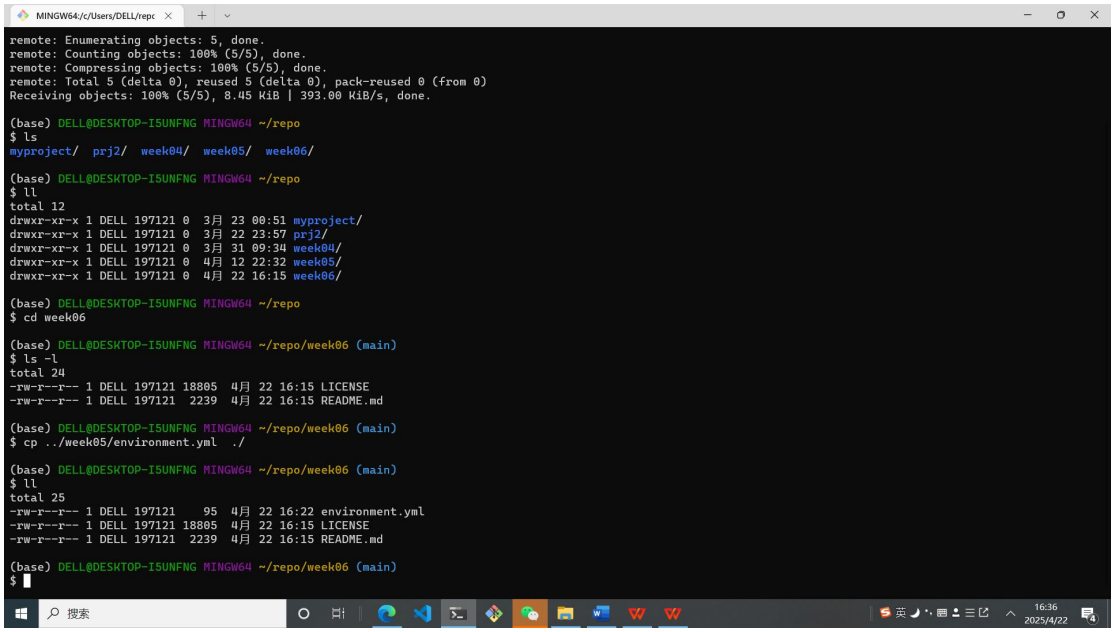
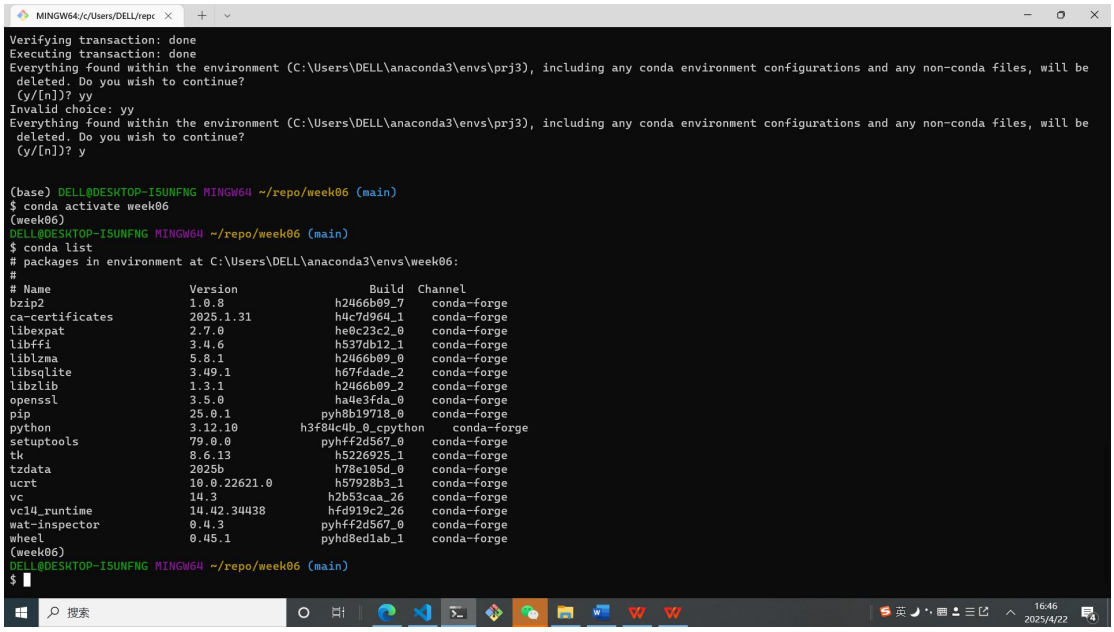
Week07 作业报告

