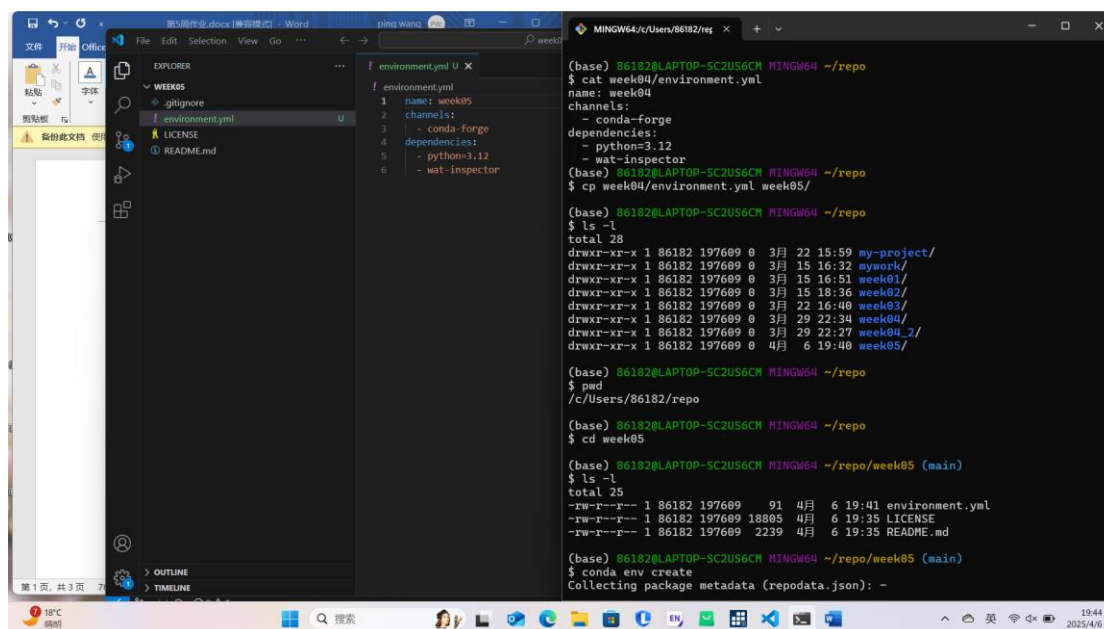


1. Fork [第 05 周打卡](#) 仓库至你的名下，然后将你名下的这个仓库 Clone 到你的本地计算机
2. 用 VS Code 打开项目目录，新建一个 environment.yml 文件，指定安装 Python 3.12，然后运行 conda env create 命令创建 Conda 环境

