1. Fork 第 04 周打卡仓库并 Clone 到本地计算机

```
MINGW64:/c/Users/16386/rep

(base) 16386@□□ □□□□□□ MINGW64 ~
$ cd repo

(base) 16386@□□ □□□□□□ MINGW64 ~/repo
$ ls -l
total 4
drwxr-xr-x 1 16386 197609 0  3月  22 16:47 myproject/
drwxr-xr-x 1 16386 197609 0  3月  15 18:23 mywork/
drwxr-xr-x 1 16386 197609 0  3月  22 15:26 prj1/

(base) 16386@□□ □□□□□□ MINGW64 ~/repo
$ git clone https://gitcode.com/niuwen123/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 | 227.00 KiB/s, done.

(base) 16386@□□ □□□□□□ MINGW64 ~/repo
$ cd week04/

(base) 16386@□□ □□□□□□ MINGW64 ~/repo/week04 (main)
$ pwd
/c/Users/16386/repo/week04

(base) 16386@□□ □□□□□□ MINGW64 ~/repo/week04 (main)
$ git remote show origin
* remote origin
```

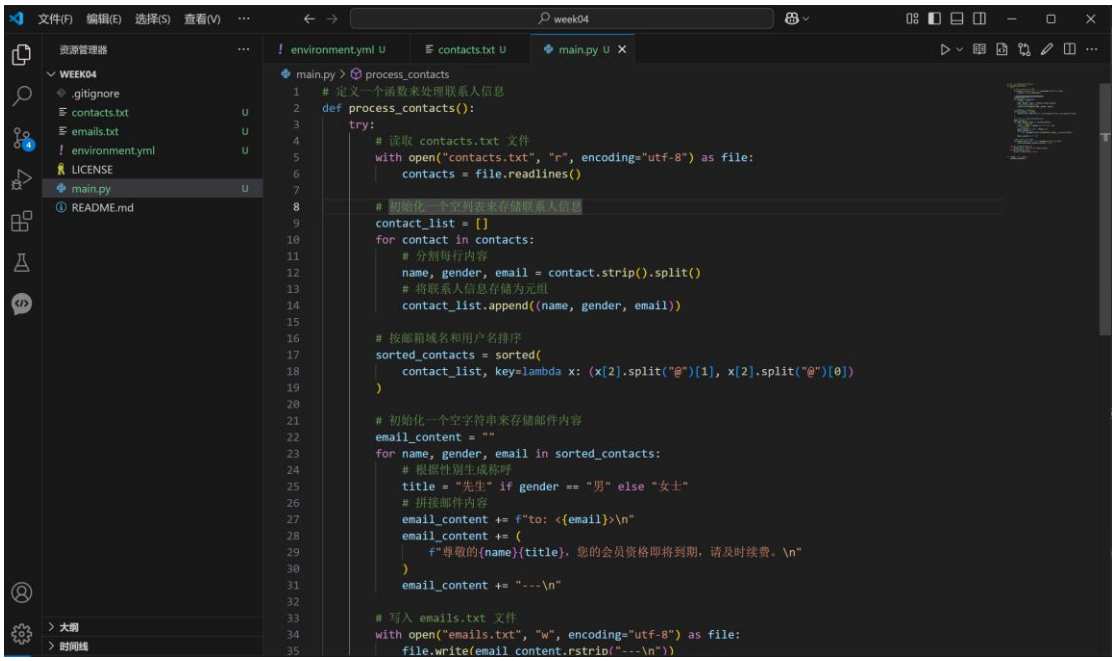
2. 新建 environment.yml 文件，安装 Python 3.12,创建 Conda 环境

