第五周学习笔记

1. 复制 enviroment.yml 文件

```yaml
name: week05
channels:
  - conda-forge
dependencies:
  - python=3.12
  - wat-inspector
```

2. 建立 use of bool 文件

```
>>> t = True
>>> f = False
>>> print(t,f)
True False
>>> print(type(t))
<class 'bool'>
>>> print(isinstance(t,int))
True
>>>
```

3. 建立 use_of_bites

```
>>> s = b"hello"
>>> print(s)
b'hello'
>>> print(s[0])
104
>>>
>>> s = "你好"
>>> b = s.encode()
>>> print(b)
b'\xe4\xbd\xa0\xe5\xa5\xbd'
>>>
>>>
>>> print("整数")
整数
>>> i = 5
>>> x = 1
>>> y = i + x
>>> print(i, x, y)
5 1 6
>>>
```

4. 建立 use_of_datetime

```
>>> from datetime import date
>>>
>>> print(date.today())
2025-04-14
>>>
>>> t1 = date.today()
>>> t2 = date(2025, 10, 1)
>>> td = t2 - t1
>>> print(td)
170 days, 0:00:00
>>> print(td.days)
170
>>>
>>> s1 = date(2024, 12, 31)
>>> s2 = date(2025, 1, 3)
>>> sd = s2 - s1
>>> print(sd)
3 days, 0:00:00
```

5. 建立 use_of_ditc

```
>>> d = {"a": 1, "b": 2, "c": 3}
>>> print(d)
{'a': 1, 'b': 2, 'c': 3}
>>> print(type(d))
<class 'dict'>
>>>
>>> for a in d:
...         print(a)
...         for a in d:
...                     print(d[a])
...
```

6. 建立 use_of_float

```
>>> import random
>>>
>>> x = 3.14159
>>> print(type(x))
<class 'float'>
>>>
>>>
>>> y = float("3.14159")
>>> print(type(y))
<class 'float'>
>>> assert x == y
>>>
>>>
>>> x = 10 / 3
>>> print(x, type(x))
3.3333333333333335 <class 'float'>
>>>
>>> x = random.random()
>>> print(x)
0.7922993189815026
>>>
>>> assert not 0.0
>>>
>>> f = float("nan")  ##特殊值
>>> print(f)
nan
>>>
>>> print(3.14e2)  ##3.14乘10的平方
314.0
>>> inf = float("inf")
>>> print(inf == inf)  ##正无穷等于正无穷
True
>>> ##浮点数不精确，支持各种运算
>>>
```

7. 建立 use_of_int

```
>>> i = 5
>>> x = 1
>>> y = i + x
>>> print(i, x, y)
5 1 6
>>>
```

8. 建立 use_of_list

```
>>> a = [2, 3]
>>> print(c1 + a)
[1, 5, 'abc', 2, 3]
>>>
>>> print(c1 + a == a + c1)  ##不支持-运算 支持*运算
False
>>> print(a * 2)
[2, 3, 2, 3]
>>>
>>> b = a * 2
>>> print(b)
[2, 3, 2, 3]
>>> a[0] = 3
>>> print(a)
[3, 3]
>>> print(b)
[2, 3, 2, 3]
>>>
>>> a = [1, 2]
>>> b = [a] * 2
>>> print(b)
[[1, 2], [1, 2]]
>>> a[0] = 3
>>> print(a)
[3, 2]
>>> print(b)
[[3, 2], [3, 2]]
>>>
>>> a = [2, 5, 3]
>>> b = [i**2 for i in a]  ##运算式
>>> print(b)
[4, 25, 9]
>>>
>>> a = [1, 2]
>>> b = [a] * 2
>>> print(b)
[[1, 2], [1, 2]]
>>> x = a.append(4)
>>> print(a)
[1, 2, 4]
>>> print(b)
[[1, 2, 4], [1, 2, 4]]
>>> print(x)
None
>>>
```

