4月 17 14:10 README.md

(base) 86153@DESKTOP-CS5HNR1 MINGW64 ~/repo/week06 (main)
$ python guessing_game.py
欢迎来到猜数字游戏！我已经想好了一个 1 到 100 之间的数字，你可以开始猜啦。
(第 1 次尝试) 请输入你猜的数字 (输入整数，或者输入 q 回车退出): 24
猜的数字太小了，再试试。
(第 2 次尝试) 请输入你猜的数字 (输入整数，或者输入 q 回车退出): 99
猜的数字太大了，再试试。
(第 3 次尝试) 请输入你猜的数字 (输入整数，或者输入 q 回车退出): 55
猜的数字太小了，再试试。
(第 4 次尝试) 请输入你猜的数字 (输入整数，或者输入 q 回车退出): |
```

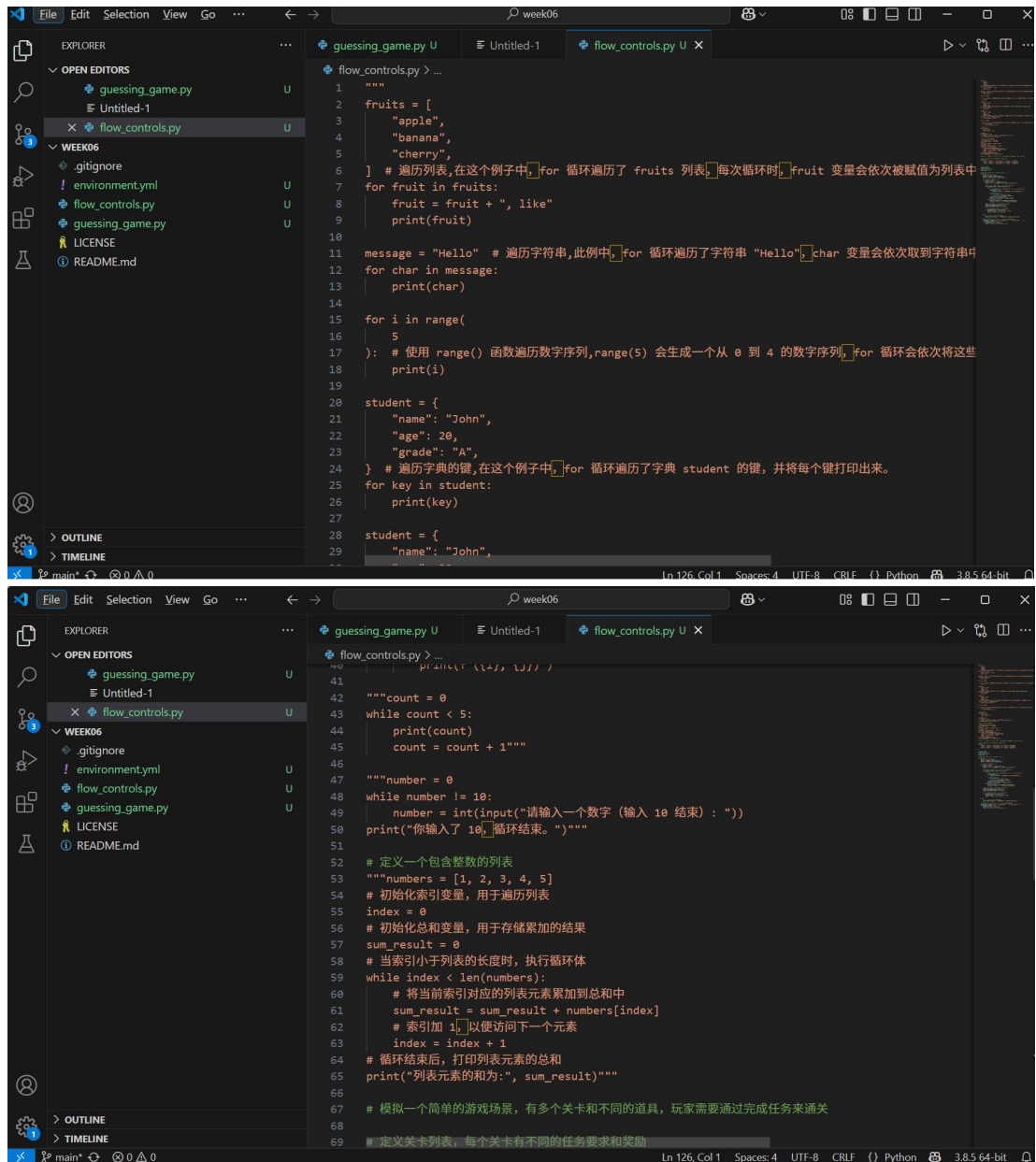
4. 创建一个 `flow_controls.py` 文件，让豆包 (或 DeepSeek 等任何大模型) 生成例子，尝试运行，体会理解以下 Python 流程控制语句：

- `for` 迭代循环 (iteration loop)
- `while` 条件循环 (conditional loop)
- `break` 打断跳出循环
- `continue` 跳至下一轮循环
- `for...else` 循环未被打断的处理
- `if` 条件分支
- `if...elif[...elif]` 多重条件分支

- `if...else` 未满足条件的处理

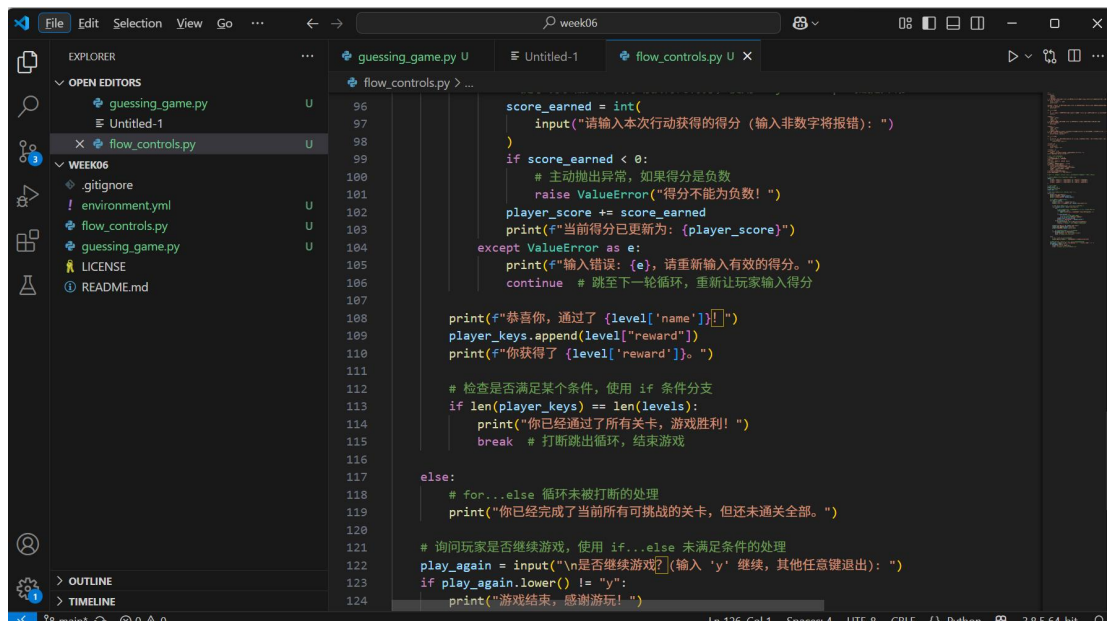
- `try...except[...except...else...finally]` 捕捉异常的处理

- `raise` 主动抛出异常



```
1 """
2 fruits = [
3     "apple",
4     "banana",
5     "cherry",
6 ] # 遍历列表,在这个例子中,for 循环遍历了 fruits 列表,每次循环时,fruit 变量会依次被赋值为列表中
7 for fruit in fruits:
8     fruit = fruit + ", like"
9     print(fruit)
10
11 message = "Hello" # 遍历字符串,此例中,for 循环遍历了字符串 "Hello",char 变量会依次取到字符串中
