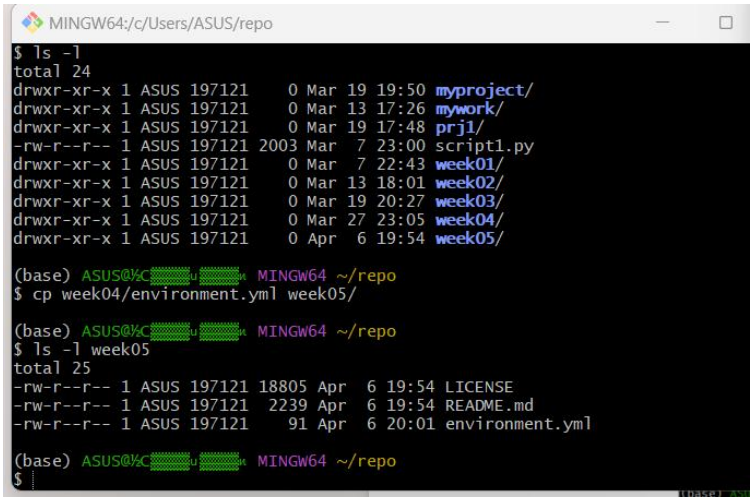
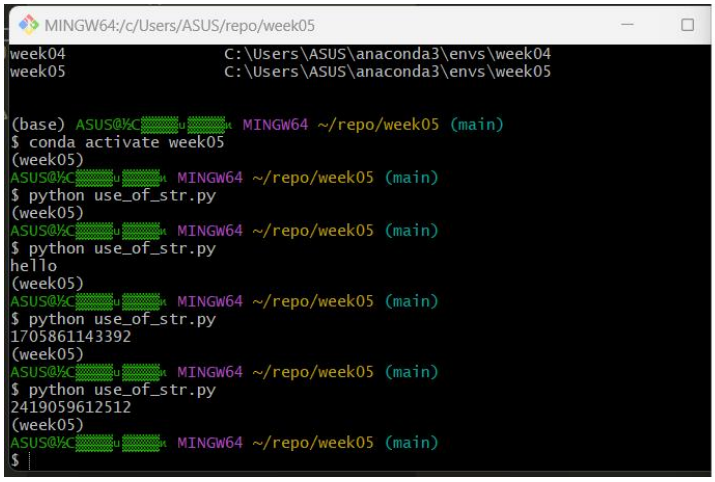
金融计算与编程第五周学习笔记

1. 把 environment.yml 文件复制到 week05，



2. 在 vs code 里创建 use_of_str.py 文件。



每次运行的结果不一样，地址是虚拟的，用来显示地址。
但是改为列表以后，id 就不重复了。虽然值是相等的，但不是同一个对象。

```python
a = "hello"
b = "hello"
x = id(a)
print(x)
y = id(b)
print(y)
```

```python
a = [2, 5]
b = [2, 5]
x = id(a)
print(x)
y = id(b)
print(y)
```

```
MINGW64:/c/Users/ASUS/repo/week05
$ conda activate week05
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
hello
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
1705861143392
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2419059612512
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2502127672160
2502127672160
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$
```

```
ASUS@%C MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
hello
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
1705861143392
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2419059612512
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2502127672160
2502127672160
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2090440857856
2090440855872
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$
```

id 是不变的。

```python
a = [2, 5]
b = [2, 5]
x = id(a)
print(x)
y = id(b)
print(y)
a[0] = 9
print(a)
print(b)
print(id(a))  # is it same as x?
print(id(b))  # is it same as y?
```

```
ASUS@%C MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2391273576704
2391273574720
[9, 5]
[2, 5]
2391273576704
2391273574720
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$
```

type 可以返回对象的类型。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。

```
ASUS@%C MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
1525042845952
1525042843968
[9, 5]
[2, 5]
1525042845952
1525042843968
<class 'list'>
False
dir(a): ['__add__', '__class__', '__class_getitem__', '__contains__', '__delattr
__', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__
etattribute__', '__getitem__', '__getstate__', '__gt__', '__hash__', '__iadd__'
 '__imul__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__',
__lt__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr
__', '__reversed__', '__rmul__', '__setattr__', '__setitem__', '__sizeof__', '__
tr__', '__subclasshook__', 'append', 'clear', 'copy', 'count', 'extend', 'index
, 'insert', 'pop', 'remove', 'reverse', 'sort']
(week05)
ASUS@%C MINGW64 ~/repo/week05 (main)
$
```

use_of_str.py > ...

```python
a = [2, 5]
b = [2, 5]
x = id(a)
print(x)
y = id(b)
print(y)
a[0] = 9
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))
print("dir(a):", dir(a))
print("isinstance(a,str):", isinstance(a, str))
print(isinstance(a, (str, float)))
print(isinstance(a, (str, float, list)))

```

```
ASUS@%C▒▒▒▒u▒▒▒▒ MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2917094398208
2917094396224
[9, 5]
[2, 5]
2917094398208
2917094396224
<class 'list'>
False
dir(a): ['__add__', '__class__', '__class_getitem__', '__contains__', '__delattr
__', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__g
etattribute__', '__getitem__', '__getstate__', '__gt__', '__hash__', '__iadd__',
 '__imul__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__', '
__lt__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr_
_', '__reversed__', '__rmul__', '__setattr__', '__setitem__', '__sizeof__', '__s
tr__', '__subclasshook__', 'append', 'clear', 'copy', 'count', 'extend', 'index'
, 'insert', 'pop', 'remove', 'reverse', 'sort']
isinstance(a,str): False
False
True
(week05)
ASUS@%C▒▒▒▒u▒▒▒▒ MINGW64 ~/repo/week05 (main)
$
```

