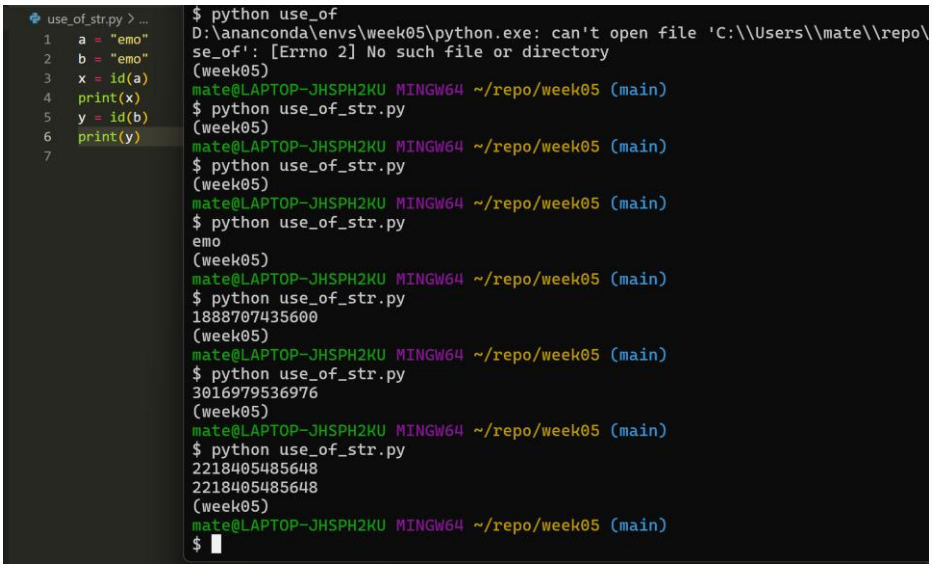
第五周学习报告

掌握最基础、最常用的几种 Python 对象类型 (type)，包括字符串 (str)、字节串 (bytes)、整数 (int)、浮点数 (float)、布尔值 (bool)、列表 (list)、字典 (dict)、元组 (tuple)、集合 (set)。这几种类型都是 Python 解释器 内置的 (built-in)，不需要任何导入 (import)。

```
(base) mate@LAPTOP-JHSPH2KU MINGW64 ~/repo
$ cd week05
done
#
# To activate this environment, use
#
#     $ conda activate week05
#
# To deactivate an active environment, use
#
#     $ conda deactivate


(base) mate@LAPTOP-JHSPH2KU MINGW64 ~/repo/week05 (main)
$
```

逐个创建 `use_of_{name}.py` 文件，其中 `{name}` 替换为上述要求掌握的对象类型，例如 `use_of_str.py`：

- 在全局作用域 (global scope) 内尝试键入 (活学活用) Python 代码，亲手验证概念 (Proof of Concept, PoC)
- 对于任何对象，都可以传给以下内置函数 (built-in function) 用于检视 (inspect):
  - `id()` -- 返回对象在虚拟内存中的地址 (正整数)，如果 `id(a) == id(b)`，那么 `a is b` (`is` 是个运算符)
  - `type()` -- 返回对象的类型
  - `isinstance()` -- 判断对象是否属于某个 (或某些) 类型
  - `dir()` -- 返回对象所支持的属性 (attributes) 的名称列表
  - `str()` -- 返回对象 `print` 时要显示在终端的字符串
- 可以调用 `print()` 函数将表达式 (expression) 输出到终端，查看结果是否符合预期
- 可以利用 `assert` 语句查验某个表达式 (expression) 为真，否则报错 (`AssertionError`) 退出
- 可以利用 `try` 语句拦截报错，避免退出，将流程 (flow) 转入 `except` 语句
- 可以调用 `breakpoint()` 函数暂停程序运行，进入 `pdb` 调试 (debug) 模式



```python
a = [2, 5]
b = [2, 5]
x = id(a)
print(x)
y = id(b)
print(y)
a[0] = 100
print(a)
print(b)
print(id(a))  # is it same as x?
print(id(b))  # is it same as y?
print(type(a))
print("isinstance(a, str): ", isinstance(a, str))

```

```
MINGW64:/c/Users/mate/repo
(week05)
mate@LAPTOP-JHSPH2KU MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
1719804172544
1719804170560
[100, 5]
[2, 5]
1719804172544
1719804170560
<class 'list'>
isinstance(a, str):  False
(week05)
mate@LAPTOP-JHSPH2KU MINGW64 ~/repo/week05 (main)
$ python
Python 3.12.9 | packaged by conda-forge | (main, Mar  4 2025,
```

```python
print("isinstance(a, str): ", isinstance(a, str))
print("dir(a):", dir(a))

```

```
MINGW64:/c/Users/mate/repo
<class 'list'>
isinstance(a, str):  False
dir(a): ['__add__', '__class__', '__class_getitem__', '__cont
ains__', '__delattr__', '__delitem__', '__dir__', '__doc__',
'__eq__', '__format__', '__ge__', '__getattribute__', '__geti
tem__', '__getstate__', '__gt__', '__hash__', '__iadd__', '__
imul__', '__init__', '__init_subclass__', '__iter__', '__le__
', '__len__', '__lt__', '__mul__', '__ne__', '__new__', '__re
duce__', '__reduce_ex__', '__repr__', '__reversed__', '__rmul
__', '__setattr__', '__setitem__', '__sizeof__', '__str__', '
__subclasshook__', 'append', 'clear', 'copy', 'count', 'exten
d', 'index', 'insert', 'pop', 'remove', 'reverse', 'sort']
(week05)
mate@LAPTOP-JHSPH2KU MINGW64 ~/repo/week05 (main)
$
```

```python
print("isinstance(a, str): ", isinstance(a, str))
print("isinstance(a), list):", isinstance(a, list))
print(isinstance(a, (str, float, list)))
assert isinstance(a, list)
print("nohappy")

