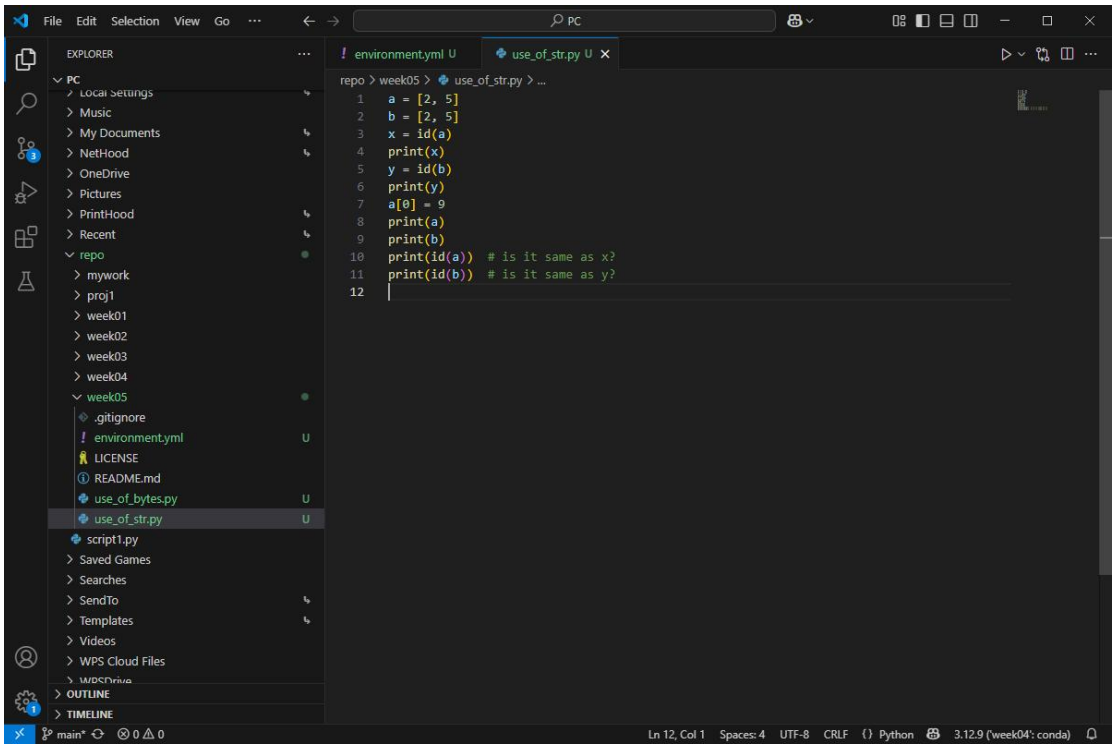
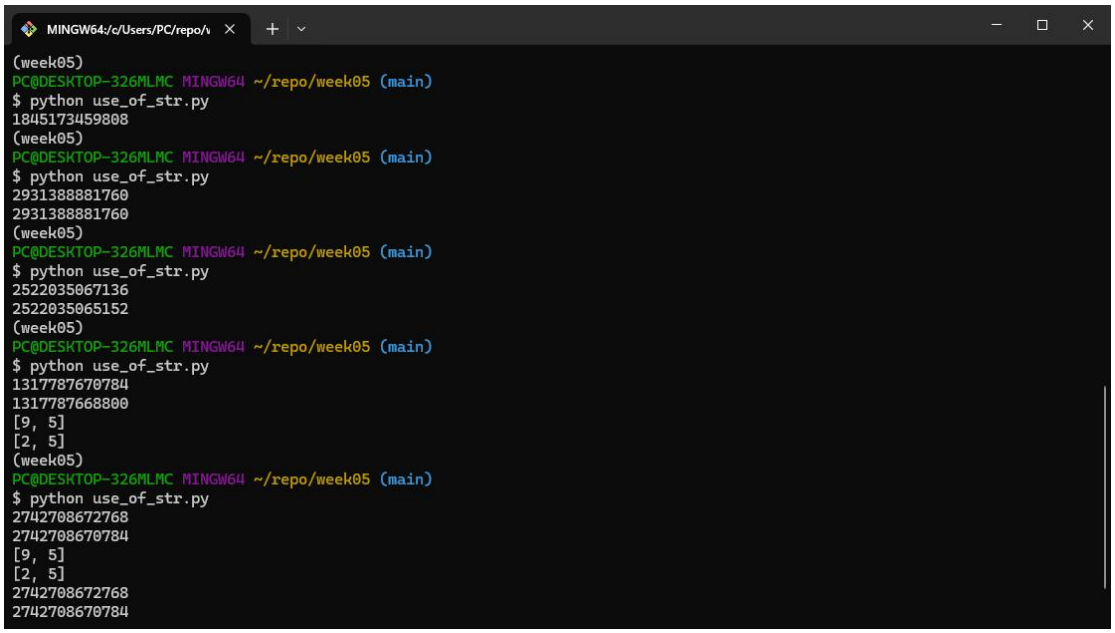
1. id() -- 返回对象在虚拟内存中的地址（正整数），如果 id(a) == id(b)，那么 a is b