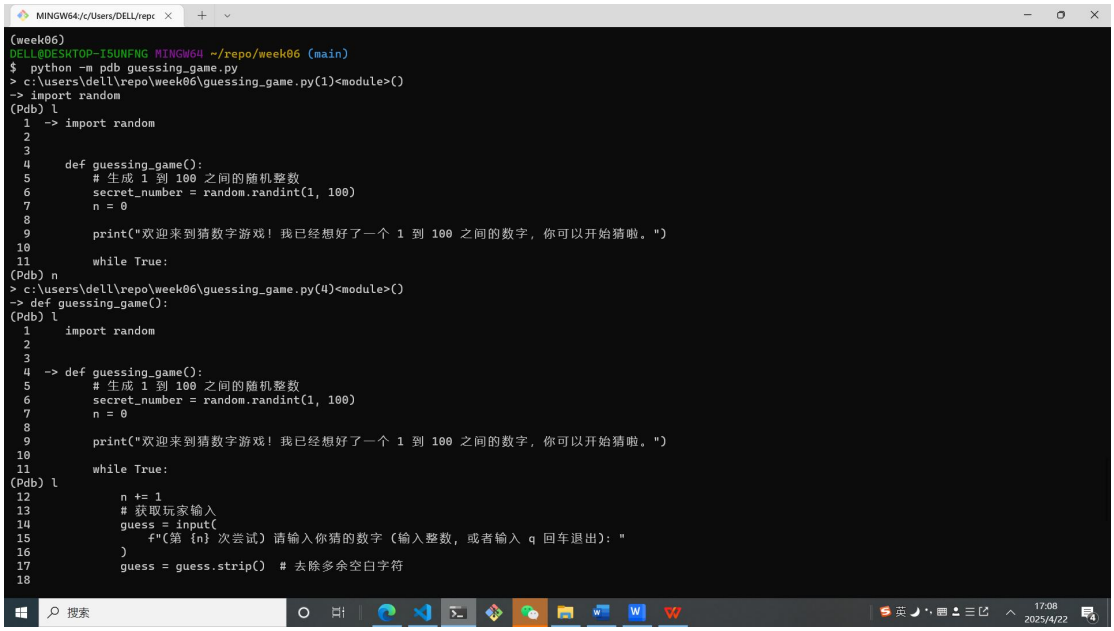
1、Fork 第 06 周打卡仓库至你的名下，然后将你名下的这个仓库 Clone 到你的本地计算机

