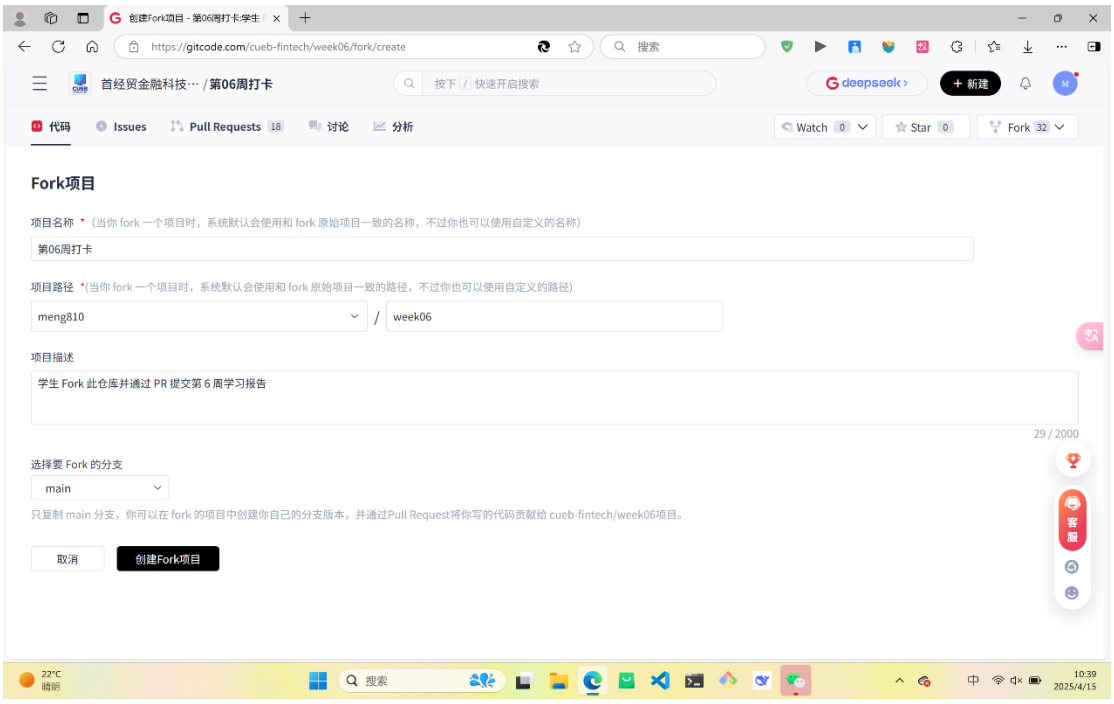
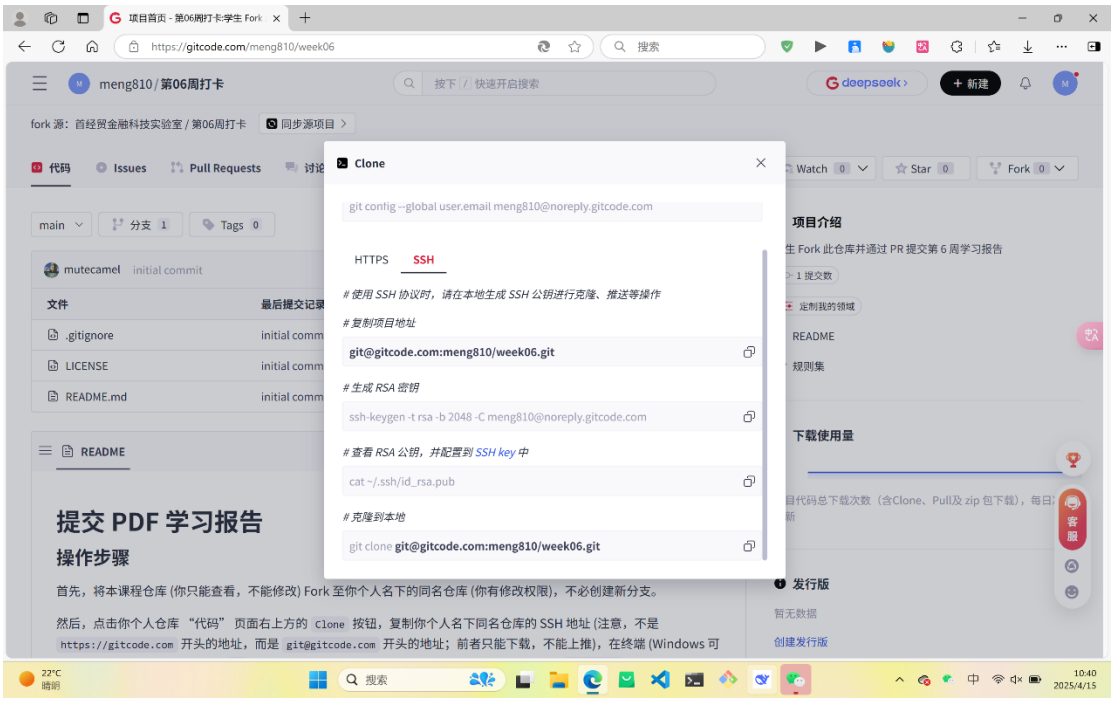
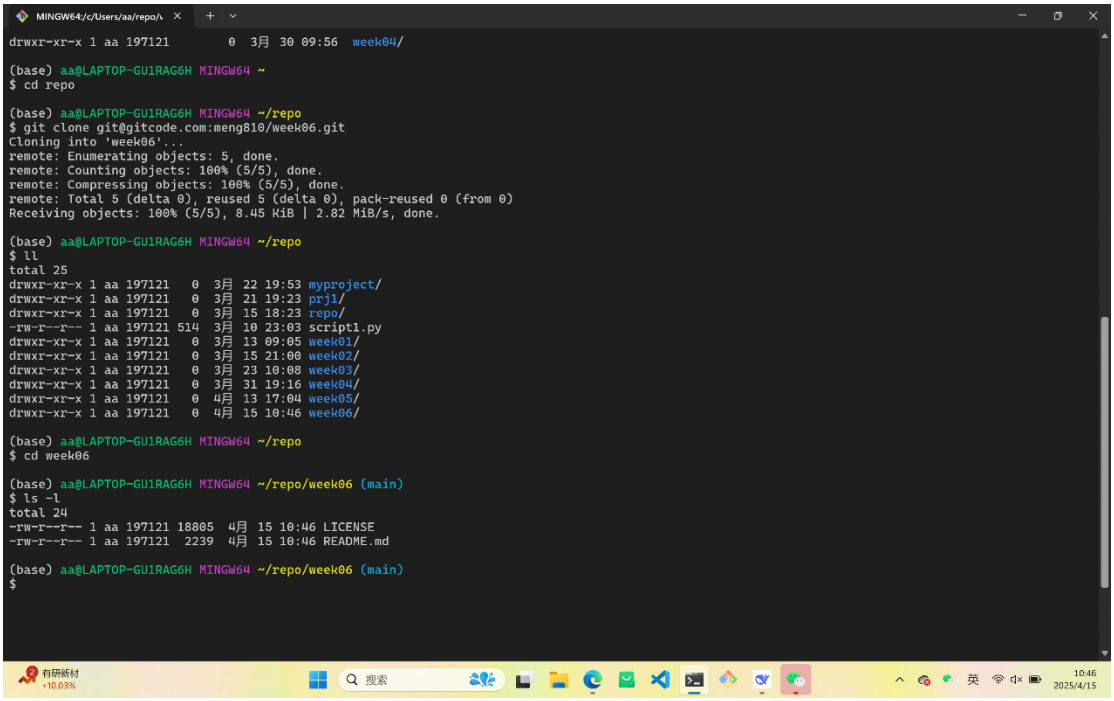
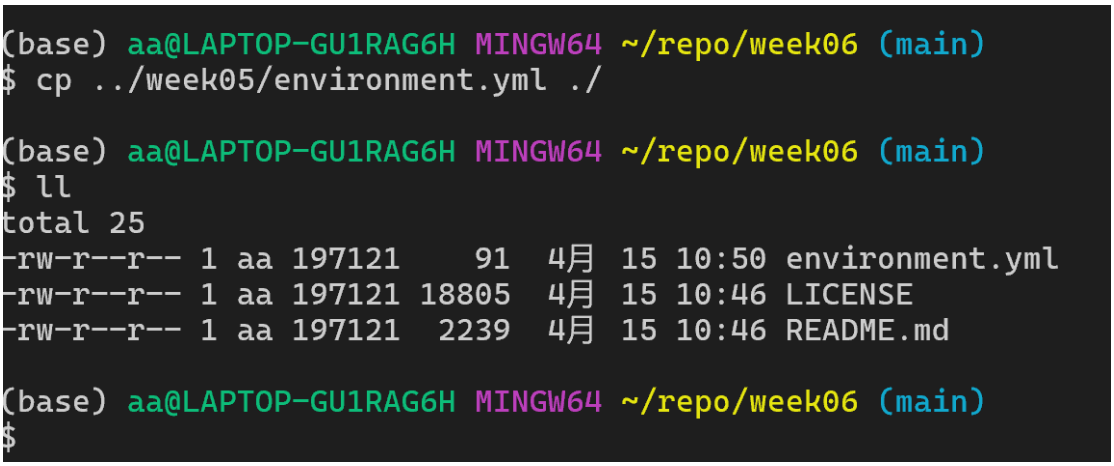
1. Fork 第 06 周打卡仓库至名下，然后将仓库 Clone 到你的本地计算机

