

# 第四周学习报告

- 1. 将第四周仓库 Clone 到本地计算机

```
MINGW64: c/Users/hp/repo x + v
(base)
hp@LAPTOP-L5E04S06 MINGW64 ~
$ cd repo
(base)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo
$ ls -l
total 12
drwxr-xr-x 1 hp 197121 0 3月 20 20:19 ccprj/
drwxr-xr-x 1 hp 197121 0 3月 14 11:52 mywork/
drwxr-xr-x 1 hp 197121 0 3月 19 22:14 prj1/
drwxr-xr-x 1 hp 197121 0 3月 7 21:41 week01/
drwxr-xr-x 1 hp 197121 0 3月 14 12:25 week02/
drwxr-xr-x 1 hp 197121 0 3月 20 20:50 week03/
(base)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo
$ git clone https://gitcode.com/crazy77/week04.git
Cloning into 'week04'...
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)
Unpacking objects: 100% (5/5), 8.43 KiB | 410.00 KiB/s, done.
(base)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo
$
```

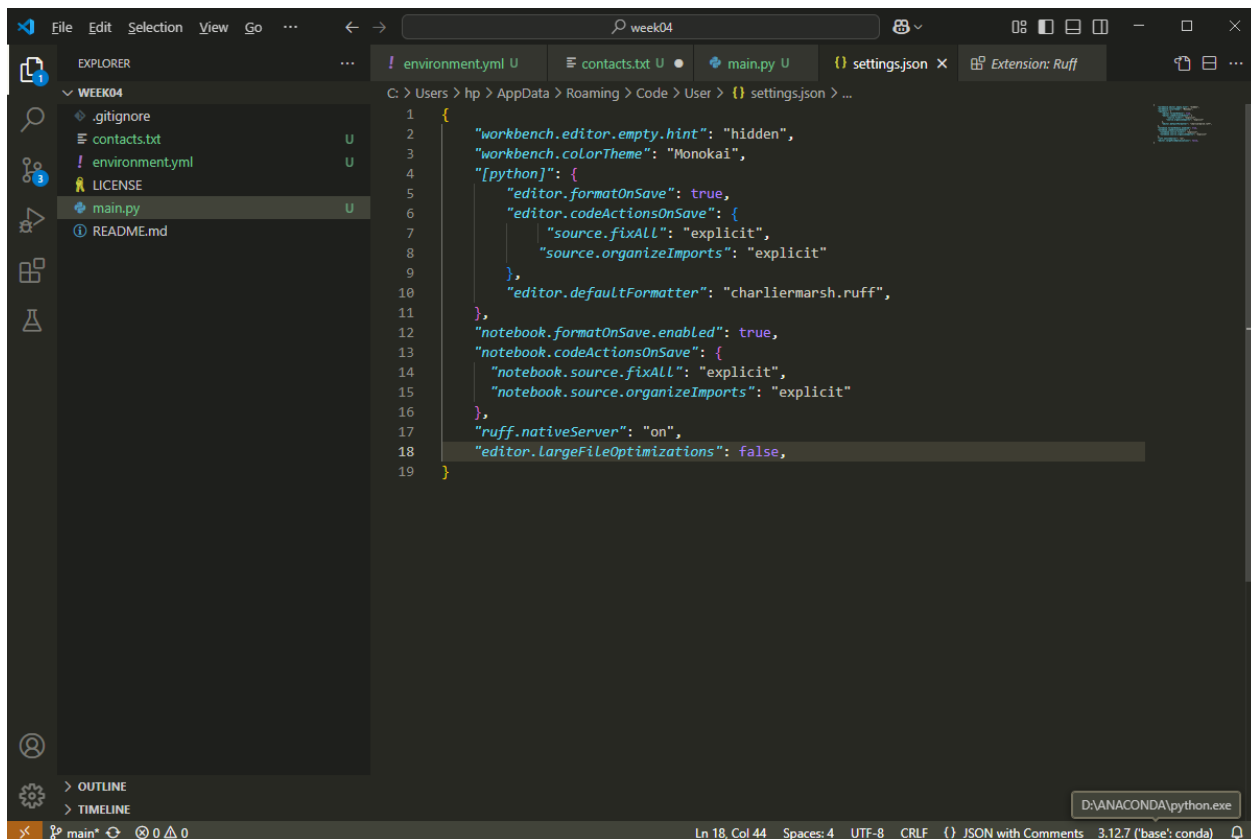
- 2. 创建 Conda 环境

```
MINGW64: c/Users/hp/repo/v x + v
(base)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$ cp ../ccprj/environment.yml ./
(base)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$ ls -l
total 25
-rw-r--r-- 1 hp 197121 83 3月 26 22:39 environment.yml
-rw-r--r-- 1 hp 197121 18805 3月 26 22:30 LICENSE
-rw-r--r-- 1 hp 197121 2239 3月 26 22:30 README.md
(base)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$ cat environment.yml
name: ccprj
done
#
# To activate this environment, use
#
# $ conda activate week04
#
# To deactivate an active environment, use
#
# $ conda deactivate
(base)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$
```