如果 assert 语句正确，就会什么也不干进行下一步，错误将会终止。

use_of_str.py > ...

```python
a = [2, 5]
b = [2, 5]
x = id(a)
print(x)
y = id(b)
print(y)
a[0] = 9
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))
print("dir(a):", dir(a))
print("isinstance(a,str):", isinstance(a, str))
print(isinstance(a, (str, float)))
print(isinstance(a, (str, float, list)))
assert isinstance(a, list)
print("goodbye")

```

```
$ python use_of_str.py
2393451403520
2393451401536
[9, 5]
[2, 5]
2393451403520
2393451401536
<class 'list'>
False
dir(a): ['__add__', '__class__', '__class_getitem__', '__contains__', '__delattr
__', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__g
etattribute__', '__getitem__', '__getstate__', '__gt__', '__hash__', '__iadd__',
 '__imul__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__', '
__lt__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr_
_', '__reversed__', '__rmul__', '__setattr__', '__setitem__', '__sizeof__', '__s
tr__', '__subclasshook__', 'append', 'clear', 'copy', 'count', 'extend', 'index'
, 'insert', 'pop', 'remove', 'reverse', 'sort']
isinstance(a,str): False
False
True
goodbye
(week05)
ASUS@½C▒▒▒u▒▒▒▒ MINGW64 ~/repo/week05 (main)
$
```

3. 字面值，一般用\n 或者\t。

```
use_of_str.py > ...
1   print("字面值")
2   s = "university"
3   print(s)
4   print(isinstance(s, str))
5   assert type(s) is str
6
7   print("f -string")
8   x = "tom"
9   s = f"name: {x}"
10  print(s)
11
12  s = "a\tb"
13  print("TAB", s)
14
15  s = "abc\ndef"
16  print("great", s)
17
```

```
ASUS@½C▒▒▒▒u▒▒▒▒▒ MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
university
True
f -string
name: tom
TAB a   b
(week05)
ASUS@½C▒▒▒▒u▒▒▒▒▒ MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
university
True
f -string
name: tom
TAB a   b
great abc
def
(week05)
ASUS@½C▒▒▒▒u▒▒▒▒▒ MINGW64 ~/repo/week05 (main)
$
```

在二进制下面，1.1+2.2 不是整数。

```
print("初始化")
s = str()
print(s)
s = str([5, 8, 2])
print(s)
assert str([5, 8, 2]) == "[5, 8, 2]"
assert str(1.1 + 2.2) == "3.3"
```

```
[5, 8, 2]
Traceback (most recent call last):
  File "C:\Users\ASUS\anaconda3\envs\week05\Lib\pdb.py", line 1960, in main
    pdb._run(target)
  File "C:\Users\ASUS\anaconda3\envs\week05\Lib\pdb.py", line 1754, in _run
    self.run(target.code)
  File "C:\Users\ASUS\anaconda3\envs\week05\Lib\bdb.py", line 627, in run
    exec(cmd, globals, locals)
  File "<string>", line 1, in <module>
  File "C:\Users\ASUS\repo\week05\use_of_str.py", line 30, in <module>
    assert str(1.1 + 2.2) == "3.3"
           ^^^^^^^^^^^^^^^^^^^^^^^
AssertionError
Uncaught exception. Entering post mortem debugging
Running 'cont' or 'step' will restart the program
> c:\users\asus\repo\week05\use_of_str.py(30)<module>()
-> assert str(1.1 + 2.2) == "3.3"
(Pdb) p str(1.1+2.2)
'3.3000000000000003'
(Pdb)
```

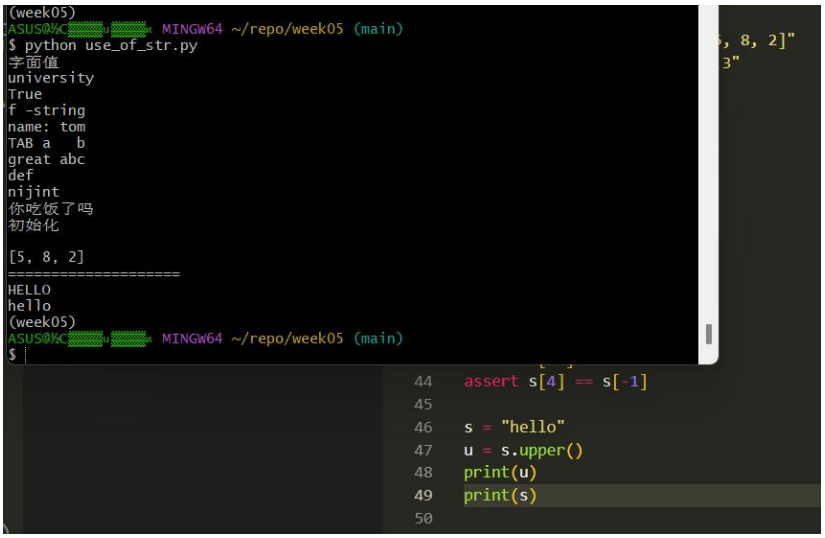
4. 索引值。正确的不报错，s[:3]是索引从左到右的三个字符。

```
s = "="
x = id(s)
s = s * 20
y = id(s)
print(s)
assert x != y

s = "hello"
assert s[3] == "l"
```

```
s = "hello"
assert s[3] == "l"
assert s[:3] == "hel"
```

字符串的对象不能被修改。



加法和乘法可以运算。比较字符大小按顺序比较，先比较第一个字符，第一个字符相等再比较第二个，与长短无关。字符串的长度为 0 才会报错。