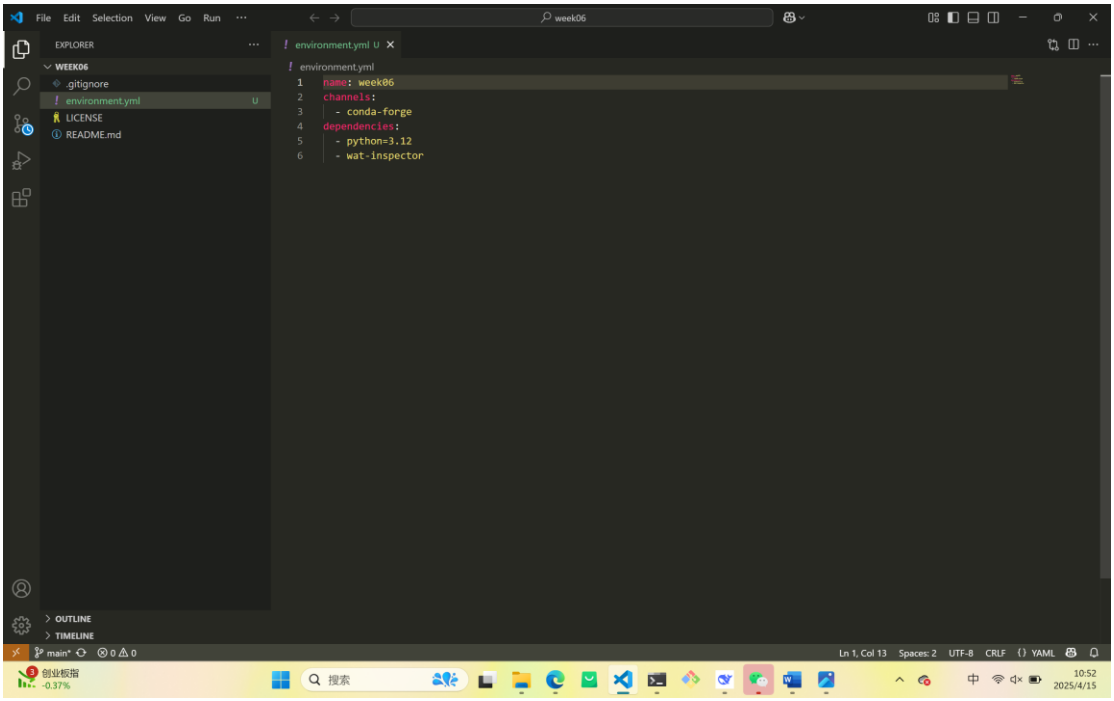


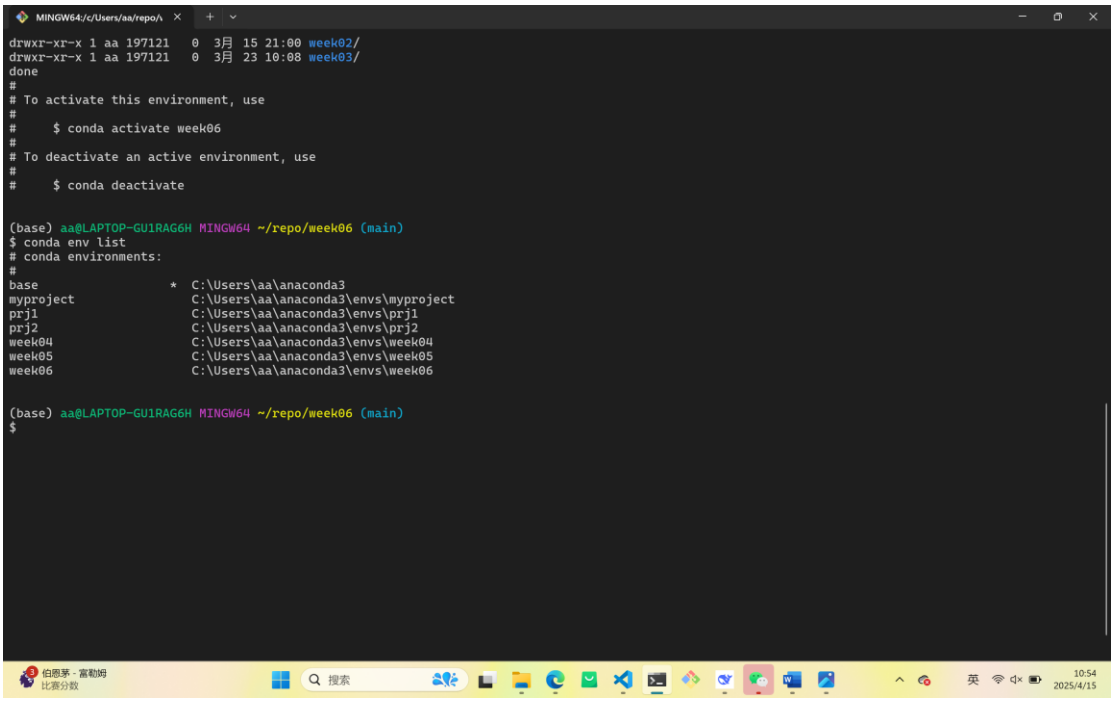


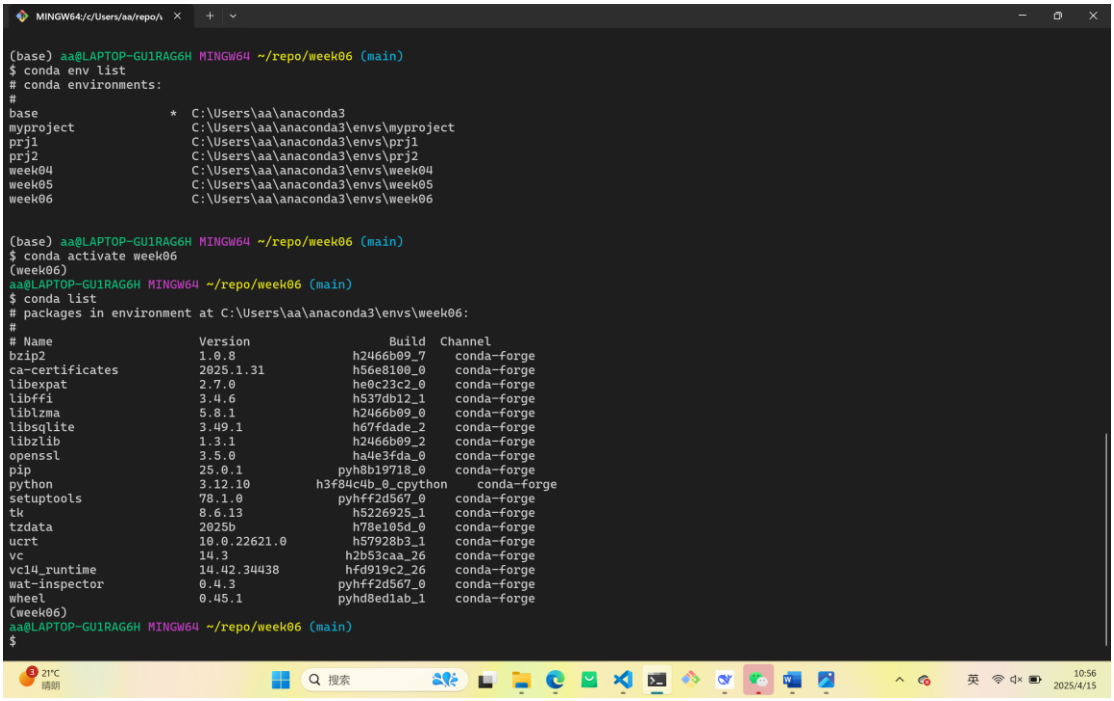


2. 用 VS Code 打开项目目录，新建一个 environment.yml 文件，指定安装 Python 3.12，然后运行 conda env create 命令创建 Conda 环境

