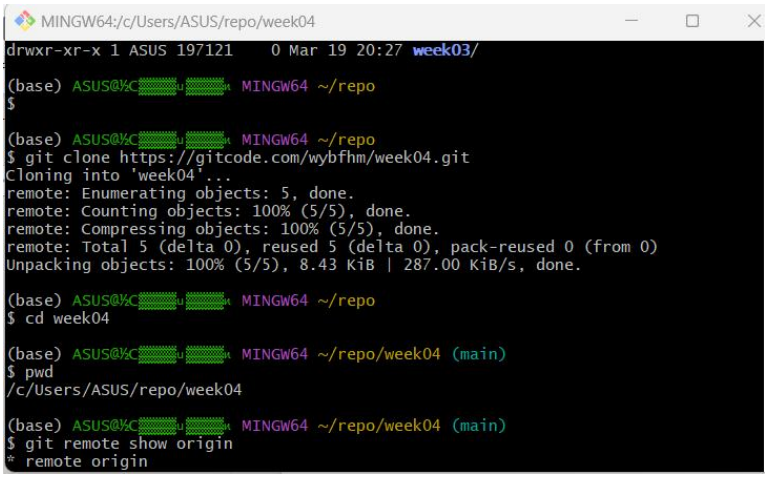
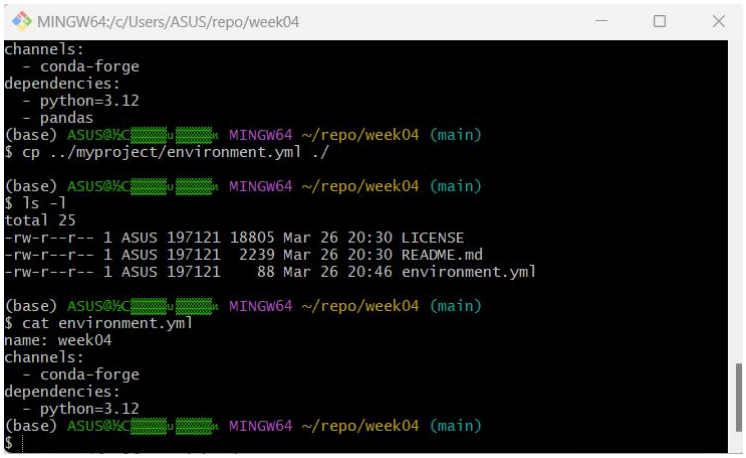
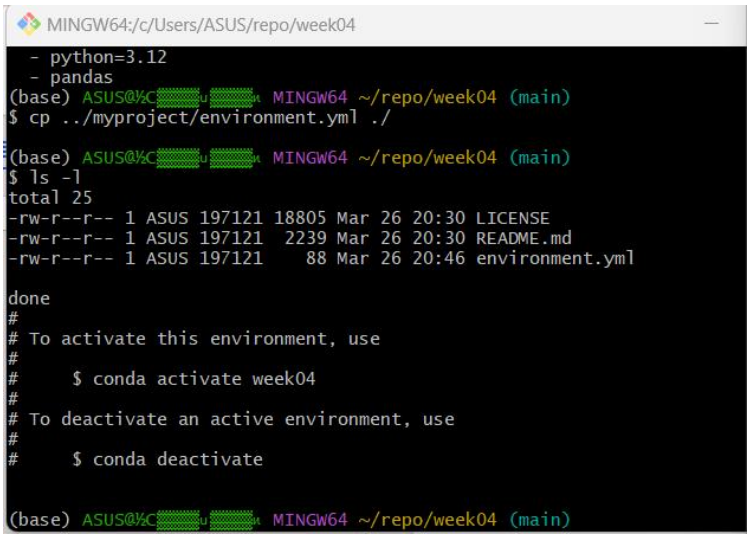
金融计算与编程第四周学习笔记

