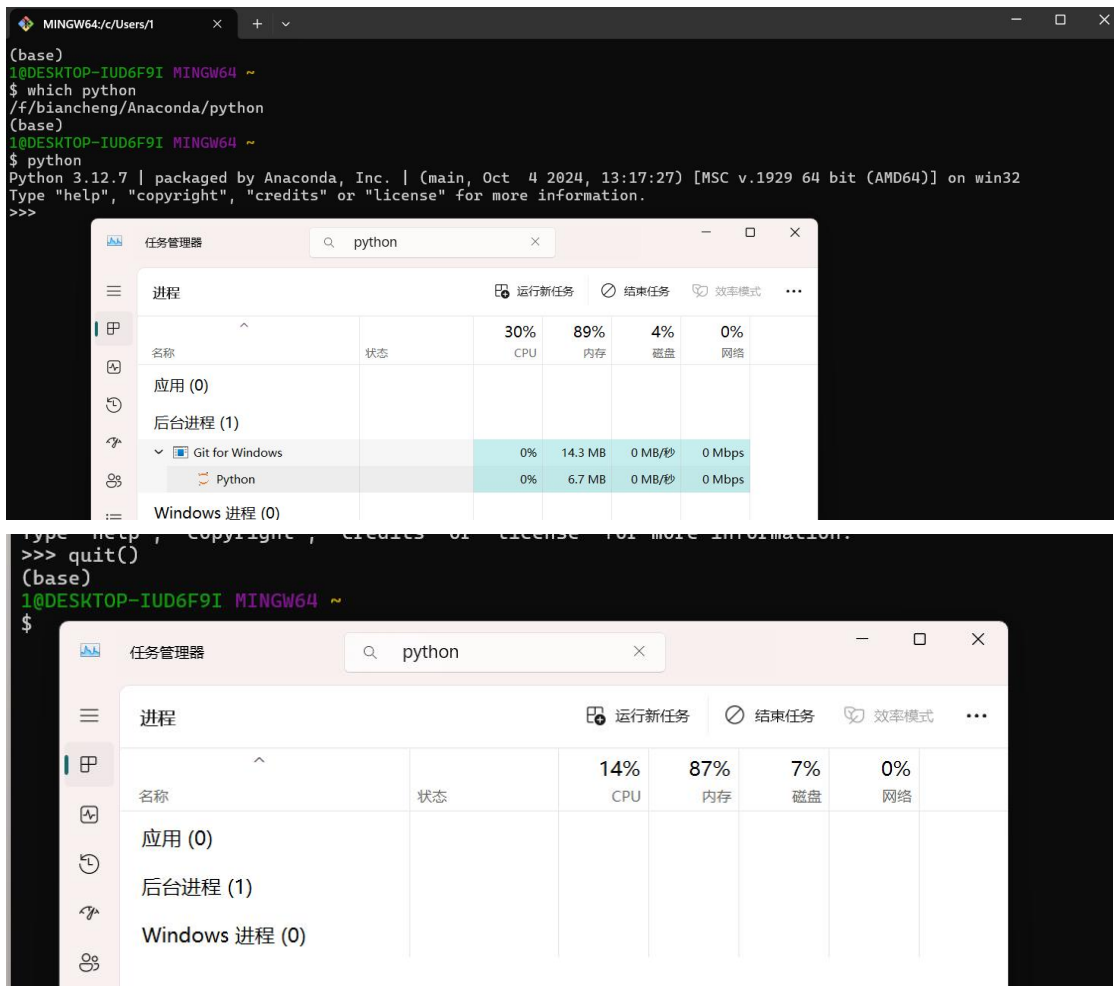
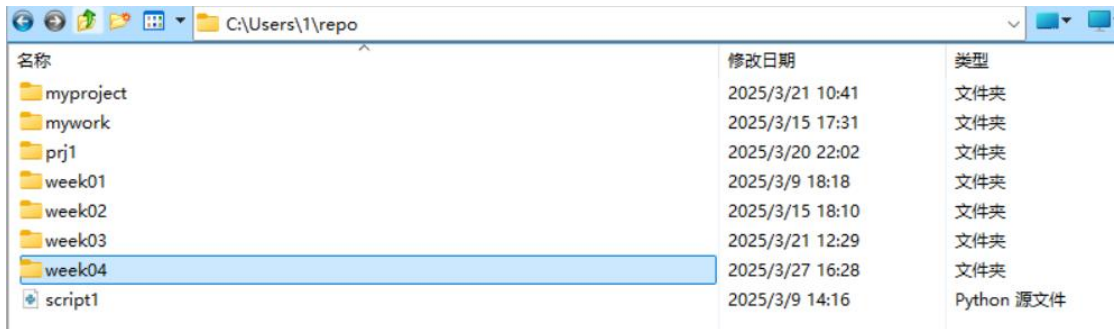


# 金融编程与计算-学习报告-week04



1.Fork 第 04 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

```

$ ls -l ../myproject
total 196846
-rw-r--r-- 1 1 197609      87  3月 21 10:27 environment.yml
-rw-r--r-- 1 1 197609 201568176  3月 21 10:50 EPA_SmartLocationDatabase_V3_Jan_2021_Final.csv
-rw-r--r-- 1 1 197609      713  3月 21 11:17 main.py
(base)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ cat ../myproject/environment.yml
name: myproject
channels:
  - conda-forge
dependencies:
  - python=3.12
  - pandas(base)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$

```

从 myproject 文件夹中将 environment.yml 复制到 week04

```

$ cp ../myproject/environment.yml ./
(base)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ ls -l
total 25
-rw-r--r-- 1 1 197609      87  3月 27 16:39 environment.yml
-rw-r--r-- 1 1 197609 18805  3月 27 16:28 LICENSE
-rw-r--r-- 1 1 197609  2239  3月 27 16:28 README.md
(base)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$