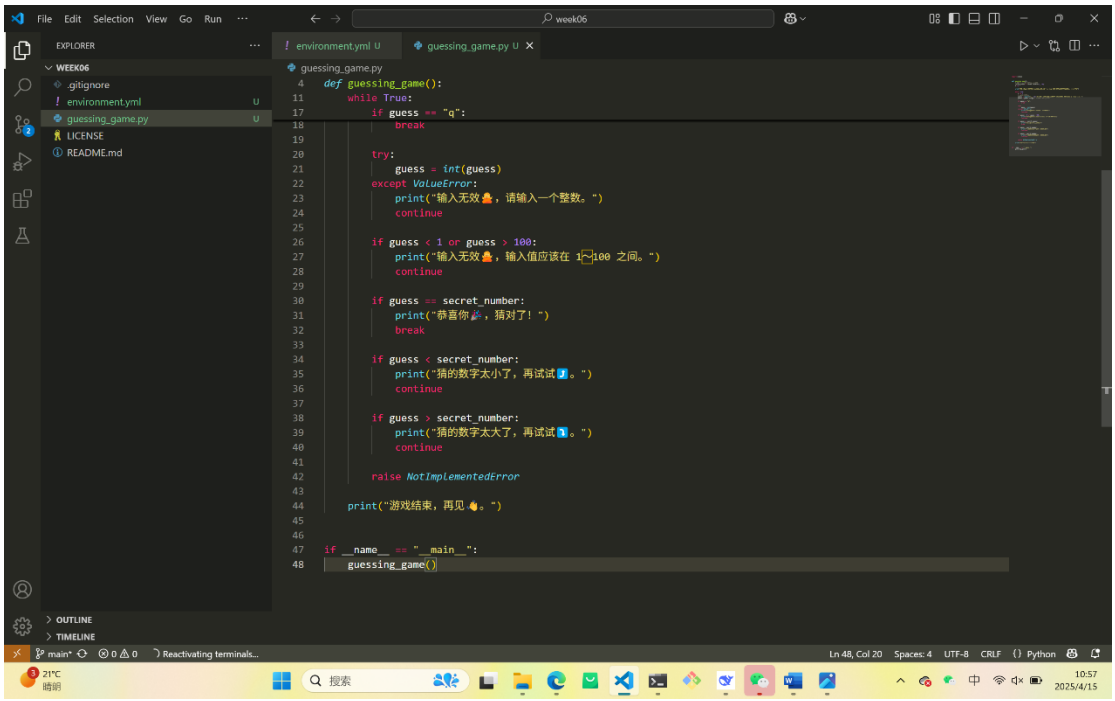



File Edit Selection View Go Run
week06
EXPLORER
WEEK06
.gitignore
environment.yml U
LICENSE
README.md
environment.yml U
environment.yml
1 name: week06
2 channels:
3 - conda-forge
4 dependencies:
5 - python=3.12
6 - wat-inspector
OUTLINE
TIMELINE
main*
Ln 1, Col 13 Spaces: 2 UTF-8 CRLF YAML
创业板指
-0.37%
搜索
10:52
2025/4/15


MINGW64:/c/Users/aa/repo/
drwxr-xr-x 1 aa 197121 0 3月 15 21:00 week02/
drwxr-xr-x 1 aa 197121 0 3月 23 10:08 week03/
done
#
To activate this environment, use
#
$ conda activate week06
#
To deactivate an active environment, use
#
$ conda deactivate
(base) aa@LAPTOP-GU1RAG6H MINGW64 ~/repo/week06 (main)
$ conda env list
# conda environments:
#
base * C:\Users\aa\anaconda3
myproject C:\Users\aa\anaconda3\envs\myproject
prj1 C:\Users\aa\anaconda3\envs\prj1
prj2 C:\Users\aa\anaconda3\envs\prj2
week04 C:\Users\aa\anaconda3\envs\week04
week05 C:\Users\aa\anaconda3\envs\week05
week06 C:\Users\aa\anaconda3\envs\week06
(base) aa@LAPTOP-GU1RAG6H MINGW64 ~/repo/week06 (main)
$
伯恩茅 - 富勒姆
比赛分数
搜索
英
10:54
2025/4/15

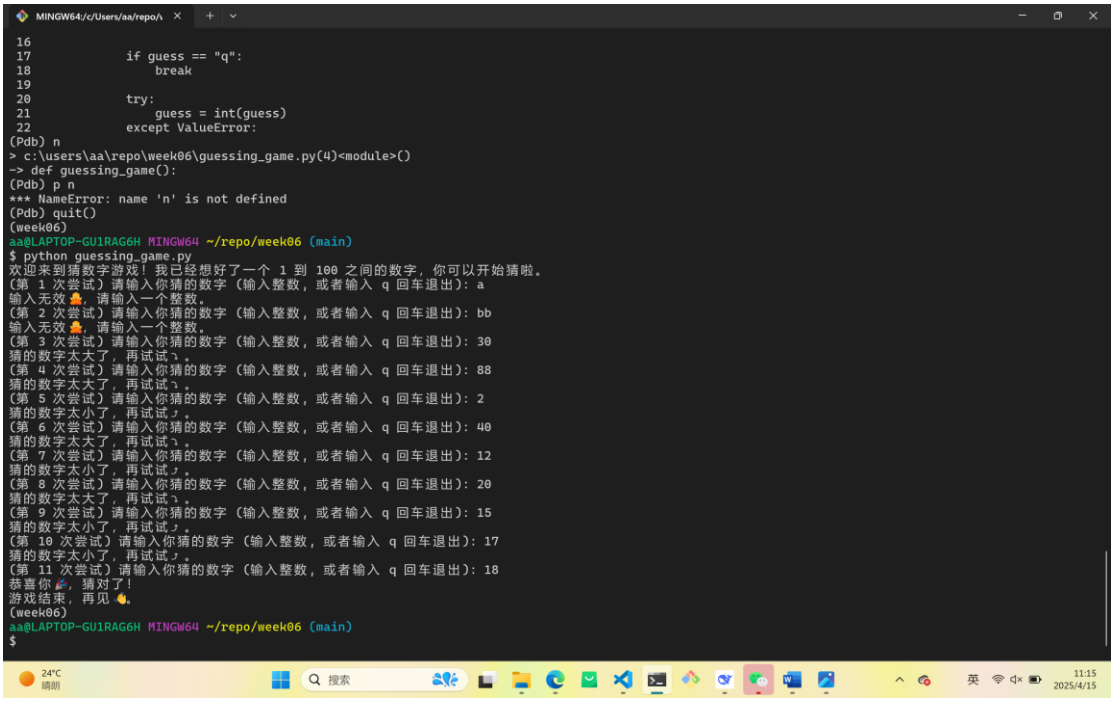


3. 创建一个 guessing_game.py 文件，复制粘贴以下代码，运用 pdb 调试器理解其运行流程：



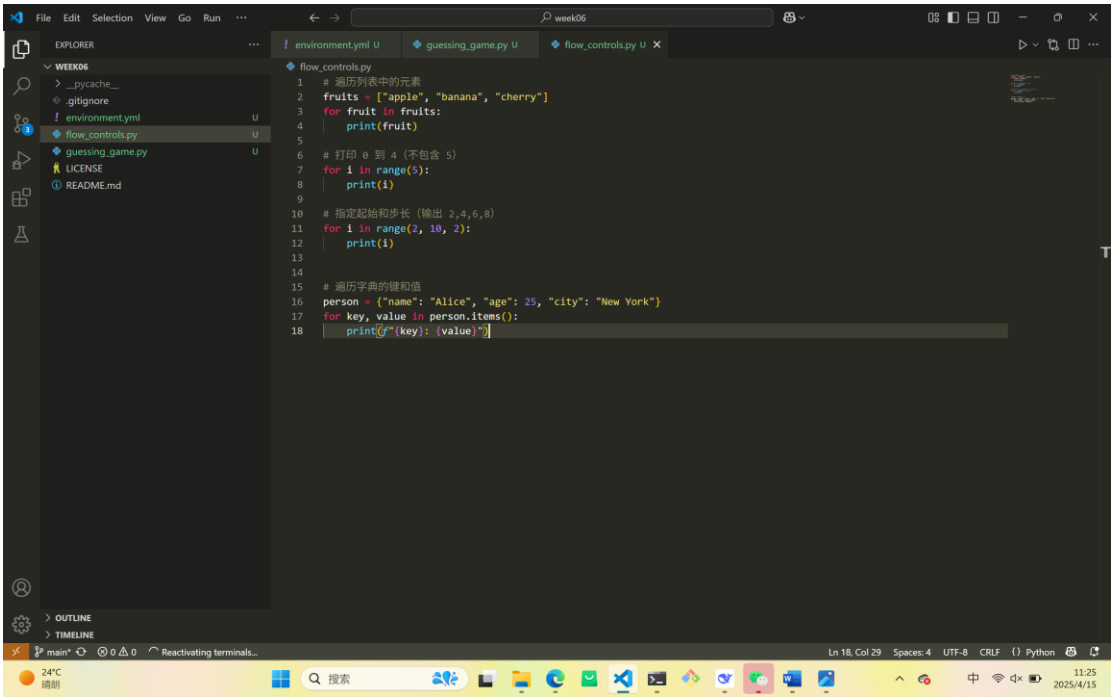
```
aa@LAPTOP-GU1RAG6H MINGW64 ~/repo/week06 (main)
$ python -m pdb guessing_game.py
> c:\users\aa\repo\week06\guessing_game.py(1)<module>()
-> import random
(Pdb) l
  1  -> import random
  2
  3
  4     def guessing_game():
  5         # 生成 1 到 100 之间的随机整数
  6         secret_number = random.randint(1, 100)
  7         n = 0
  8
  9         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。")
 10
 11         while True:
(Pdb) n
> c:\users\aa\repo\week06\guessing_game.py(4)<module>()
-> def guessing_game():
(Pdb)
> c:\users\aa\repo\week06\guessing_game.py(47)<module>()
-> if __name__ == "__main__":
(Pdb)
> c:\users\aa\repo\week06\guessing_game.py(48)<module>()
-> guessing_game()
```