```

```
MINGW64:/c/Users/mate/repo
$ python use_of_str.py
2596761049344
2596761047360
[100, 5]
[2, 5]
2596761049344
2596761047360
<class 'list'>
isinstance(a, str):  False
isinstance(a), list): True
True
nohappy
(week05)
mate@LAPTOP-JHSPH2KU MINGW64 ~/repo/week05 (main)
$
```

4. 对于 每一个 上述要求掌握的对象类型 (将来遇到新的对象类型也应该如此)，我们首先应该熟悉如何通过 **表达式** (expression) 得到他们的 **实例** (instance)，一般包括以下途径:

    - 字面值 (literal) (包括 f-string 语法)
    - 推导式 (comprehension) (仅限 `list`、`dict`、`set`)
    - 初始化 (init)
    - 运算值 (operator)
    - 索引值 (subscription)
    - 返回值 (return value of function/method call)

5. 对于 每一个 上述要求掌握的对象类型 (将来遇到新的对象类型也应该如此)，我们也要尝试验证其以下几个方面的 **属性** (attributes):

    - 对数学运算符 (`+`、`-`、`*`、`/`、`//`、`%`、`@`) 有没有支持
    - 如何判断相等 (`==`)
    - 对于比较运算符 (`>`、`<`、`>=`、`<=`) 有没有支持
    - 什么值被当作 `True`，什么值被当作 `False`
    - 是否可迭代 (iterable)，如何做迭代 (`for` 循环)
    - 是否支持返回长度 (`len`)
    - 是否 (如何) 支持索引操作 (subscription) (`[]` 运算符)
    - 拥有哪些常用方法 (method) 可供调用 (`()` 运算符)

    建议先在 `pdb` 里试验，然后把确定能够运行的代码写在 `use_of_{name}.py` 文件里

use_of_str.py > ...

```python
print("字面値")
s = "library"
print(s)
print(isinstance(s, str))
assert type(s) is str
m = [2, 4]
print(m)
print(isinstance(m, list))

```

MINGW64:/c/Users/mate/repo

```
$ python use_of_str.py
字面値
library
True
(week05)
mate@LAPTOP-JHSPH2KU MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面値
library
True
[2, 4]
True
(week05)
mate@LAPTOP-JHSPH2KU MINGW64 ~/repo/week05 (main)
$
```

environment.yml U | use_of_str.py U | use_of_bytes.py U

use_of_str.py > ...

```python
print("字面値")
s = "library"
print(s)
print(isinstance(s, str))
assert type(s) is str
m = [2, 4]
print(m)
print(isinstance(m, list))

print("f-string")
x = "cat"
s = f"name: {x}"
print(s)

print("f-string")
x = "meet"
s = f"brunch: {x}"
print(s)

```

MINGW64:/c/Users/mate/repo

```
(week05)
mate@LAPTOP-JHSPH2KU MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面値
library
True
[2, 4]
True
f-string
name: cat
f-string
brunch: meet
(week05)
mate@LAPTOP-JHSPH2KU MINGW64 ~/repo/week05 (main)
$
```

```python
aaa
"""
print(s)


print("初始化")
s = str()
print(s)

s = str([2, 4, 5, 6, 999999
print(s)


assert str([2, 4, 5, 6, 999
assert str([1.1 + 2.2]) ==

```

```
Running 'cont' or 'step' will restart the program
> c:\users\mate\repo\week05\use_of_str.py(44)<module>()
-> assert str([1.1 + 2.2]) == "3.3"
(Pdb) l.
 39     s = str([2, 4, 5, 6, 9999999999])
 40     print(s)
 41
 42
 43     assert str([2, 4, 5, 6, 9999999999]) == "[2, 4, 5, 6,
 9999999999]"
 44  -> assert str([1.1 + 2.2]) == "3.3"
[EOF]
(Pdb) p str(1.1 + 2.2)
'3.3000000000000003'
(Pdb)
```

```
100
101 assert s == "aaaa"  # 两个字
102
103 print("abc" > "ABC")  # TRU
104 print("abcdfff" > "12567367
105 print("&^$^&" > "buou*&^79"
106 print("9" < ":")  # true
107 print("book" < "?")  # fals
108 assert "book"
109 assert "buibui"  # 不是空的
110 assert not ""
111
112 s = "flower"
113 print(iter(s))  # 可迭代
114
115 breakpoint()
116 for c in s:
117     print(c)
118
```