```

```

! environment.yml U X
! environment.yml
1  name: week04
2  channels:
3    - conda-forge
4  dependencies:
5    - python=3.12
6

```

```

$ ls -l
total 25
-rw-r--r-- 1 1 197609      76  3月 27 16:43 environment.yml
-rw-r--r-- 1 1 197609 18805  3月 27 16:28 LICENSE
-rw-r--r-- 1 1 197609  2239  3月 27 16:28 README.md
(base)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ cat environment.yml
name: week04
channels:
  - conda-forge
dependencies:
  - python=3.12
  (base)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$

```

```

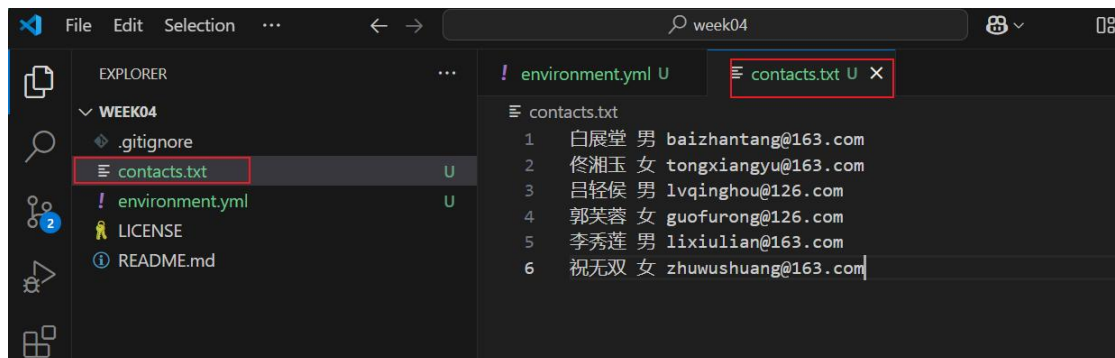
$ conda env create
Retrieving notices: done
Channels:
- conda-forge
- defaults
- https://repo.anaconda.com/pkgs/main
- https://repo.anaconda.com/pkgs/r
- https://repo.anaconda.com/pkgs/msys2
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

Downloading and Extracting Packages:

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate week04
#
# To deactivate an active environment, use
#
#     $ conda deactivate

```

3.新建一个 contacts.txt 文件，每行写一个联系人，每个联系人都包含姓名、性别、邮箱三个字段，用空格分隔



```

$ ls -l
total 26
-rw-r--r-- 1 1 197609 202 3月 27 16:53 contacts.txt
-rw-r--r-- 1 1 197609 76 3月 27 16:43 environment.yml
-rw-r--r-- 1 1 197609 18805 3月 27 16:28 LICENSE
-rw-r--r-- 1 1 197609 2239 3月 27 16:28 README.md
(base)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ cat contacts.txt
白展堂 男 baizhantang@163.com
佟湘玉 女 tongxiangyu@163.com
吕轻侯 男 lvqinghou@126.com
郭芙蓉 女 guofurong@126.com
李秀莲 男 lixiulian@163.com
祝无双 女 zhuwushuang@163.com
(base)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$

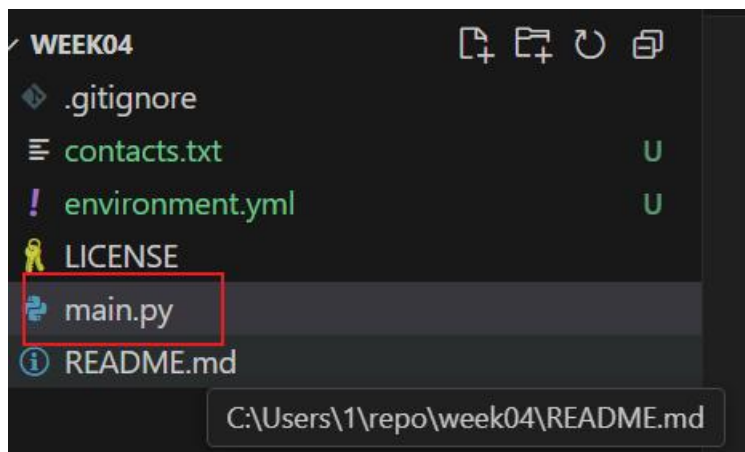
```

```

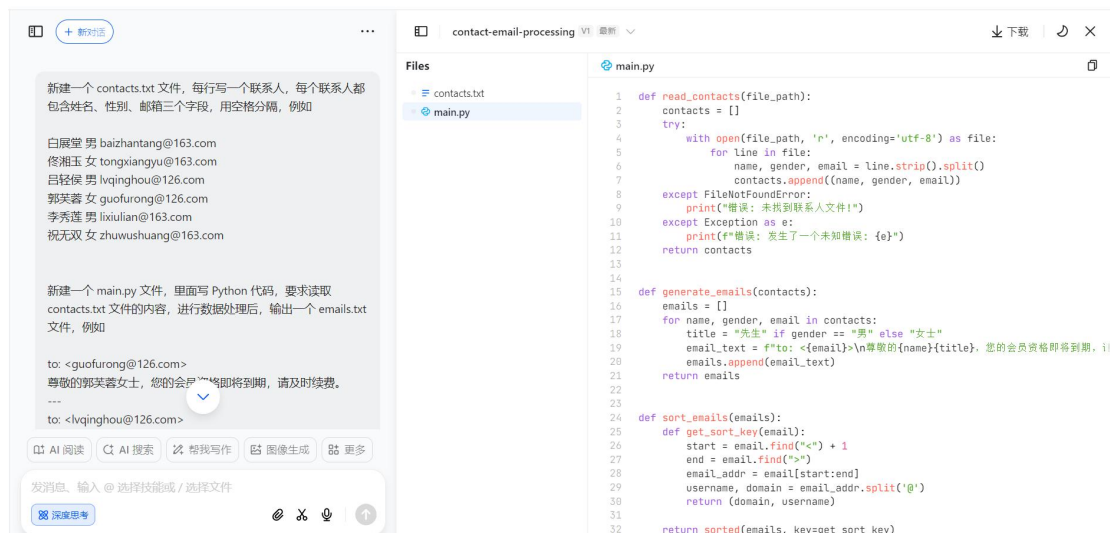
$ cat contacts.txt environment.yml
白展堂 男 baizhantang@163.com
佟湘玉 女 tongxiangyu@163.com
吕轻侯 男 lvqinghou@126.com
郭芙蓉 女 guofurong@126.com
李秀莲 男 lixiulian@163.com
祝无双 女 zhuwushuang@163.com name: week04
channels:
  - conda-forge
dependencies:
  - python=3.12
  (base)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ cat contacts.txt environment.yml
白展堂 男 baizhantang@163.com
佟湘玉 女 tongxiangyu@163.com
吕轻侯 男 lvqinghou@126.com
郭芙蓉 女 guofurong@126.com
李秀莲 男 lixiulian@163.com
祝无双 女 zhuwushuang@163.com
name: week04
channels:
  - conda-forge
dependencies:
  - python=3.12
  (base)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$