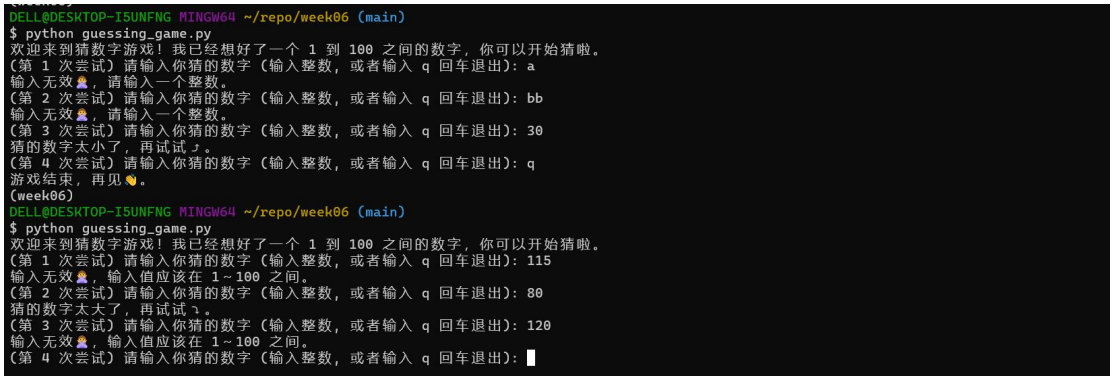


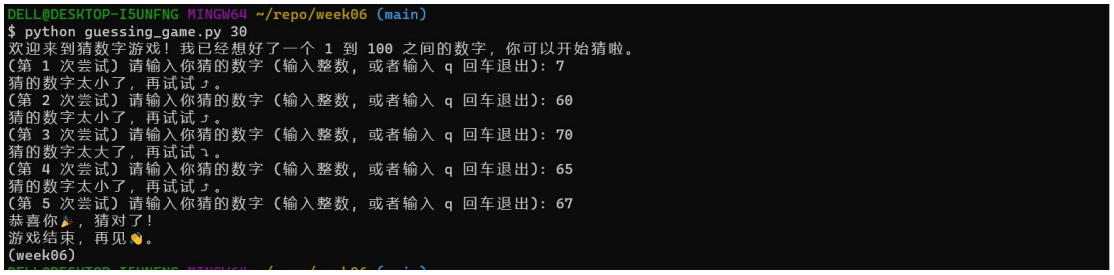
2、用 VS Code 打开项目目录，新建一个 environment.yml 文件，指定安装 Python 3.12，然后运行 conda env create 命令创建 Conda 环境



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：

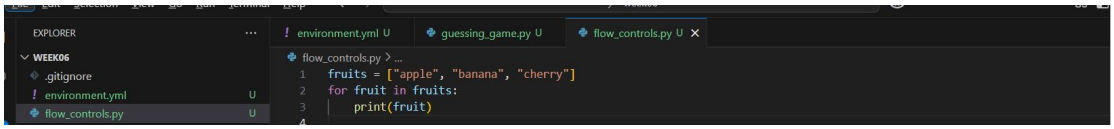






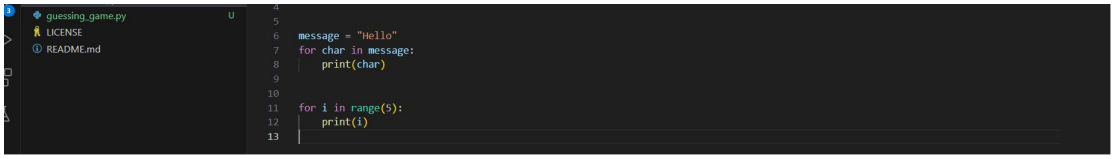
4、创建一个 flow_controls.py 文件，让豆包 (或 DeepSeek 等任何大模型) 生成例子，尝试运行，体会理解以下 Python 流程控制语句：

For 循环



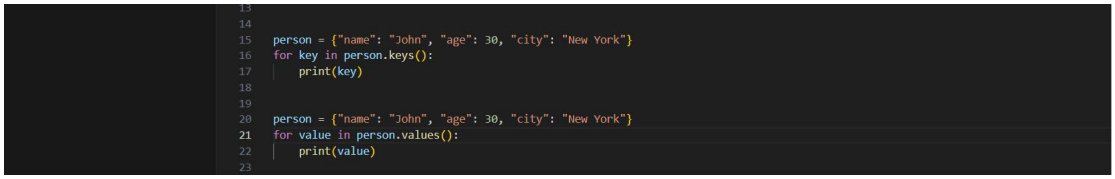

```
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$ python flow_controls.py
apple
banana
cherry
(week06)
```

字符串



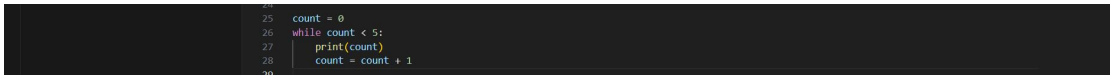

```
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$ python flow_controls.py
apple
banana
cherry
H
e
l
l
o
0
1
2
3
4
(week06)
```

字典




```
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$ python flow_controls.py
apple
banana
cherry
H
e
l
l
o
0
1
2
3
4
name
age
city
John
30
New York
(week06)
```