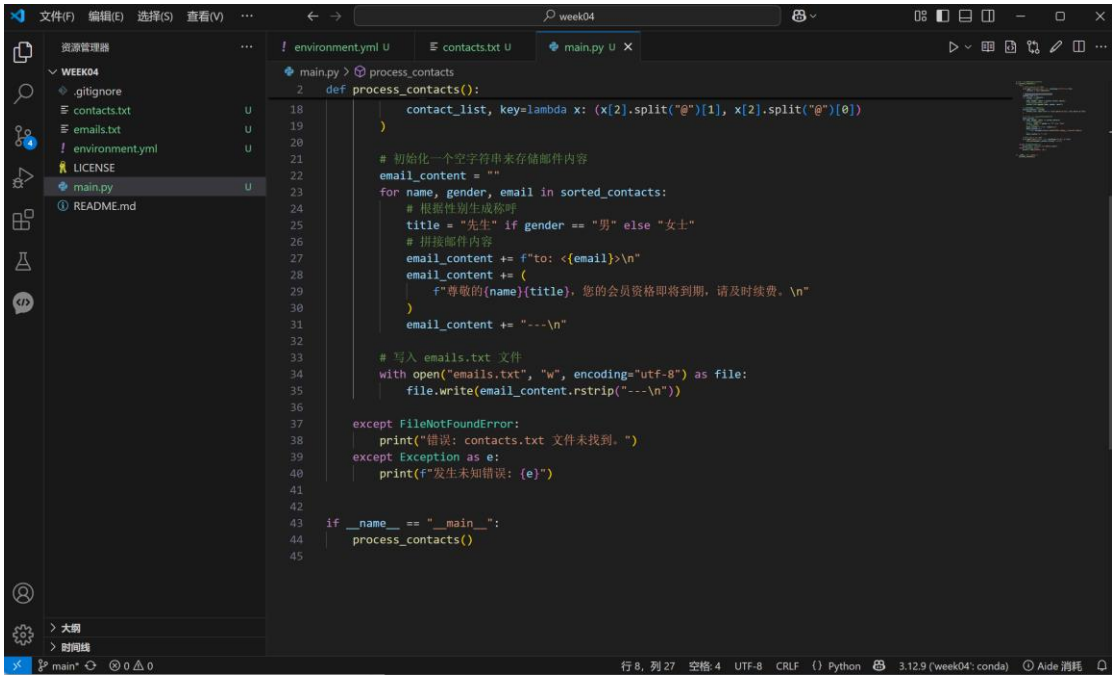
```
(base) 16386@□□ □□□□□□ MINGW64 ~/repo/week04 (main)
$ cp ../myproject/environment.yml
cp: missing destination file operand after '../myproject/environment.yml'
Try 'cp --help' for more information.

(base) 16386@□□ □□□□□□ MINGW64 ~/repo/week04 (main)
$ cp ../myproject/environment.yml ./

(base) 16386@□□ □□□□□□ MINGW64 ~/repo/week04 (main)
$ cat environment.yml
name: week04
channels:
  - conda-forge
dependencies:
  - python=3.12
(base) 16386@□□ □□□□□□ MINGW64 ~/repo/week04 (main)
$ conda env create
C:\Users\16386\anaconda3\Lib\argparse.py:2006: FutureWarning: `remote_definition` is deprecated and will be removed in 2
5.9. Use `conda env create --file=URL` instead.
  action(self, namespace, argument_values, option_string)
Retrieving notices: ...working... done
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: -
```