12 for char in message:
13     print(char)
14
15 for i in range(
16     5
17 ): # 使用 range() 函数遍历数字序列,range(5) 会生成一个从 0 到 4 的数字序列,for 循环会依次将这些
18     print(i)
19
20 student = {
21     "name": "John",
22     "age": 20,
23     "grade": "A",
24 } # 遍历字典的键,在这个例子中,for 循环遍历了字典 student 的键,并将每个键打印出来。
25 for key in student:
26     print(key)
27
28 student = {
29     "name": "John",
30 }
```

```
40 print(f"你输入了 {count}, 循环结束。")
41
42 count = 0
43 while count < 5:
44     print(count)
45     count = count + 1
46
47 number = 0
48 while number != 10:
49     number = int(input("请输入一个数字 (输入 10 结束) : "))
50 print("你输入了 10, 循环结束。")
51
52 # 定义一个包含整数的列表
53 numbers = [1, 2, 3, 4, 5]
54 # 初始化索引变量,用于遍历列表
55 index = 0
56 # 初始化总和变量,用于存储累加的结果
57 sum_result = 0
58 # 当索引小于列表的长度时,执行循环体
59 while index < len(numbers):
60     # 将当前索引对应的列表元素累加到总和
61     sum_result = sum_result + numbers[index]
62     # 索引加 1,以便访问下一个元素
63     index = index + 1
64 # 循环结束后,打印列表元素的总和
65 print("列表元素的和为:", sum_result)
66
67 # 模拟一个简单的游戏场景,有多个关卡和不同的道具,玩家需要通过完成任务来通关