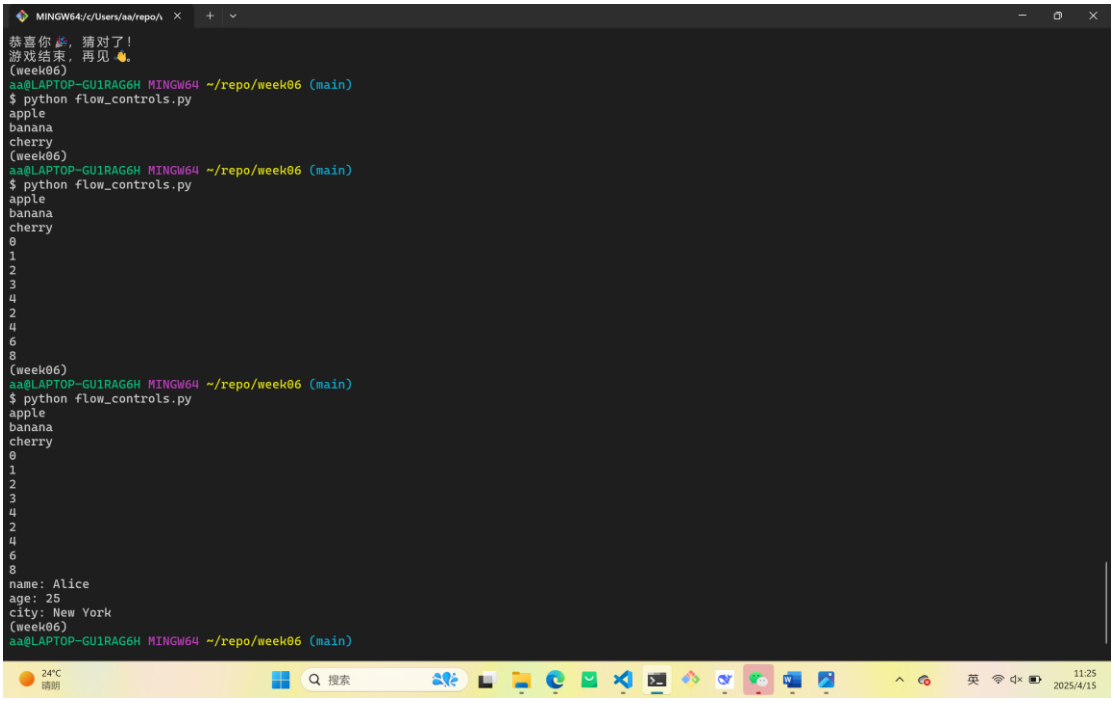
```
 18
 19             if guess == "q":
 20                 break
 21
 22  ->         try:
 23                 guess = int(guess)
 24             except ValueError:
 25                 print("输入无效🙅, 请输入一个整数。")
 26                 continue
 27
(Pdb) n
> c:\users\16386\repo\week06\guessing_game.py(23)guessing_game()
-> guess = int(guess)
(Pdb) l
 18
 19             if guess == "q":
 20                 break
 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:
(Pdb) p guess
'50'
(Pdb) p int(guess)
50
(Pdb)
```



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：

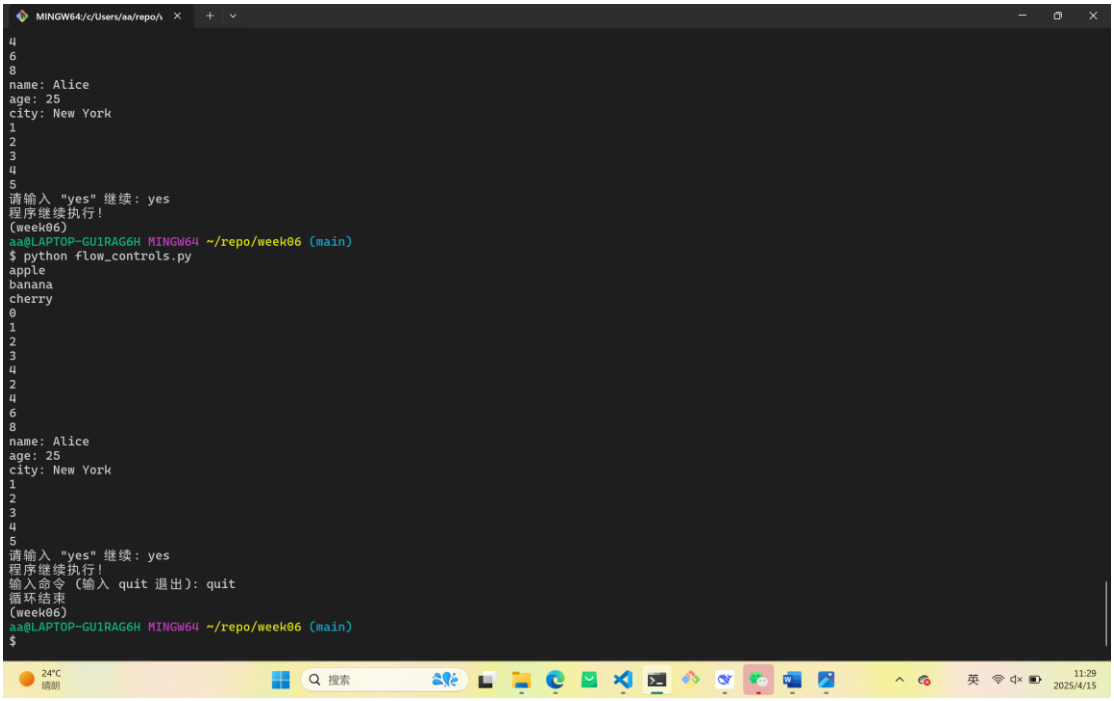
   for 迭代循环（iteration loop）





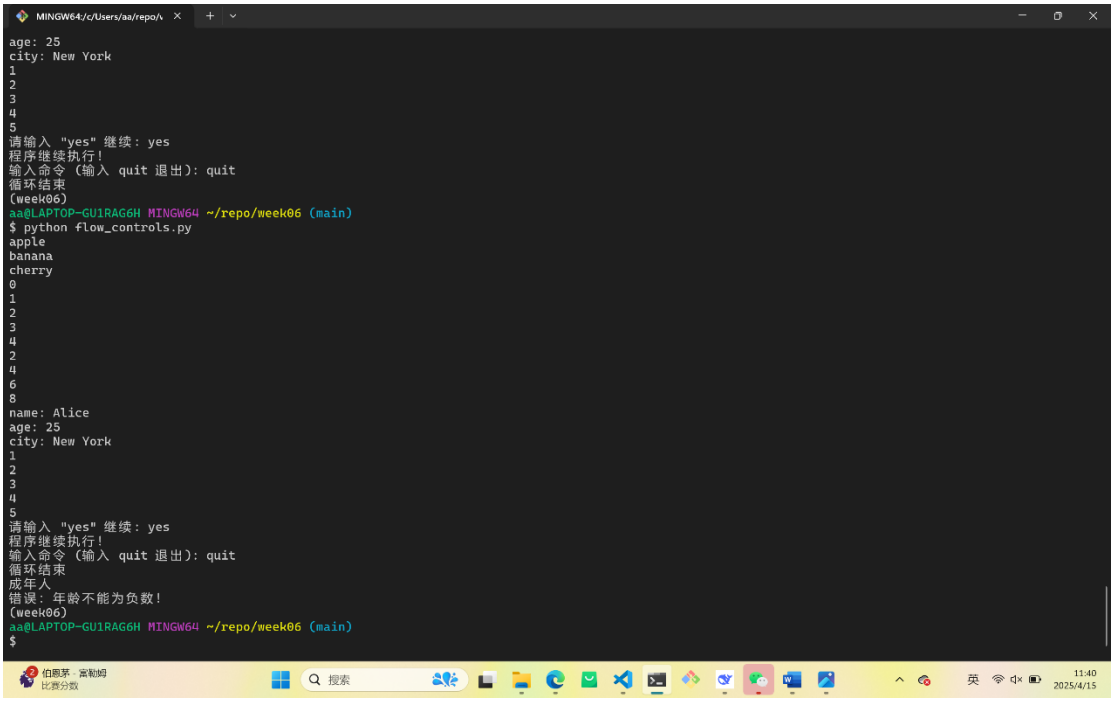
while 条件循环 (conditional loop)

```python
# 从 1 数到 5
count = 1
while count <= 5:
    print(count)
    count += 1  # 等价于 count = count + 1


# 要求用户输入 "yes" 才继续
user_input = ""
while user_input.lower() != "yes":
    user_input = input('请输入 "yes" 继续: ')
print("程序继续执行! ")


# 输入 "quit" 退出循环
While True:
    command = input("输入命令 (输入 quit 退出): ")
    if command == "quit":
        break
    print(f"执行命令: {command}")
print("循环结束")
```