3. 运用 ai 大模型生成代码，成功运行可以产生 emails.txt





4. python -m pdb main.py 命令：启动代码解释器

pdb 提示符下运行 l 命令：显示代码运行到什么地方

n 命令：执行当前行

ll 命令：显示所有代码

p 命令：打印表达式

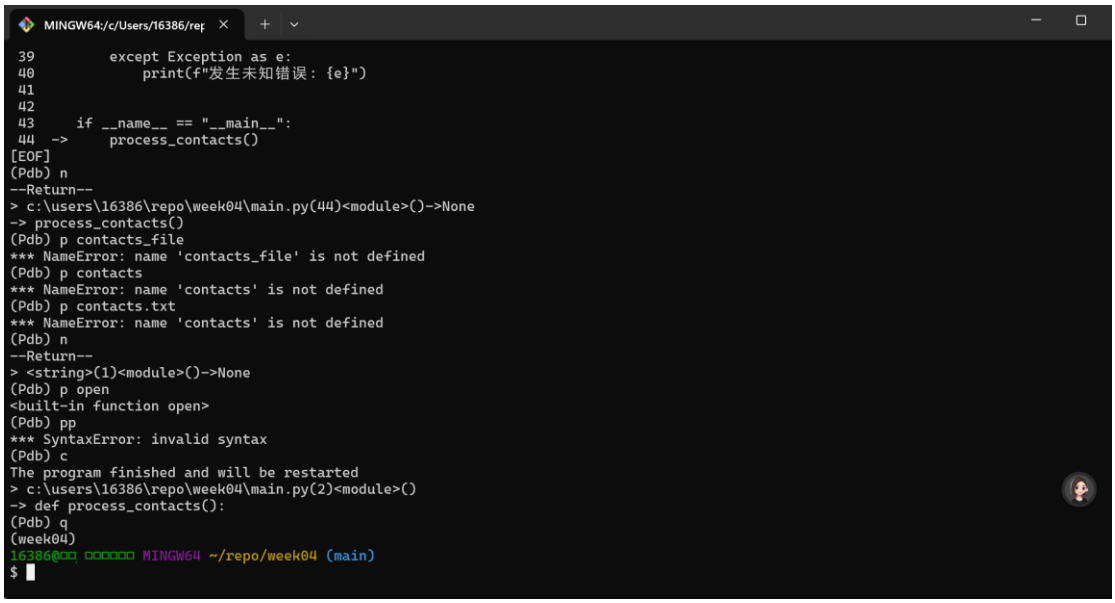
s 命令：步入调用

c 命令：继续运行

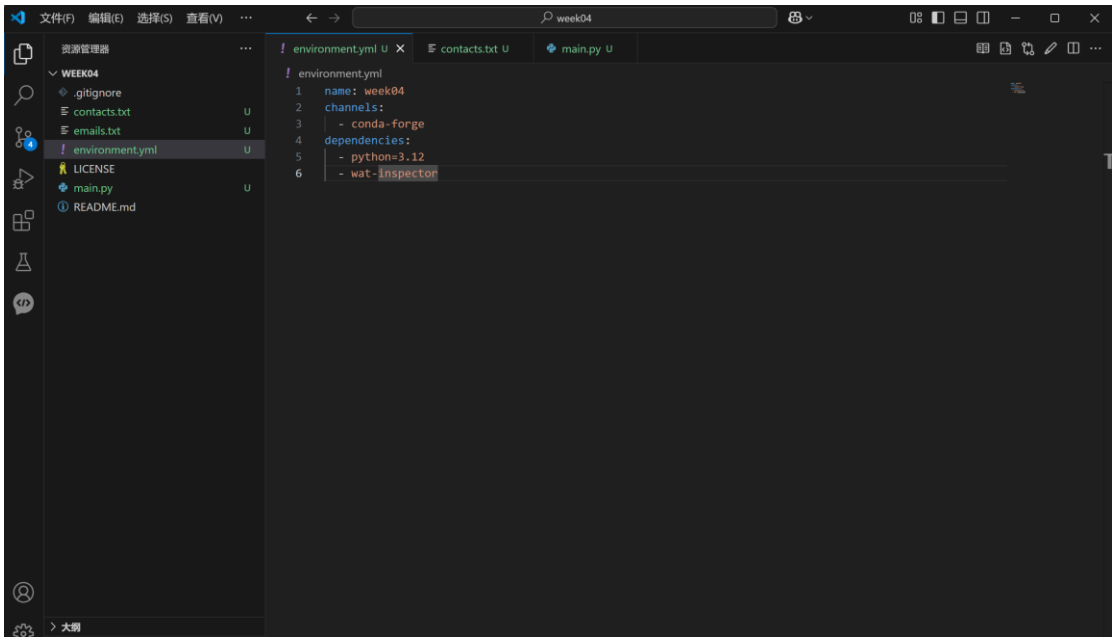
pp 命令：美观打印

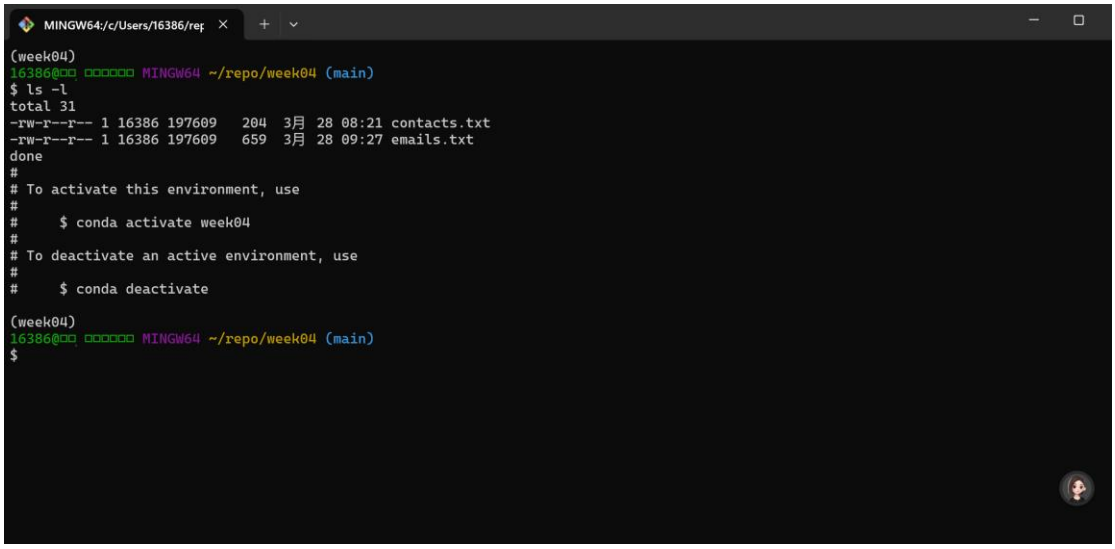
```
MINGW64:/c/Users/16386/rep
16386@□□ □□□□□□ MINGW64 ~/repo/week04 (main)
$ python -m pdb main.py
> c:\users\16386\repo\week04\main.py(2)<module>()
-> def process_contacts():
(Pdb) l
  1     # 定义一个函数来处理联系人信息
  2  -> def process_contacts():
  3         try:
  4             # 读取 contacts.txt 文件
  5             with open("contacts.txt", "r", encoding="utf-8") as file:
  6                 contacts = file.readlines()
  7
  8             # 初始化一个空列表来存储联系人信息
  9             contact_list = []
 10             for contact in contacts:
 11                 # 分割每行内容
(Pdb) n
> c:\users\16386\repo\week04\main.py(43)<module>()
-> if __name__ == "__main__":
(Pdb) l
 38             print("错误: contacts.txt 文件未找到。")
 39         except Exception as e:
 40             print(f"发生未知错误: {e}")
 41
 42
 43  -> if __name__ == "__main__":
 44         process_contacts()
[EOF]
(Pdb) ll
  1     # 定义一个函数来处理联系人信息
```

```
MINGW64:/c/Users/16386/rep
 44         process_contacts()
(Pdb) p process_contacts()
None
(Pdb) s
> c:\users\16386\repo\week04\main.py(44)<module>()
-> process_contacts()
(Pdb) l .
 39         except Exception as e:
 40             print(f"发生未知错误: {e}")
 41
 42
 43     if __name__ == "__main__":
 44  ->     process_contacts()
[EOF]
(Pdb) l 1,5
  1     # 定义一个函数来处理联系人信息
  2     def process_contacts():
  3         try:
  4             # 读取 contacts.txt 文件
  5             with open("contacts.txt", "r", encoding="utf-8") as file:
(Pdb) l .
 39         except Exception as e:
 40             print(f"发生未知错误: {e}")
 41
 42
 43     if __name__ == "__main__":
 44  ->     process_contacts()
[EOF]
(Pdb) n
--Return--
```