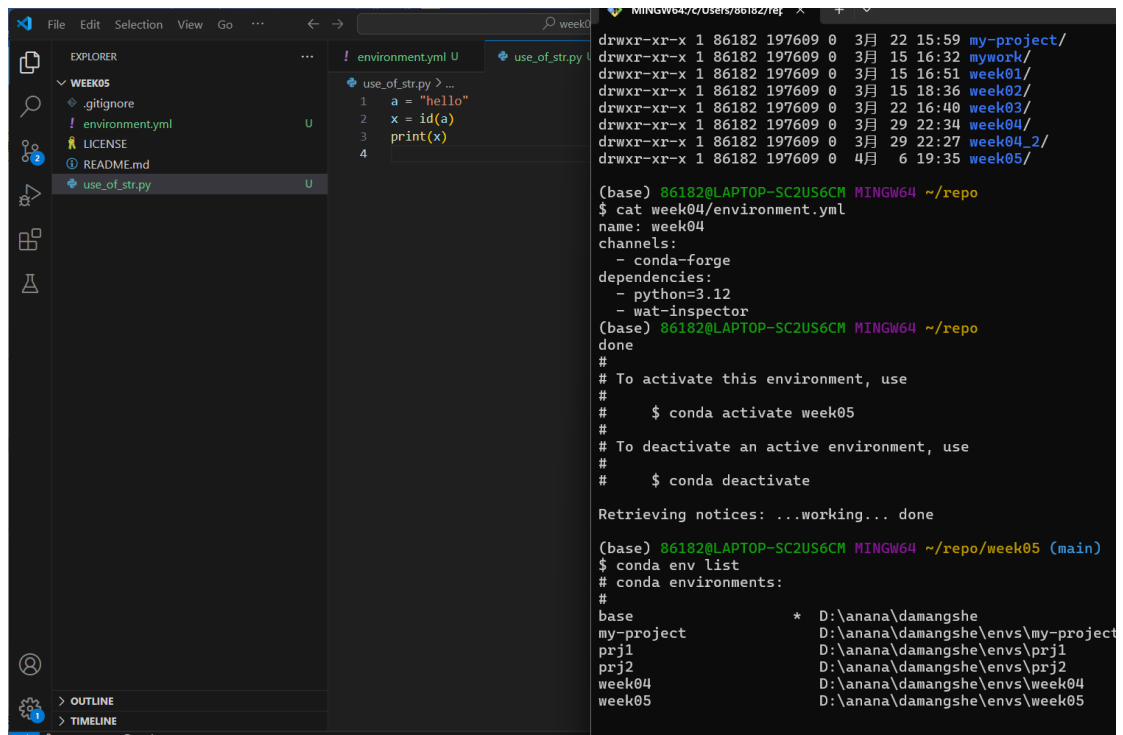


3. 逐个创建 use_of_{name}.py 文件，其中 {name} 替换为上述要求掌握的对象类型，例如 use_of_str.py:
 - 在全局作用域 (global scope) 内尝试键入 (活学活用) Python 代码，

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)



The screenshot shows a code editor with a file explorer on the left, a code editor in the center, and a terminal on the right. The file explorer shows a project named 'WEEK05' with files like '.gitignore', 'environment.yml', 'LICENSE', 'README.md', and 'use_of_str.py'. The code editor shows the content of 'use_of_str.py', which is a simple Python script:

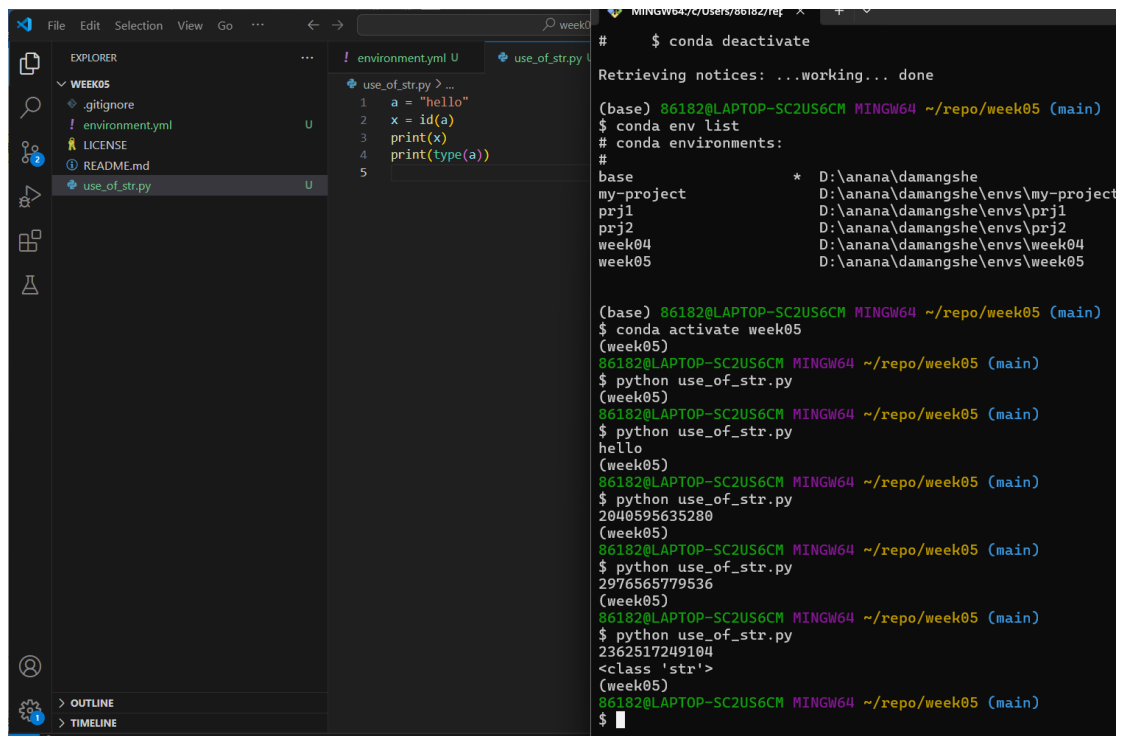
```
1 a = "hello"
2 x = id(a)
3 print(x)
4
```