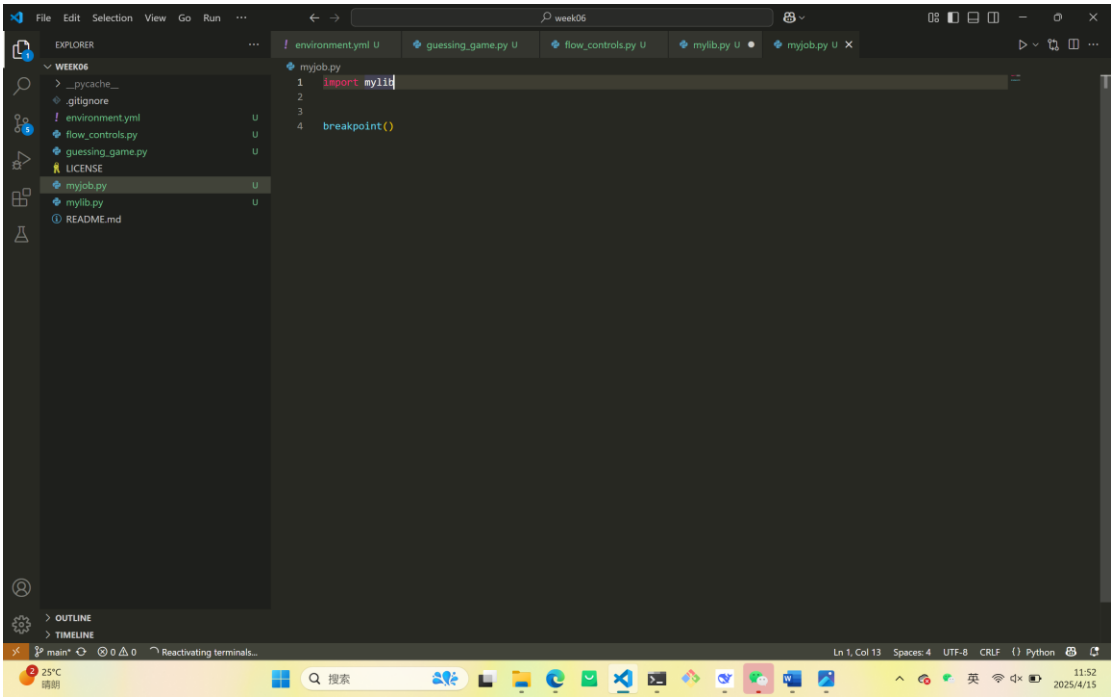


try...except[...except...else...finally] 捕捉异常的处理、raise 主动抛出异常

```python
def check_age(age):
    try:
        if age < 0:
            raise ValueError("年龄不能为负数! ")  # 主动抛出异常
        elif age < 18:
            print("未成年人")
        else:
            print("成年人")
    except ValueError as e:
        print(f"错误: {e}")

check_age(25)  # 输出: 成年人
check_age(-5)  # 输出: 错误: 年龄不能为负数!
```

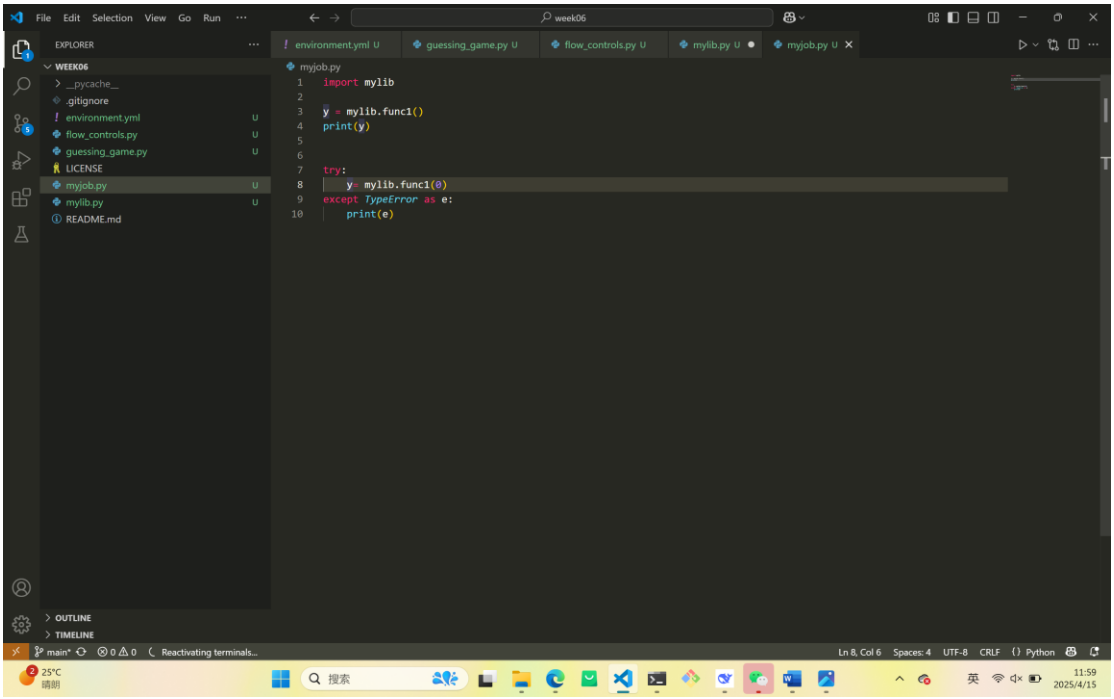


5. 创建一个 mylib.py 模块（module），在里面定义以下函数，再创建一个 myjob.py 脚本（script），从 mylib.py 导入函数并尝试调用：