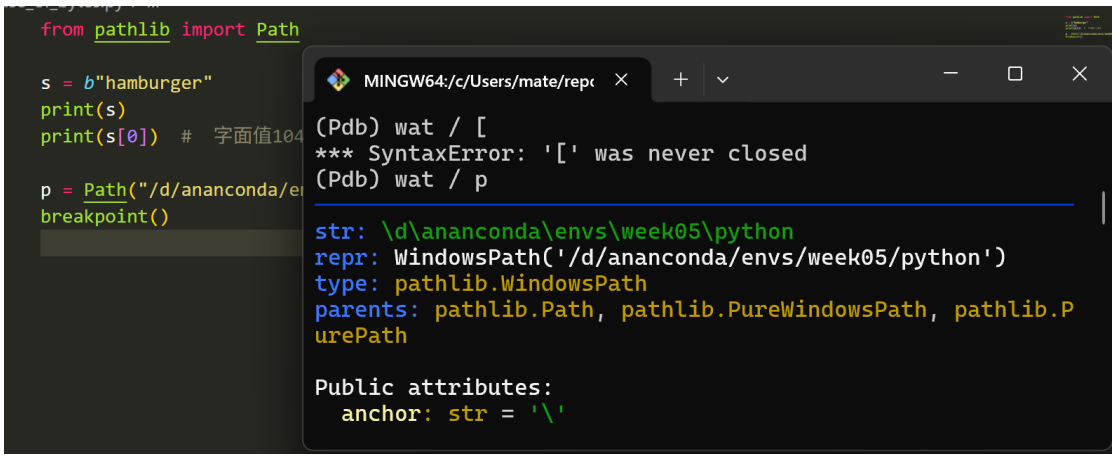
```
(Pdb) p next(g)
'f'
(Pdb) p next(g)
'l'
(Pdb) p next(g)
'o'
(Pdb) p next(g)
'w'
(Pdb) p next(g)
'e'
(Pdb) p next(g)
'r'
(Pdb) p next(g)
*** StopIteration
(Pdb)
```

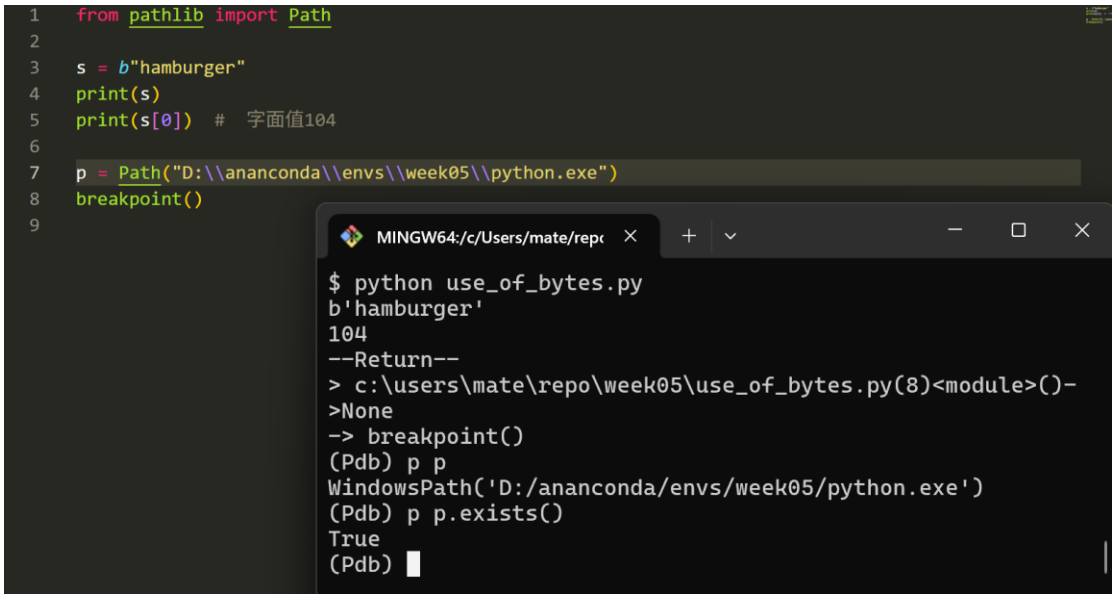
```
115
116 for c in s:  # 做迭代
117     print(c)
118
119 print(len(s))  # 返回长度
120
121 s = "reader"
122 assert s[1:5] == "eade"  #
123
124 s = "1y6656389"
125 assert s[2:8] == "665638"
126 breakpoint()
127
```

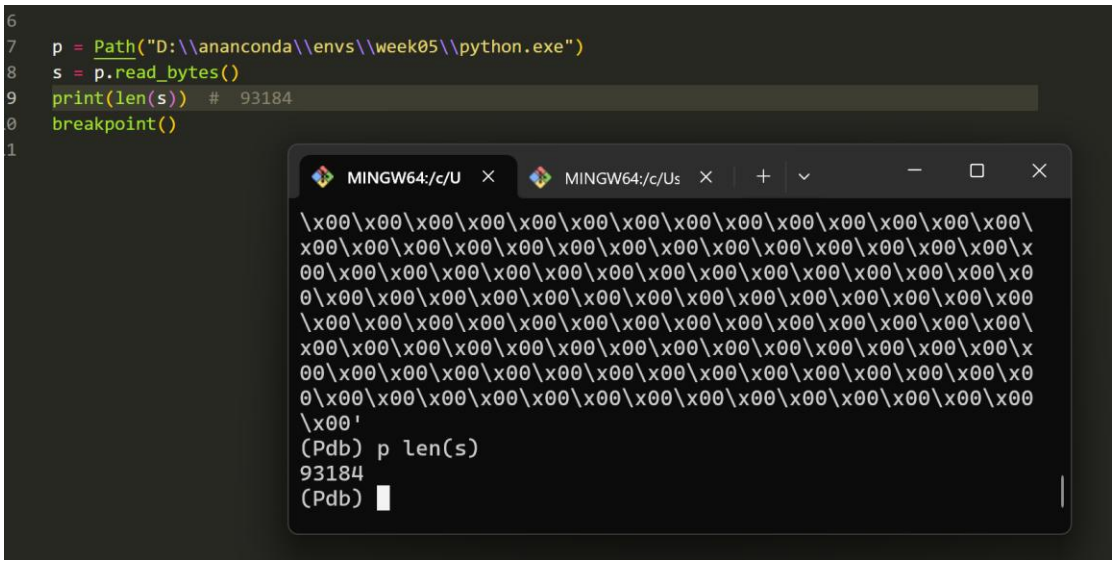
```
--Return--
> c:\users\mate\repo\week05\use_of_str.py(126)<module>()->Non
e
-> breakpoint()
(Pdb) p s
'1y6656389'
(Pdb) import wat
(Pdb) wat / s
value: '1y6656389'
type: str
len: 9

Public attributes:
```