While 循环

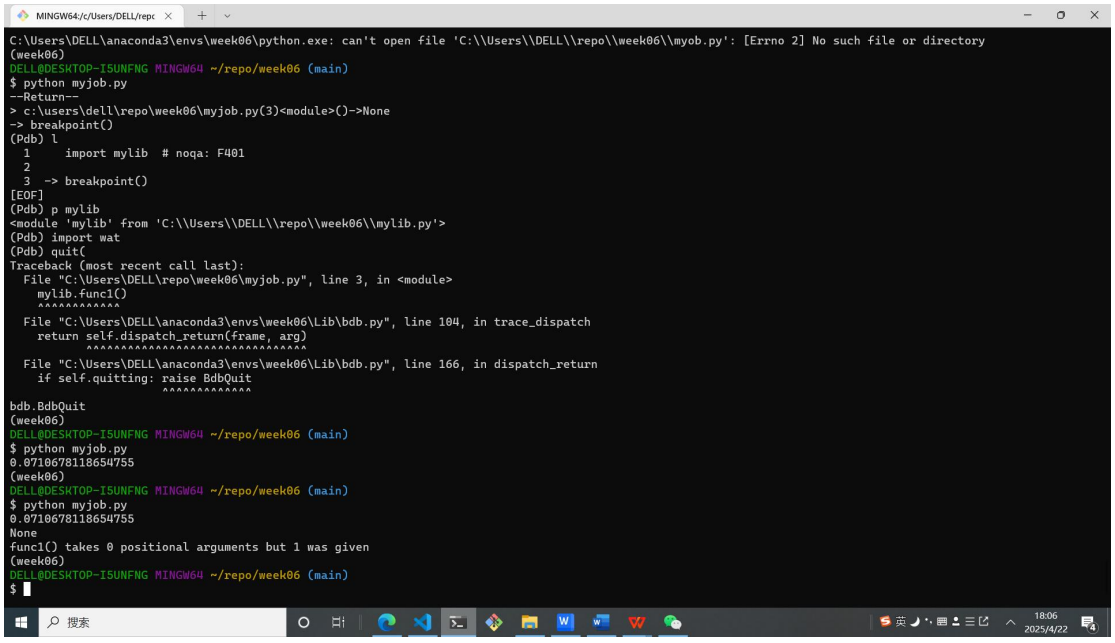


```
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$ python flow_controls.py
apple
banana
cherry
H
e
l
l
o
0
1
2
3
4
name
age
city
John
30
New York
0
1
2
3
4
(week06)
```

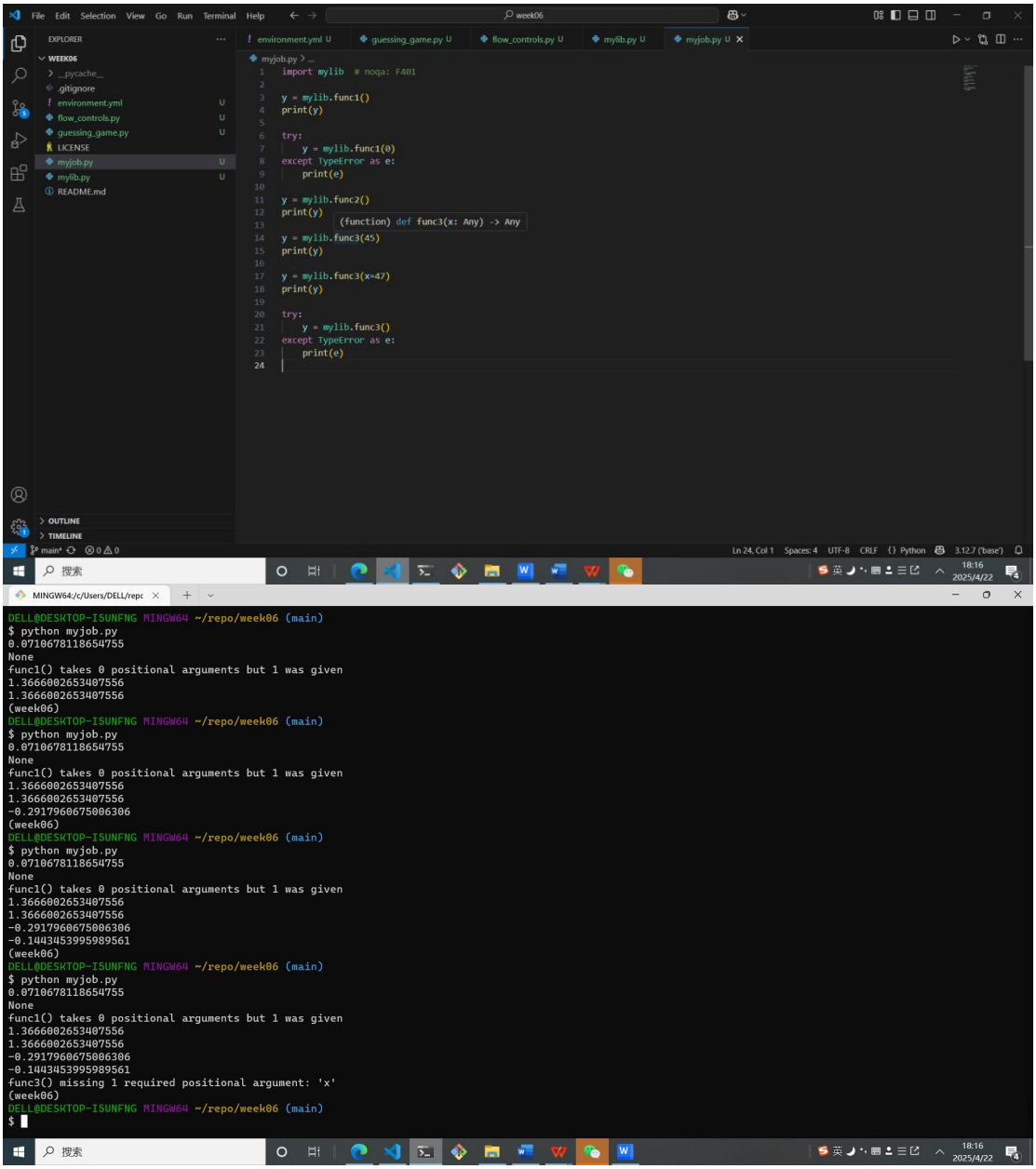
for 循环：会自动对可迭代对象中的每个元素进行迭代，每次迭代时，将当前元素赋值给指定的变量，然后执行循环体代码，直至遍历完所有元素。