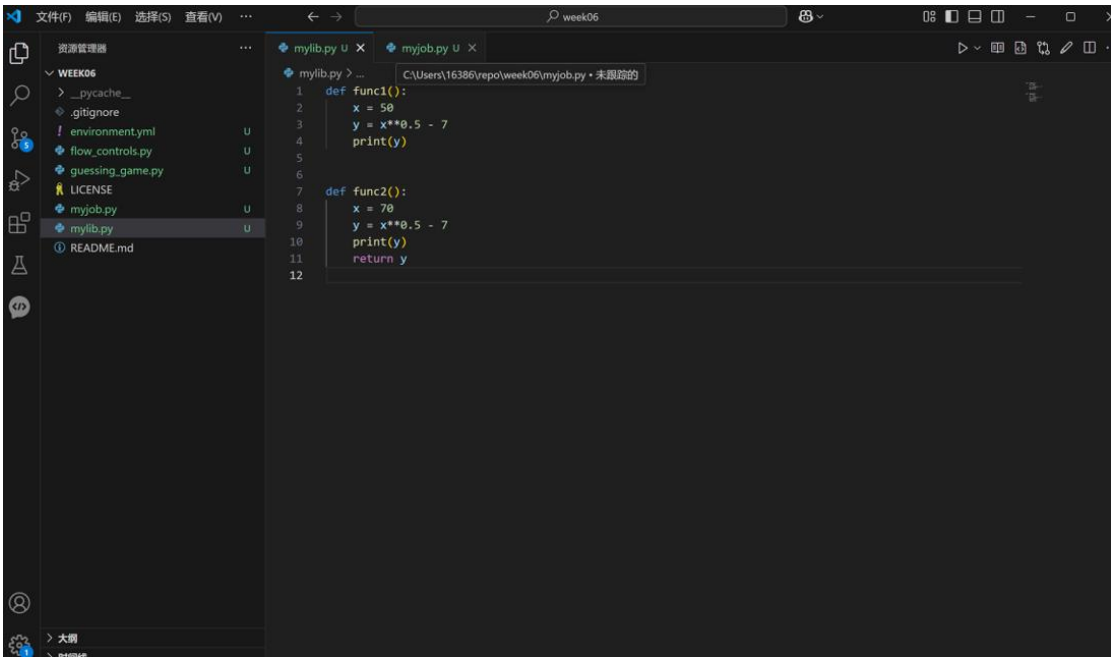


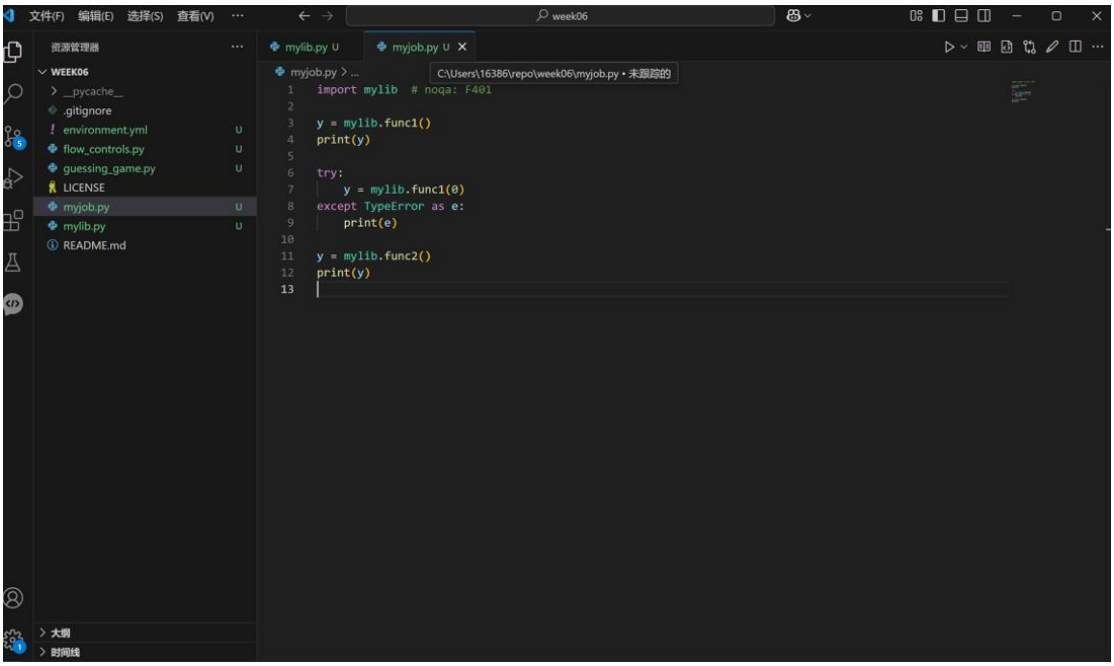
```
aa@LAPTOP-GU1RAG6H MINGW64 ~/repo/week06 (main)
$ python myjob.py
--Return--
> c:\users\aa\repo\week06\myjob.py(4)<module>()->None
-> breakpoint()
(Pdb) p mylib
<module 'mylib' from 'C:\\Users\\aa\\repo\\week06\\mylib.py'>
(Pdb) l
  1     import mylib
  2
  3
  4  -> breakpoint()
[EOF]
(Pdb) p mylib
<module 'mylib' from 'C:\\Users\\aa\\repo\\week06\\mylib.py'>
(Pdb) wat / mylib
```