1. 把名下的仓库 Clone 到我的电脑。

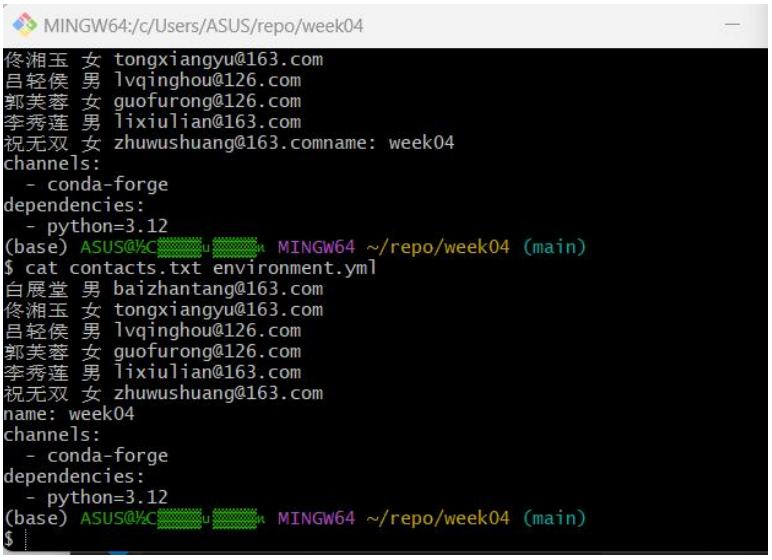


2. 创建 conda 环境。Ctrl+E 可以直接切换到代码的最后，ctrl+A 切换到最前面。

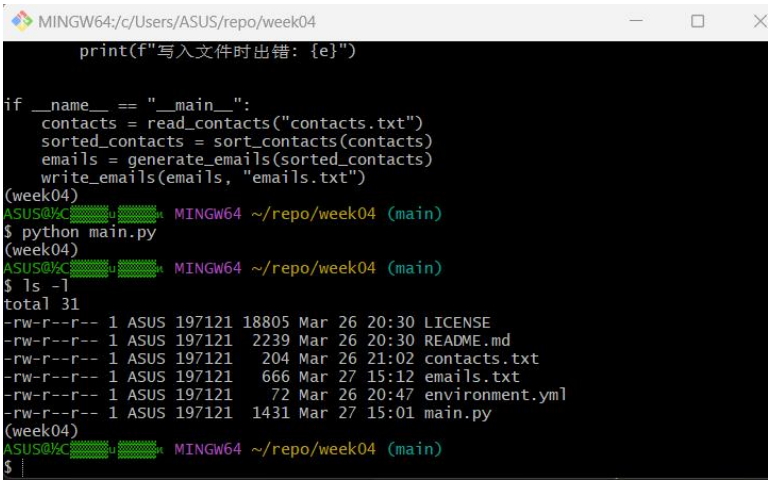




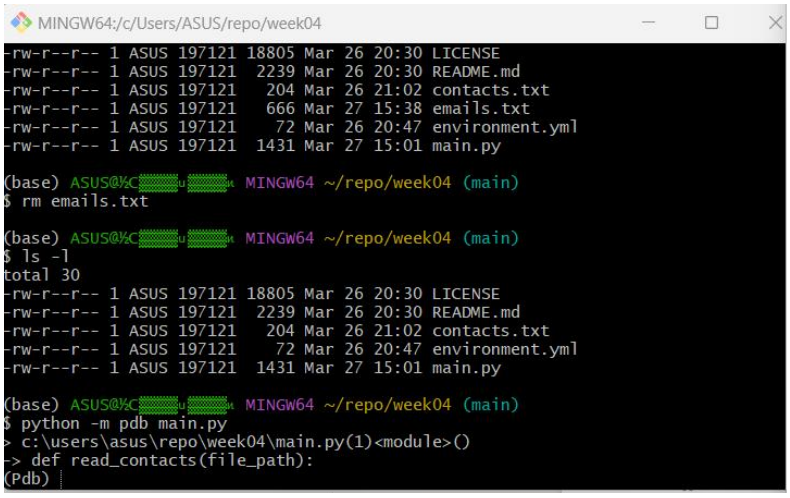
3. 文本文件的最后最好加一个空行。



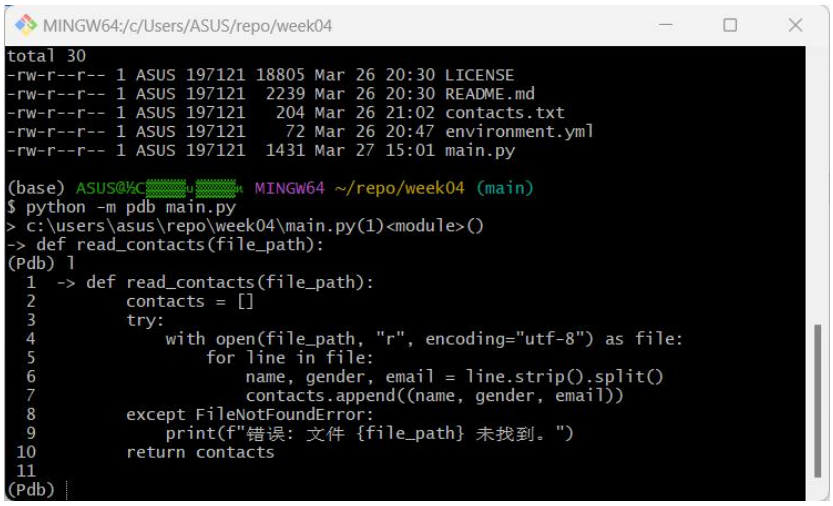
4. 生成 txt 文件。



5. python 基础概念学习：

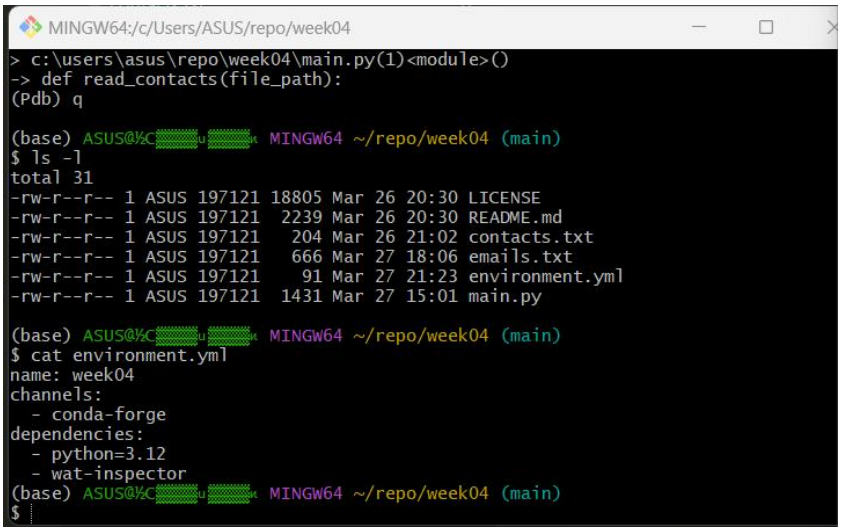


让调试器运行，l 有显示代码的作用



箭头指向的是即将要运行的代码，n 是执行代码的意思，ll 显示全部代码，p 打印表达式，s 是步入调用，定义是定义，运行是运行，l .是显示运行的上面五行加下面五行的代码，pp 是美观打印。

6. 安装 wat。



发现没有切换到 week04 环境，用 conda activate week04 切换。

```
  - python=3.12
  - wat-inspector(week04)
ASUS@½C▓▓▓▓u▓▓▓▓ 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)