. The terminal shows the output of the script, which is the memory address of the string 'hello'. The output is:

```
drwxr-xr-x 1 86182 197609 0 3月 22 15:59 my-project/
drwxr-xr-x 1 86182 197609 0 3月 15 16:32 mywork/
drwxr-xr-x 1 86182 197609 0 3月 15 16:51 week01/
drwxr-xr-x 1 86182 197609 0 3月 15 18:36 week02/
drwxr-xr-x 1 86182 197609 0 3月 22 16:40 week03/
drwxr-xr-x 1 86182 197609 0 3月 29 22:34 week04/
drwxr-xr-x 1 86182 197609 0 3月 29 22:27 week04_2/
drwxr-xr-x 1 86182 197609 0 4月 6 19:35 week05/

(base) 86182@LAPTOP-SC2U56CM MINGW64 ~/repo
$ cat week04/environment.yml
name: week04
channels:
  - conda-forge
dependencies:
  - python=3.12
  - wat-inspector
(base) 86182@LAPTOP-SC2U56CM MINGW64 ~/repo
done
#
# To activate this environment, use
#
#   $ conda activate week05
#
# To deactivate an active environment, use
#
#   $ conda deactivate
Retrieving notices: ...working... done
(base) 86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ conda env list
# conda environments:
#
base                  * D:\anana\damangshe
my-project            D:\anana\damangshe\envs\my-project
prj1                  D:\anana\damangshe\envs\prj1
prj2                  D:\anana\damangshe\envs\prj2
week04                D:\anana\damangshe\envs\week04
week05                D:\anana\damangshe\envs\week05
```

- `type()` -- 返回对象的类型



The screenshot shows a VS Code editor with a file explorer on the left showing a project named 'WEEK05' with files like '.gitignore', 'environment.yml', 'LICENSE', 'README.md', and 'use_of_str.py'. The main editor shows the 'use_of_str.py' file with the following code:

```
1 a = "hello"
2 x = id(a)
3 print(x)
4 print(type(a))
5
```

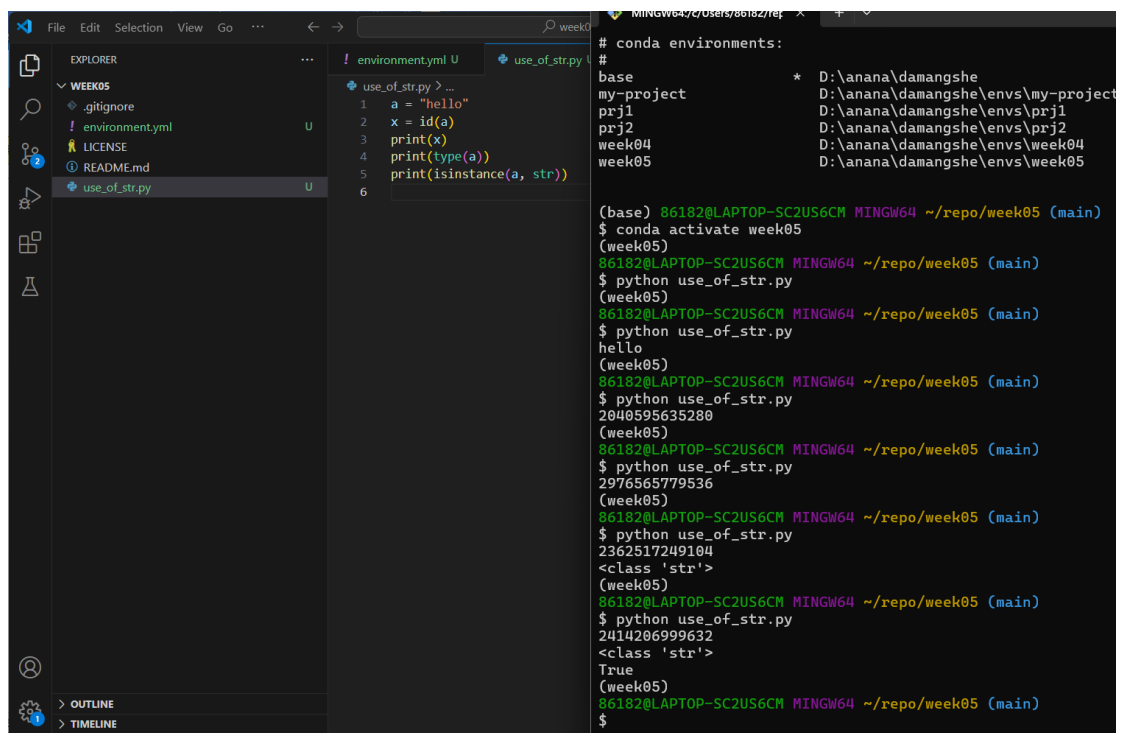
The terminal on the right shows the following commands and output:

```
# $ conda deactivate
Retrieving notices: ...working... done

(base) 86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ conda env list
# conda environments:
#
base                    * D:\anana\damangshe
my-project              D:\anana\damangshe\envs\my-project
prj1                   D:\anana\damangshe\envs\prj1
prj2                   D:\anana\damangshe\envs\prj2
week04                 D:\anana\damangshe\envs\week04
week05                 D:\anana\damangshe\envs\week05

(base) 86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ conda activate week05
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
hello
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2040595635280
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2976565779536
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2362517249104
<class 'str'>
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$
```