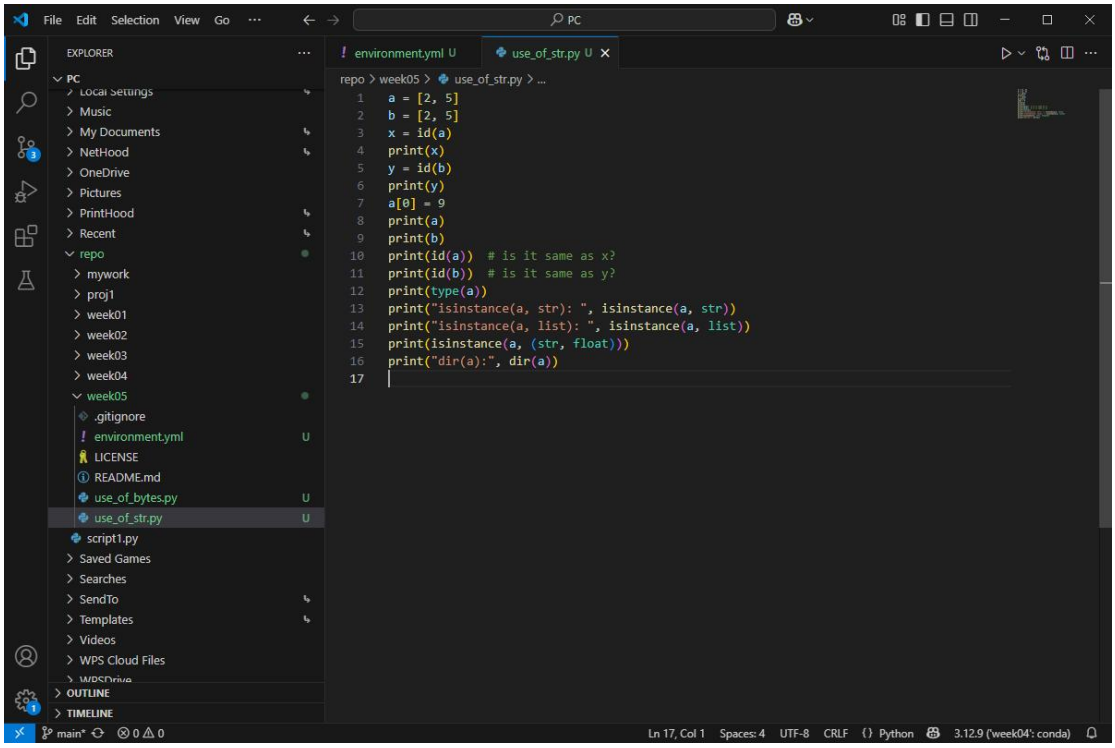


2. type() -- 返回对象的类型

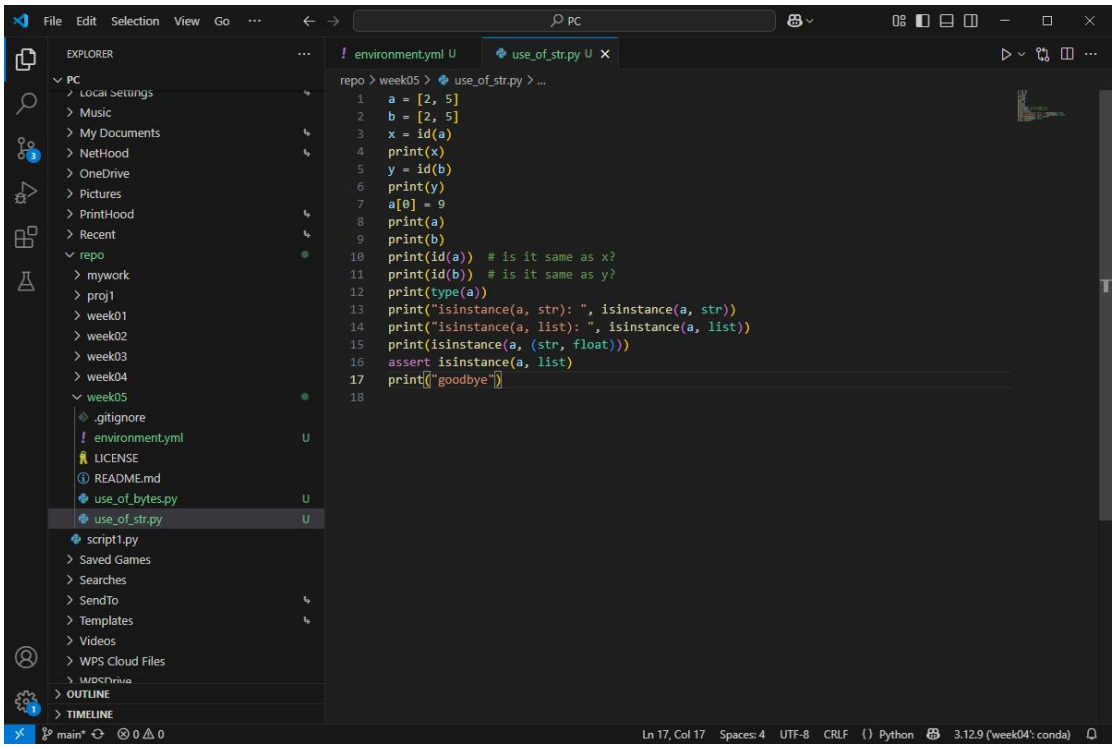
isinstance() -- 判断对象是否属于某个(或某些)类型

dir() -- 返回对象所支持的属性的名称列表

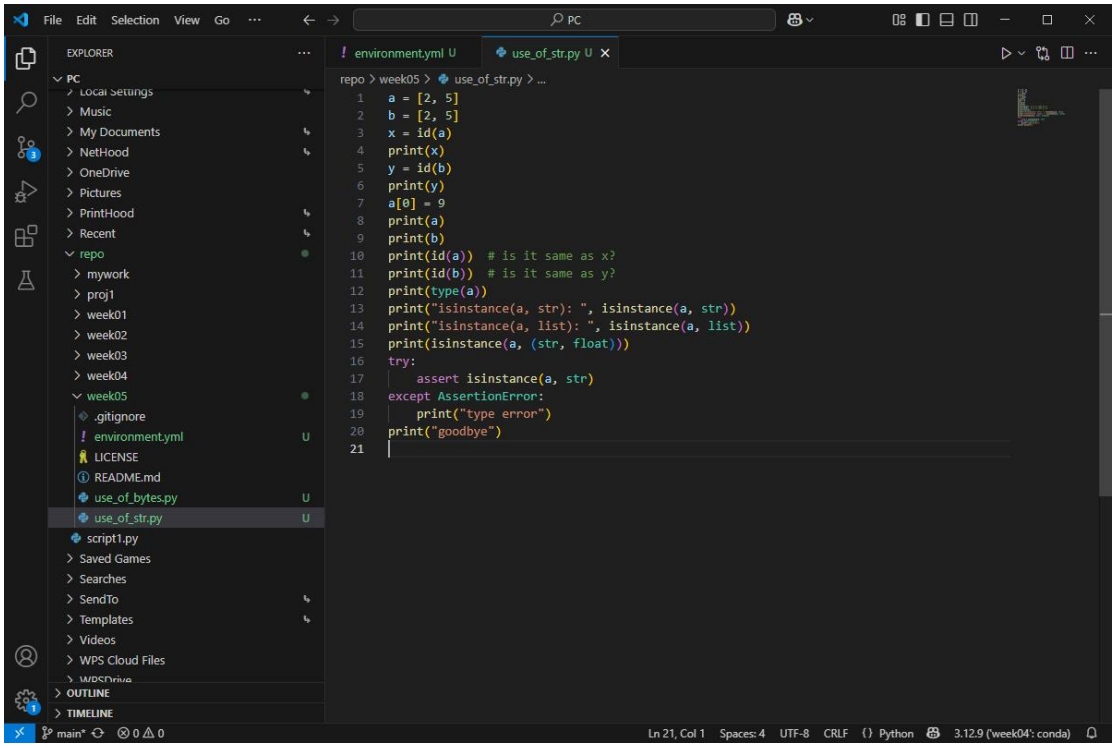
调用 print() 函数将表达式输出到终端，查看结果是否符合预期



3. 利用 assert 语句查验某个表达式为真，否则报错（AssertionError）退出



利用 try 语句拦截报错，避免退出，将流程转入 except 语句



```
MINGW64:/c/Users/PC/repo/w
$ python use_of_str.py
1517922949376
1517922947392
[9, 5]
[2, 5]
1517922949376
1517922947392
<class 'list'>
isinstance(a, str):  False
isinstance(a, list):  True
False
Traceback (most recent call last):
  File "C:\Users\PC\repo\week05\use_of_str.py", line 16, in <module>
    assert isinstance(a, str)
           ^^^^^^^^^^^^^^^^^^
AssertionError
(week05)
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
1732525562112
1732525560128
[9, 5]
[2, 5]
1732525562112
1732525560128
<class 'list'>
isinstance(a, str):  False
isinstance(a, list):  True
False
goodbye
```

4. 调用 breakpoint() 函数暂停程序运行，进入 pdb 调试模式

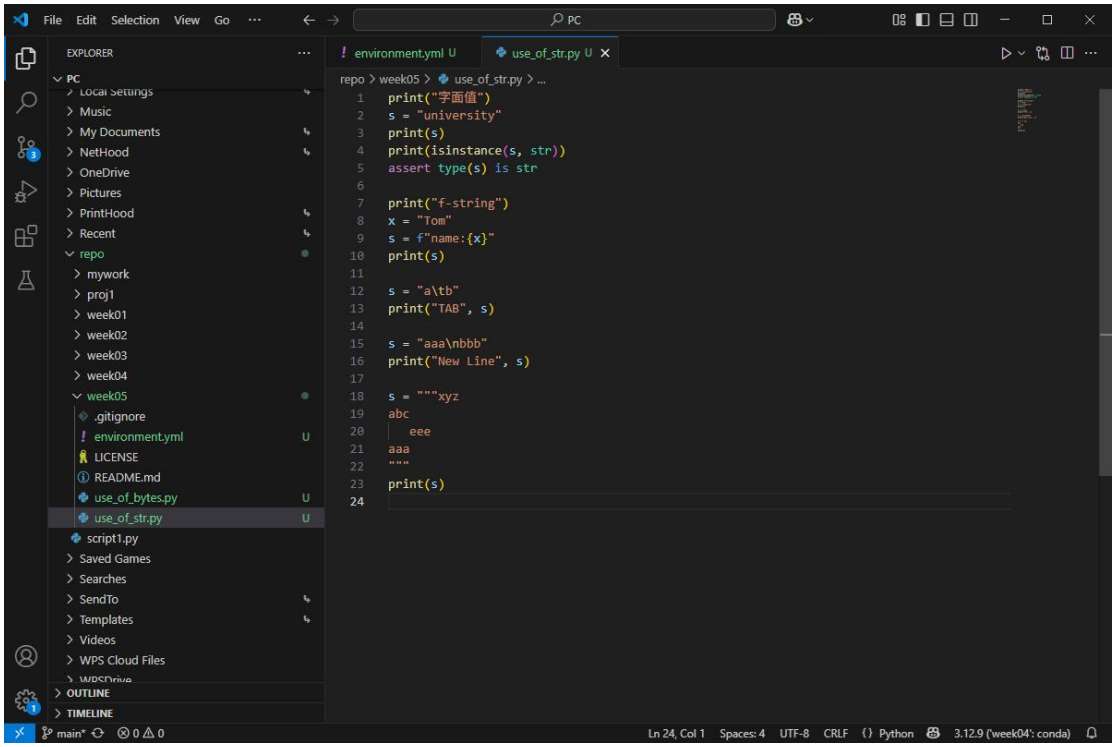
```
(week05)
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2621753727232
2621753725248
[9, 5]
[2, 5]
2621753727232
2621753725248
<class 'list'>
isinstance(a, str):  False
isinstance(a, list):  True
False
> c:\users\pc\repo\week05\use_of_str.py(20)<module>()
-> print("type error")
(Pdb) l .
 15     print(isinstance(a, (str, float)))
 16     try:
 17         assert isinstance(a, str)
 18     except AssertionError:
 19         breakpoint()
 20  ->     print("type error")
 21     print("goodbye")
[EOF]
(Pdb) p a
[9, 5]
(Pdb) p isinstance(a,str)
False
(Pdb)
```

