第四周学习笔记：

```
(base) wdhwl@wcx MINGW64 ~
$ cd hello
(base) wdhwl@wcx MINGW64 ~/hello
$ ls -l
total 6929
-rw-r--r-- 1 wdhwl 197609     162  3月 30 13:57 '~$四周学习笔记.docx'
drwxr-xr-x 1 wdhwl 197609       0  3月 24 00:00  myproject/
drwxr-xr-x 1 wdhwl 197609       0  3月 23 23:26  prj1/
-rw-r--r-- 1 wdhwl 197609       0  3月 11 14:48  sccipt1.py
drwxr-xr-x 1 wdhwl 197609       0  3月 11 16:08  week01/
-rw-r--r-- 1 wdhwl 197609  665362  3月 16 21:46  第二周学习笔记.docx
-rw-r--r-- 1 wdhwl 197609 1199452  3月 16 21:57  第二周学习笔记.pdf
-rw-r--r-- 1 wdhwl 197609 1776432  3月 24 00:17  第三周学习笔记.docx
-rw-r--r-- 1 wdhwl 197609 3162251  3月 24 00:18  第三周学习笔记.pdf
-rw-r--r-- 1 wdhwl 197609       0  3月 30 13:57  第四周学习笔记.docx
-rw-r--r-- 1 wdhwl 197609  274726  3月 11 15:58  第一周学习笔记.docx
(base) wdhwl@wcx MINGW64 ~/hello
$ git clone https://gitcode.com/wolfxiao/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 | 345.00 KiB/s, done.
(base) wdhwl@wcx MINGW64 ~/hello
$ cd week04/
(base) wdhwl@wcx MINGW64 ~/hello/week04 (main)
$ pwd
/c/Users/wdhwl/hello/week04
(base) wdhwl@wcx MINGW64 ~/hello/week04 (main)
$ git remote show origin
* remote origin
  Fetch URL: https://gitcode.com/wolfxiao/week04.git
  Push  URL: https://gitcode.com/wolfxiao/week04.git
  HEAD branch: main
  Remote branch:
    main tracked
  Local branch configured for 'git pull':
    main merges with remote main
  Local ref configured for 'git push':
    main pushes to main (up to date)
```

```
(base) wdhwl@wcx MINGW64 ~/hello/week04 (main)
$ conda remove -n myproject --all

Remove all packages in environment C:\Users\wdhwl\anaconda3\envs\myproject:


## Package Plan ##

  environment location: C:\Users\wdhwl\anaconda3\envs\myproject


The following packages will be REMOVED:

  bzip2-1.0.8-h2466b09_7
  ca-certificates-2025.1.31-h56e8100_0
  libexpat-2.6.4-he0c23c2_0
  libffi-3.4.6-h537db12_0
  liblzma-5.6.4-h2466b09_0
  libsqlite-3.49.1-h67fdade_2
done
#
# To activate this environment, use
#
#     $ conda activate myproject
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

```
(base) wdhwl@wcx MINGW64 ~/hello/week04 (main)
$ cat contacts.txt
(base) wdhwl@wcx MINGW64 ~/hello/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
done
#
# To activate this environment, use
#
#     $ conda activate week04
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(base) wdhwl@wcx MINGW64 ~/hello/week04 (main)
$ conda activate week04
(week04) wdhwl@wcx MINGW64 ~/hello/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(week04) wdhwl@wcx MINGW64 ~/hello/week04 (main)
$ cat main.py
# 打开 contacts.txt 文件进行读取
with open('contacts.txt', 'r', encoding='utf-8') as file:
    # 读取文件的每一行并去除首尾空格
    lines = file.read().splitlines()

contacts = []
# 遍历每一行数据
for line in lines:
    # 按空格分割每一行，得到姓名、性别和邮箱
    name, gender, email = line.split()
```

```
for line in lines:
    # 按空格分割每一行，得到姓名、性别和邮箱
    name, gender, email = line.split()
    # 将联系人信息存储为元组
    contacts.append((name, gender, email))

# 定义排序函数，先按邮箱域名排序，再按邮箱用户名排序
def sort_key(contact):
    email = contact[2]
    username, domain = email.split('@')
    return (domain, username)