```

4.新建一个 main.py 文件，里面写 Python 代码，要求读取 contacts.txt 文件的内容，进行数据处理后，输出一个 emails.txt 文件

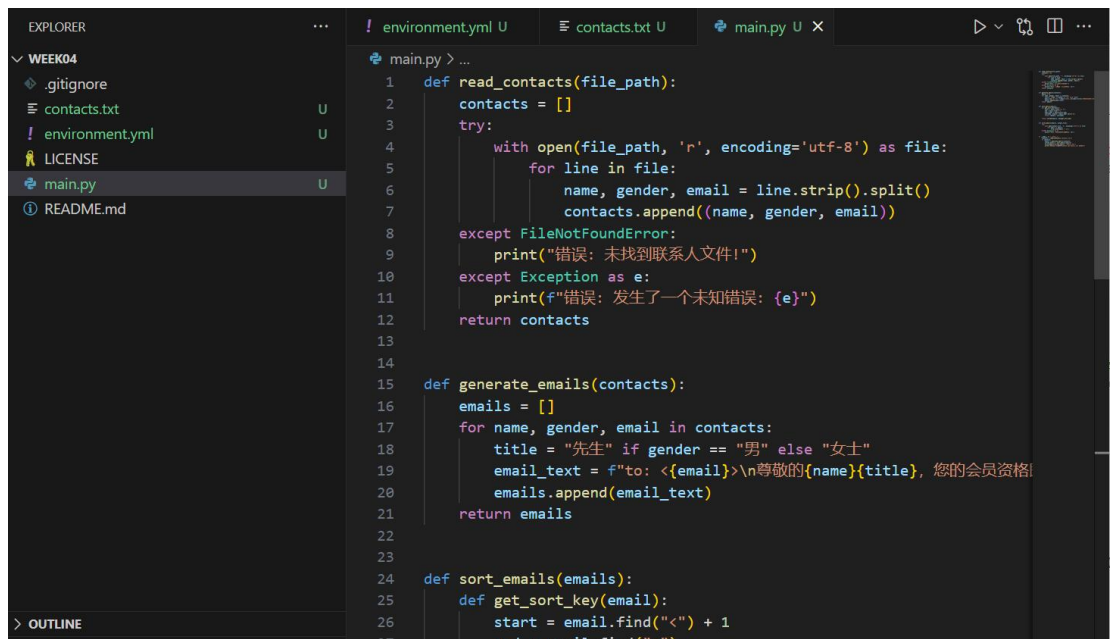


5.可以将以上“任务要求”的文本，复制粘贴到大模型（比如豆包、DeepSeek）里，请 AI 来帮助编写程序初稿