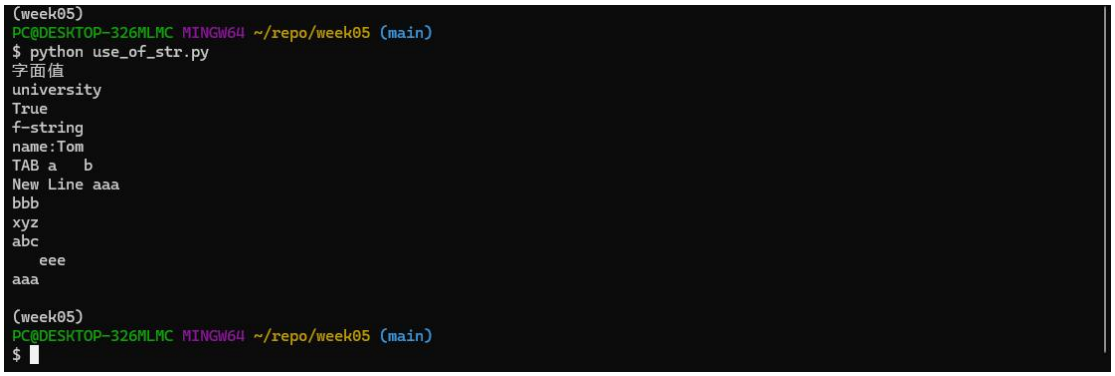
5. 字符串的表达式（expression）和实例（instance）

（1）字面值（Literal）是直接写在代码里的「固定值」，无需计算即可确定内容，包括 f-string 这种「动态字面值」，即运行时计算表达式。

基础字面值：123（整数）、3.14（浮点数）、'hello'（字符串）、True 、None

f-string 字面值：用 f"" 包裹，含 {表达式}，运行时动态渲染。

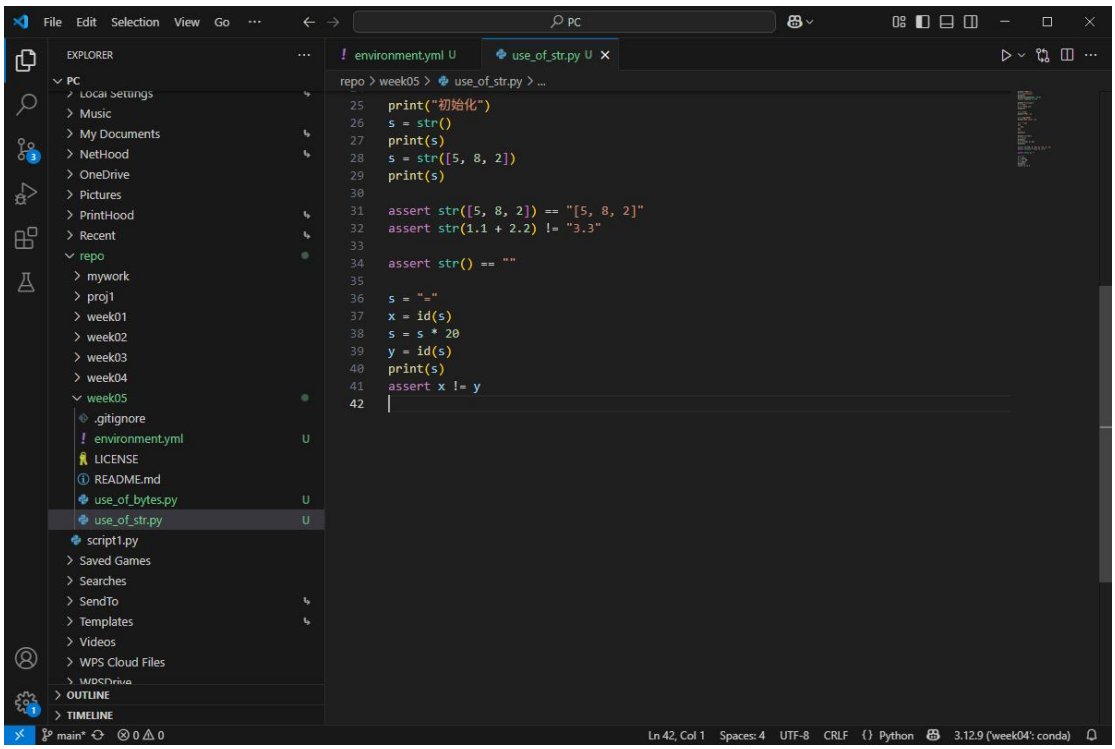


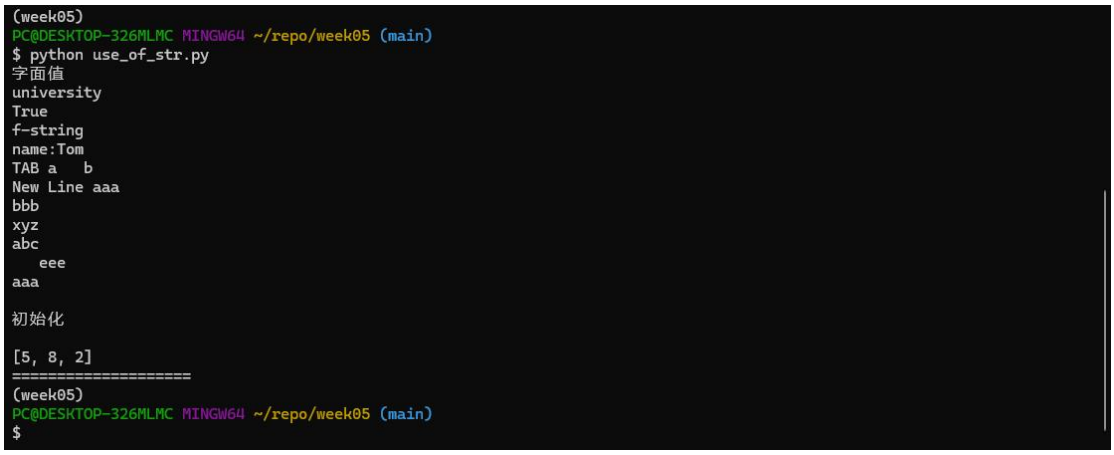


（2）推导式（Comprehension）是一行代码创建容器，替代 for 循环的冗长写法，仅限 list/dict/set 三种类型。

（3）初始化（Init）是创建对象并赋予初始值的过程，分「字面量初始化」和「构造函数初始化」。

（4）运算值（Operator）是运算符作用于操作数后得到的结果，分「算术运算」和「特殊运算」（如拼接、比较）。





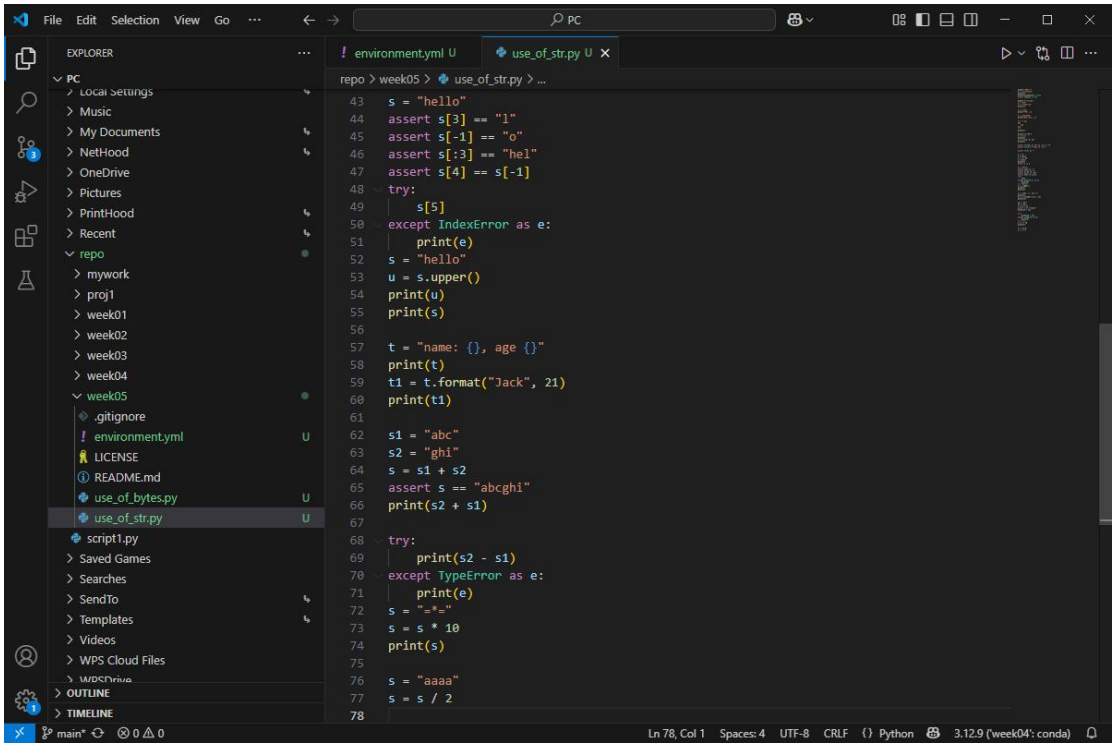
（5）索引值（Subscription）是通过［］运算符获取对象中的元素，支持「索引」和「切片」。

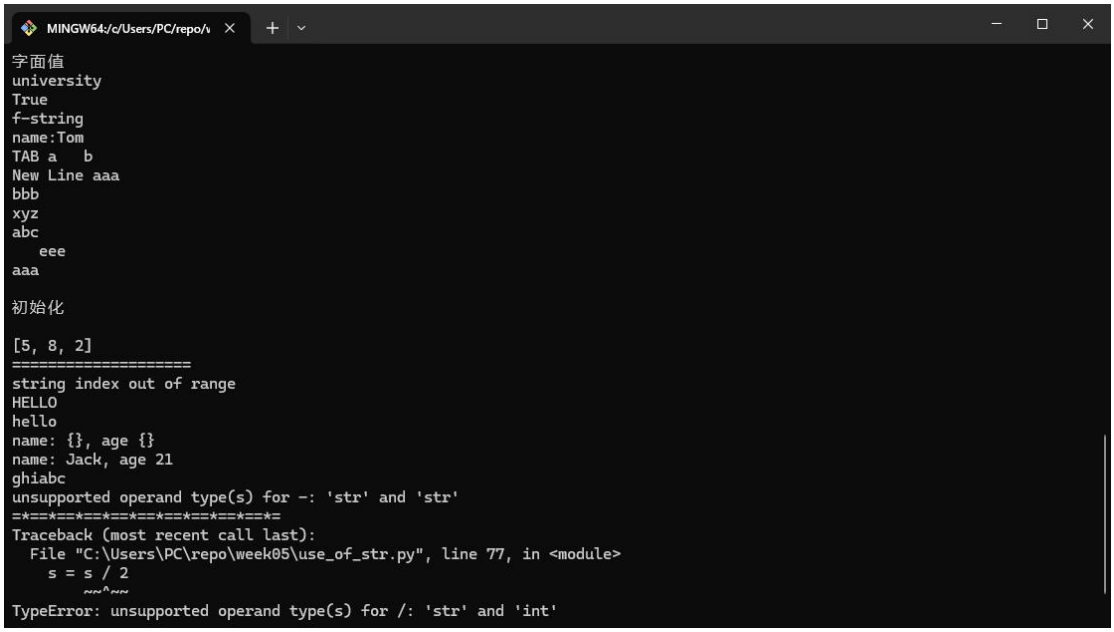
序列类型：列表、元组、字符串（按位置索引）

映射类型：字典（按键索引）

（6）返回值（Return Value）是函数或方法执行后返回的结果，用 return 语句指定，默认返回 None。

区分「修改原对象的方法」（如 list.append() 返回 None）和「纯计算方法」（如 str.strip() 返回新字符串）。





6. 字符串的属性

（1）数学运算符支持情况

+：加法运算符，用于数值相加或字符串拼接

-：减法运算符，用于数值相减

*：乘法运算符，用于数值相乘或字符串重复

/：除法运算符，执行普通除法，结果为浮点数

（2）判断相等，== 用于比较两个对象的值是否相等

（3）比较运算符支持情况

>、<、>=、<= 这些比较运算符，用于比较两个对象的大小关系

（4）在 Python 中，以下值被当作 False，其他值一般被当作 True：

False、None、数值 0（包括 0、0.0、0j）、空序列（如 ''、[]、()）、空映射（如 {}）、空集合（如 set()）

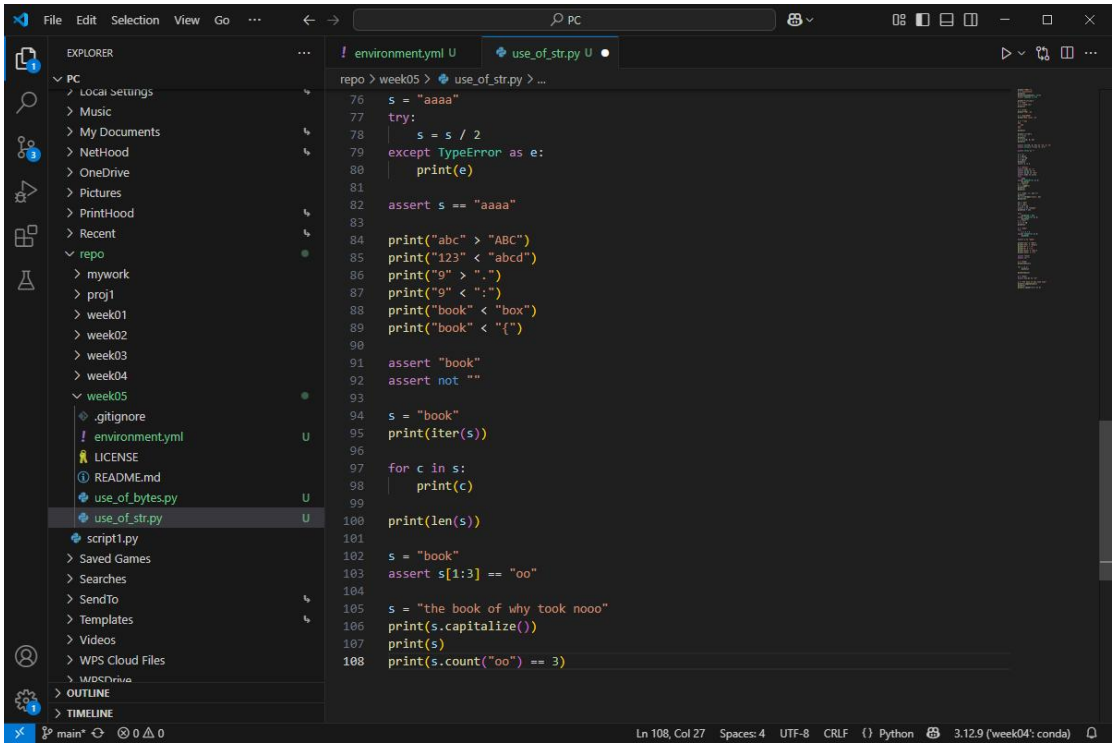
（5）可迭代对象与迭代

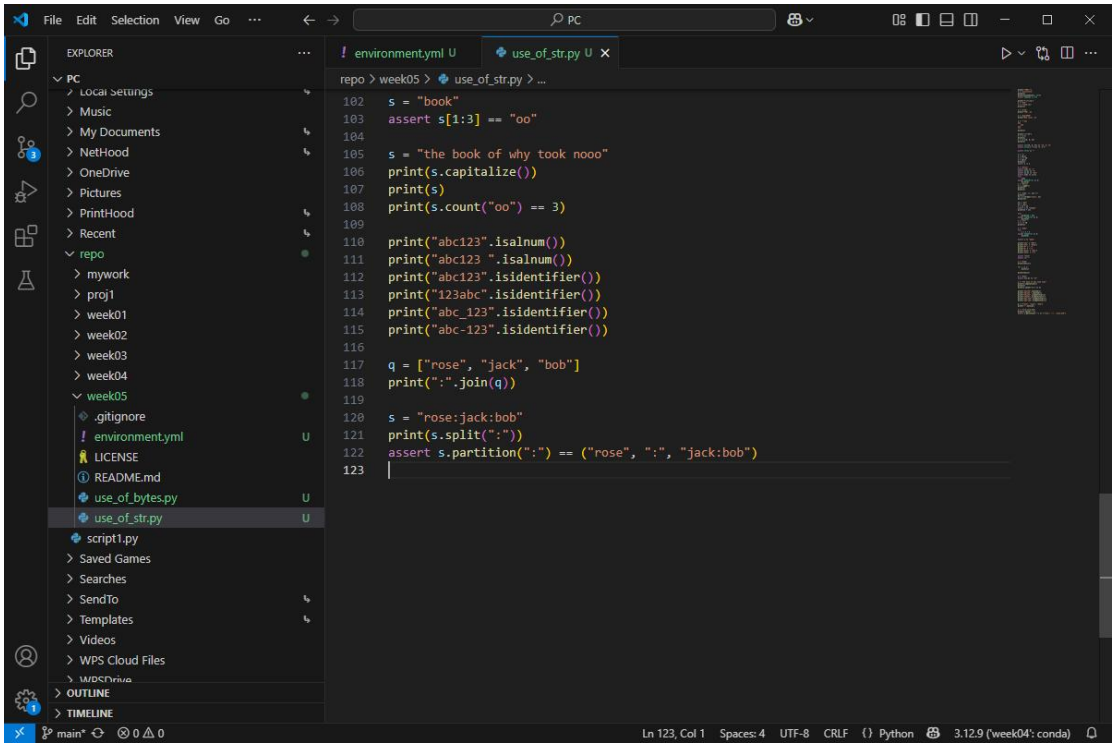
在 Python 中，可迭代对象是指实现了 iter() 方法的对象，像列表、元组、字符串、字典等都是可迭代对象，可以使用 for 循环对可迭代对象进行迭代。