68
69 # 定义关卡列表,每个关卡有不同的任务要求和奖励
```



5. 创建一个 `mylib.py` **模块** (module), 在里面定义以下函数, 再创建一个 `myjob.py` **脚本** (script), 从 `mylib.py` 导入函数并尝试调用:

- 定义函数 `func1`, 没有形参, 没有返回值
- 定义函数 `func2`, 没有形参, 有返回值
- 定义函数 `func3`, 只有一个 **位置形参** (positional parameter), 先尝试传入 **位置实参** (positional argument) 调用, 再尝试传入 **命名实参** (named argument) 调用, 再尝试不传实参 (会报错)
- 定义函数 `func4`, 只有一个 **命名形参** (named parameter), 先传入 **位置实参** 调用, 再传入 **命名实参** 调用, 再尝试不传实参 (取默认值)

- 定义函数 ``func5``，接受多个位置形参和命名形参，尝试以位置/命名各种不同方式传入实参，注意位置参数必须排在命名参数之前

- 定义函数 ``func6``，在形参列表中使用 ``/`` 来限定只接受位置实参的形参

- 定义函数 ``func7``，在形参列表中使用 ``*`` 来限定只接受命名实参的形参

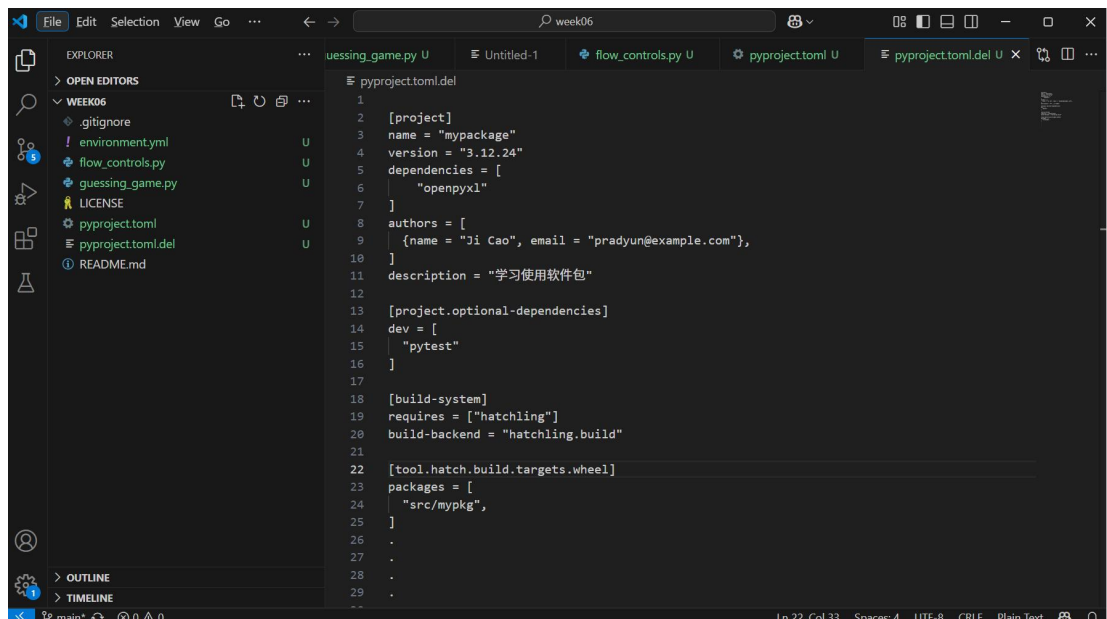
- 定义函数 ``func8``，在位置形参的最后，在形参名称前使用 ``*`` 允许传入任意数量的位置实参 (被打包为元组)

- 定义函数 ``func9``，在命名形参的最后，在形参名称前使用 ``**`` 允许传入任意数量的命名实参 (被打包为字典)

- 定义函数 ``func10``，接受两个位置形参，一个命名形参，尝试在调用时使用 ``*`` 将可迭代对象 (如元组或列表) 自动解包，按位置实参传入

- 定义函数 ``func11``，接受一个命名形参，两个命名形参，尝试在调用时使用 ``**`` 将映射对象 (如字典) 自动解包，按命名实参传入

- 定义函数 ``func12``，给函数添加 ``**内嵌文档**`` (docstring)，给形参和返回值添加 ``**类型注解**`` (type annotation)，提高函数签名的可读性