# 对联系人列表进行排序
contacts.sort(key=sort_key)

# 打开 emails.txt 文件进行写入
with open('emails.txt', 'w', encoding='utf-8') as file:
    # 遍历排序后的联系人列表
    for name, gender, email in contacts:
        # 根据性别生成称呼
        title = '先生' if gender == '男' else '女士'
        # 生成通知内容
        message = 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---\n"
        # 将通知内容写入文件
        file.write(message)

print("emails.txt 文件已生成。")(week04) wdhwl@wcx MINGW64 ~/hello/week04 (main)
$ python main.py
emails.txt 文件已生成。
(week04) wdhwl@wcx MINGW64 ~/hello/week04 (main)
$ ls -l
total 31
-rw-r--r-- 1 wdhwl 197609   202  3月 30 14:23 contacts.txt
-rw-r--r-- 1 wdhwl 197609   666  3月 30 14:41 emails.txt
-rw-r--r-- 1 wdhwl 197609    72  3月 30 14:23 environment.yml
-rw-r--r-- 1 wdhwl 197609 18805  3月 30 14:04 LICENSE
-rw-r--r-- 1 wdhwl 197609  1246  3月 30 14:29 main.py
-rw-r--r-- 1 wdhwl 197609  2239  3月 30 14:04 README.md
```

```
(week04) wdhwl@wcx MINGW64 ~/hello/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: <lixiulian@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。
---
```

通过老师对 ai 的这段文字与讲解，这场景让我想起小时候玩乐高，ai 就像现成的积木套装，而我们要做的是拆掉说明书，搭出会飞的房子。听说硅谷有些公司开始把 ai 生成的代码标注为 "辅助创作"，就像电影字幕里的 "动作指导"，主角还是得人类自己来。

Ai 确实在改变创作的游戏规则。就像外卖软件没让厨师失业，反而催生了更多创意料理。或许未来的创作者会分成两派：一派用 ai 批量生产标准化作品，另一派则专门寻找 "ai 做不到的事"。就像现在有人用 ai 画商稿，也有人坚持手绘限量版。毕竟在这个什么都能复制的时代，那些带着人类体温的 "不完美"，反而成了最珍贵的奢侈品。突然想到《头号玩家》里的彩蛋，真正的宝藏往往藏在那些笨拙的、反效率的细节里。或许这就是人类对抗 ai 的终极武器 —— 我们永远会为了某个 "没必要" 的坚持，在数字世界里留下独特的生命痕迹。

```
  2  -> with open('contacts.txt', 'r', encoding='utf-8') as file:
  3         # 读取文件的每一行并去除首尾空格
  4         lines = file.read().splitlines()
  5
  6     contacts = []
  7     # 遍历每一行数据
  8     for line in lines:
  9         # 按空格分割每一行，得到姓名、性别和邮箱
 10         name, gender, email = line.split()
 11         # 将联系人信息存储为元组
(Pdb) n
> c:\users\wdhwl\hello\week04\main.py(4)<module>()
-> lines = file.read().splitlines()
(Pdb) p contacts
*** NameError: name 'contacts' is not defined
(Pdb) p read_contacts
*** NameError: name 'read_contacts' is not defined
(Pdb) s
--Call--
> <frozen codecs>(319)decode()
(Pdb) ll
*** could not get source code
```

```
 22
 23     # 打开 emails.txt 文件进行写入
 24     with open('emails.txt', 'w', encoding='utf-8') as file:
 25         # 遍历排序后的联系人列表
 26         for name, gender, email in contacts:
 27             # 根据性别生成称呼
 28             title = '先生' if gender == '男' else '女士'
 29             # 生成通知内容
 30             message = 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---\n"
 31             # 将通知内容写入文件
 32             file.write(message)
 33
 34     print("emails.txt 文件已生成。")
(Pdb) l 23,34
 23     # 打开 emails.txt 文件进行写入
 24     with open('emails.txt', 'w', encoding='utf-8') as file:
 25         # 遍历排序后的联系人列表
 26         for name, gender, email in contacts:
 27             # 根据性别生成称呼
 28             title = '先生' if gender == '男' else '女士'
 29             # 生成通知内容
 30             message = 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---\n"
 31             # 将通知内容写入文件
 32             file.write(message)
 33
 34     print("emails.txt 文件已生成。")
(Pdb) q
(week04) wdhwl@wcx MINGW64 ~/hello/week04 (main)
$
```