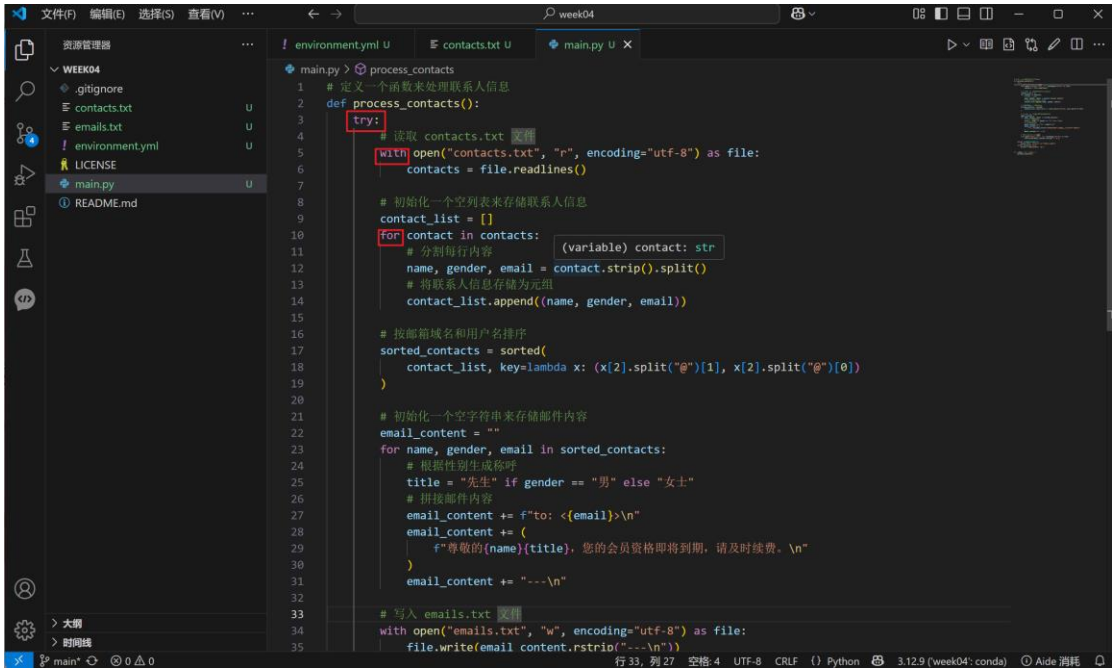
5. 利用 wat-inspector 检查文档





6. python 基本概念的学习

Python 语法保留字：下图粉色字体，表示在语法上有特殊含义



```
MINGW64:/c/Users/16386/rep
done
#
# To activate this environment, use
#
#     $ conda activate week04
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(week04)
16386@□□ □□□□□□ MINGW64 ~/repo/week04 (main)
$ 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 = 'Wen Niu'
>>> print(name)
Wen Niu
>>> def = 'Wen Niu'
  File "<stdin>", line 1
    def = 'Wen Niu'
        ^
SyntaxError: invalid syntax
>>> which = 'Qiang Gao'
>>> print(which)
Qiang Gao
>>>
```