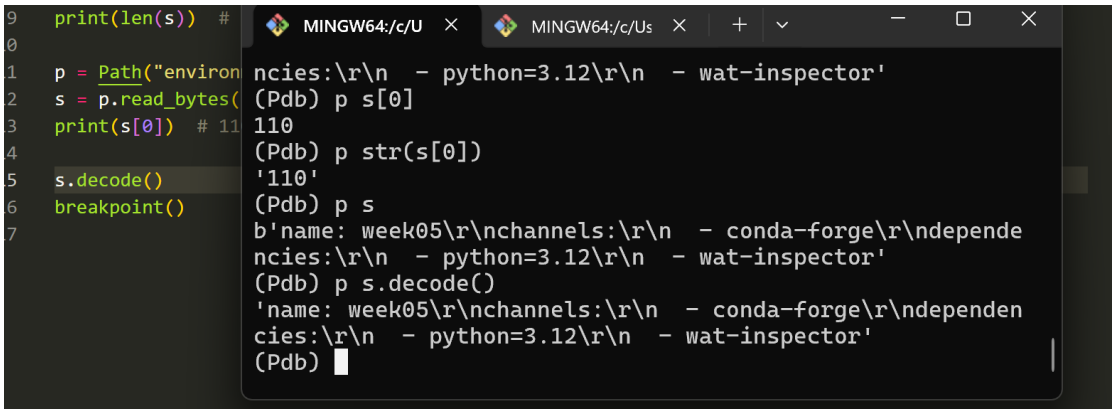
```
回长度

ade"  #

65638"

(Pdb) p s
'1y6656389'
(Pdb) p s.translate({'o': 'x'})
'1y6656389'
(Pdb) p s.translate({'y': 'x'})
'1y6656389'
(Pdb) p ord(y)
*** TypeError: ord() expected string of length 1, but int fou
nd
(Pdb) p ord('y')
121
(Pdb) p s.translate({ord('y'):ord('x')})
'1x6656389'
(Pdb)
```

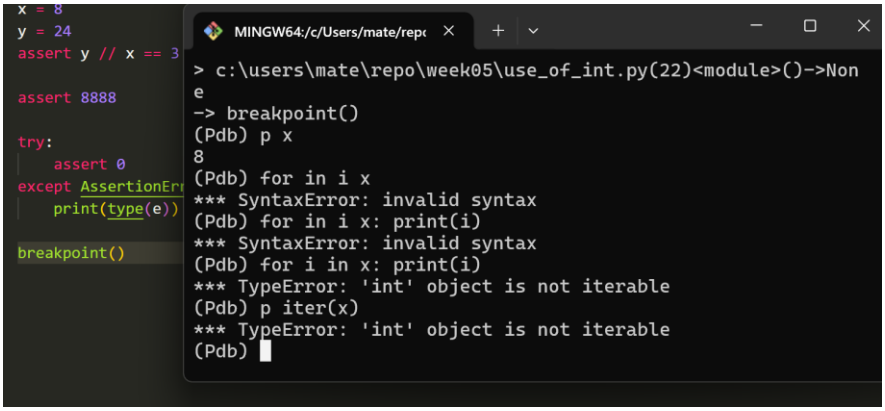
```
(Pdb) p s
'the emo of day'
(Pdb) p s
'the emo of day'
(Pdb) p s endswith('day')
*** SyntaxError: invalid syntax
(Pdb) p s.endswith('day')
True
(Pdb) p s.endswith('ay')
True
(Pdb) p s.endswith('y')
True
(Pdb) p s.endswith('emo of day')
True
(Pdb)
```


from pathlib import Path
s = b"hamburger"
print(s)
print(s[0]) # 字面值104
p = Path("/d/ananconda/e
breakpoint()
MINGW64:/c/Users/mate/repo
(Pdb) wat / [
*** SyntaxError: '[' was never closed
(Pdb) wat / p
str: \d\ananconda\envs\week05\python
repr: WindowsPath('/d/ananconda/envs/week05/python')
type: pathlib.WindowsPath
parents: pathlib.Path, pathlib.PureWindowsPath, pathlib.P
urePath
Public attributes:
anchor: str = '\'