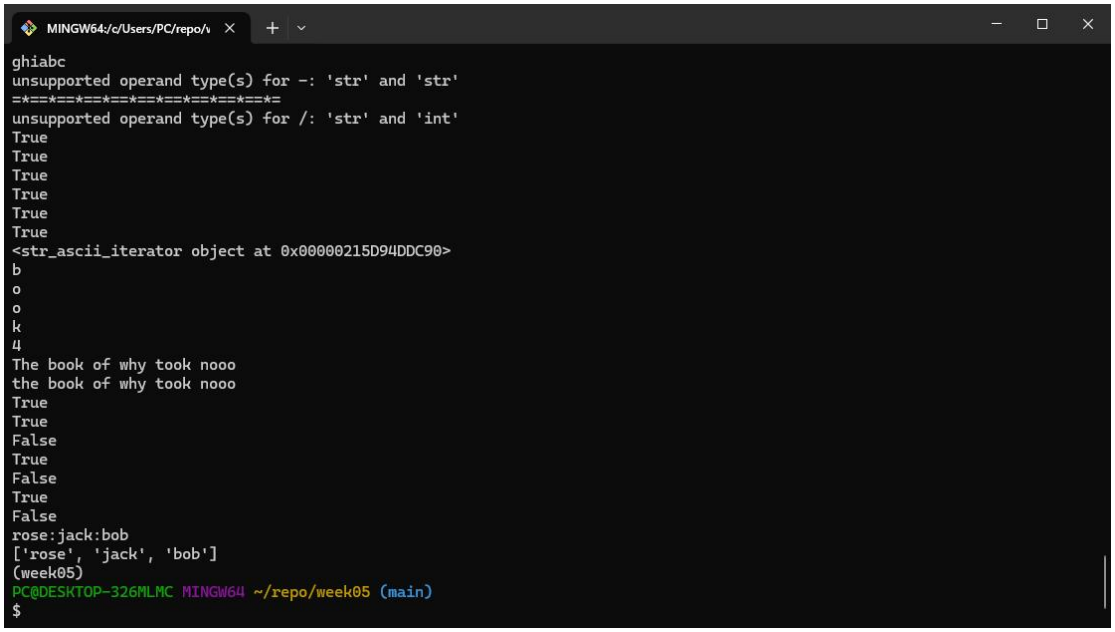
（6）Python 支持使用 len() 函数返回可迭代对象的长度

（7）索引操作

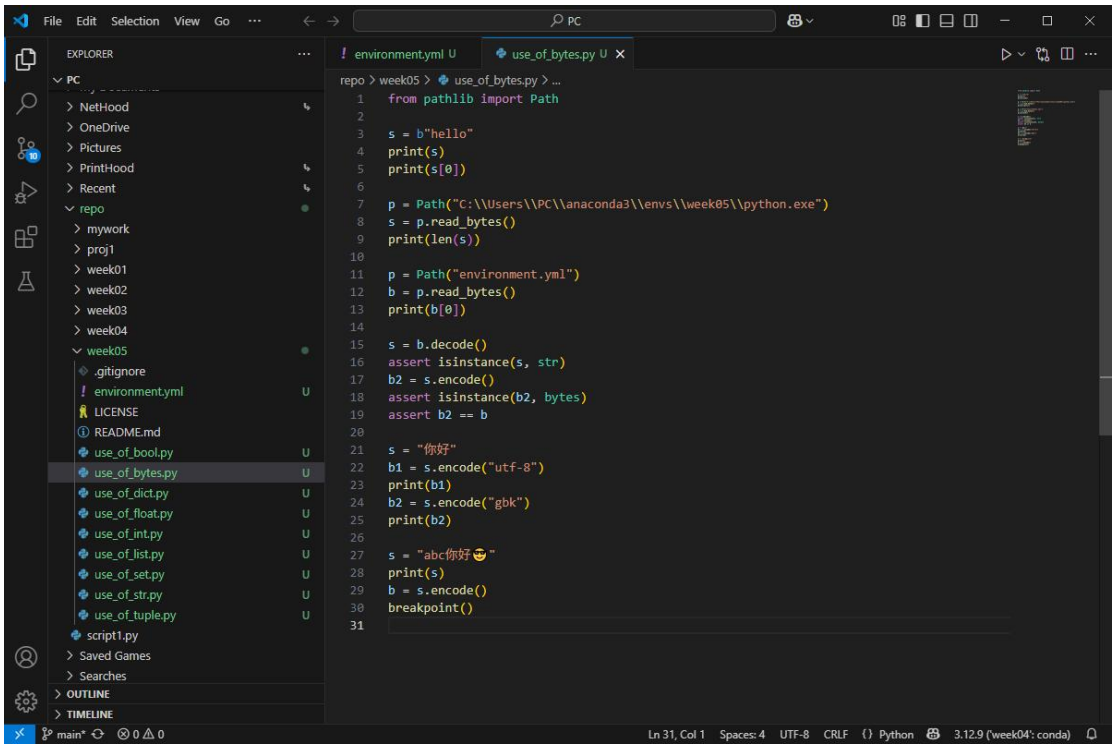
Python 支持使用 [] 运算符对序列（如列表、元组、字符串）和映射（如字典）进行索引操作。

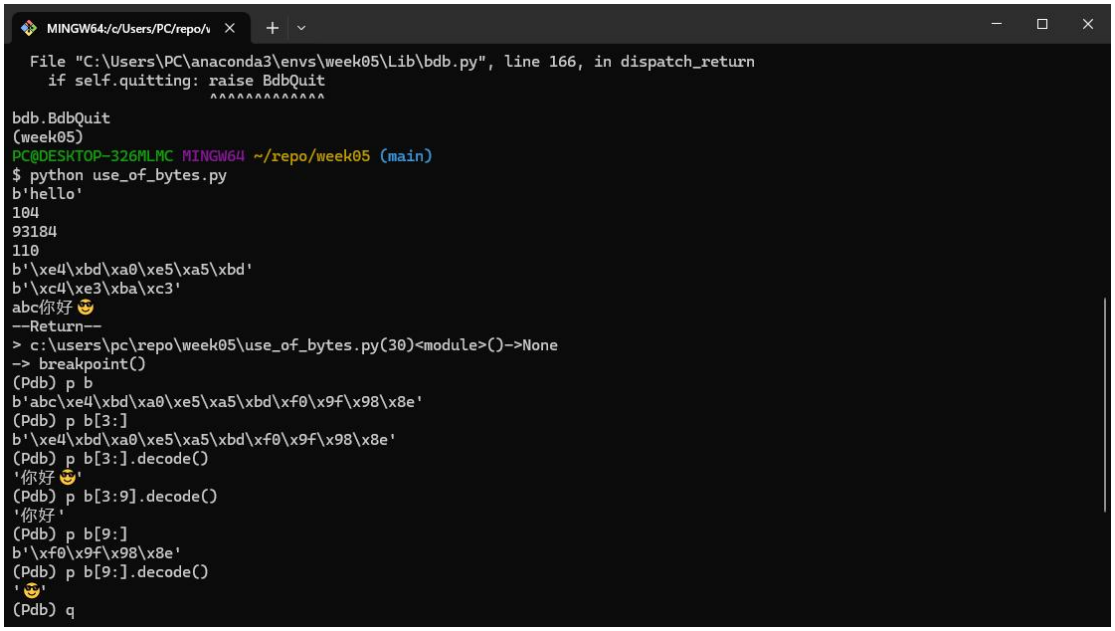



File Edit Selection View Go
PC
EXPLORER
environment.yml U
use_of_str.py U
repo > week05 > use_of_str.py > ...
Music
My Documents
NetHood
OneDrive
Pictures
PrintHood
Recent
repo
mywork
proj1
week01
week02
week03
week04
week05
.gitignore
environment.yml
LICENSE
README.md
use_of_bytes.py
use_of_str.py
script1.py
Saved Games
Searches
SendTo
Templates
Videos
WPS Cloud Files
OUTLINE
TIMELINE
s = "book"
assert s[1:3] == "oo"
s = "the book of why took nooo"
print(s.capitalize())
print(s)
print(s.count("oo") == 3)
print("abc123".isalnum())
print("abc123 ".isalnum())
print("abc123".isidentifier())
print("123abc".isidentifier())
print("abc_123".isidentifier())
print("abc-123".isidentifier())
q = ["rose", "jack", "bob"]
print(":".join(q))
s = "rose:jack:bob"
print(s.split(":"))
assert s.partition(":") == ("rose", ":", "jack:bob")
main*
Ln 123, Col 1
Spaces: 4
UTF-8
CRLF
Python
3.12.9 ('week04': conda)

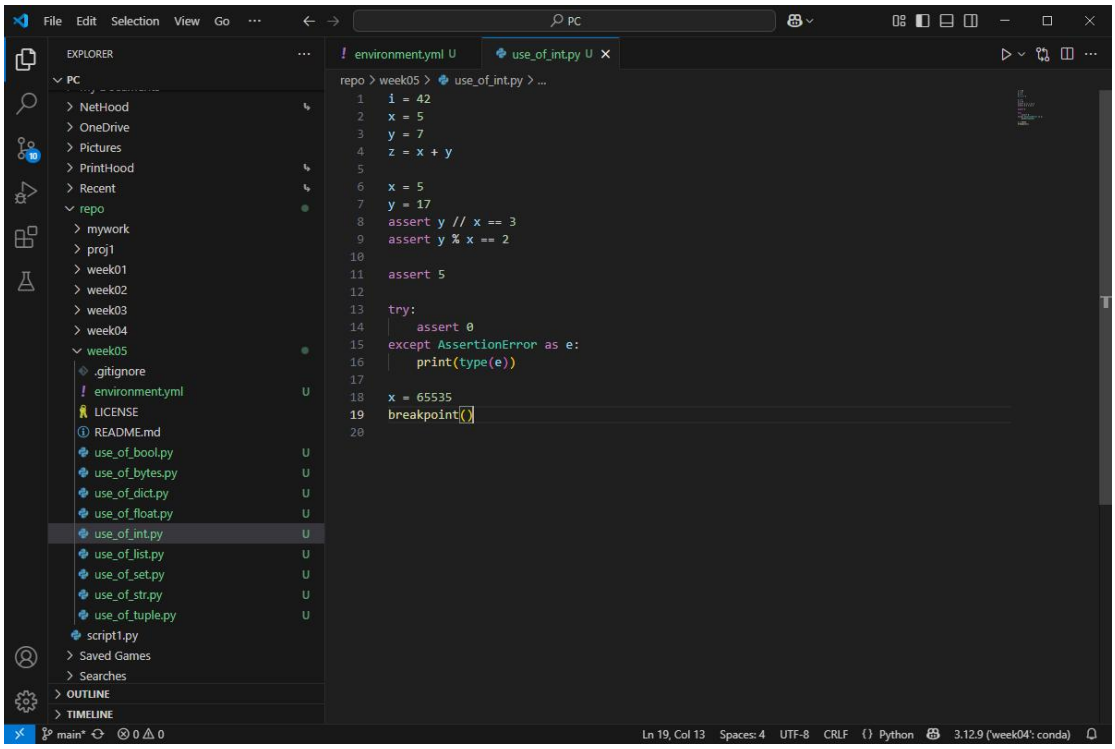

MINGW64:/c/Users/PC/repo/
ghiabc
unsupported operand type(s) for -: 'str' and 'str'
=*==========*==*==*==*==*==*==*
unsupported operand type(s) for /: 'str' and 'int'
True
True
True
True
True
True
<str_ascii_iterator object at 0x00000215D94DDC90>
b
o
o
k
4
The book of why took nooo
the book of why took nooo
True
True
False
True
False
True
False
rose:jack:bob
['rose', 'jack', 'bob']
(week05)
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$