while 循环：会先判断条件表达式的真假，如果为 `True` 则执行循环体代码，执行完后再次判断条件表达式，若仍为 `True` 则继续执行，直到条件表达式为 `False` 才停止循环。

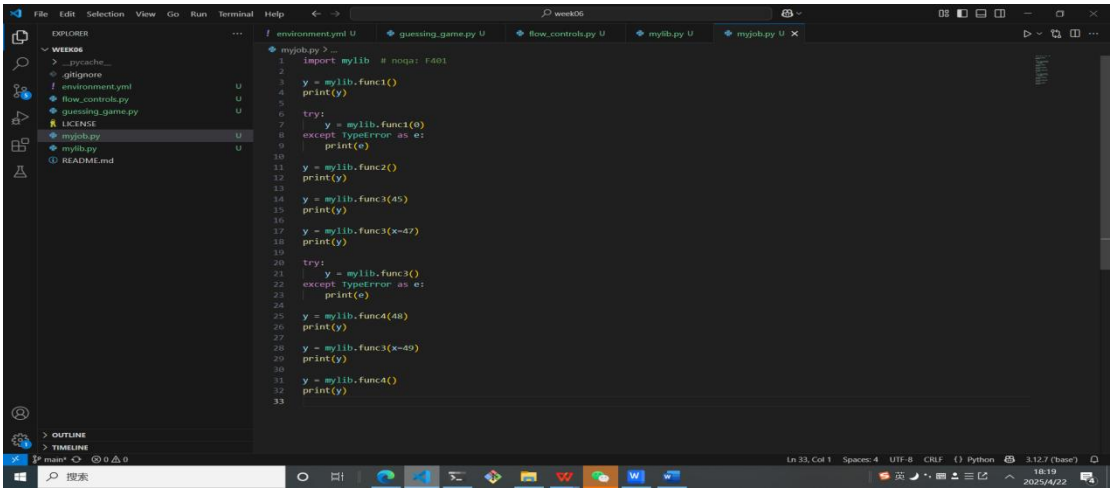
5、创建一个 mylib.py 模块（module），在里面定义以下函数，再创建一个 myjob.py 脚本（script），从 mylib.py 导入函数并尝试调用