ASUS@½C▓▓▓▓u▓▓▓▓ MINGW64 ~/repo/week04 (main)
$
```

7. def，while 为保留字，不能作为变量名，python 代码里的红色字体是保留字，不能作为变量名。

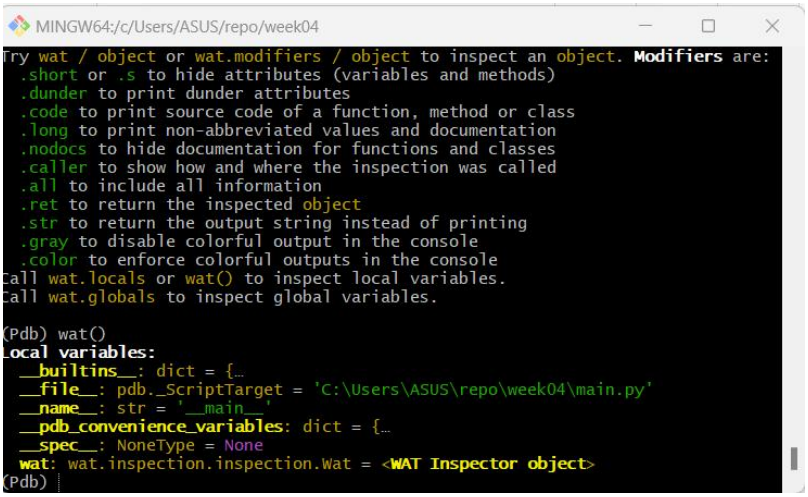
```
#
# To activate this environment, use
#
#     $ conda activate week04
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(week04)
ASUS@½C▓▓▓▓u▓▓▓▓ MINGW64 ~/repo/week04 (main)
$ python
Python 3.12.9 | packaged by conda-forge | (main, Mar  4 2025, 22:37:18) [MSC v.1
943 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> name = 'Yun Wang'
>>> print(name)
Yun Wang
>>> def = 'Yun Wang'
  File "<stdin>", line 1
    def = 'Yun Wang'
        ^
SyntaxError: invalid syntax
>>>
```