9. 建立 use_of_path

```
>>> p = Path(".")
>>> print(p)
.
>>> print(p.exists())
True
>>> print(p.absolute())
D:\Program Files\Python314
>>> pprint(list(p.iterdir()))
[WindowsPath('DLLs'),
 WindowsPath('Doc'),
 WindowsPath('include'),
 WindowsPath('Lib'),
 WindowsPath('libs'),
 WindowsPath('LICENSE.txt'),
 WindowsPath('NEWS.txt'),
 WindowsPath('python.exe'),
 WindowsPath('python3.dll'),
 WindowsPath('python314.dll'),
 WindowsPath('pythonw.exe'),
 WindowsPath('Scripts'),
 WindowsPath('tcl'),
 WindowsPath('vcruntime140.dll'),
 WindowsPath('vcruntime140_1.dll')]
>>>
>>> p = Path("./data")
>>> print(p.exists())
False
>>> p.mkdir(exist_ok=True)
>>> print(p.exists())
True
>>> print(p.is_dir())
True
>>>
>>>
>>> p = Path(".")
>>> p1 = p / "README.md"
>>> print(p1)
README.md
>>> p2 = p1.absolute()
>>> print(p2)
D:\Program Files\Python314\README.md
>>>
```

10. 建立 use_of_set

```
SyntaxError: invalid syntax
>>>
>>> q = [1, 2, 1, 2, 4, 5, 6]
>>> print(q)
[1, 2, 1, 2, 4, 5, 6]
>>> s = set(q)
>>> print(s)
{1, 2, 4, 5, 6}
>>>
>>> print(3 in s)  ##判断
False
>>>
>>> s1 = {1, 2, 3}
>>> print(s | s1)
{1, 2, 3, 4, 5, 6}
>>> print(s & s1)
{1, 2}
>>>
```

11. 建立 use_of_str

```
>>> print(3 in s)  ##判断
False
>>>
>>> s1 = {1, 2, 3}
>>> print(s | s1)
{1, 2, 3, 4, 5, 6}
>>> print(s & s1)
{1, 2}
>>> a = [2, 5]
>>> b = [2, 5]
>>> x = id(a)
>>> print(x)
1579886754176
>>> y = id(b)
>>> print(y)
1579915955456
>>> a[0] = 9
>>> print(a)
[9, 5]
>>> print(b)
[2, 5]
>>> print(id(a))
1579886754176
>>> print(id(b))
1579915955456
>>> print(type(a))
<class 'list'>
>>> print("isinstance(a,str):", isinstance(a, str))
isinstance(a,str): False
>>> print("dir(a):", dir(a))
dir(a): ['__add__', '__class__', '__class_getitem__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__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__', '__str__', '__subclasshook__', 'append', 'clear', 'copy', 'count', 'extend', 'index', 'insert', 'pop', 'remove', 'reverse', 'sort']
>>> try:
...                     assert isinstance(a, str)
...                     except AssertionError:
...                                         print("type error")
...                                         print("hello")
```

12. 建立 use_of_str_1

```
Python 3.14 (64-bit)
>>> s = "a\tb"
>>> print("TAB", s)
TAB a   b
>>>
>>> s = "aaa\nbbb"
>>> print("New Line", s)
New Line aaa
bbb
>>>
>>>
>>> print("初始化")
初始化
>>> s = str()
>>> print(s)
>>>
>>> s = str([5, 8, 2])
>>> print(s)
[5, 8, 2]
>>>
>>> assert str(1 + 2) == "3"
>>>
>>> print("运算值")
运算值
>>> s = "="
>>> s = s * 20
>>> print(s)
====================
>>>
>>> print("索引值")
索引值
>>> s = "hello"
>>> print(s[3])
l
>>> assert s[3] == "l"
>>> assert s[:3] == "hel"
>>>
>>> print("返回值")
返回值
>>> s = "hello"
>>> s = s.upper()
>>> print(s)
HELLO
>>>
>>> t = "name:{},age:{}"
>>> print(t)
name:{},age:{}
>>> t1 = t.format("Jack", 21)
>>> print(t1)
name:Jack,age:21
```

13. 建立 use_of_str_2

```
...
b
o
o
k
>>> print(len(s))
4
>>> s = "book"
>>> assert s[1:3] == "oo"
>>>
>>>
>>> q = ["rose", "jack", "bob"]
>>> print(":".join(q))
rose:jack:bob
>>>
```

14. 建立 use_of_tuple

```
>>> t = (1, "a", 3.14159)
>>> print(t)
(1, 'a', 3.14159)
>>> print(type(t))
<class 'tuple'>
>>>
>>> print(t[0])
1
>>>
```