6.AI 回复的只是静态代码，而且可能含有错误，所以我们必须在 Conda 环境里运行代码，逐行调试，检查每一行代码的运行都符合我们的期望

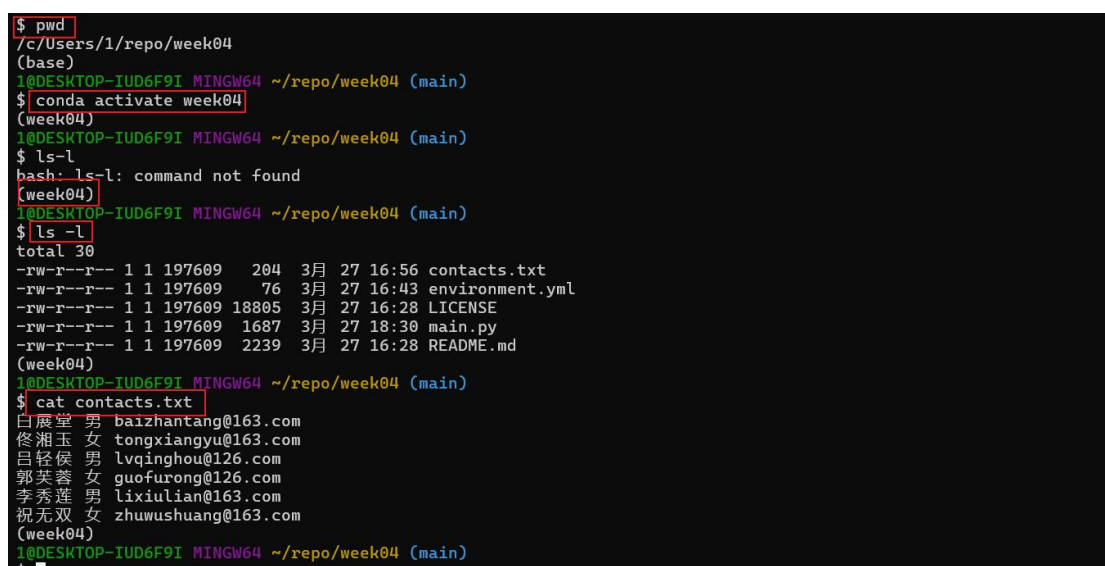
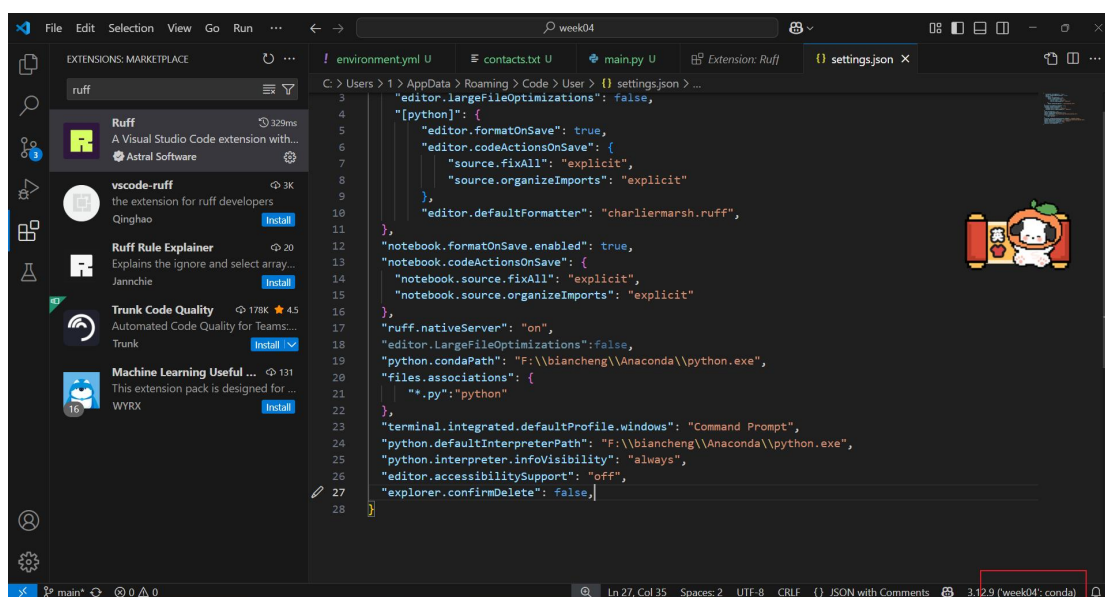
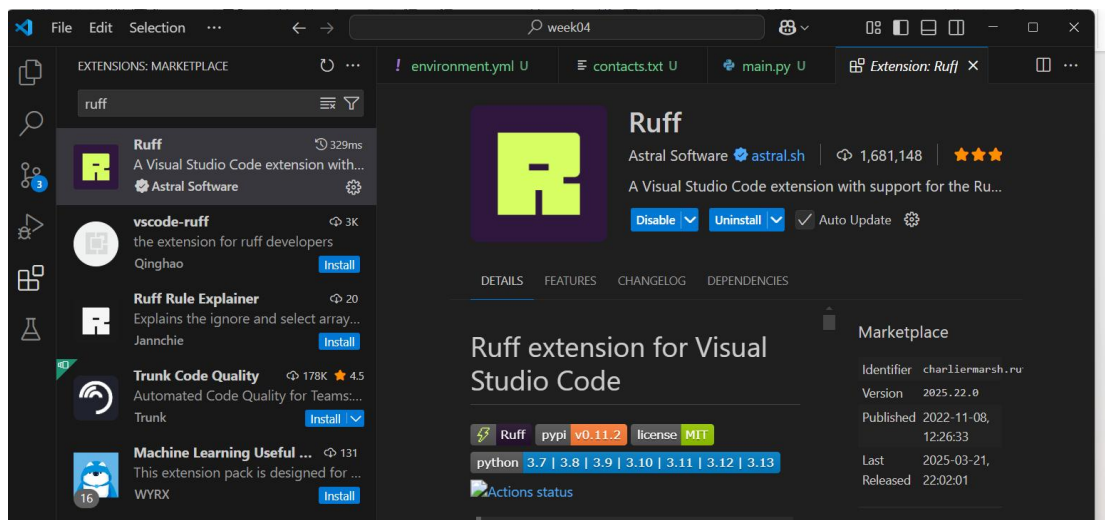
①将大模型提供的代码复制粘贴进 main.py 文件，记得保存