语句和表达式：表达式是构成语句的元素

```python
# 定义一个函数来处理联系人信息
def process_contacts():
    try:
        # 读取 contacts.txt 文件
        with open("contacts.txt", "r", encoding="utf-8") as file:
            contacts = file.readlines()

        # 初始化一个空列表来存储联系人信息
        contact_list = []
        for contact in contacts:
            # 分割每行内容
            name, gender, email = contact.strip().split()
            # 将联系人信息存储为元组
            contact_list.append((name, gender, email))

        # 按邮箱域名和用户名排序
        sorted_contacts = sorted(
            contact_list, key=lambda x: (x[2].split("@")[1], x[2].split("@")[0])
        )

        # 初始化一个空字符串来存储邮件内容
        email_content = ""
        for name, gender, email in sorted_contacts:
            # 根据性别生成称呼
            title = "先生" if gender == "男" else "女士"
            # 拼接邮件内容
            email_content += f"to: <{email}>\n"
            email_content += (
                f"尊敬的{name}{title}，您的会员资格即将到期，请及时续费。\n"
            )
            email_content += "---\n"

        # 写入 emails.txt 文件
        with open("emails.txt", "w", encoding="utf-8") as file:
            file.write(email_content.rstrip("---\n"))
```

```python
# 定义一个函数来处理联系人信息
def process_contacts():
    try:
        # 读取 contacts.txt 文件
        with open("contacts.txt", "r", encoding="utf-8") as file:
            contacts = file.readlines()

        # 初始化一个空列表来存储联系人信息
        contact_list = []
        for contact in contacts:
            # 分割每行内容
            name, gender, email = contact.strip().split()
            # 将联系人信息存储为元组
            contact_list.append((name, gender, email))

        # 按邮箱域名和用户名排序
        sorted_contacts = sorted(
            contact_list, key=lambda x: (x[2].split("@")[1], x[2].split("@")[0])
        )

        # 初始化一个空字符串来存储邮件内容
        email_content = ""
        for name, gender, email in sorted_contacts:
            # 根据性别生成称呼
            title = "先生" if gender == "男" else "女士"
            # 拼接邮件内容
            email_content += f"to: <{email}>\n"
            email_content += (
                f"尊敬的{name}{title}，您的会员资格即将到期，请及时续费。\n"
            )
            email_content += "---\n"

        # 写入 emails.txt 文件
        with open("emails.txt", "w", encoding="utf-8") as file:
            file.write(email_content.rstrip("---\n"))
```

函数

变量名

字符串

缩进：代表层级

局部变量、全局变量

```
MINGW64:/c/Users/16386/rep
 8            # 初始化一个空列表来存储联系人信息
 9            contact_list = []
10            for contact in contacts:
11                # 分割每行内容
(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
  .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\16386\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\\16386\\repo\\week04\\main.py'
(Pdb)
```

```
MINGW64:/c/Users/16386/rep
(Pdb) q
(week04)
16386@□□ □□□□□□ MINGW64 ~/repo/week04 (main)
$ python -m pdb main.py
> c:\users\16386\repo\week04\main.py(2)<module>()
-> def process_contacts():
(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
  .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.globals
Global variables:
  __builtins__: dict = {…
  __file__: pdb._ScriptTarget = 'C:\Users\16386\repo\week04\main.py'
  __name__: str = '__main__'
  __pdb_convenience_variables: dict = {…
  __spec__: NoneType = None
  wat: wat.inspection.inspection.Wat = <WAT Inspector object>
(Pdb)
```

LEGB 规则：“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）

函数定义和调用

```python
# 定义一个函数来处理联系人信息
def process_contacts():
    try:
        # 读取 contacts.txt 文件
        with open("contacts.txt", "r", encoding="utf-8") as file:
            contacts = file.readlines()

        # 初始化一个空列表来存储联系人信息
        contact_list = []
        for contact in contacts:
            # 分割每行内容
            name, gender, email = contact.strip().split()
            # 将联系人信息存储为元组
            contact_list.append((name, gender, email))

        # 按邮箱域名和用户名排序
        sorted_contacts = sorted(
            contact_list, key=lambda x: (x[2].split("@")[1], x[2].split("@")[0])
        )

        # 初始化一个空字符串来存储邮件内容
        email_content = ""
        for name, gender, email in sorted_contacts:
            # 根据性别生成称呼
            title = "先生" if gender == "男" else "女士"
            # 拼接邮件内容
            email_content += f"to: <{email}>\n"
            email_content += (
                f"尊敬的{name}{title}，您的会员资格即将到期，请及时续费。\n"
            )
            email_content += "---\n"

        # 写入 emails.txt 文件
        with open("emails.txt", "w", encoding="utf-8") as file:
            file.write(email_content.rstrip("---\n"))
```

```
(Pdb) p print
<built-in function print>
(Pdb) p print('aaa')
aaa
None
(Pdb)
```

对象：在内存里的数据

属性：类型有的值方面的一些特点

方法：类型能干什么事