6. 把 `mylib` 模块转变为 ****软件包** (package)** 安装进当前的 Conda 环境来使用

- 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

- 将 `mylib.py` 模块移动至 `src/mypkg/mylib.py`, 创建 `src/mypkg/__init__.py` 文件, 准备好软件包的源代码

- 创建 `pyproject.toml` 配置文件, 按照 [文档](<https://packaging.python.org/en/latest/guides/writing-pyproject-toml/>) 填写基本的软件包信息

- 在 `pyproject.toml` 配置文件里, 按照 [文档](<https://hatch.pypa.io/latest/config/build/>) 填写软件包的 ******

构建** (build) 配置

- 使用 `pip install -e .` 以本地可编辑模式把当前软件包安装进当前 Conda 环境
- 修改 `environment.yml` 文件, 使得 `conda env create` 自动安装本地可编辑软件包

```
pyproject.toml
1
2 [project]
3 name = "mypackage"
4 version = "3.12.24"
5 dependencies = [
6     "openpyxl"
7 ]
8 authors = [
9     {name = "Ji Cao", email = "pradyun@example.com"},
10 ]
11 description = "学习使用软件包"
12
13 [project.optional-dependencies]
14 dev = [
15     "pytest"
16 ]
17
18 [build-system]
19 requires = ["hatchling"]
20 build-backend = "hatchling.build"
21
22 [tool.hatch.build.targets.wheel]
23 packages = [
24     "src/mypkg",
25 ]
26 .
27 .
28 .
29 .
```