- 3.新建一个 contacts.txt 文件

```
(base)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$ ls -l
total 25
-rw-r--r-- 1 hp 197121 0 3月 26 22:51 contacts.txt
-rw-r--r-- 1 hp 197121 76 3月 26 22:43 environment.yml
-rw-r--r-- 1 hp 197121 18805 3月 26 22:30 LICENSE
-rw-r--r-- 1 hp 197121 2239 3月 26 22:30 README.md
(base)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$ cat contacts.txt
燕小六 男 yanxiaoliu@126.com
杨蕙兰 女 yanghuilan@163.com
莫小宝 男 moxiaobao@126.com
慕容嫣 女 murongyan@163.com
邢育森 男 xingyusen@126.com
凌腾云 女 lingtengyun@163.com
(base)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$
```

- 4.安装 Ruff 扩展并按照文档配置 Ruff



- 5.运行AI生成的代码

```
MINGW64/c/Users/hp/repo/v X + v
邢育森 男 xingyusen@126.com
凌腾云 女 lingtengyun@163.com
(week04)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$ python main.py
邮件内容已成功写入 emails.txt 文件。
(week04)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$ cat emails.txt
to: <moxiaobao@126.com>
尊敬的莫小宝先生，您的会员资格即将到期，请及时续费。
---
to: <xingyusen@126.com>
尊敬的邢育森先生，您的会员资格即将到期，请及时续费。
---
to: <yanxiaoliu@126.com>
尊敬的燕小六先生，您的会员资格即将到期，请及时续费。
---
to: <lingtengyun@163.com>
尊敬的凌腾云女士，您的会员资格即将到期，请及时续费。
---
to: <murongyan@163.com>
尊敬的慕容嫣女士，您的会员资格即将到期，请及时续费。
---
to: <yanghuilan@163.com>
尊敬的杨惠兰女士，您的会员资格即将到期，请及时续费。
---
(week04)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$
```

- 6.在 (pdb) 提示符下练习使用相关命令

```
MINGW64/c/Users/hp/repo/v X + v
(week04)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$ python -m pdb main.py
> c:\users\hp\repo\week04\main.py(1)<module>()
-> def read_contacts(file_path):
(Pdb) l
1  -> def read_contacts(file_path):
2      try:
3          contacts = []
4          with open(file_path, "r", encoding="utf-8") as file:
5              for line in file:
6                  name, gender, email = line.strip().split()
7                  contacts.append((name, gender, email))
8          return contacts
9      except FileNotFoundError:
10         print(f"错误：文件 {file_path} 未找到!")
11         return []
(Pdb) n
> c:\users\hp\repo\week04\main.py(17)<module>()
-> def generate_emails(contacts):
(Pdb) l
12         except Exception as e:
13             print(f"错误：发生了一个未知错误：{e}")
14             return []
15
16
17 -> def generate_emails(contacts):
18     emails = []
19     for name, gender, email in contacts:
20         title = "先生" if gender == "男" else "女士"
```

(1) l (显示代码)