位置实参


import mylib # noqa: F401
y = mylib.func1()
print(y)
try:
y = mylib.func1(0)
except TypeError as e:
print(e)
y = mylib.func2()
print(y)
(function) def func3(x: Any) -> Any
y = mylib.func3(45)
print(y)
y = mylib.func3(x=47)
print(y)
try:
y = mylib.func3()
except TypeError as e:
print(e)
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$ python myjob.py
0.07106781186547551
None
func1() takes 0 positional arguments but 1 was given
1.3666002653407556
1.3666002653407556
(week06)
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$ python myjob.py
0.07106781186547551
None
func1() takes 0 positional arguments but 1 was given
1.3666002653407556
1.3666002653407556
-0.2917960675006306
(week06)
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$ python myjob.py
0.07106781186547551
None
func1() takes 0 positional arguments but 1 was given
1.3666002653407556
1.3666002653407556
-0.2917960675006306
-0.14434539959895561
(week06)
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$ python myjob.py
0.07106781186547551
None
func1() takes 0 positional arguments but 1 was given
1.3666002653407556
1.3666002653407556
-0.2917960675006306
-0.14434539959895561
func3() missing 1 required positional argument: 'x'
(week06)
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$

命名实参


import mylib # noqa: F401
y = mylib.func1()
print(y)
try:
y = mylib.func1(0)
except TypeError as e:
print(e)
y = mylib.func2()
print(y)
y = mylib.func3(45)
print(y)
y = mylib.func3(x=47)
print(y)
try:
y = mylib.func3()
except TypeError as e:
print(e)
y = mylib.func4(48)
print(y)
y = mylib.func3(x=49)
print(y)
y = mylib.func4()
print(y)

```
$ python myjob.py
0.07106781186654755
None
func1() takes 0 positional arguments but 1 was given
1.3666002653407556
1.3666002653407556
-0.2917960675006306
-0.14434539959895561
func3() missing 1 required positional argument: 'x'
-0.07179676972449123
(week06)
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$ python myjob.py
0.07106781186654755
None
func1() takes 0 positional arguments but 1 was given
1.3666002653407556
1.3666002653407556
-0.2917960675006306
-0.14434539959895561
func3() missing 1 required positional argument: 'x'
-0.07179676972449123
0.0
(week06)
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$ python myjob.py
0.07106781186654755
None
func1() takes 0 positional arguments but 1 was given
1.3666002653407556
1.3666002653407556
-0.2917960675006306
-0.14434539959895561
func3() missing 1 required positional argument: 'x'
-0.07179676972449123
0.0
0.07106781186654755
(week06)
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$
```

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，接受一个位置形参，两个命名形参，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

```python
def func11(arg1, arg2):
    print(f"arg1 的值是: {arg1}")
    print(f"arg2 的值是: {arg2}")
```

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

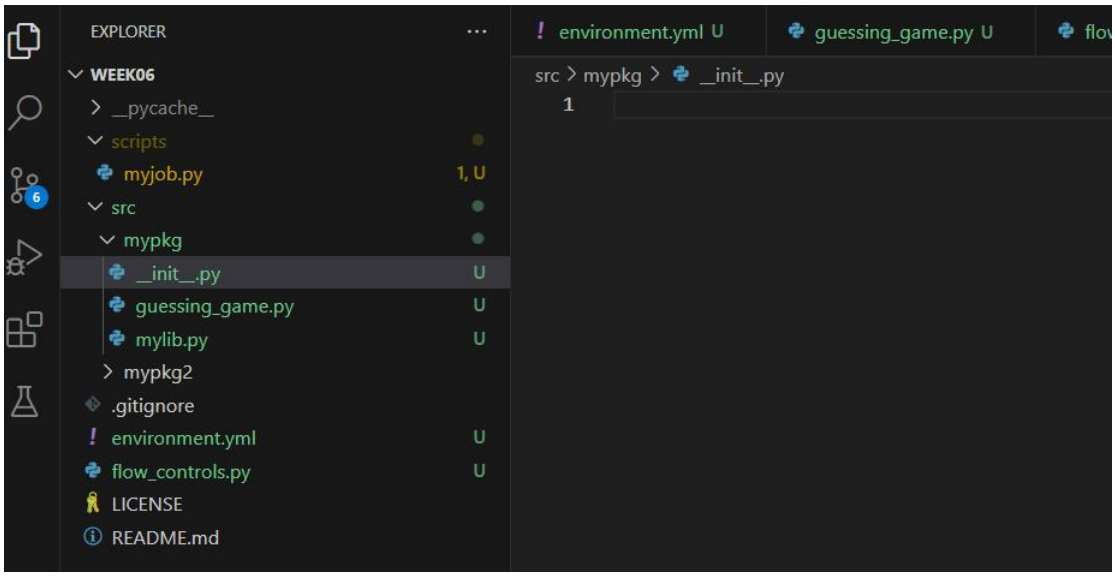
```python
def func12(arg1: str, arg2: int, named_arg: str = "default") -> None:
    "多个参数的调用例子"
    print(f"位置实参  arg1): {arg1}")
    print(f"位置实参  arg2): {arg2}")
    print(f"命名实参  named_arg): {named_arg}")
```

```
MINGW64:/c/Users/DELL/repo
0.07106781186547 55
None
func1() takes 0 positional arguments but 1 was given
1.3666002653407556
1.3666002653407556
-0.2917960675006306
-0.14434539959895 61
func3() missing 1 required positional argument: 'x'
-0.07179676972449123
0.0
0.07106781186547 55
15
15
15
-5
-3
func6() got some positional-only arguments passed as keyword arguments: 'a'
5
35
0
name: Alice
age: 25
city: New York
位置实参  arg1): 10
位置实参  arg2): 20
命名实参  named_arg): default
位置实参  arg1): 10
位置实参  arg2): 20
命名实参  named_arg): default
位置实参  arg1): 50
位置实参  arg2): 60
命名实参  named_arg): new value
arg1 的值是: 100
arg2 的值是: 200
位置实参  arg1): 7
位置实参  arg2): 8
命名实参  named_arg): 9
(week06)
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$
```