②在 VS Code 扩展商店里安装 Python 扩展，使得在编写.py 文件时能够显示和选择 Python 解释器（需要绕过防火墙）

③在 VS Code 扩展商店里安装 Ruff 扩展，按照文档配置 Ruff，实现在保存.py 文件时能够自动规范化 Python 代码





④运行 `python main.py` 命令(作用是启动 Python 解释器, 执行 `main.py` 里的代码直至结束 (EOF) 或报错 (Exception)), 检查运行结果是否符合预期

```

1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ cat main.py
def read_contacts(file_path):
    contacts = []
    try:
        with open(file_path, "r", encoding="utf-8") as file:
            for line in file:
                name, gender, email = line.strip().split()
                contacts.append((name, gender, email))
    except FileNotFoundError:
        print("错误: 未找到联系人文件!")
    except Exception as e:
        print(f"错误: 发生了一个未知错误: {e}")
    return contacts

def generate_emails(contacts):
    emails = []
    for name, gender, email in contacts:
        title = "先生" if gender == "男" else "女士"
        email_text = f"to: <{email}>\n尊敬的{name}{title}, 您的会员资格即将到期, 请及时续费。\\n---"
        emails.append(email_text)
    return emails

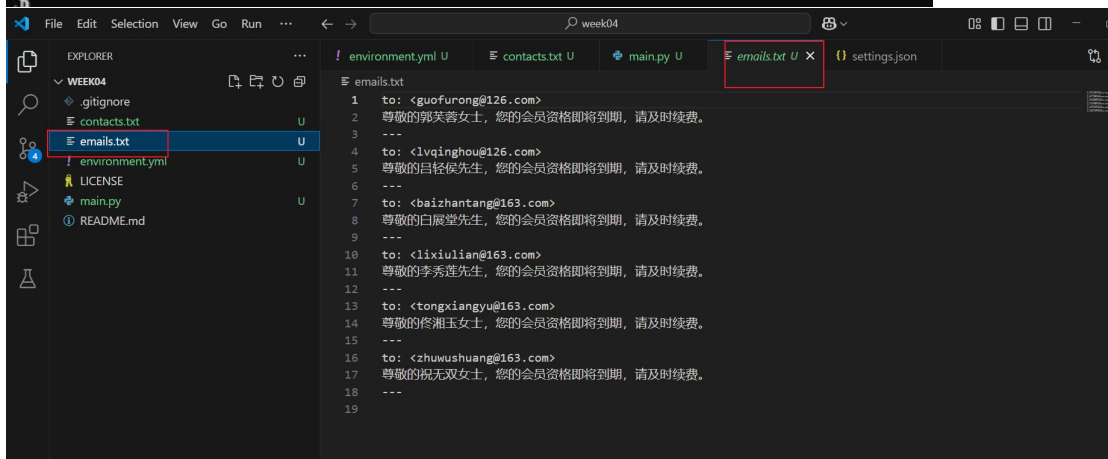
def sort_emails(emails):
    def get_sort_key(email):
        start = email.find("<") + 1

```

```

1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ python main.py
邮件提醒已生成并排序输出到 emails.txt 文件。
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$

```



```

$ ls -l
total 31
-rw-r--r-- 1 1 197609 204 3月 27 16:56 contacts.txt
-rw-r--r-- 1 1 197609 666 3月 27 19:34 emails.txt
-rw-r--r-- 1 1 197609 76 3月 27 16:43 environment.yml
-rw-r--r-- 1 1 197609 18805 3月 27 16:28 LICENSE
-rw-r--r-- 1 1 197609 1687 3月 27 18:30 main.py
-rw-r--r-- 1 1 197609 2239 3月 27 16:28 README.md
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ cat emails.txt
to: <guofurong@126.com>
尊敬的郭芙蓉女士, 您的会员资格即将到期, 请及时续费。
---
to: <lvqinghou@126.com>
尊敬的吕轻侯先生, 您的会员资格即将到期, 请及时续费。
---
to: <baizhantang@163.com>
尊敬的白展堂先生, 您的会员资格即将到期, 请及时续费。
---
to: <lixuilian@163.com>
尊敬的李秀莲先生, 您的会员资格即将到期, 请及时续费。
---
to: <tongxiangyu@163.com>
尊敬的佟湘玉女士, 您的会员资格即将到期, 请及时续费。
---
to: <zhuwushuang@163.com>
尊敬的祝无双女士, 您的会员资格即将到期, 请及时续费。

```

⑤运行 `python -m pdb main.py` 命令(作用是以调试模式 (debug mode) 启动 Python 解释器, 准备执行 main.py 里的代码)

```
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ ls -l
total 31
-rw-r--r-- 1 1 197609 204 3月 27 16:56 contacts.txt
-rw-r--r-- 1 1 197609 666 3月 27 19:34 emails.txt
-rw-r--r-- 1 1 197609 76 3月 27 16:43 environment.yml
-rw-r--r-- 1 1 197609 18805 3月 27 16:28 LICENSE
-rw-r--r-- 1 1 197609 1687 3月 27 18:30 main.py
-rw-r--r-- 1 1 197609 2239 3月 27 16:28 README.md
(week04)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ rm emails.txt
(week04)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ ls -l
total 30
-rw-r--r-- 1 1 197609 204 3月 27 16:56 contacts.txt
-rw-r--r-- 1 1 197609 76 3月 27 16:43 environment.yml
-rw-r--r-- 1 1 197609 18805 3月 27 16:28 LICENSE
-rw-r--r-- 1 1 197609 1687 3月 27 18:30 main.py
-rw-r--r-- 1 1 197609 2239 3月 27 16:28 README.md
(week04)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$
```

```
(week04)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ python -m pdb main.py
> c:\users\1\repo\week04\main.py(1)<module>()
-> def read_contacts(file_path):
(Pdb)
```