显示当前正在调试的代码段。-> 指向的是将要执行的行。

l. : 显示将要执行的行的前后五行代码内容

l 20,25 : 显示第20行-25行代码

## (2) n (执行当前行)

执行当前行，然后停在接下来要执行的行。如果当前行调用了函数，它不会进入函数内部，而是直接执行完该函数调用并停在调用之后的行。

## (3) p (打印表达式)

计算并打印指定表达式的值，可以使用它来查看变量的值、表达式的计算结果等。

```
MINGW64: c:/Users/hp/repo/v x + v
38
39 if __name__ == "__main__":
40     contacts = read_contacts("contacts.txt")
41     sorted_contacts = sort_contacts(contacts)
42     emails = generate_emails(sorted_contacts)
(Pdb) p sort_contacts
*** NameError: name 'sort_contacts' is not defined
(Pdb) n
> c:\users\hp\repo\week04\main.py(30)<module>()
-> def write_emails(emails, output_file):
(Pdb) p sort_contacts
<function sort_contacts at 0x0000016F91BDF600>
(Pdb) l .
25
26 def sort_contacts(contacts):
27     return sorted(contacts, key=lambda x: (x[2].split("@")[1], x[2].split("@")[0]))
28
29
30 -> def write_emails(emails, output_file):
31     try:
32         with open(output_file, "w", encoding="utf-8") as file:
33             for email in emails:
34                 file.write(email + "\n")
35     except Exception as e:
(Pdb) n
> c:\users\hp\repo\week04\main.py(39)<module>()
-> if __name__ == "__main__":
(Pdb) p write_emails
<function write_emails at 0x0000016F91BDF7E0>
(Pdb) █
```

## (4) s (步入调用)

执行当前行，如果当前行是一个函数调用，它会进入该函数内部，停在函数的第一行，以便可以逐行调试函数内部的代码。

```
MINGW64: c:/Users/hp/repo/v x + v
38
39 -> if __name__ == "__main__":
40     contacts = read_contacts("contacts.txt")
41     sorted_contacts = sort_contacts(contacts)
42     emails = generate_emails(sorted_contacts)
43     write_emails(emails, "emails.txt")
44     print("邮件内容已成功写入 emails.txt 文件。")
(Pdb) s
> c:\users\hp\repo\week04\main.py(40)<module>()
-> contacts = read_contacts("contacts.txt")
(Pdb) s
--Call--
> c:\users\hp\repo\week04\main.py(1)read_contacts()
-> def read_contacts(file_path):
(Pdb) s
> c:\users\hp\repo\week04\main.py(2)read_contacts()
-> try:
(Pdb) l .
1 def read_contacts(file_path):
2 -> try:
3     contacts = []
4     with open(file_path, "r", encoding="utf-8") as file:
5         for line in file:
6             name, gender, email = line.strip().split()
7             contacts.append((name, gender, email))
8     return contacts
9 except FileNotFoundError:
10     print(f"错误: 文件 {file_path} 未找到!")
11     return []
(Pdb) █
```

### (5) pp (美观打印)

和 p 命令类似，都是用于打印表达式的值，但 pp 会以更美观、易读的格式输出复杂的数据结构。

### (6) c (继续执行)

继续执行代码，直到遇到下一个断点或者程序结束。

```
MINGW64/c/Users/hp/repo/v x + v
(Pdb) n
> c:\users\hp\repo\week04\main.py(4)read_contacts()
-> with open(file_path, "r", encoding="utf-8") as file:
(Pdb) n
> c:\users\hp\repo\week04\main.py(8)read_contacts()
-> return contacts
(Pdb) n
--Return--
> c:\users\hp\repo\week04\main.py(8)read_contacts()->[('燕小六', '男', 'yanxiaoliu@126.com'), ('杨蕙兰', '女', 'yanghuilan@163.com'), ('莫小宝', '男', 'moxiaobao@126.com'), ('慕容嫣', '女', 'murongyan@163.com'), ('邢育森', '男', 'xingyusen@126.com'), ('凌腾云', '女', 'lingtengyun@163.com')]
-> return contacts
(Pdb) pp read_contacts
<function read_contacts at 0x0000016F918DF9C0>
(Pdb) pp contacts
[('燕小六', '男', 'yanxiaoliu@126.com'),
 ('杨蕙兰', '女', 'yanghuilan@163.com'),
 ('莫小宝', '男', 'moxiaobao@126.com'),
 ('慕容嫣', '女', 'murongyan@163.com'),
 ('邢育森', '男', 'xingyusen@126.com'),
 ('凌腾云', '女', 'lingtengyun@163.com')]
(Pdb) c
邮件内容已成功写入 emails.txt 文件。
The program finished and will be restarted
> c:\users\hp\repo\week04\main.py(1)<module>()
-> def read_contacts(file_path):
(Pdb) q
(week04)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$
```