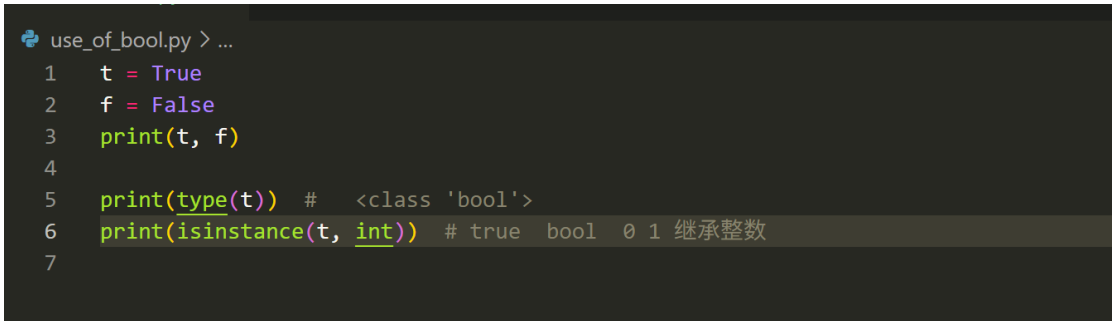

1 from pathlib import Path
2
3 s = b"hamburger"
4 print(s)
5 print(s[0]) # 字面值104
6
7 p = Path("D:\\ananconda\\envs\\week05\\python.exe")
8 breakpoint()
9
MINGW64:/c/Users/mate/repo
$ python use_of_bytes.py
b'hamburger'
104
--Return--
> c:\users\mate\repo\week05\use_of_bytes.py(8)<module>()-
>None
-> breakpoint()
(Pdb) p p
WindowsPath('D:/ananconda/envs/week05/python.exe')
(Pdb) p p.exists()
True
(Pdb)

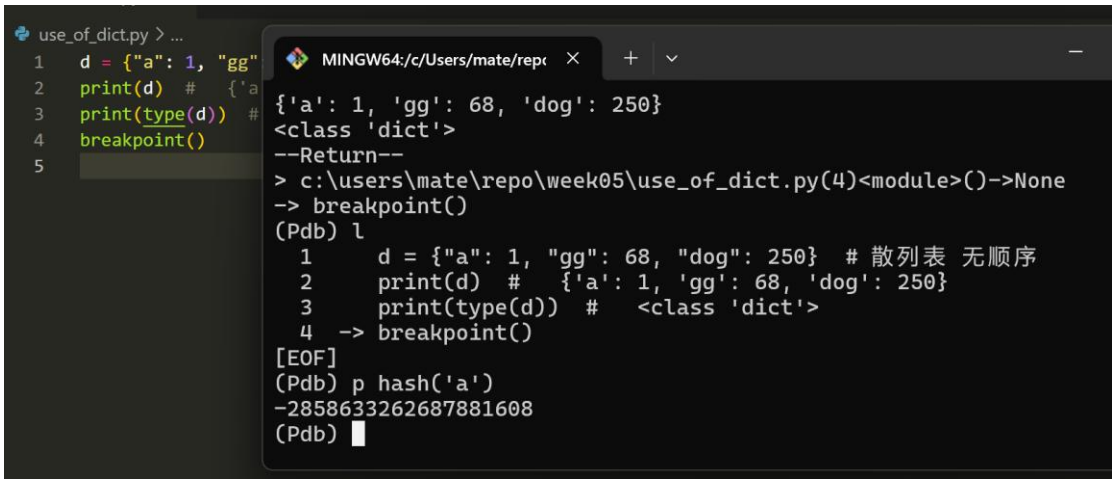

6
7 p = Path("D:\\ananconda\\envs\\week05\\python.exe")
8 s = p.read_bytes()
9 print(len(s)) # 93184
0 breakpoint()
1
MINGW64:/c/U
MINGW64:/c/Us
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x00'
(Pdb) p len(s)
93184
(Pdb)