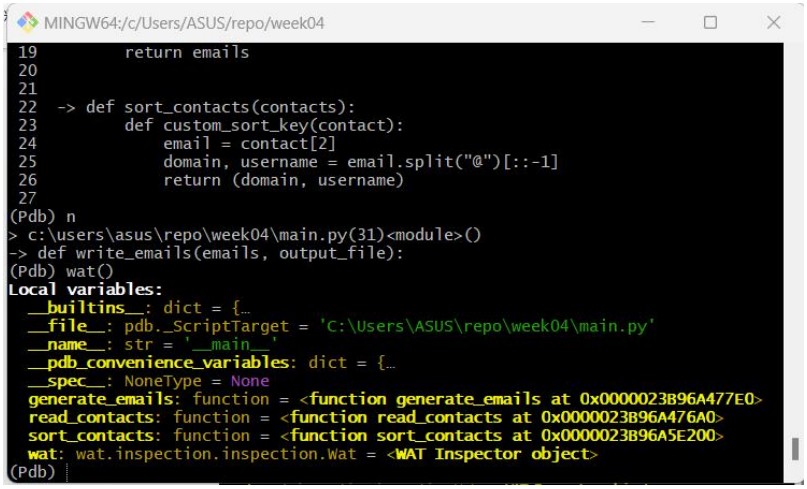
语句：逻辑上完整的一句话，for 循环语句，赋值语句，if 语句，return 语句，with 语句，表达式是构成语句的元素，相当于词语。语句可以嵌套，语句里可以有表达式，file 是表达式，if 语句里的条件判断也是表达式。一个表达式自己也构成语句，表达式是也可以嵌套，如__name__ == “__main__”，name 和 main 两个语句构成了新语句。在 vs code 里，白色代码是变量名，黄色是字符串，类型比较明确的是绿色，橙色的是参数。

缩进在 python 里很重要，缩进必须严格对齐，缩进代表层级。

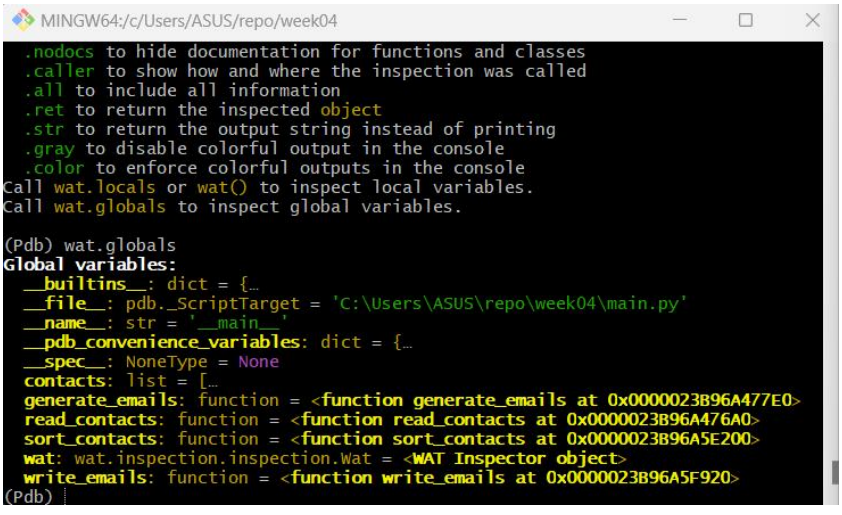
局部变量：



随着代码的运行，wat()里的局部变量会随之更新。



全局变量：全局变量总能访问到，是顶级的。



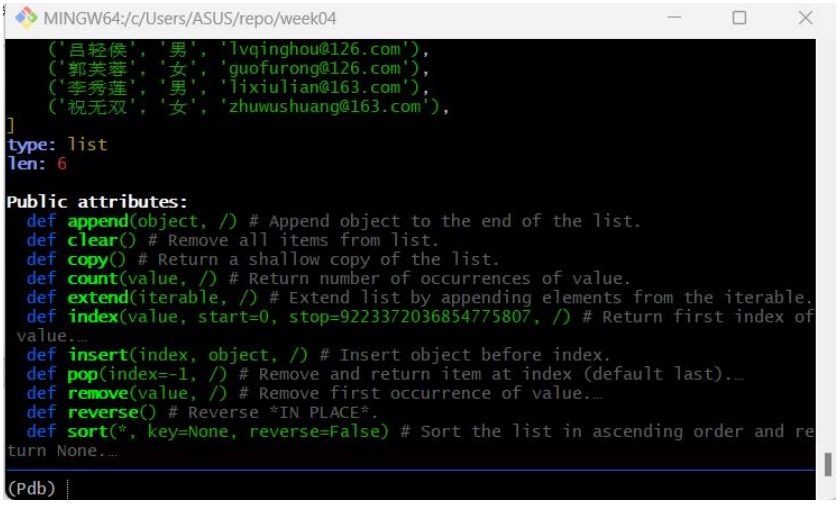
一旦 return，中途里面的东西就会被清理。相当于的房间里的东西就是局部变量，一旦运行完毕就会被清理。

函数里的函数是可以访问外层函数的变量的，legb 概念。

调用函数必须满足参数。

两个等号是运算符，“.”是名称访问运算符。Python 所管理的内存都是对

象。对象都有自己的类型。


MINGW64:/c/Users/ASUS/repo/week04
('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'),
]
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 re
turn None.…
(Pdb)