```
(base) wdhwl@wcx MINGW64 ~
$ python
Python 3.12.7 | packaged by Anaconda, Inc. | (main, Oct  4 2024, 13:17:27) [MSC v.1929 64 bit (AMD64
Type "help", "copyright", "credits" or "license" for more information.
>>> # 这是一个全局变量，在整个模块中都可以访问
>>> global_variable = 10
>>>
>>> # 定义一个函数
>>> def example_function(local_parameter):
...     # 这是一个局部变量，只能在函数内部访问
...     local_variable = 20
...     # 打印局部变量
...     print(f"局部变量的值：{local_variable}")
...     # 打印传入的参数
...     print(f"传入的参数值：{local_parameter}")
...     # 访问全局变量
...     print(f"全局变量的值：{global_variable}")
...     # 定义一个字典
...     my_dict = {'key1': 'value1', 'key2': 2}
...     # 定义一个列表
...     my_list = [1, 2, 3]
...     # 定义一个元组
...     my_tuple = (4, 5, 6)
...     # 定义一个字符串
...     my_string = "Hello, World!"
...     # 打印不同类型的字面值
...     print(f"字典：{my_dict}")
...     print(f"列表：{my_list}")
...     print(f"元组：{my_tuple}")
...     print(f"字符串：{my_string}")
```

```
全局变量的值：10
字典：{'key1': 'value1', 'key2': 2}
列表：[1, 2, 3]
元组：(4, 5, 6)
字符串：Hello, World!
运算结果：25
>>> print(f"函数返回值：{returned_value}")
函数返回值：25
>>>
>>> # 下面解释一些其他概念
>>>
>>> # 语句：Python 中可以独立执行的一行代码就是一个语句，例如上面的赋值语句、函数调用语句等
>>> # 表达式：表达式是由变量、字面值和运算符组成的，会计算出一个值，例如 local_variable + local_parameter 就是一个表达式

>>>
>>> # 缩进：Python 使用缩进来表示代码块，例如函数体就是通过缩进来界定的，如果缩进不正确，会导致语法错误
>>>
>>> # LEGB 规则：在查找变量时，Python 会按照 Local（局部作用域）、Enclosing（闭包作用域）、Global（全局作用域）、Built-i
n（内置作用域）的顺序进行查找
>>>
>>> # 对象和类型：Python 中一切都是对象，每个对象都有其类型，例如 my_dict 是一个字典对象，类型是 dict
>>> print(f"my_dict 的类型：{type(my_dict)}")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'my_dict' is not defined
>>>
>>> # 属性和方法：对象可以有属性和方法，例如列表对象有 append 方法，可以用来添加元素
```

代码解释

Python 语法保留字：Python 中有一些保留字，如 def 用于定义函数，return 用于从函数返回值等，这些保留字不能用作变量名。

语句和表达式：语句是可以独立执行的代码，如赋值语句、函数调用语句等；表达式是由变量、字面值和运算符组成的，会计算出一个值。

缩进：Python 使用缩进来表示代码块，函数体、条件语句块等都是通过缩进来界定的。

局部变量、全局变量、LEGB 规则：局部变量在函数内部定义，只能在函数内部访问；全局变量在模块级别定义，可以在整个模块中访问。LEGB 规则是 Python 查找变量的顺序。

函数的定义和调用：使用 def 关键字定义函数，然后通过函数名和括号来调用函数。

字面值：代码中定义的字符串、整数、列表、字典、元组等都是字面值。

运算符：代码中使用了 + 运算符进行加法运算。

形参、实参、返回值：形参是函数定义时的参数，实参是函数调用时传入的参数，返回值是函数通过 return 语句返回的值。

对象、类型、属性、方法：Python 中一切都是对象，每个对象都有其类型。对象可以有属性和方法，属性是对象的特征，方法是对象的行为。

安装 wat 无法用 python 打开，目前未发现问题所在，可能是版本兼容性问题