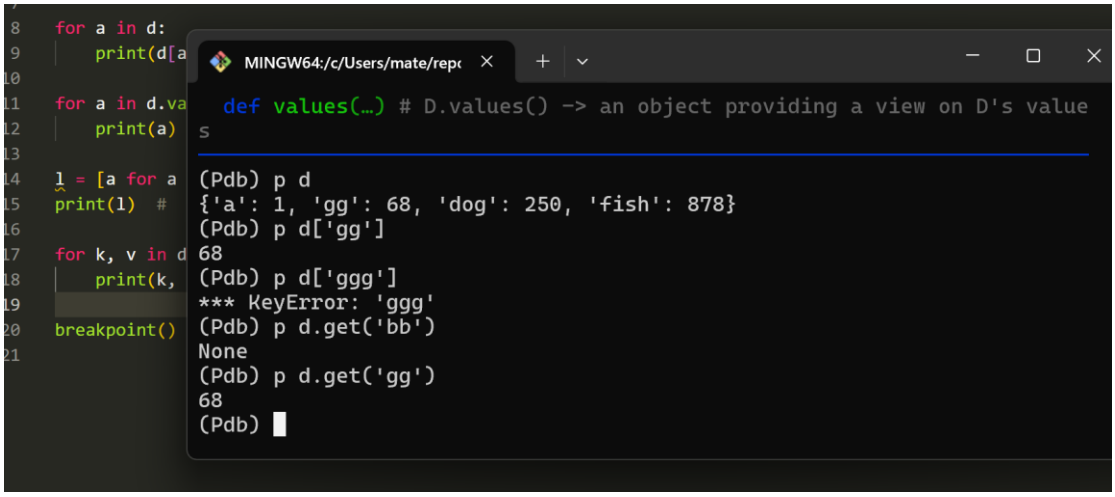


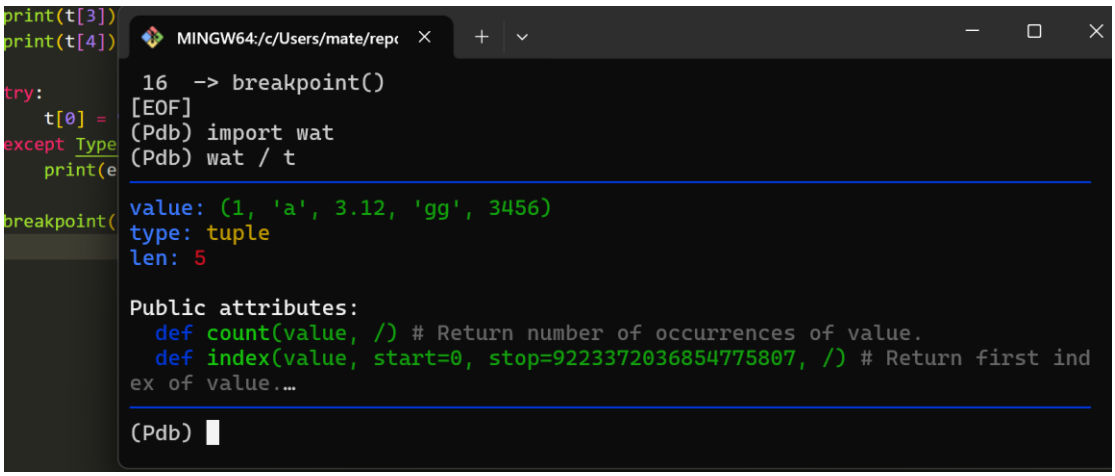
整数不循环迭代，不能返回长度，不能索取

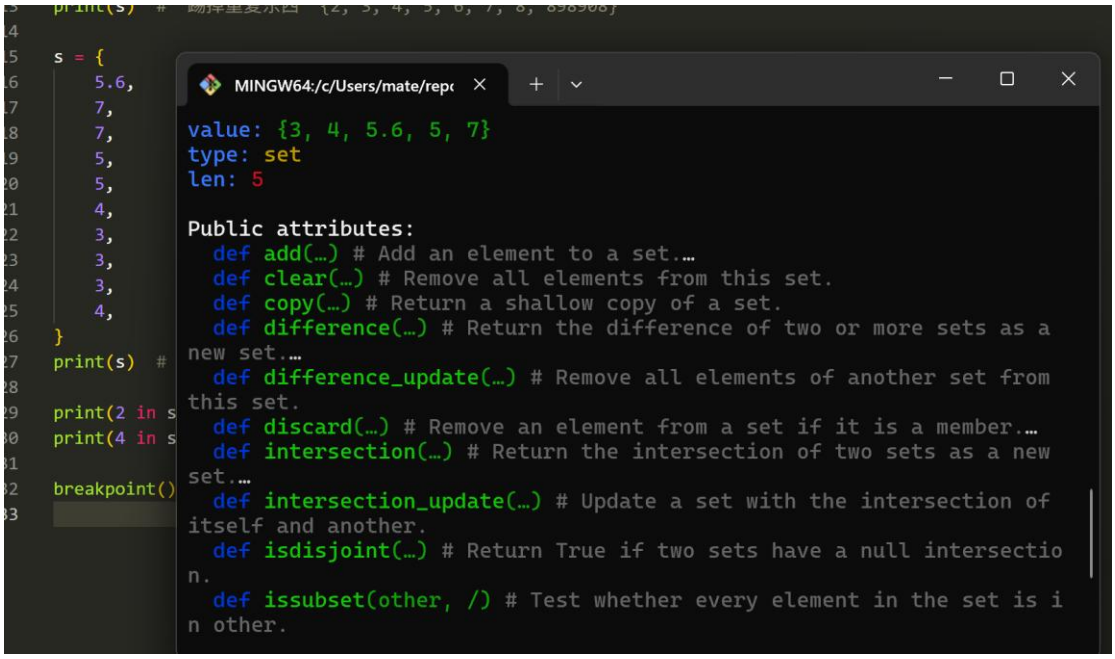


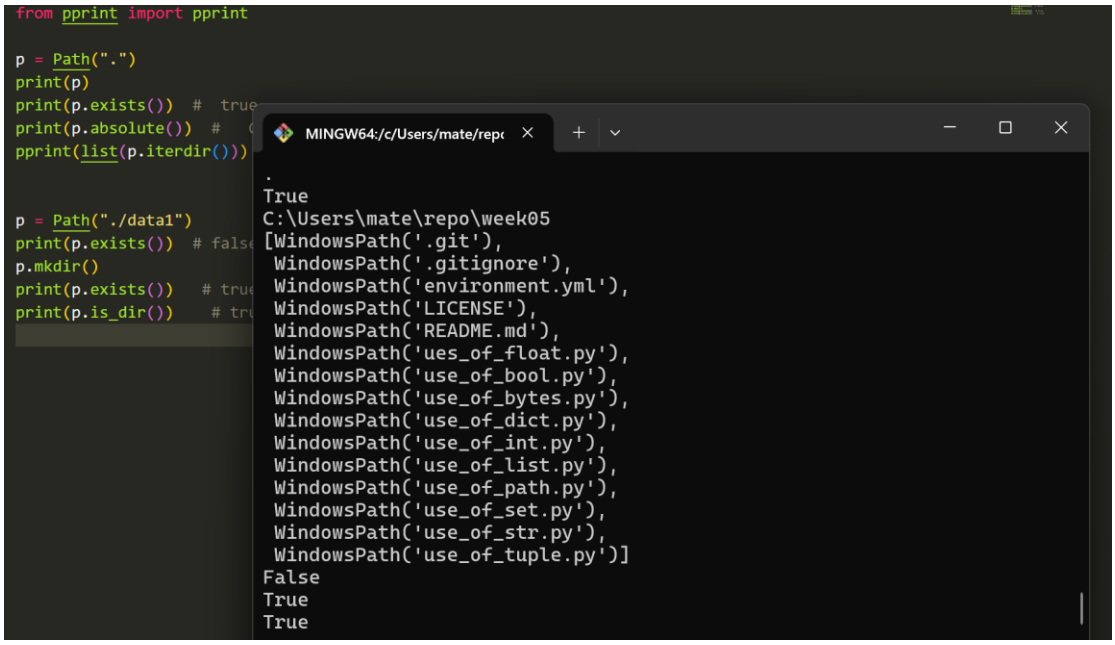


哈希值


for a in d:
print(d[a
for a in d.va
print(a)
l = [a for a
print(l) #
for k, v in d
print(k,
breakpoint()
MINGW64:/c/Users/mate/repo
def values(…) # D.values() -> an object providing a view on D's value
s
(Pdb) p d
{'a': 1, 'gg': 68, 'dog': 250, 'fish': 878}
(Pdb) p d['gg']
68
(Pdb) p d['ggg']
*** KeyError: 'ggg'
(Pdb) p d.get('bb')
None
(Pdb) p d.get('gg')
68
(Pdb)

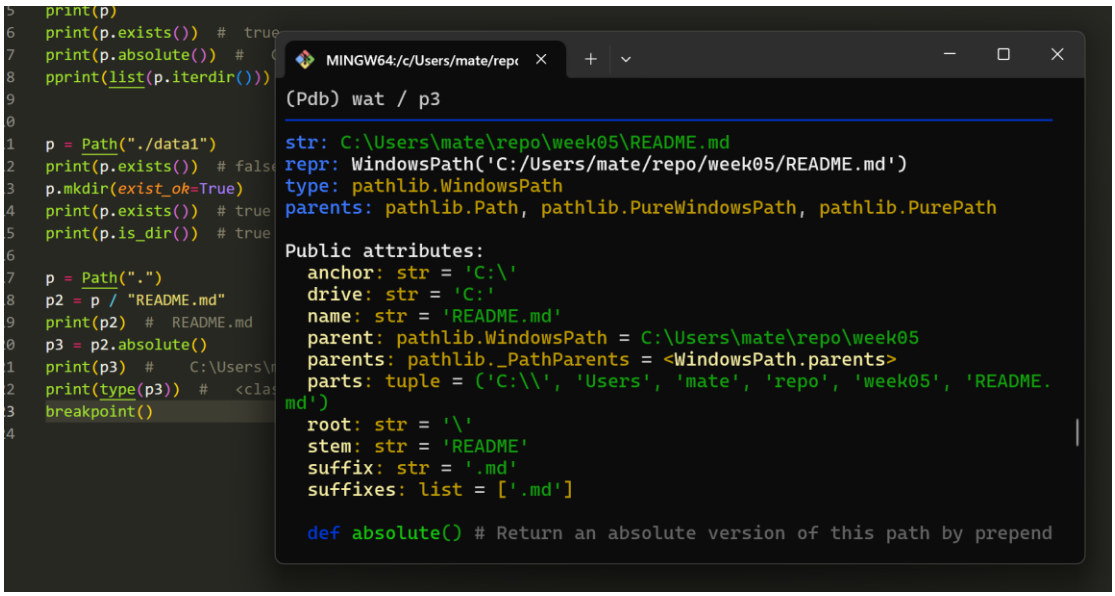

print(t[3])
print(t[4])
try:
t[0] =
except Type
print(e
breakpoint(
MINGW64:/c/Users/mate/repo
16 -> breakpoint()
[EOF]
(Pdb) import wat
(Pdb) wat / t
value: (1, 'a', 3.12, 'gg', 3456)
type: tuple
len: 5
Public attributes:
def count(value, /) # Return number of occurrences of value.
def index(value, start=0, stop=9223372036854775807, /) # Return first ind
ex of value.…
(Pdb)


s = {
5.6,
7,
7,
5,
5,
4,
3,
3,
3,
4,
}
print(s) #
print(2 in s
print(4 in s
breakpoint()
MINGW64:/c/Users/mate/repo
value: {3, 4, 5.6, 5, 7}
type: set
len: 5
Public attributes:
def add(…) # Add an element to a set.…
def clear(…) # Remove all elements from this set.
def copy(…) # Return a shallow copy of a set.
def difference(…) # Return the difference of two or more sets as a
new set.…
def difference_update(…) # Remove all elements of another set from
this set.
def discard(…) # Remove an element from a set if it is a member.…
def intersection(…) # Return the intersection of two sets as a new
set.…
def intersection_update(…) # Update a set with the intersection of
itself and another.
def isdisjoint(…) # Return True if two sets have a null intersectio
n.
def issubset(other, /) # Test whether every element in the set is i
n other.