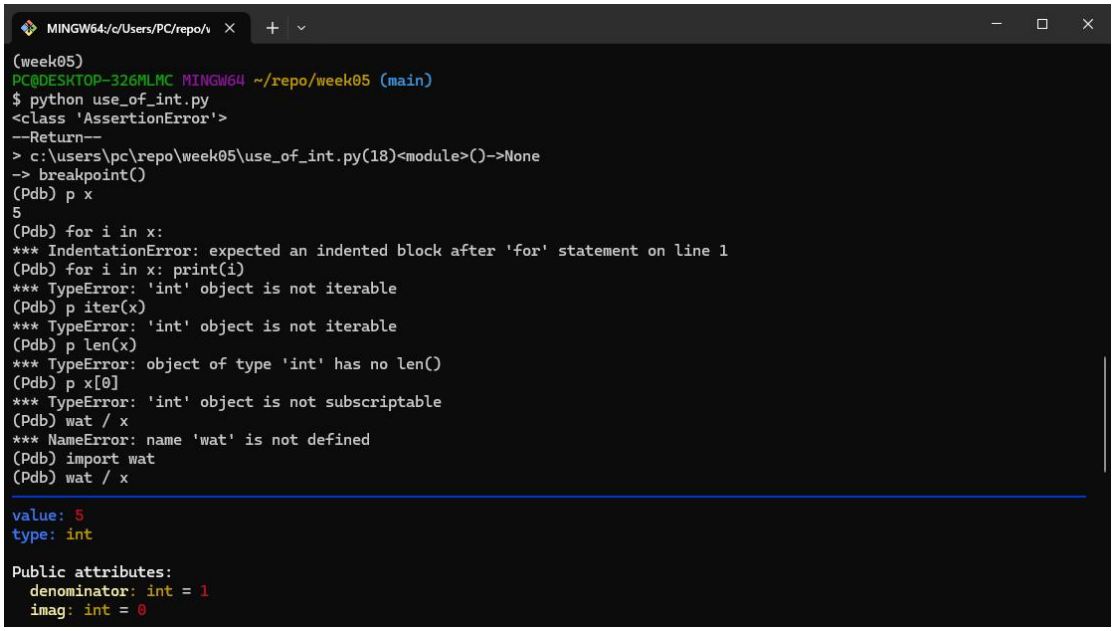
7. 字节串


File Edit Selection View Go
PC
EXPLORER
environment.yml U
use_of_bytes.py U
repo > week05 > use_of_bytes.py > ...
NetHood
OneDrive
Pictures
PrintHood
Recent
repo
mywork
proj1
week01
week02
week03
week04
week05
.gitignore
environment.yml
LICENSE
README.md
use_of_bool.py
use_of_bytes.py
use_of_dict.py
use_of_float.py
use_of_int.py
use_of_list.py
use_of_set.py
use_of_str.py
use_of_tuple.py
script1.py
Saved Games
Searches
OUTLINE
TIMELINE
from pathlib import Path
s = b"hello"
print(s)
print(s[0])
p = Path("C:\\Users\\PC\\anaconda3\\envs\\week05\\python.exe")
s = p.read_bytes()
print(len(s))
p = Path("environment.yml")
b = p.read_bytes()
print(b[0])
s = b.decode()
assert isinstance(s, str)
b2 = s.encode()
assert isinstance(b2, bytes)
assert b2 == b
s = "你好"
b1 = s.encode("utf-8")
print(b1)
b2 = s.encode("gbk")
print(b2)
s = "abc你好😊"
print(s)
b = s.encode()
breakpoint()
main*
Ln 31, Col 1
Spaces: 4
UTF-8
CRLF
Python
3.12.9 ('week04': conda)

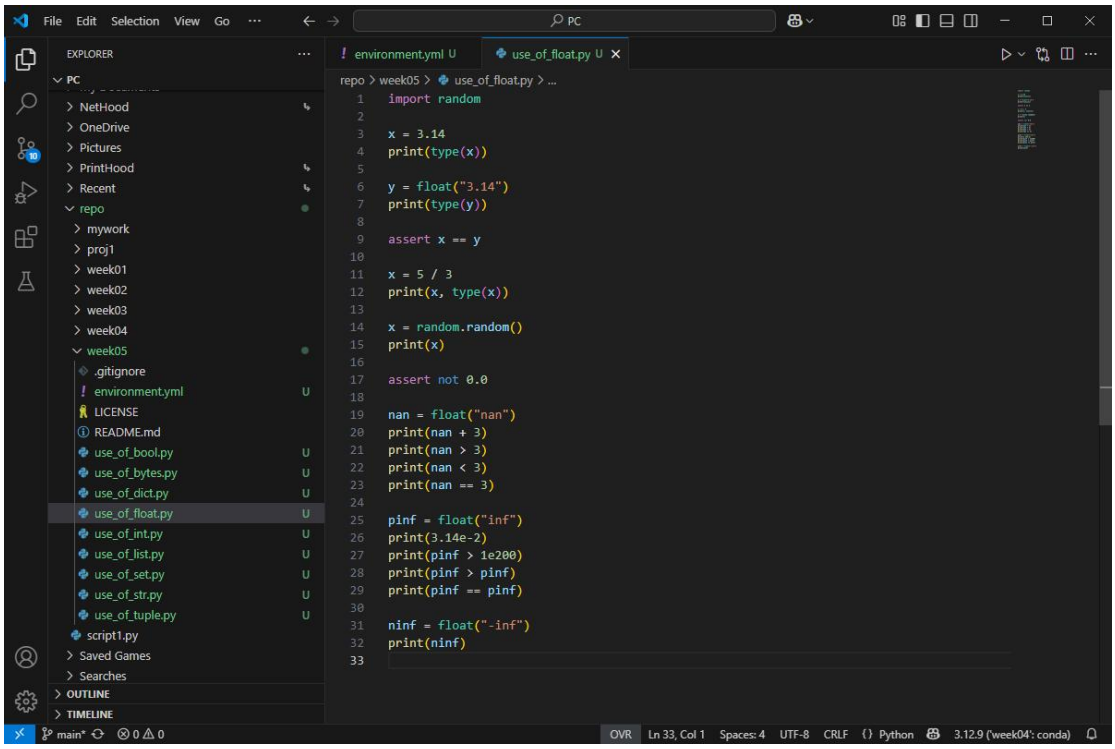

MINGW64:/c/Users/PC/repo/
File "C:\Users\PC\anaconda3\envs\week05\Lib\bdb.py", line 166, in dispatch_return
if self.quitting: raise BdbQuit
^^^^^^^^^^^^^
bdb.BdbQuit
(week05)
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$ python use_of_bytes.py
b'hello'
104
93184
110
b'\xe4\xbd\xa0\xe5\xa5\xbd'
b'\xc4\xe3\xba\xc3'
abc你好😊
--Return--
> c:\users\pc\repo\week05\use_of_bytes.py(30)<module>()->None
-> breakpoint()
(Pdb) p b
b'abc\xe4\xbd\xa0\xe5\xa5\xbd\xf0\x9f\x98\x8e'
(Pdb) p b[3:]
b'\xe4\xbd\xa0\xe5\xa5\xbd\xf0\x9f\x98\x8e'
(Pdb) p b[3:].decode()
'你好😊'
(Pdb) p b[3:9].decode()
'你好'
(Pdb) p b[9:]
b'\xf0\x9f\x98\x8e'
(Pdb) p b[9:].decode()
'😊'
(Pdb) q

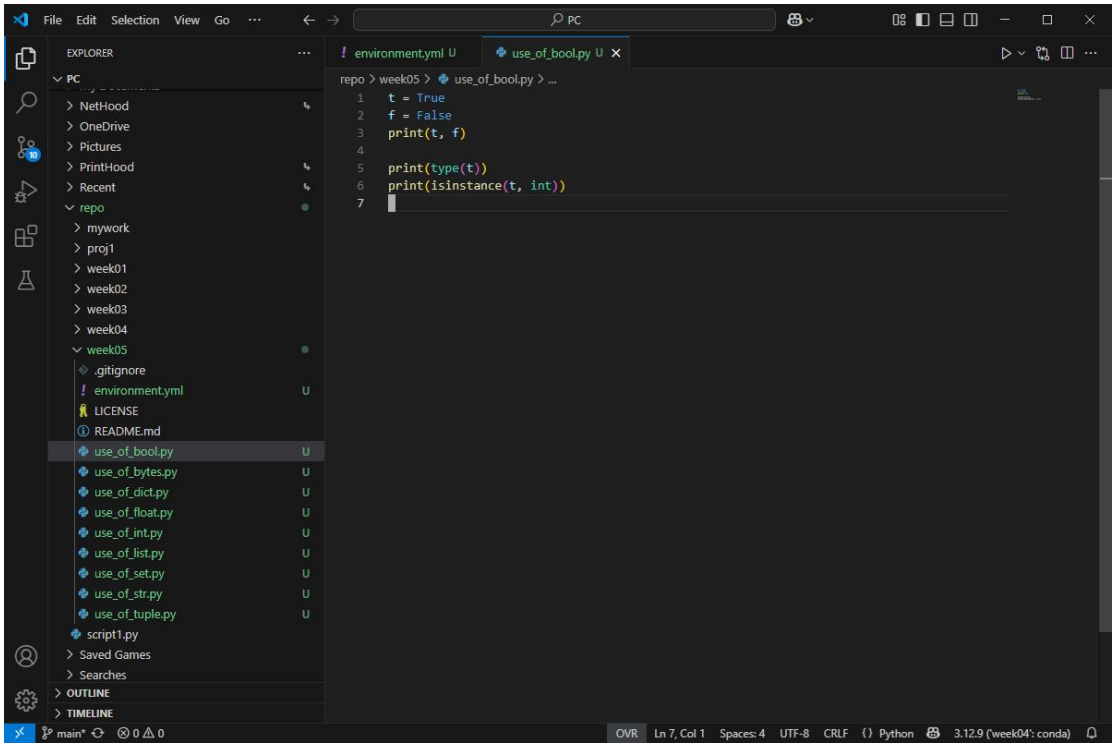
8. 整数


File Edit Selection View Go
PC
EXPLORER
environment.yml U
use_of_int.py U
repo > week05 > use_of_int.py > ...
NetHood
OneDrive
Pictures
PrintHood
Recent
repo
mywork
proj1
week01
week02
week03
week04
week05
.gitignore
environment.yml
LICENSE
README.md
use_of_bool.py
use_of_bytes.py
use_of_dict.py
use_of_float.py
use_of_int.py
use_of_list.py
use_of_set.py
use_of_str.py
use_of_tuple.py
script1.py
Saved Games
Searches
OUTLINE
TIMELINE
i = 42
x = 5
y = 7
z = x + y
x = 5
y = 17
assert y // x == 3
assert y % x == 2
assert 5
try:
assert 0
except AssertionError as e:
print(type(e))
x = 65535
breakpoint()
main*
Ln 19, Col 13
Spaces: 4
UTF-8
CRLF
Python
3.12.9 ('week04': conda)


MINGW64:/c/Users/PC/repo/
(week05)
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$ python use_of_int.py
<class 'AssertionError'>
--Return--
> c:\users\pc\repo\week05\use_of_int.py(18)<module>()->None
-> breakpoint()
(Pdb) p x
5
(Pdb) for i in x:
*** IndentationError: expected an indented block after 'for' statement on line 1
(Pdb) for i in x: print(i)
*** TypeError: 'int' object is not iterable
(Pdb) p iter(x)
*** TypeError: 'int' object is not iterable
(Pdb) p len(x)
*** TypeError: object of type 'int' has no len()
(Pdb) p x[0]
*** TypeError: 'int' object is not subscriptable
(Pdb) wat / x
*** NameError: name 'wat' is not defined
(Pdb) import wat
(Pdb) wat / x
value: 5
type: int
Public attributes:
denominator: int = 1
imag: int = 0