### (7) q (退出 pdb 调试器)

- 7.安装wat-inspector检查 (inspect) 各种对象

```
(base)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$ cat environment.yml
name: week04
channels:
  - conda-forge
dependencies:
  - python=3.12
  - wat-inspector

hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$ conda env update
Channels:
  - conda-forge
  - https://repo.anaconda.com/pkgs/main
  - https://repo.anaconda.com/pkgs/r
  - https://repo.anaconda.com/pkgs/msys2
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done
#
# To activate this environment, use
#
#     $ conda activate week04
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(week04)
hp@LAPTOP-L5E04S06 MINGW64 ~/repo/week04 (main)
$
```

## • 8.学习理解以下 Python 基本概念

### ❖ Python 语法保留字 (reserved key words)

保留字是 Python 语言中具有特殊意义的单词，不能将它们用作变量名、函数名或其他标识符。代码中标红的

'False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield'

### ❖ 语句 (statement) 和表达式 (expression)

语句是 Python 程序中的一个完整指令，它执行某个操作，但不一定有返回值。常见的语句包括赋值语句、if 语句、for 循环等。

大语句里可以嵌套子语句 语句里可以包含表达式

表达式是由值、变量和运算符组成的组合。表达式不能包含语句，表达式里可以嵌套表达式。vscode中白色（变量名）、黄色（字符串）、绿色（明确的函数）、橙色（形参）

### ❖ 缩进 (indent)

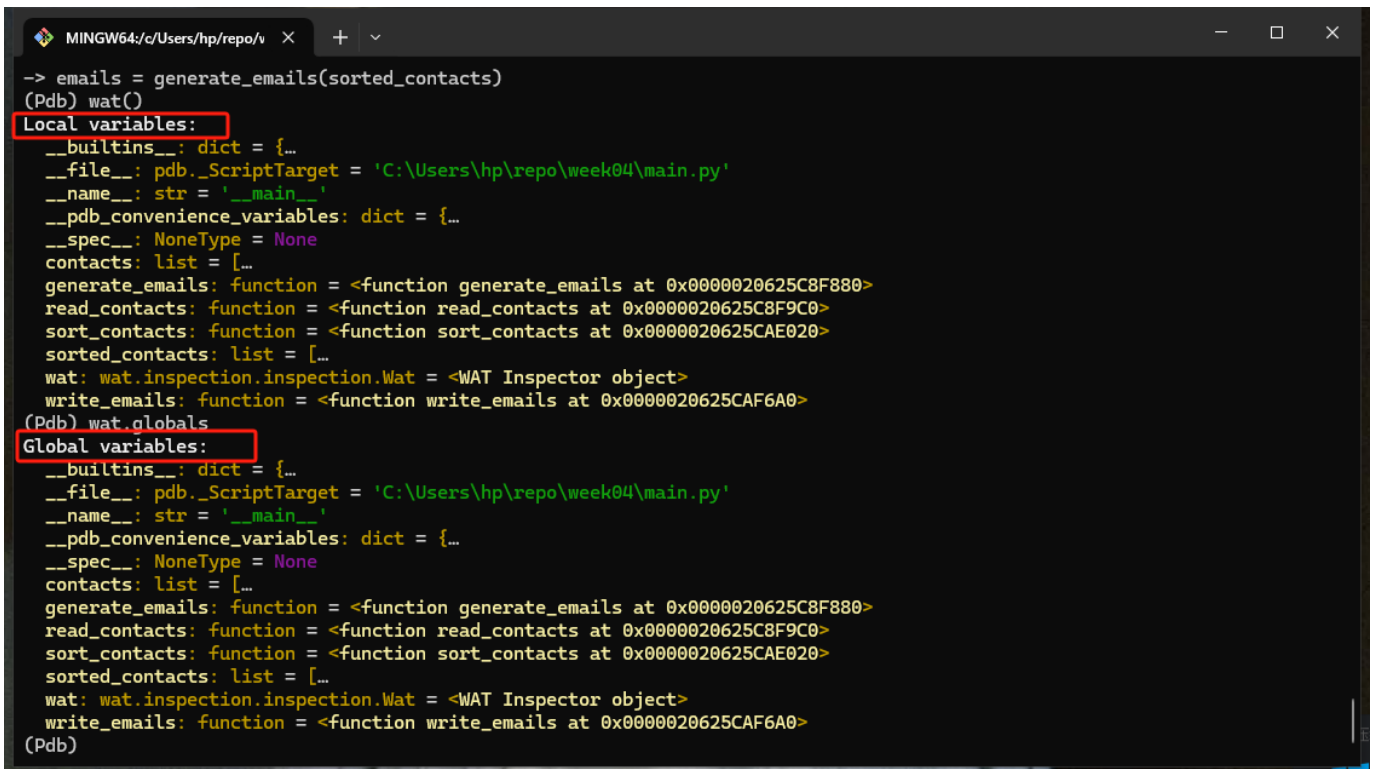
在 Python 中，缩进用于表示代码块的层次结构。

### ❖ 局部变量 (local variable)、全局变量 (global variable)、LEGB 规则

局部变量：定义在函数内部的变量，只能在该函数内部访问。

全局变量：定义在函数外部的变量，可以在整个程序中访问。

LEGB 规则：Python 查找变量的顺序，即 Local（局部作用域）、Enclosing（嵌套作用域）、Global（全局作用域）、Built-in（内置作用域）。当你引用一个变量时，Python 会按照这个顺序依次查找。



```
MINGW64~/c/Users/hp/repo/v x + v
-> emails = generate_emails(sorted_contacts)
(Pdb) wat()
Local variables:
__builtins__: dict = {...}
__file__: pdb._ScriptTarget = 'C:\Users\hp\repo\week04\main.py'
__name__: str = '__main__'
__pdb_convenience_variables__: dict = {...}
__spec__: NoneType = None
contacts: list = [...]