定义函数 func1，没有形参，没有返回值

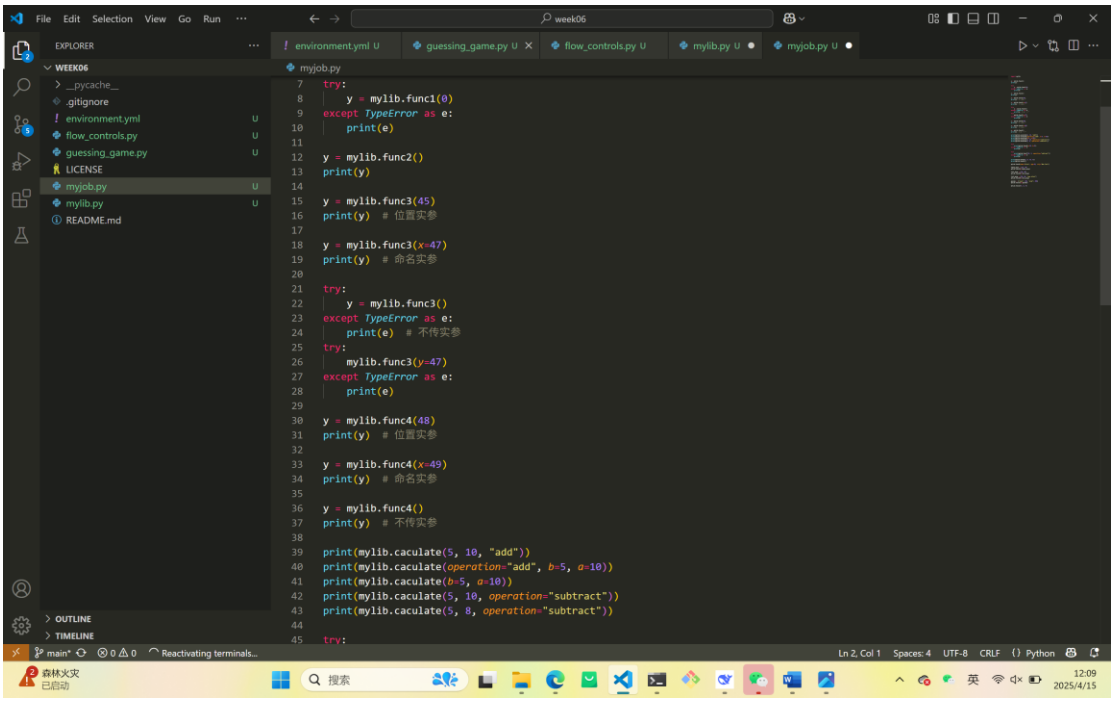


定义函数 func2，没有形参，有返回值

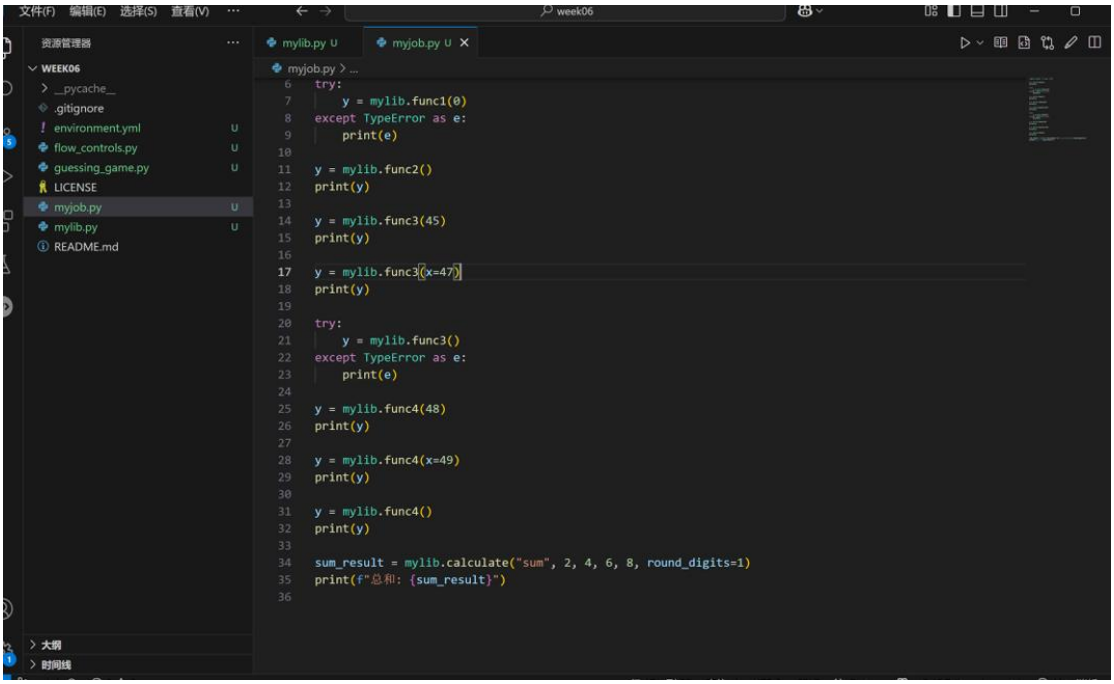




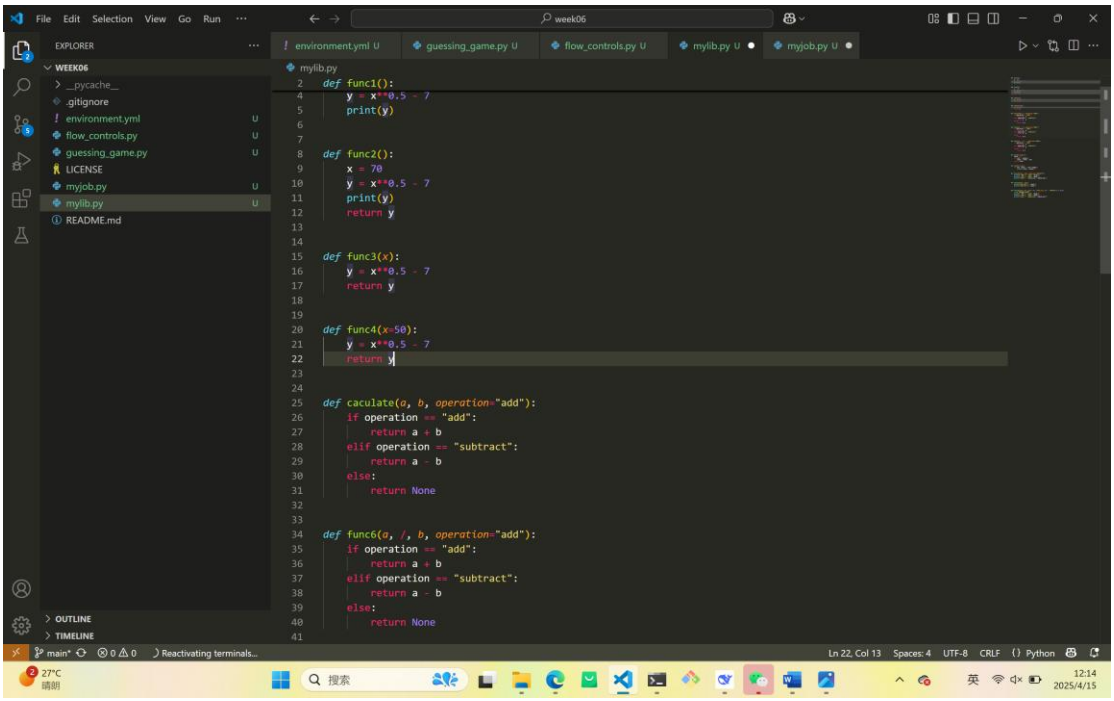
定义函数 func4，只有一个 命名形参 (named parameter)，先传入 位置实参 调用，再传入 命名实参 调用，再尝试不传实参 (取默认值)

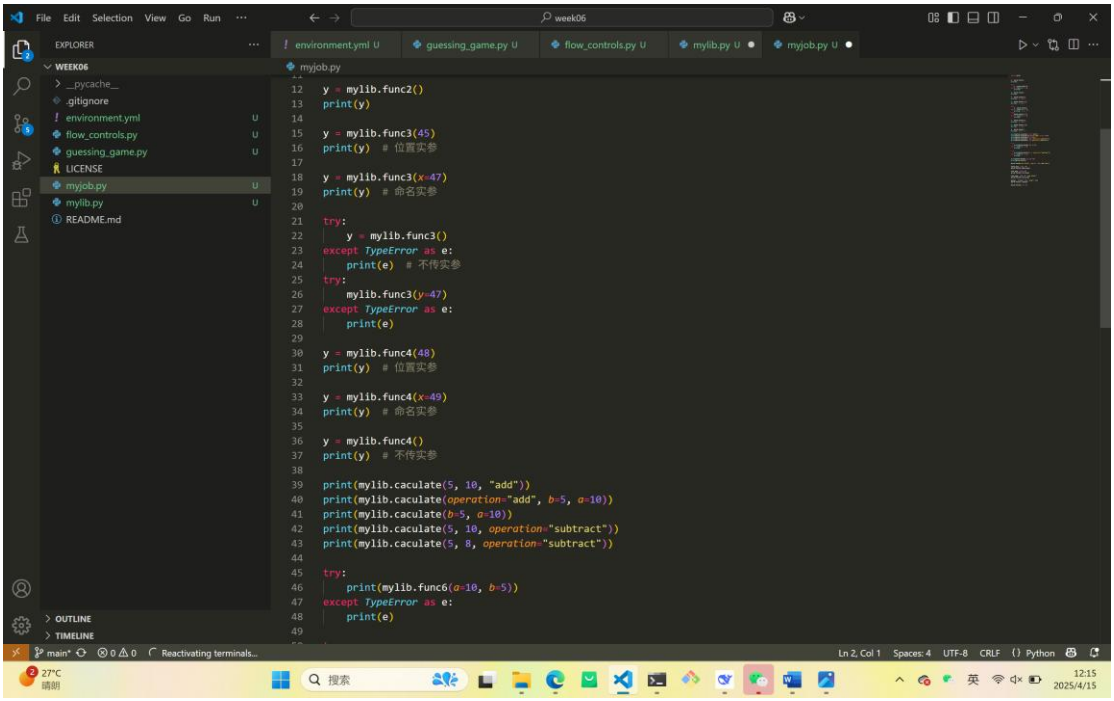


定义函数 func5，接受多个位置形参和命名形参，尝试以位置/命名各种不同方式传入实参，注意位置参数必须排在命名参数之前



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

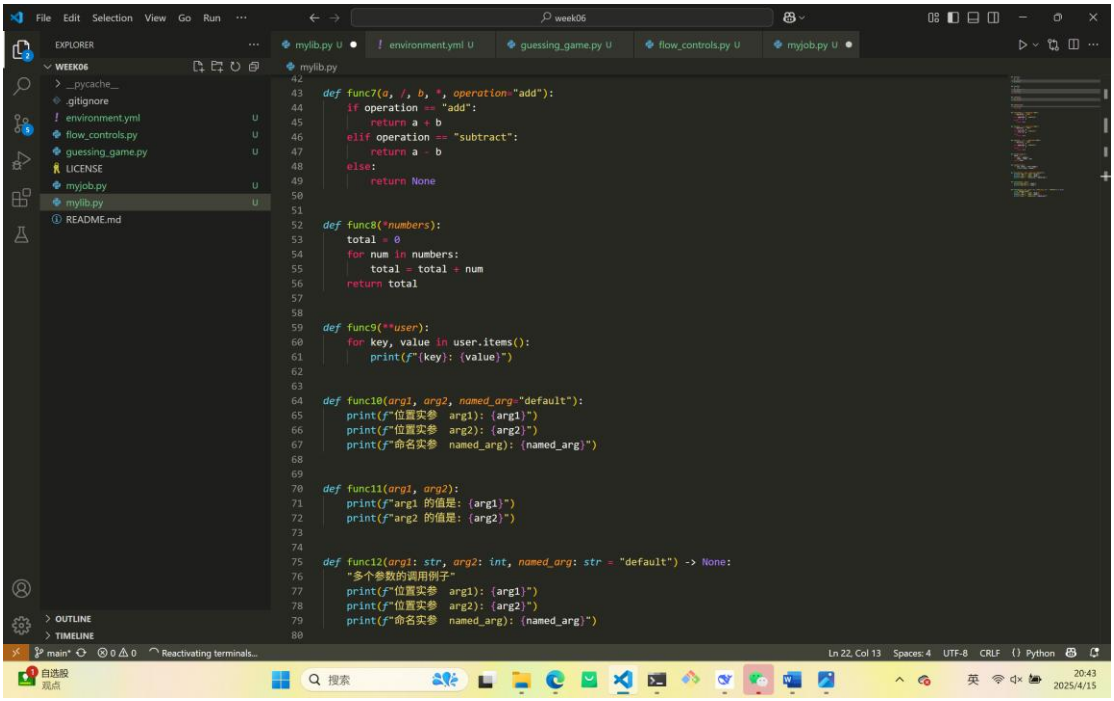
定义函数 func8，在位置形参的最后，在形参名称前使用 * 允许传入任意数量的位置实参 (被打包为元组)

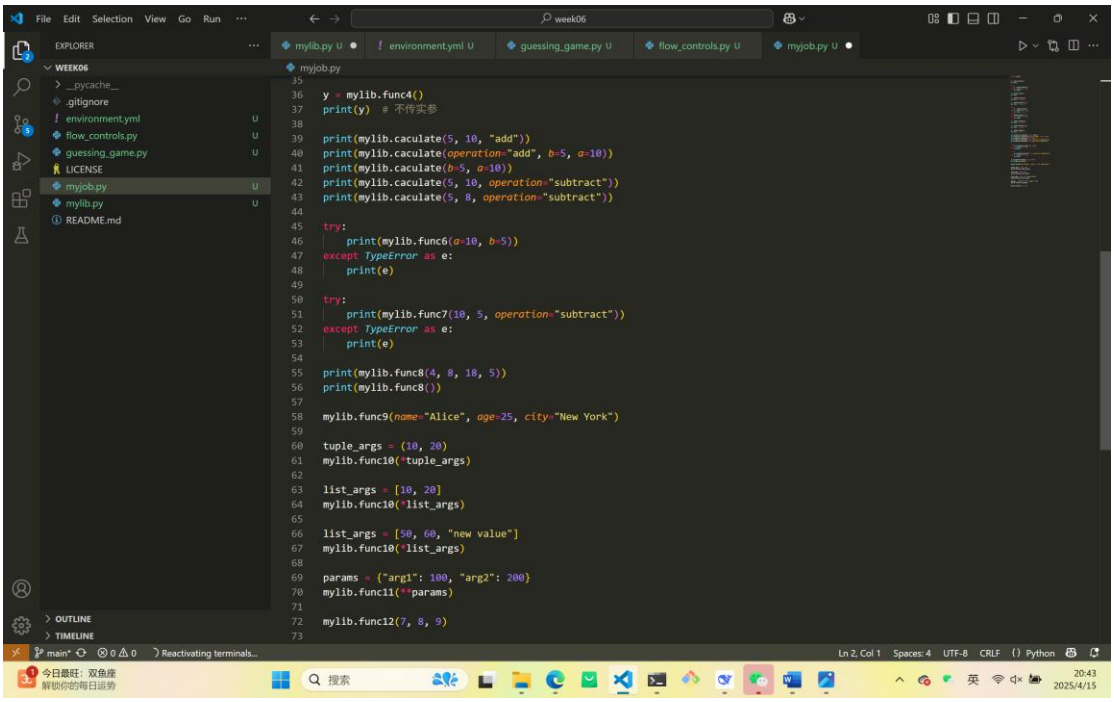
定义函数 func9，在命名形参的最后，在形参名称前使用 ** 允许传入任意数量的命名实参 (被打包为字典)