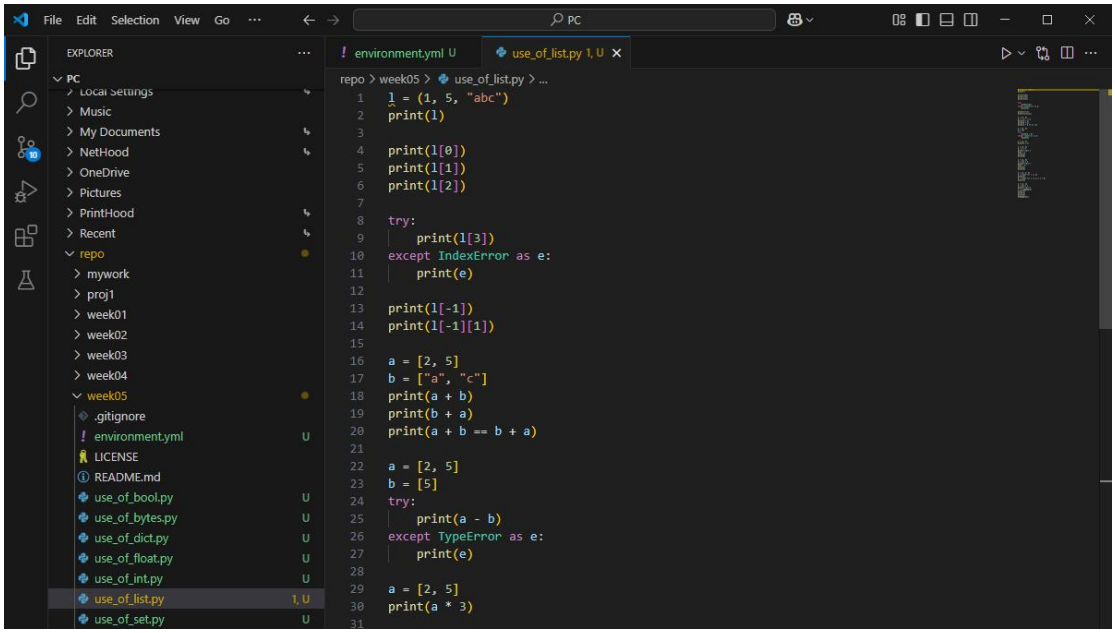
9. 浮点数和布尔值

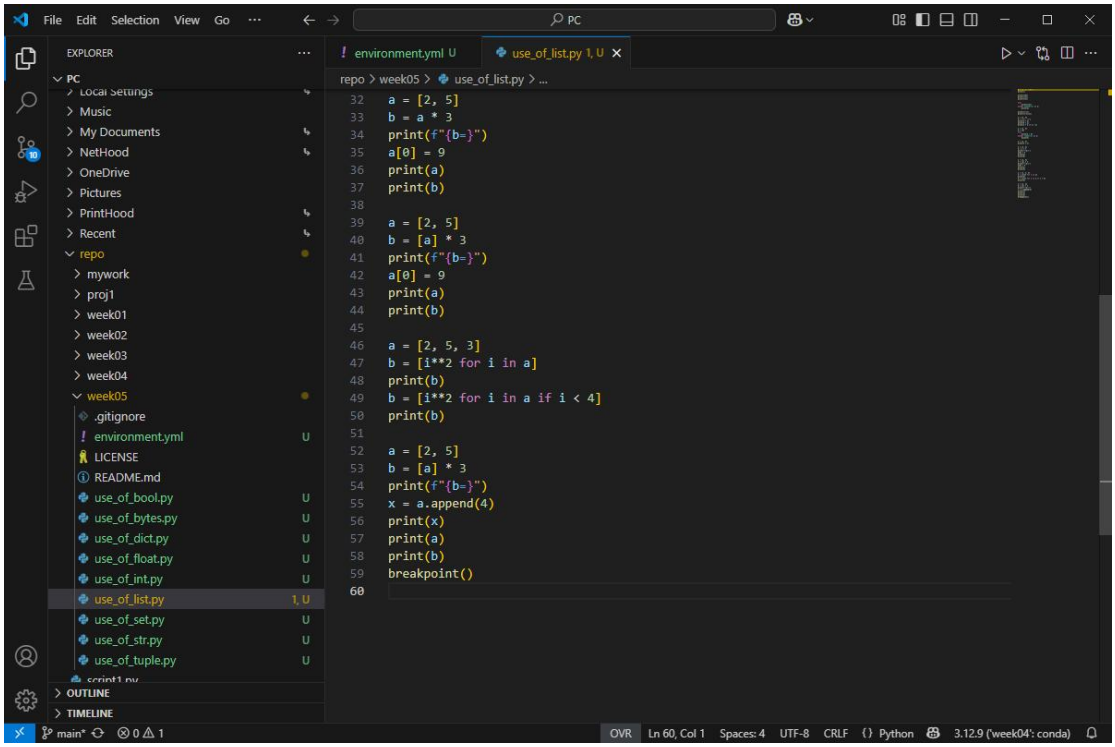

File Edit Selection View Go
PC
EXPLORER
environment.yml U
use_of_float.py U
repo > week05 > use_of_float.py > ...
import random
x = 3.14
print(type(x))
y = float("3.14")
print(type(y))
assert x == y
x = 5 / 3
print(x, type(x))
x = random.random()
print(x)
assert not 0.0
nan = float("nan")
print(nan + 3)
print(nan > 3)
print(nan < 3)
print(nan == 3)
pinf = float("inf")
print(3.14e-2)
print(pinf > 1e200)
print(pinf > pinf)
print(pinf == pinf)
ninf = float("-inf")
print(ninf)
NetHood
OneDrive
Pictures
PrintHood
Recent
repo
mywork
proj1
week01
week02
week03
week04
week05
.gitignore
environment.yml
LICENSE
README.md
use_of_bool.py
use_of_bytes.py
use_of_dict.py
use_of_float.py
use_of_int.py
use_of_list.py
use_of_set.py
use_of_str.py
use_of_tuple.py
script1.py
Saved Games
Searches
OUTLINE
TIMELINE
main*
OVR Ln 33, Col 1 Spaces: 4 UTF-8 CRLF Python 3.12.9 ('week04': conda)

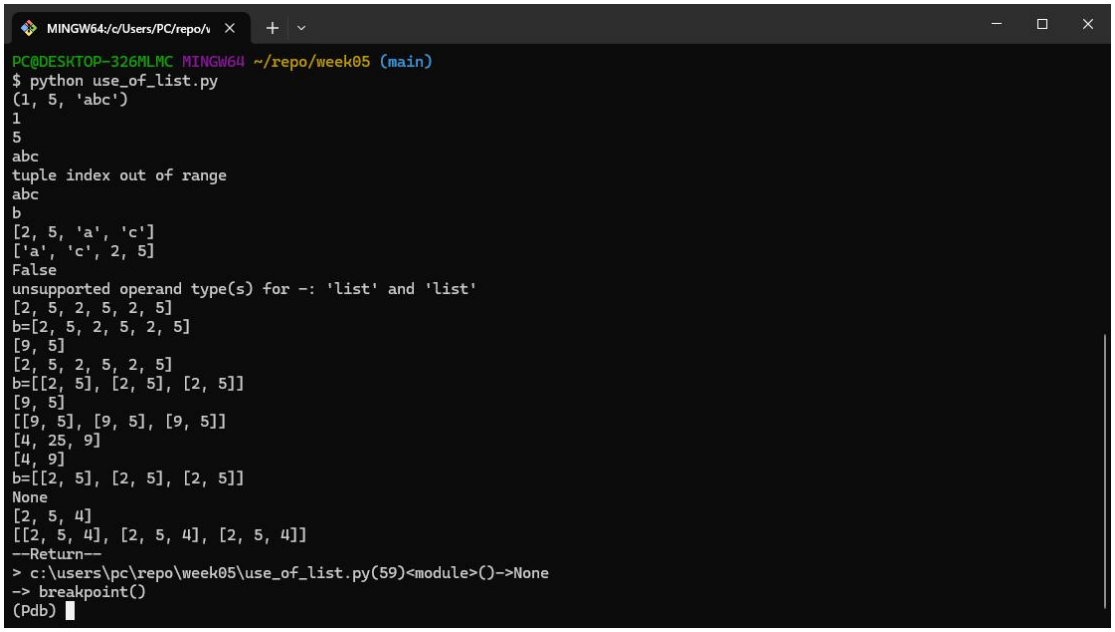

File Edit Selection View Go
PC
EXPLORER
environment.yml U
use_of_bool.py U
repo > week05 > use_of_bool.py > ...
t = True
f = False
print(t, f)
print(type(t))
print(isinstance(t, int))
NetHood
OneDrive
Pictures
PrintHood
Recent
repo
mywork
proj1
week01
week02
week03
week04
week05
.gitignore
environment.yml
LICENSE
README.md
use_of_bool.py
use_of_bytes.py
use_of_dict.py
use_of_float.py
use_of_int.py
use_of_list.py
use_of_set.py
use_of_str.py
use_of_tuple.py
script1.py
Saved Games
Searches
OUTLINE
TIMELINE
main*
OVR Ln 7, Col 1 Spaces: 4 UTF-8 CRLF Python 3.12.9 ('week04': conda)

```
(week05)
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$ python use_of_float.py
<class 'float'>
<class 'float'>
1.6666666666666667 <class 'float'>
0.2174845816334181
nan
False
False
False
0.0314
True
False
True
-inf
(week05)
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$ python use_of_bool.py
True False
<class 'bool'>
True
(week05)
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$
```