generate_emails: function = <function generate_emails at 0x0000020625C8F880>
read_contacts: function = <function read_contacts at 0x0000020625C8F9C0>
sort_contacts: function = <function sort_contacts at 0x0000020625CAE020>
sorted_contacts: list = [...]
wat: wat.inspection.inspection.Wat = <WAT Inspector object>
write_emails: function = <function write_emails at 0x0000020625CAF6A0>
(Pdb) wat.globals
Global variables:
__builtins__: dict = {...}
__file__: pdb._ScriptTarget = 'C:\Users\hp\repo\week04\main.py'
__name__: str = '__main__'
__pdb_convenience_variables__: dict = {...}
__spec__: NoneType = None
contacts: list = [...]
generate_emails: function = <function generate_emails at 0x0000020625C8F880>
read_contacts: function = <function read_contacts at 0x0000020625C8F9C0>
sort_contacts: function = <function sort_contacts at 0x0000020625CAE020>
sorted_contacts: list = [...]
wat: wat.inspection.inspection.Wat = <WAT Inspector object>
write_emails: function = <function write_emails at 0x0000020625CAF6A0>
(Pdb)
```

❖ 函数 (function) 的定义 (define) 和调用 (call)

调用函数：通过函数名和传递相应的参数来调用函数。

❖ 字面值 (literal) (字符串 (str)、整数 (int)、列表 (list)、字典 (dict)、元组 (tuple))

❖ 运算符 (operator)

“等于” ==

赋值语句 =

. (名称访问运算符) () (调用运算符)

❖ 形参 (parameter)、实参 (argument)、返回值 (return value)

形参：在函数定义时，位于函数名后面括号内的变量，用于接收传递给函数的值。

实参：在函数调用时，传递给函数的具体值。

返回值：函数执行完毕后返回给调用者的值，使用 return 语句返回。没有返回值：none

❖ 对象 (object)、类型 (type)、属性 (attribute)、方法 (method)

对象：Python 中一切皆对象，每个对象都有自己的类型和属性。

类型：决定了对象可以执行的操作和具有的属性。可以使用 type() 函数查看对象的类型。

属性：对象所拥有的变量，用于存储对象的状态。

方法：对象可以执行的操作。