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。

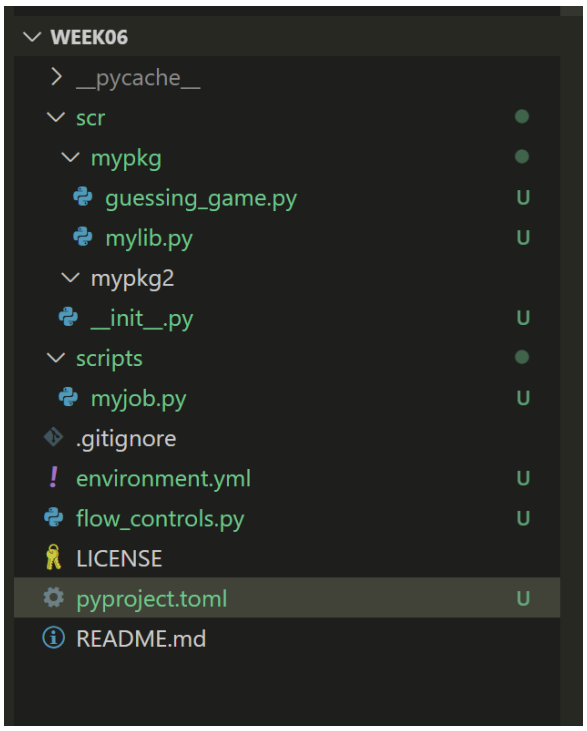




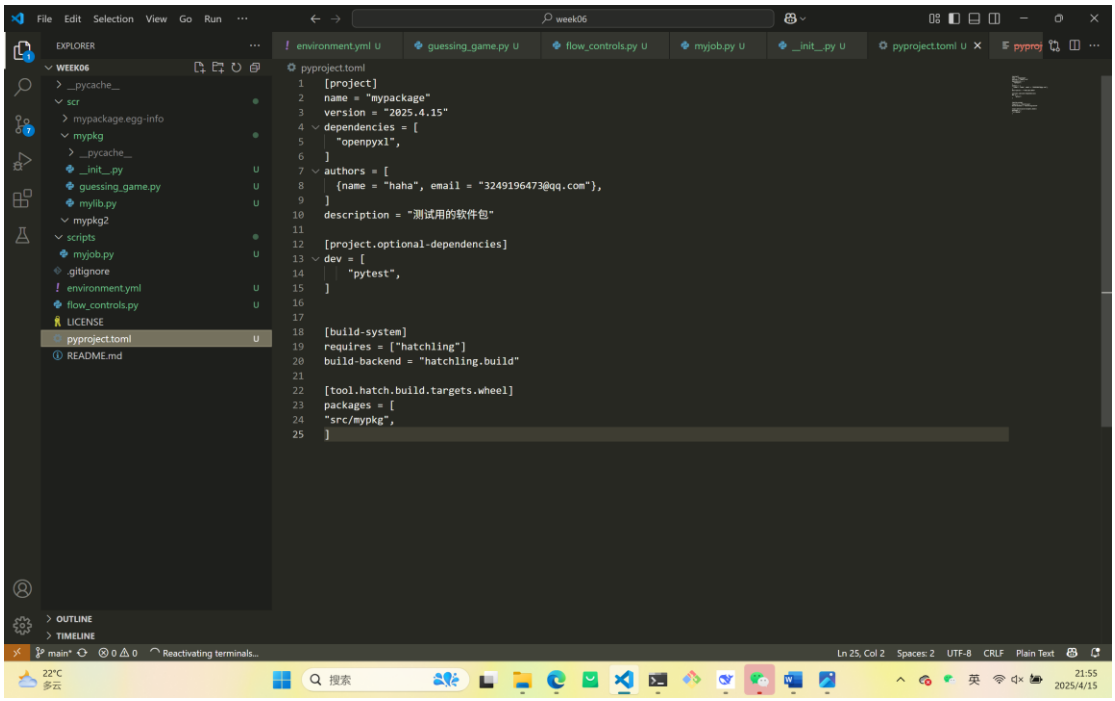
6. 把 mylib 模块转变为 **软件包** (package) 安装进当前的 Conda 环境来使用

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

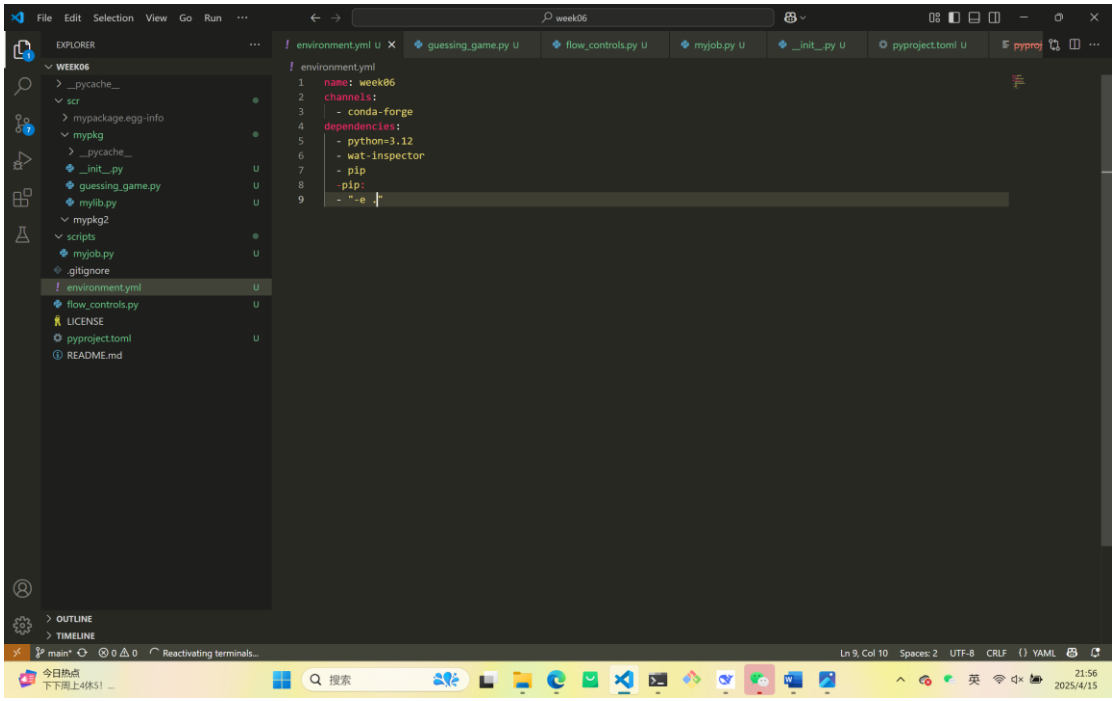
将 mylib.py 模块移动至 src/mypkg/mylib.py，创建 src/mypkg/__init__.py 文件，准备好软件包的源代码



创建 pyproject.toml 配置文件，按照文档填写基本的软件包信息



修改 environment.yml 文件，使得 conda env create 自动安装本地可编辑软件包


File Edit Selection View Go Run
week06
EXPLORER
WEEK06
pycache
scr
mypackage.egg-info
mypkg
pycache
__init__.py
guessing_game.py
mylib.py
mypkg2
scripts
myjob.py
.gitignore
environment.yml
flow_controls.py
LICENSE
pyproject.toml
README.md
environment.yml
guessing_game.py
flow_controls.py
myjob.py
__init__.py
pyproject.toml
name: week06
channels:
- conda-forge
dependencies:
- python=3.12
- wat-inspector
- pip
- pip:
- "-e .[
OUTLINE
TIMELINE
main*
Reactivating terminals...
Ln 9, Col 10
Spaces: 2
UTF-8
CRLF
YAML
今日热点
下下周上4休5！...
搜索
英
21:56
2025/4/15