⑥在 (pdb) 提示符下练习使用 l(显示代码)、n(执行当前行)、p(打印表达式)、s(步入调用)、pp(美观打印)、c(继续执行) 等命令 (参考文档)

```
(Pdb) l
1 -> def read_contacts(file_path):
2     contacts = []
3     try:
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     except FileNotFoundError:
9         print("错误: 未找到联系人文件!")
10    except Exception as e:
11        print(f"错误: 发生了一个未知错误: {e}")
(Pdb) n
> c:\users\1\repo\week04\main.py(15)<module>()
-> def generate_emails(contacts):
(Pdb) l
10    except Exception as e:
11        print(f"错误: 发生了一个未知错误: {e}")
12    return contacts
13
14
15 -> def generate_emails(contacts):
16     emails = []
17     for name, gender, email in contacts:
18         title = "先生" if gender == "男" else "女士"
19         email_text = f"to: <{email}>\n尊敬的{name}{title}, 您的会员资格即将到期, 请及时续费。 \n---"
20         emails.append(email_text)
(Pdb)
```



```

(Pdb) ll
1 def read_contacts(file_path):
2     contacts = []
3     try:
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     except FileNotFoundError:
9         print("错误: 未找到联系人文件!")
10    except Exception as e:
11        print(f"错误: 发生了一个未知错误: {e}")
12    return contacts
13
14
15 -> def generate_emails(contacts):
16     emails = []
17     for name, gender, email in contacts:
18         title = "先生" if gender == "男" else "女士"
19         email_text = f"to: <{email}>\n尊敬的{name}{title}, 您的会员资格即将到期, 请及时续费。\\n---"
20         emails.append(email_text)
21     return emails
22
23
24     def sort_emails(emails):
25         def get_sort_key(email):
26             start = email.find("<") + 1
27             end = email.find(">")

```

```

(Pdb) p contacts
*** NameError: name 'contacts' is not defined
(Pdb) p read_contacts
<function read_contacts at 0x00000210216B3240>
(Pdb)

```

```

def sort_emails(emails):
(Pdb) ll
19         email_text = f"to: <{email}>\n尊敬的{name}{title}, 您的会员资格即将到期, 请及时续费。\\n---"
20         emails.append(email_text)
21     return emails
22
23
24 -> def sort_emails(emails):
25     def get_sort_key(email):
26         start = email.find("<") + 1
27         end = email.find(">")
28         email_addr = email[start:end]
29         username, domain = email_addr.split("@")
(Pdb) l 1,5
1     def read_contacts(file_path):
2         contacts = []
3         try:
4             with open(file_path, "r", encoding="utf-8") as file:
5                 for line in file:

```

"l": 上5行下5行

"l 1,5": 第1行到第5行

```

(Pdb) p contacts
[('白展堂', '男', 'baizhantang@163.com'), ('佟湘玉', '女', 'tongxiangyu@163.com'), ('吕轻侯', '男', 'lvqinghou@126.com'), ('郭芙蓉', '女', 'guofurong@126.com'), ('李秀莲', '男', 'lixliulan@163.com'), ('祝无双', '女', 'zhuwushuang@163.com')]
(Pdb) pp contacts
[('白展堂', '男', 'baizhantang@163.com'), ('佟湘玉', '女', 'tongxiangyu@163.com'), ('吕轻侯', '男', 'lvqinghou@126.com'), ('郭芙蓉', '女', 'guofurong@126.com'), ('李秀莲', '男', 'lixliulan@163.com'), ('祝无双', '女', 'zhuwushuang@163.com')]
(Pdb)

```

```

(Pdb) p type(contacts)
<class 'list'>
(Pdb) p len(contacts)
6
(Pdb) q 退出
(week04)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$

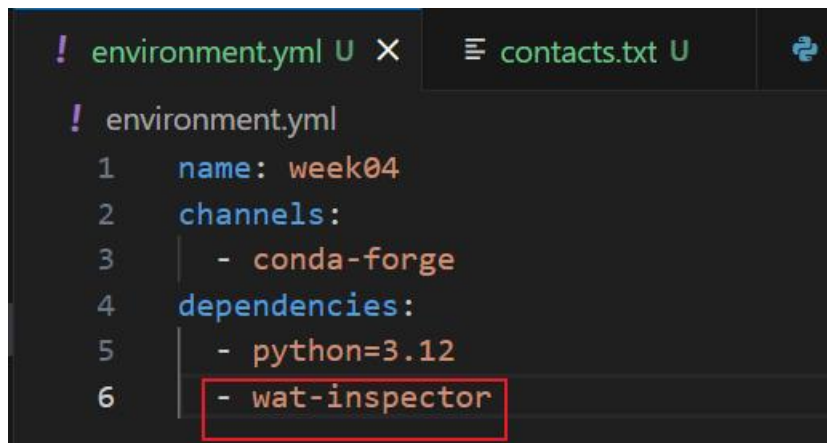
```

```

(Pdb) c      一股脑儿走到底
邮件提醒已生成并排序输出到 emails.txt 文件。
The program finished and will be restarted
> c:\users\1\repo\week04\main.py(1)<module>()
-> def read_contacts(file_path):
(Pdb) q
(week04)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$

```

⑦在调试过程中，利用 wat-inspector (第三方软件包，需要安装) 检查(inspect)各种对象 (参考文档)



```

! environment.yml U X  contacts.txt U
! environment.yml
1  name: week04
2  channels:
3    - conda-forge
4  dependencies:
5    - python=3.12
6    - wat-inspector

```

```

$ ls -l
total 31
-rw-r--r-- 1 1 197609 204 3月 27 16:56 contacts.txt
-rw-r--r-- 1 1 197609 666 3月 27 20:46 emails.txt
-rw-r--r-- 1 1 197609 91 3月 27 20:52 environment.yml
-rw-r--r-- 1 1 197609 18805 3月 27 16:28 LICENSE
-rw-r--r-- 1 1 197609 1687 3月 27 18:30 main.py
-rw-r--r-- 1 1 197609 2239 3月 27 16:28 README.md
(week04)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ cat environment.yml
name: week04
channels:
  - conda-forge
dependencies:
  - python=3.12
  - wat-inspector(week04)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$

```

```
$ conda env update
Channels:
- conda-forge
- defaults
- https://repo.anaconda.com/pkg/main
- https://repo.anaconda.com/pkg/r
- https://repo.anaconda.com/pkg/msys2
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

Downloading and Extracting Packages:

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate week04
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(week04)
```