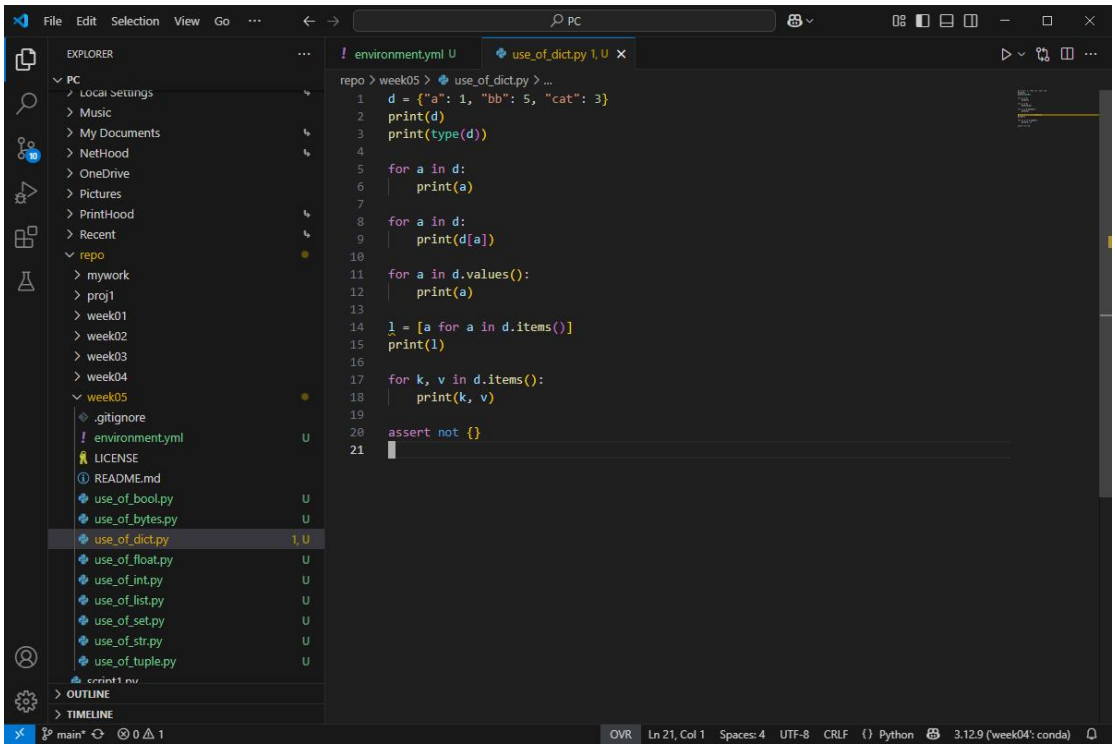
10. 列表

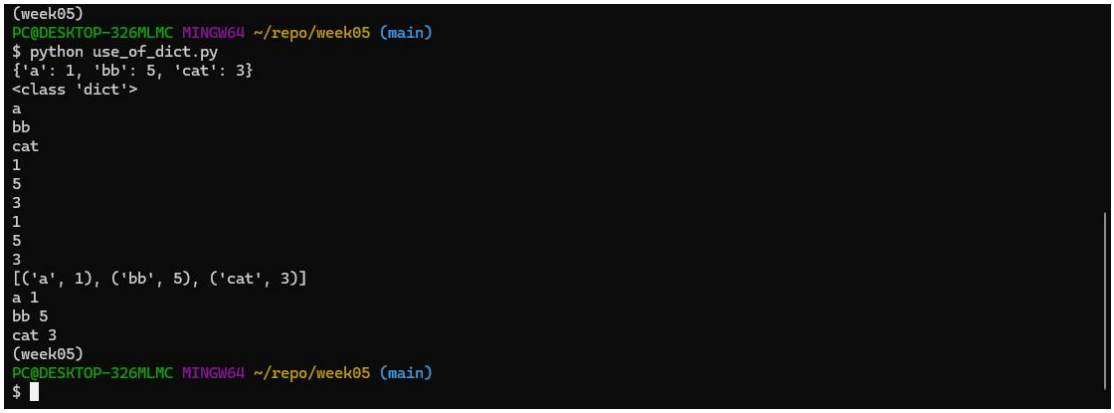



File Edit Selection View Go
PC
EXPLORER
environment.yml U
use_of_list.py 1, U
repo > week05 > use_of_list.py > ...
Music
My Documents
NetHood
OneDrive
Pictures
PrintHood
Recent
repo
mywork
proj1
week01
week02
week03
week04
week05
.gitignore
environment.yml
LICENSE
README.md
use_of_bool.py
use_of_bytes.py
use_of_dict.py
use_of_float.py
use_of_int.py
use_of_list.py
use_of_set.py
use_of_str.py
use_of_tuple.py
OUTLINE
TIMELINE
a = [2, 5]
b = a * 3
print(f"{b=}")
a[0] = 9
print(a)
print(b)
a = [2, 5]
b = [a] * 3
print(f"{b=}")
a[0] = 9
print(a)
print(b)
a = [2, 5, 3]
b = [i**2 for i in a]
print(b)
b = [i**2 for i in a if i < 4]
print(b)
a = [2, 5]
b = [a] * 3
print(f"{b=}")
x = a.append(4)
print(x)
print(a)
print(b)
breakpoint()
main*
OVR
Ln 60, Col 1
Spaces: 4
UTF-8
CRLF
Python
3.12.9 ('week04': conda)


MINGW64:/c/Users/PC/repo/
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$ python use_of_list.py
(1, 5, 'abc')
1
5
abc
tuple index out of range
abc
b
[2, 5, 'a', 'c']
['a', 'c', 2, 5]
False
unsupported operand type(s) for -: 'list' and 'list'
[2, 5, 2, 5, 2, 5]
b=[2, 5, 2, 5, 2, 5]
[9, 5]
[2, 5, 2, 5, 2, 5]
b=[[2, 5], [2, 5], [2, 5]]
[9, 5]
[[9, 5], [9, 5], [9, 5]]
[4, 25, 9]
[4, 9]
b=[[2, 5], [2, 5], [2, 5]]
None
[2, 5, 4]
[[2, 5, 4], [2, 5, 4], [2, 5, 4]]
--Return--
> c:\users\pc\repo\week05\use_of_list.py(59)<module>()->None
-> breakpoint()
(Pdb)

11. 字典





12. 元组及与列表的区别

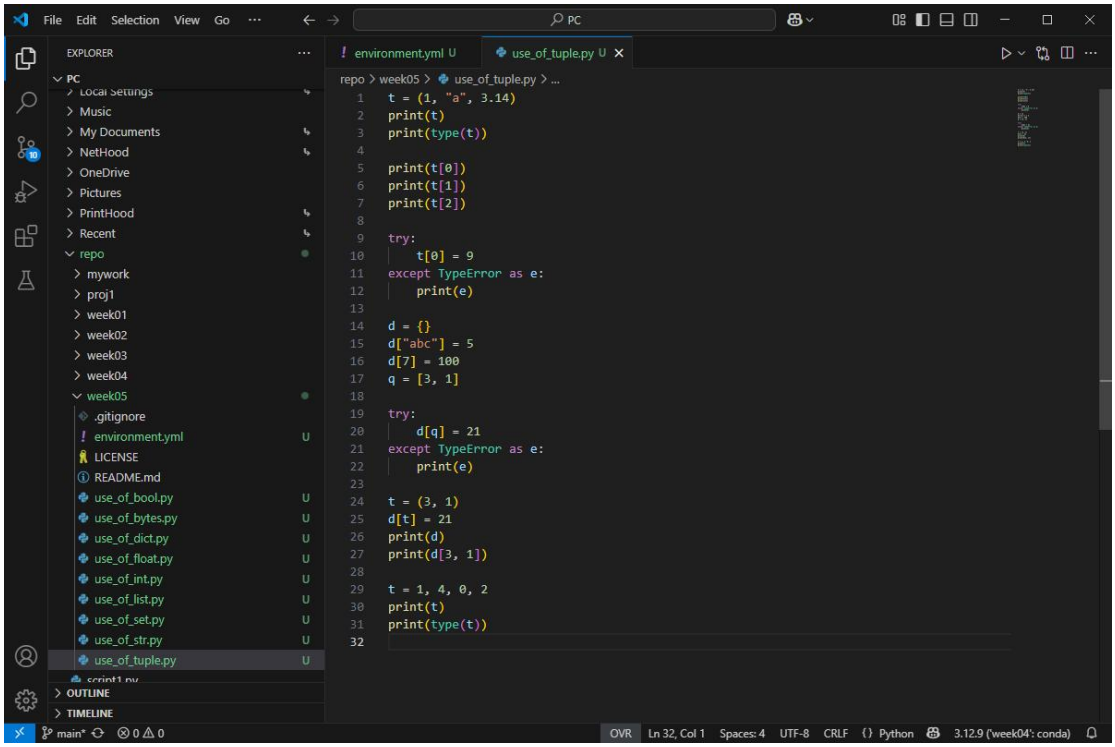
（1）列表使用方括号［］来定义，元素之间用逗号分隔；元组使用圆括号（）来定义，元素之间用逗号分隔。定义只有一个元素的元组时，需要在元素后面加逗号。

（2）列表是可变对象，这意味着能够对其元素进行添加、删除、修改操作；元组是不可变对象，一旦创建，其元素就不能被修改、添加或删除。

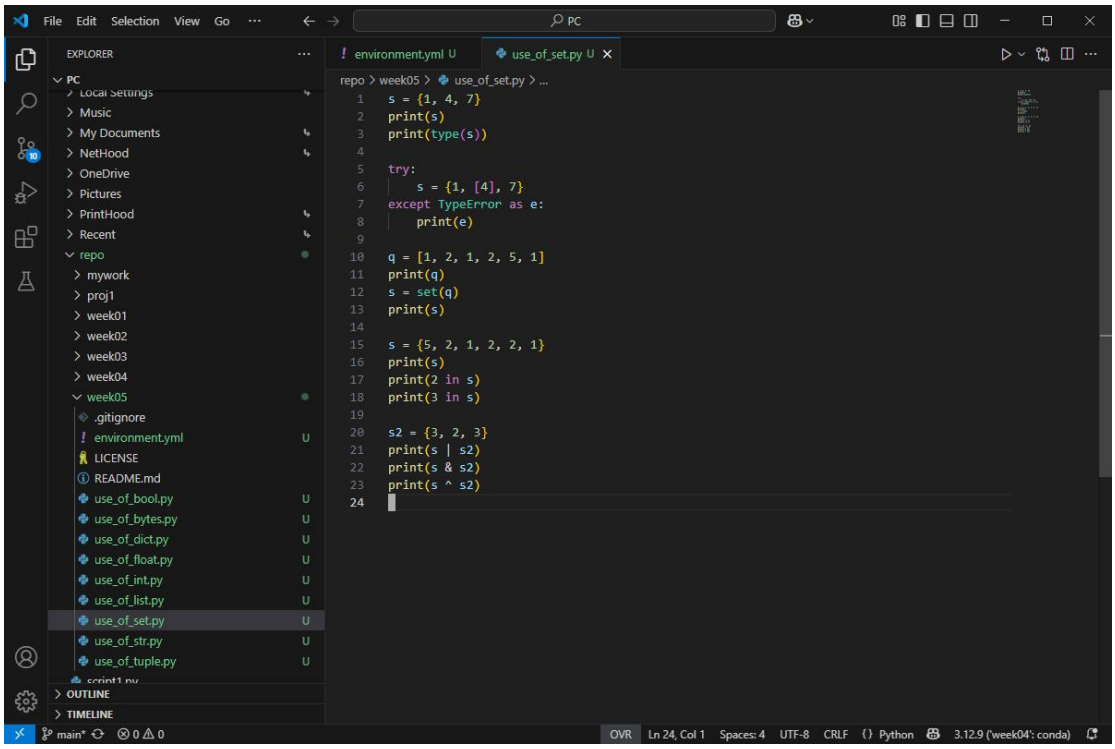
（3）列表适用于需要动态修改元素的场景，像存储用户输入的一系列数据、动态更新的数据集等；元组适合存储不可变的数据，如坐标、数据库查询结果等，

也可用于函数返回多个值。

（4）列表拥有较多的内置方法，如 append()、extend()、remove()、sort() 等，方便对列表进行操作；元组方法较少，只有 count() 和 index() 两个常用方法。