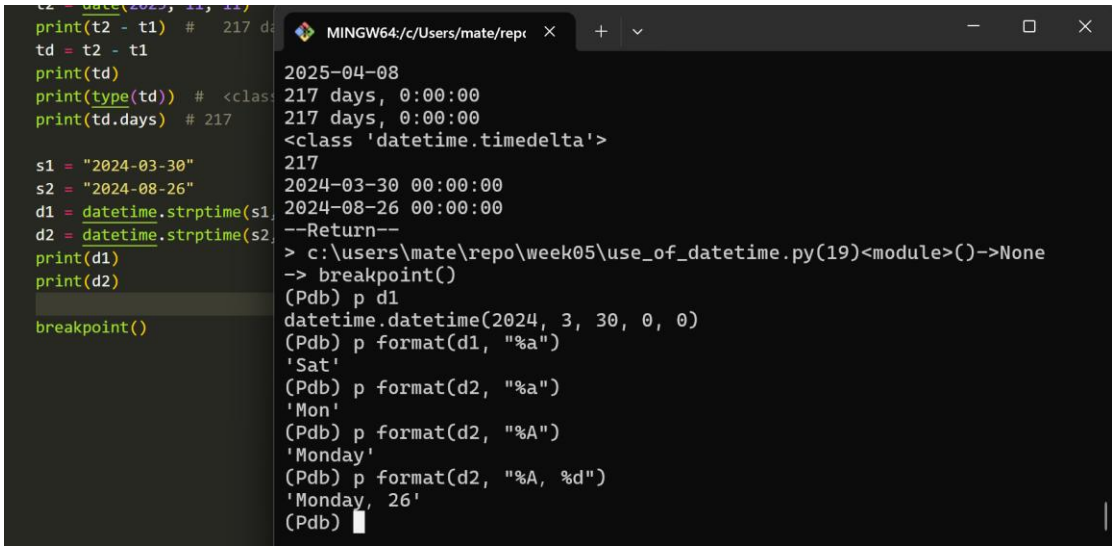
```
from pprint import pprint

p = Path(".")
print(p)
print(p.exists())  #  true
print(p.absolute())  #
pprint(list(p.iterdir()))


p = Path("./data1")
print(p.exists())  # false
p.mkdir()
print(p.exists())  # true
print(p.is_dir())  # tr
```

```
MINGW64:/c/Users/mate/repo
.
True
C:\Users\mate\repo\week05
[WindowsPath('.git'),
 WindowsPath('.gitignore'),
 WindowsPath('environment.yml'),
 WindowsPath('LICENSE'),
 WindowsPath('README.md'),
 WindowsPath('ues_of_float.py'),
 WindowsPath('use_of_bool.py'),
 WindowsPath('use_of_bytes.py'),
 WindowsPath('use_of_dict.py'),
 WindowsPath('use_of_int.py'),
 WindowsPath('use_of_list.py'),
 WindowsPath('use_of_path.py'),
 WindowsPath('use_of_set.py'),
 WindowsPath('use_of_str.py'),
 WindowsPath('use_of_tuple.py')]
False
True
True
```



```
print(p)
print(p.exists())  #  true
print(p.absolute())  #
pprint(list(p.iterdir()))


p = Path("./data1")
print(p.exists())  # false
p.mkdir(exist_ok=True)
print(p.exists())  # true
print(p.is_dir())  # true

p = Path(".")
p2 = p / "README.md"
print(p2)  #  README.md
p3 = p2.absolute()
print(p3)  #    C:\Users\
print(type(p3))  #   <cla
breakpoint()
```

```
MINGW64:/c/Users/mate/repo
(Pdb) wat / p3
str: C:\Users\mate\repo\week05\README.md
repr: WindowsPath('C:/Users/mate/repo/week05/README.md')
type: pathlib.WindowsPath
parents: pathlib.Path, pathlib.PureWindowsPath, pathlib.PurePath

Public attributes:
  anchor: str = 'C:\'
  drive: str = 'C:'
  name: str = 'README.md'
  parent: pathlib.WindowsPath = C:\Users\mate\repo\week05
  parents: pathlib._PathParents = <WindowsPath.parents>
  parts: tuple = ('C:\\', 'Users', 'mate', 'repo', 'week05', 'README.
md')
  root: str = '\'
  stem: str = 'README'
  suffix: str = '.md'
  suffixes: list = ['.md']

  def absolute() # Return an absolute version of this path by prepend
```



```
print(t2 - t1)  #   217 da
td = t2 - t1
print(td)
print(type(td))  #  <class
print(td.days)  # 217

s1 = "2024-03-30"
s2 = "2024-08-26"
d1 = datetime.strptime(s1,
d2 = datetime.strptime(s2,
print(d1)
print(d2)

breakpoint()
```

```
MINGW64:/c/Users/mate/repo
2025-04-08
217 days, 0:00:00
217 days, 0:00:00
<class 'datetime.timedelta'>
217
2024-03-30 00:00:00
2024-08-26 00:00:00
--Return--
> c:\users\mate\repo\week05\use_of_datetime.py(19)<module>()->None
-> breakpoint()
(Pdb) p d1
datetime.datetime(2024, 3, 30, 0, 0)
(Pdb) p format(d1, "%a")
'Sat'
(Pdb) p format(d2, "%a")
'Mon'
(Pdb) p format(d2, "%A")
'Monday'
(Pdb) p format(d2, "%A, %d")
'Monday, 26'
(Pdb)
```