⑧在调试过程中，观察代码逐步运行的效果，学习理解以下 Python 基本概念

(1) Python 语法保留字 (reserved key words)

```
← → week04
! environment.yml U
contacts.txt U
main.py U X
emails.txt U
settings.json

main.py > read_contacts
1 def read_contacts(file_path):
2     contacts = []
3     try:
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     except FileNotFoundError:
9         print("错误: 未找到联系人文件!")
10    except Exception as e:
11        print(f"错误: 发生了一个未知错误: {e}")
12    return contacts
13
14
15 def generate_emails(contacts):
16     emails = []
17     for name, gender, email in contacts:
18         title = "先生" if gender == "男" else "女士"
19         email_text = f"to: <{email}>\n尊敬的{name}{title}, 您的会员资格即将到期, 请及时续费.\n---"
20         emails.append(email_text)
21     return emails
22
23
24 def sort_emails(emails):
25     def get_sort_key(email):
26         start = email.find("<") + 1
27         end = email.find(">")
28         email_addr = email[start:end]
29         username, domain = email_addr.split("@")
30         return (domain, username)

$ python
Python 3.12.9 | packaged by conda-forge | (main, Mar  4 2025, 22:37:18) [MSC v.1943 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> name = 'DK'
>>> print(name)
DK
>>> def = 'DK'    保留字
File "<stdin>", line 1
    def = 'DK'
      ^
SyntaxError: invalid syntax
>>> which = 'DK'
>>> print(which)
DK
>>> while = 'DK'    保留字
File "<stdin>", line 1
    while = 'DK'
      ^
IndentationError: unexpected indent
>>>
```

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

语句里能够包含表达式,但是表达式里是不能够包含语句的,表达式内也可嵌套表达式。

## (3) 缩进 (indent)

## (4) 局部变量 (local variable) vs. 全局变量 (global variable)



```

1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ python -m pdb main.py
> c:\users\1\repo\week04\main.py(1)<module>()
-> def read_contacts(file_path):
(Pdb) l
1  -> def read_contacts(file_path):
2      contacts = []
3      try:
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      except FileNotFoundError:
9          print("错误: 未找到联系人文件!")
10     except Exception as e:
11         print(f"错误: 发生了一个未知错误: {e}")
(Pdb) wat
*** NameError: name 'wat' is not defined
(Pdb) import wat
(Pdb) wat
Try wat / object or wat.modifiers / object to inspect an object. Modifiers are:
.short or .s to hide attributes (variables and methods)
.dunder to print dunder attributes
.code to print source code of a function, method or class
.long to print non-abbreviated values and documentation
.nodocs to hide documentation for functions and classes
.caller to show how and where the inspection was called
.all to include all information
.ret to return the inspected object
.str to return the output string instead of printing

```

```

(Pdb) wat()
Local variables:
__builtins__: dict = {...}
__file__: pdb._ScriptTarget = 'C:\Users\1\repo\week04\main.py'
__name__: str = '__main__'
__pdb_convenience_variables__: dict = {...}
__spec__: NoneType = None
wat: wat.inspection.inspection.Wat = <WAT Inspector object>
(Pdb) p wat
Try wat / object or wat.modifiers / object to inspect an object. Modifiers are:
.short or .s to hide attributes (variables and methods)
.dunder to print dunder attributes
.code to print source code of a function, method or class
.long to print non-abbreviated values and documentation
.nodocs to hide documentation for functions and classes
.caller to show how and where the inspection was called
.all to include all information
.ret to return the inspected object
.str to return the output string instead of printing
.gray to disable colorful output in the console
.color to enforce colorful outputs in the console
Call wat.locals or wat() to inspect local variables.
Call wat.globals to inspect global variables.
(Pdb)

```

```

(Pdb) wat
Try wat / object or wat.modifiers / object to inspect an object. Modifiers are:
.short or .s to hide attributes (variables and methods)
.dunder to print dunder attributes
.code to print source code of a function, method or class
.long to print non-abbreviated values and documentation
.nodocs to hide documentation for functions and classes
.caller to show how and where the inspection was called
.all to include all information
.ret to return the inspected object
.str to return the output string instead of printing
.gray to disable colorful output in the console
.color to enforce colorful outputs in the console
Call wat.locals or wat() to inspect local variables.
Call wat.globals to inspect global variables.

```

```

(Pdb) wat()
Local variables:
  __builtins__: dict = {...
  __file__: pdb._ScriptTarget = 'C:\Users\1\repo\week04\main.py'
  __name__: str = '__main__'
  __pdb_convenience_variables: dict = {...
  __spec__: NoneType = None
  wat: wat.inspection.inspection.Wat = <WAT Inspector object>
(Pdb) p __name__
'__main__'
(Pdb) q
(week04)
1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$

```

```

1@DESKTOP-IUD6F9I MINGW64 ~/repo/week04 (main)
$ python -m pdb main.py
> c:\users\1\repo\week04\main.py(1)<module>()
-> def read_contacts(file_path):
(Pdb) l
1  -> def read_contacts(file_path):
2      contacts = []
3      try:
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      except FileNotFoundError:
9          print("错误: 未找到联系人文件!")
10     except Exception as e:
11         print(f"错误: 发生了一个未知错误: {e}")
(Pdb) import wat
(Pdb) wat()
Local variables:
  __builtins__: dict = {...
  __file__: pdb._ScriptTarget = 'C:\Users\1\repo\week04\main.py'
  __name__: str = '__main__'
  __pdb_convenience_variables: dict = {...
  __spec__: NoneType = None
  wat: wat.inspection.inspection.Wat = <WAT Inspector object>
(Pdb) p __file__
'C:\Users\1\repo\week04\main.py'
(Pdb) p __spec__
None
(Pdb)