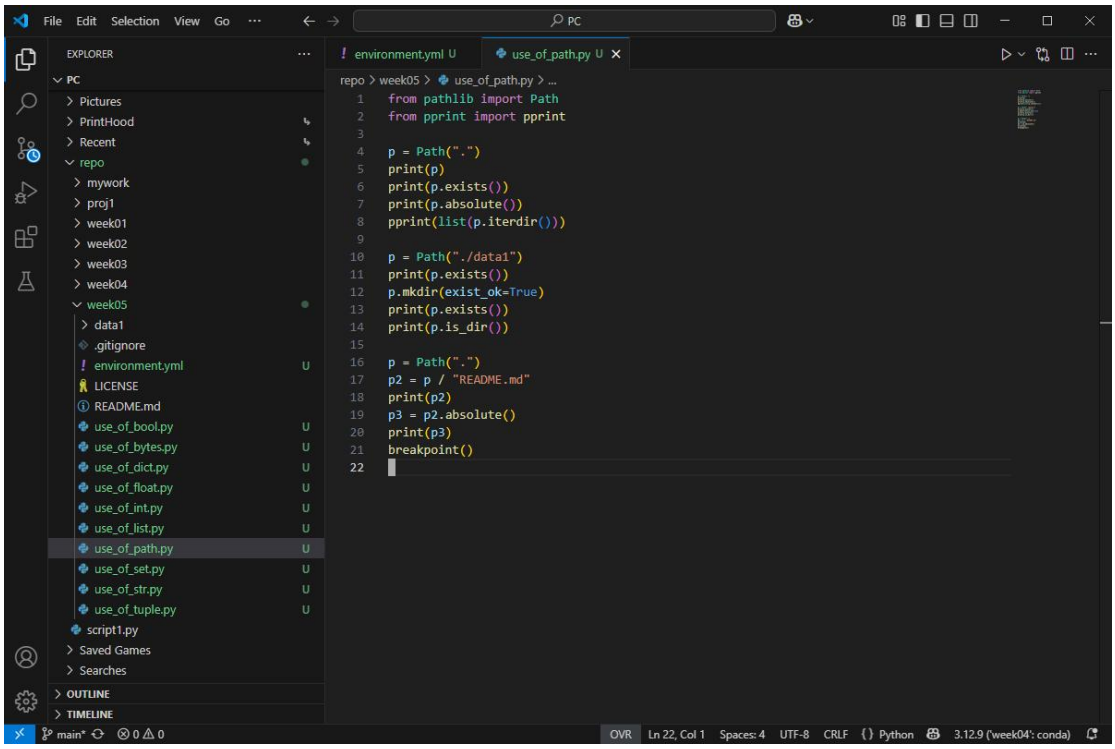


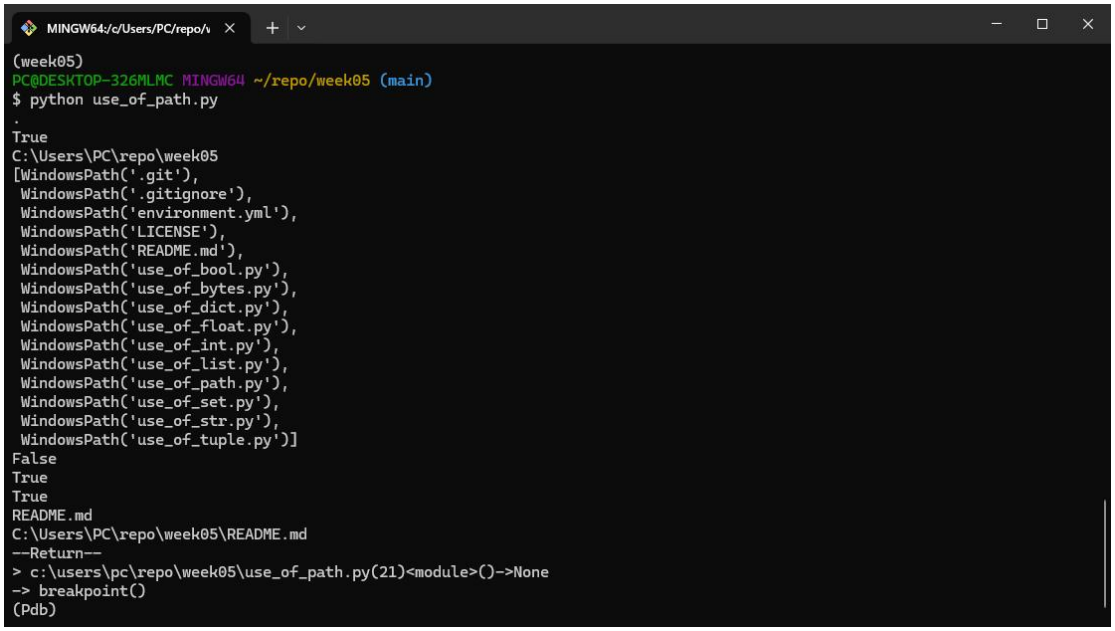
13. 集合



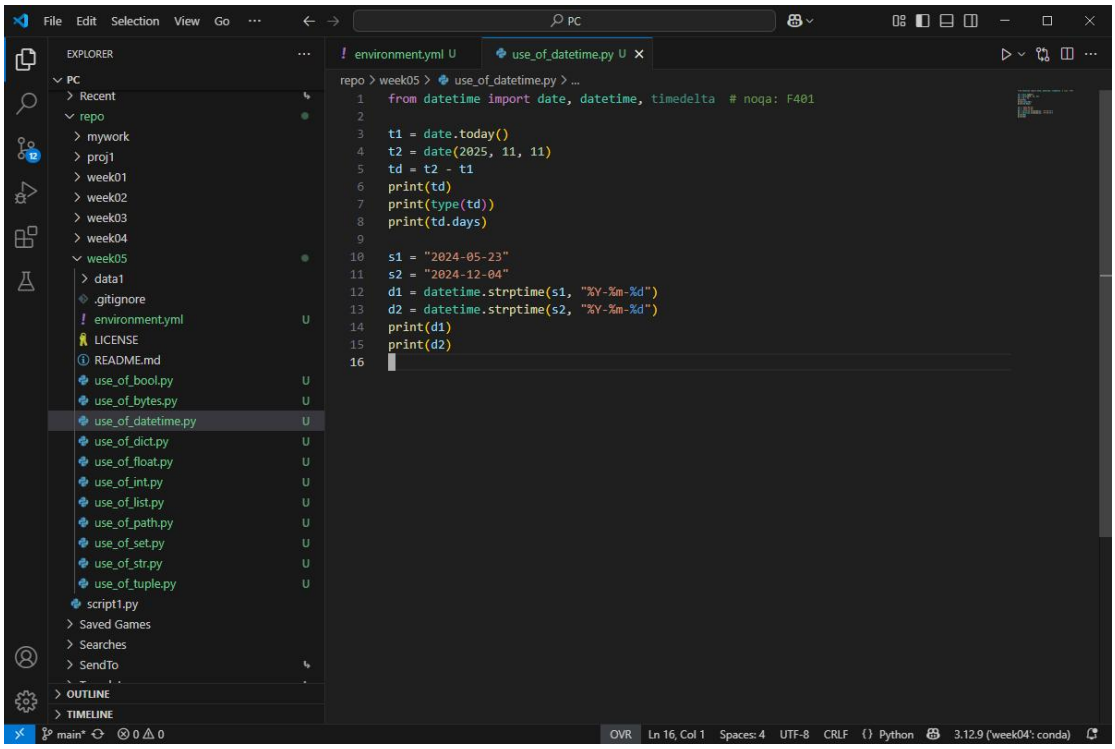

```
(week05)
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$ python use_of_set.py
{1, 4, 7}
<class 'set'>
unhashable type: 'list'
[1, 2, 1, 2, 5, 1]
{1, 2, 5}
{1, 2, 5}
True
False
{1, 2, 3, 5}
{2}
{1, 3, 5}
(week05)
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$
```

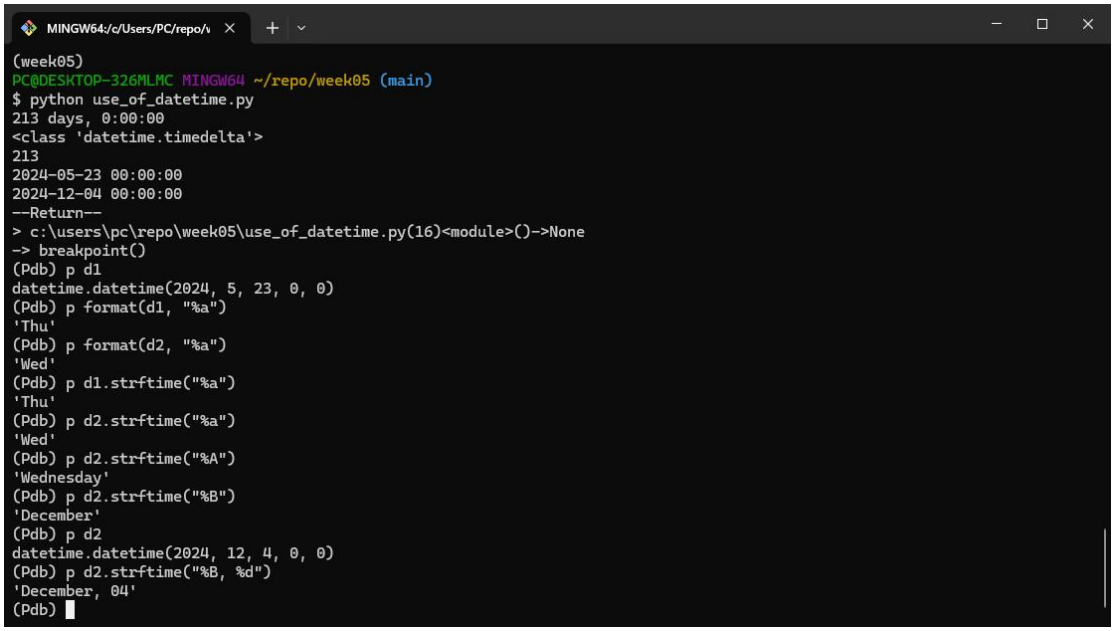
14. 路径


File Edit Selection View Go
PC
EXPLORER
PC
Pictures
PrintHood
Recent
repo
mywork
proj1
week01
week02
week03
week04
week05
data1
.gitignore
environment.yml
LICENSE
README.md
use_of_bool.py
use_of_bytes.py
use_of_dict.py
use_of_float.py
use_of_int.py
use_of_list.py
use_of_path.py
use_of_set.py
use_of_str.py
use_of_tuple.py
script1.py
Saved Games
Searches
OUTLINE
TIMELINE
environment.yml U
use_of_path.py U
repo > week05 > use_of_path.py > ...
from pathlib import Path
from pprint import pprint
p = Path(".")
print(p)
print(p.exists())
print(p.absolute())
pprint(list(p.iterdir()))
p = Path("./data1")
print(p.exists())
p.mkdir(exist_ok=True)
print(p.exists())
print(p.is_dir())
p = Path(".")
p2 = p / "README.md"
print(p2)
p3 = p2.absolute()
print(p3)
breakpoint()
main*
OVR
Ln 22, Col 1
Spaces: 4
UTF-8
CRLF
Python
3.12.9 ('week04': conda)


MINGW64:/c/Users/PC/repo/
(week05)
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$ python use_of_path.py
.
True
C:\Users\PC\repo\week05
[WindowsPath('.git'),
 WindowsPath('.gitignore'),
 WindowsPath('environment.yml'),
 WindowsPath('LICENSE'),
 WindowsPath('README.md'),
 WindowsPath('use_of_bool.py'),
 WindowsPath('use_of_bytes.py'),
 WindowsPath('use_of_dict.py'),
 WindowsPath('use_of_float.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
README.md
C:\Users\PC\repo\week05\README.md
--Return--
> c:\users\pc\repo\week05\use_of_path.py(21)<module>()->None
-> breakpoint()
(Pdb)

15. 日期时间


File Edit Selection View Go
PC
EXPLORER
PC
Recent
repo
mywork
proj1
week01
week02
week03
week04
week05
data1
.gitignore
environment.yml
LICENSE
README.md
use_of_bool.py
use_of_bytes.py
use_of_datetime.py
use_of_dict.py
use_of_float.py
use_of_int.py
use_of_list.py
use_of_path.py
use_of_set.py
use_of_str.py
use_of_tuple.py
script1.py
Saved Games
Searches
SendTo
OUTLINE
TIMELINE
environment.yml U
use_of_datetime.py U
repo > week05 > use_of_datetime.py > ...
from datetime import date, datetime, timedelta # noqa: F401
t1 = date.today()
t2 = date(2025, 11, 11)
td = t2 - t1
print(td)
print(type(td))
print(td.days)
s1 = "2024-05-23"
s2 = "2024-12-04"
d1 = datetime.strptime(s1, "%Y-%m-%d")
d2 = datetime.strptime(s2, "%Y-%m-%d")
print(d1)
print(d2)
main*
0 0
OVR
Ln 16, Col 1
Spaces: 4
UTF-8
CRLF
Python
3.12.9 ('week04': conda)


MINGW64:/c/Users/PC/repo/
(week05)
PC@DESKTOP-326MLMC MINGW64 ~/repo/week05 (main)
$ python use_of_datetime.py
213 days, 0:00:00
<class 'datetime.timedelta'>
213
2024-05-23 00:00:00
2024-12-04 00:00:00
--Return--
> c:\users\pc\repo\week05\use_of_datetime.py(16)<module>()->None
-> breakpoint()
(Pdb) p d1
datetime.datetime(2024, 5, 23, 0, 0)
(Pdb) p format(d1, "%a")
'Thu'
(Pdb) p format(d2, "%a")
'Wed'
(Pdb) p d1.strftime("%a")
'Thu'
(Pdb) p d2.strftime("%a")
'Wed'
(Pdb) p d2.strftime("%A")
'Wednesday'
(Pdb) p d2.strftime("%B")
'December'
(Pdb) p d2
datetime.datetime(2024, 12, 4, 0, 0)
(Pdb) p d2.strftime("%B, %d")
'December, 04'
(Pdb)