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)
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$ python scripts/myjob.py
Traceback (most recent call last):
  File "C:\Users\DELL\repo\week06\scripts\myjob.py", line 1, in <module>
    import mylib  # noqa: F401
    ^^^^^^^^^^^
ModuleNotFoundError: No module named 'mylib'
(week06)
DELL@DESKTOP-I5UNFNG MINGW64 ~/repo/week06 (main)
$
```

```
EXPLORER
WEEK06
  __pycache__
    mylib.cpython-312.pyc
  scripts
    myjob.py 1, U
  src
  mypkg
    __init__.py U
    guessing_game.py U
flow_controls.py U
pyproject.toml U
environment.yml U
mylib.py U
myjob.py 1, U
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 ."
10
```



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

在 pyproject.toml 配置文件里，按照 文档 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

```
DELL@DESKTOP-ISUNFNG MINGW64 ~/repo/week06 (main)
$ pip install -e.
Looking in indexes: https://mirrors.tuna.tsinghua.edu.cn/pypi/web/simple
Obtaining file:///C:/Users/DELL/repo/week06
  Installing build dependencies ... done
  Checking if build backend supports build_editable ... done
  Getting requirements to build editable ... done
  Installing backend dependencies ... done
  Preparing editable metadata (pyproject.toml) ... done
Collecting openpyxl (from mypackage==2025.4.14)
  Downloading https://mirrors.tuna.tsinghua.edu.cn/pypi/web/packages/c0/da/977ded879c29cbd04de313843e76868e6e13408a94ed6b987245dc7c8506/openpyxl-3.1.5-py2.p
y3-none-any.whl (250 kB)
Collecting et-xmlfile (from openpyxl->mypackage==2025.4.14)
  Downloading https://mirrors.tuna.tsinghua.edu.cn/pypi/web/packages/c1/8b/5fe2cc11fee489817272089c4203e679c63b570a5aaeb18d852ae3cbba6a/et_xmlfile-2.0.0-py3
-none-any.whl (18 kB)
Building wheels for collected packages: mypackage
  Building editable for mypackage (pyproject.toml) ... done
  Created wheel for mypackage: filename=mypackage-2025.4.14-py2.py3-none-any.whl size=7276 sha256=2b773119a98eee80a4c137fee60693614bbc6f7fa94e99af2b1a0342b8
510aac
  Stored in directory: C:\Users\DELL\AppData\Local\Temp\pip-ephem-wheel-cache-75_csrf5\wheels\78\26\3b\3047d49ce5791ac0f90411a5b5d9923295d59e0cde27b493e4
Successfully built mypackage
Installing collected packages: et-xmlfile, openpyxl, mypackage
Successfully installed et-xmlfile-2.0.0 mypackage-2025.4.14 openpyxl-3.1.5
(week06)
```