```

```

(Pdb) p __file__
'C:\\Users\\1\\repo\\week04\\main.py'
(Pdb) p __spec__
None
(Pdb) p while 保留字
*** SyntaxError: invalid syntax
(Pdb) p which
*** NameError: name 'which' is not defined
(Pdb)

```

```

(Pdb) p print
<built-in function print>
(Pdb) p type
<class 'type'>
(Pdb) p list
<class 'list'>
(Pdb)

```

```

(Pdb) n
> c:\users\1\repo\week04\main.py(15)<module>()
-> def generate_emails(contacts):
(Pdb) wat()
Local variables:
__builtins__: dict = {...
__file__: pdb._ScriptTarget = 'C:\Users\1\repo\week04\main.py'
__name__: str = '__main__'
__pdb_convenience_variables__: dict = {...
__spec__: NoneType = None
read_contacts: function = <function read_contacts at 0x000001B970C74180>
wat: wat.inspection.inspection.Wat = <WAT Inspector object>
(Pdb) l .
10         except Exception as e:
11             print(f"错误: 发生了一个未知错误: {e}")
12             return contacts
13
14
15     -> def generate_emails(contacts):
16         emails = []
17         for name, gender, email in contacts:
18             title = "先生" if gender == "男" else "女士"
19             email_text = f"to: <{email}>\n尊敬的{name}{title}, 您的会员资格即将到期, 请及时续费.\n---"
20             emails.append(email_text)
(Pdb) n
> c:\users\1\repo\week04\main.py(24)<module>()
-> def sort_emails(emails):
(Pdb) wat()
Local variables:

```

```

(Pdb) n
> c:\users\1\repo\week04\main.py(24)<module>()
-> def sort_emails(emails):
(Pdb) wat()
Local variables:
__builtins__: dict = {...
__file__: pdb._ScriptTarget = 'C:\Users\1\repo\week04\main.py'
__name__: str = '__main__'
__pdb_convenience_variables__: dict = {...
__spec__: NoneType = None
generate_emails: function = <function generate_emails at 0x000001B9710BF060>
read_contacts: function = <function read_contacts at 0x000001B970C74180>
wat: wat.inspection.inspection.Wat = <WAT Inspector object>
(Pdb)

```

```

29         username, domain = email_addr.split('@')
(Pdb) n
> c:\users\1\repo\week04\main.py(35)<module>()
-> def write_emails(emails, output_file):
(Pdb) wat()
Local variables:
__builtins__: dict = {...
__file__: pdb._ScriptTarget = 'C:\Users\1\repo\week04\main.py'
__name__: str = '__main__'
__pdb_convenience_variables__: dict = {...
__spec__: NoneType = None
generate_emails: function = <function generate_emails at 0x000001B9710BF060>
read_contacts: function = <function read_contacts at 0x000001B970C74180>
sort_emails: function = <function sort_emails at 0x000001B9710F1D00>
wat: wat.inspection.inspection.Wat = <WAT Inspector object>
(Pdb)

```

```

(Pdb) wat.globals
Global variables:
__builtins__: dict = {...
__file__: pdb._ScriptTarget = 'C:\Users\1\repo\week04\main.py'
__name__: str = '__main__'
__pdb_convenience_variables__: dict = {...
__spec__: NoneType = None
contacts: list = [...
generate_emails: function = <function generate_emails at 0x000001B9710BF060>
read_contacts: function = <function read_contacts at 0x000001B970C74180>
sort_emails: function = <function sort_emails at 0x000001B9710F1D00>
wat: wat.inspection.inspection.Wat = <WAT Inspector object>
write_emails: function = <function write_emails at 0x000001B9710F1DA0>

```

全局变量总是能够访问到的，局部变量只有走进“房间”里才会有且用完会被清理掉。



在 Python 里，变量查找遵循 LEGB 规则，此规则描述了 Python 解释器查找变量的顺序。LEGB 分别代表 Local（局部作用域）、Enclosing（闭包作用域）、Global（全局作用域）、Built-in（内置作用域）。下面为你详细解释：

### 1. Local（局部作用域）

局部作用域是指函数内部定义的变量所处的作用域。当在函数内部定义一个变量时，这个变量只能在该函数内部被访问，函数外部无法直接访问。

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

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

(7) 运算符 (operator)

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

形参：抽象

形参：具体的值

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

```
(Pdb) p contacts
[('白展堂', '男', 'baizhantang@163.com'), ('佟湘玉', '女', 'tongxiangyu@163.com'), ('吕轻侯', '男', 'lvqinghou@126.com'), ('郭芙蓉', '女', 'guofurong@126.com'), ('李秀莲', '男', 'lixiliu@163.com'), ('祝无双', '女', 'zhuwushuang@163.com')]
(Pdb) wat / contacts
value: [
  ('白展堂', '男', 'baizhantang@163.com'),
  ('佟湘玉', '女', 'tongxiangyu@163.com'),
  ('吕轻侯', '男', 'lvqinghou@126.com'),
  ('郭芙蓉', '女', 'guofurong@126.com'),
  ('李秀莲', '男', 'lixiliu@163.com'),
  ('祝无双', '女', 'zhuwushuang@163.com'),
]
type: list
len: 6

Public attributes:
def append(object, /) # Append object to the end of the list.
def clear() # Remove all items from list.
def copy() # Return a shallow copy of the list.
def count(value, /) # Return number of occurrences of value.
def extend(iterable, /) # Extend list by appending elements from the iterable.
def index(value, start=0, stop=9223372036854775807, /) # Return first index of value...
def insert(index, object, /) # Insert object before index.
def pop(index=-1, /) # Remove and return item at index (default last)...
def remove(value, /) # Remove first occurrence of value...
def reverse() # Reverse *IN PLACE*.
def sort(*, key=None, reverse=False) # Sort the list in ascending order and return None...
```