- `isinstance()` -- 判断对象是否属于某个 (或某些) 类型



The screenshot shows a VS Code editor with a file explorer on the left showing a project named 'WEEK05' with files like '.gitignore', 'environment.yml', 'LICENSE', 'README.md', and 'use_of_str.py'. The main editor shows the 'use_of_str.py' file with the following code:

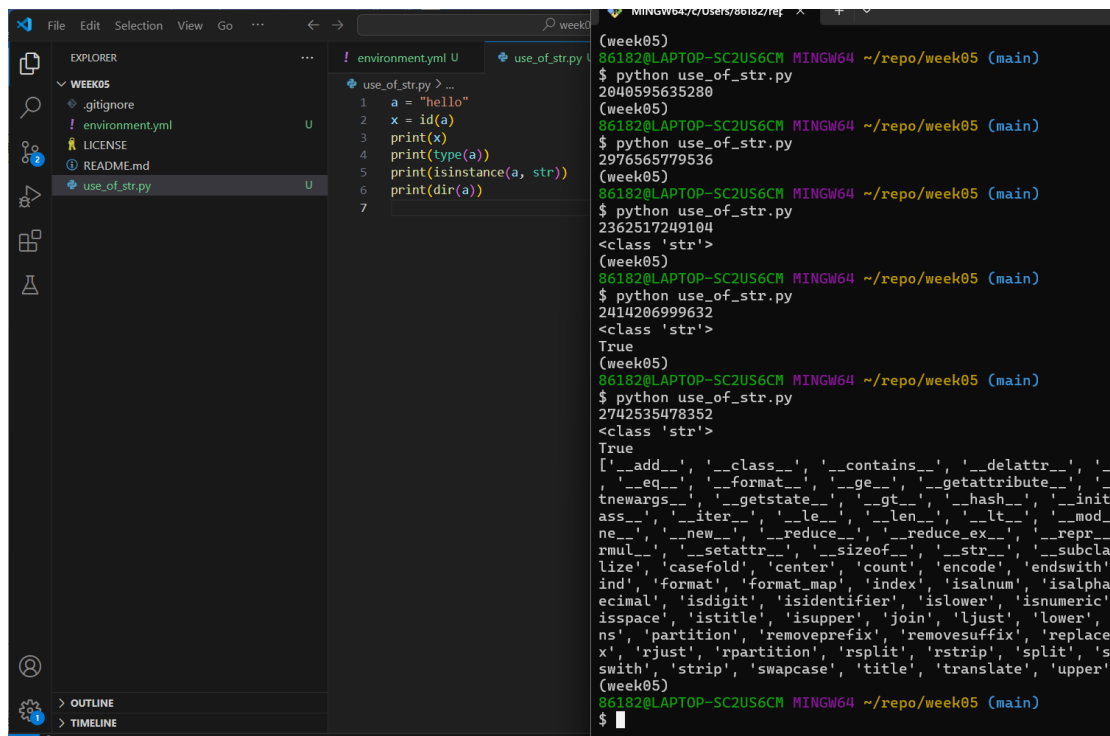
```
1 a = "hello"
2 x = id(a)
3 print(x)
4 print(type(a))
5 print(isinstance(a, str))
6
```

The terminal on the right shows the following commands and output:

```
# conda environments:
#
base                    * D:\anana\damangshe
my-project              D:\anana\damangshe\envs\my-project
prj1                   D:\anana\damangshe\envs\prj1
prj2                   D:\anana\damangshe\envs\prj2
week04                 D:\anana\damangshe\envs\week04
week05                 D:\anana\damangshe\envs\week05

(base) 86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ conda activate week05
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
hello
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2040595635280
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2976565779536
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2362517249104
<class 'str'>
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2414206999632
True
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$
```

- `dir()` -- 返回对象所支持的属性 (attributes) 的名称列表



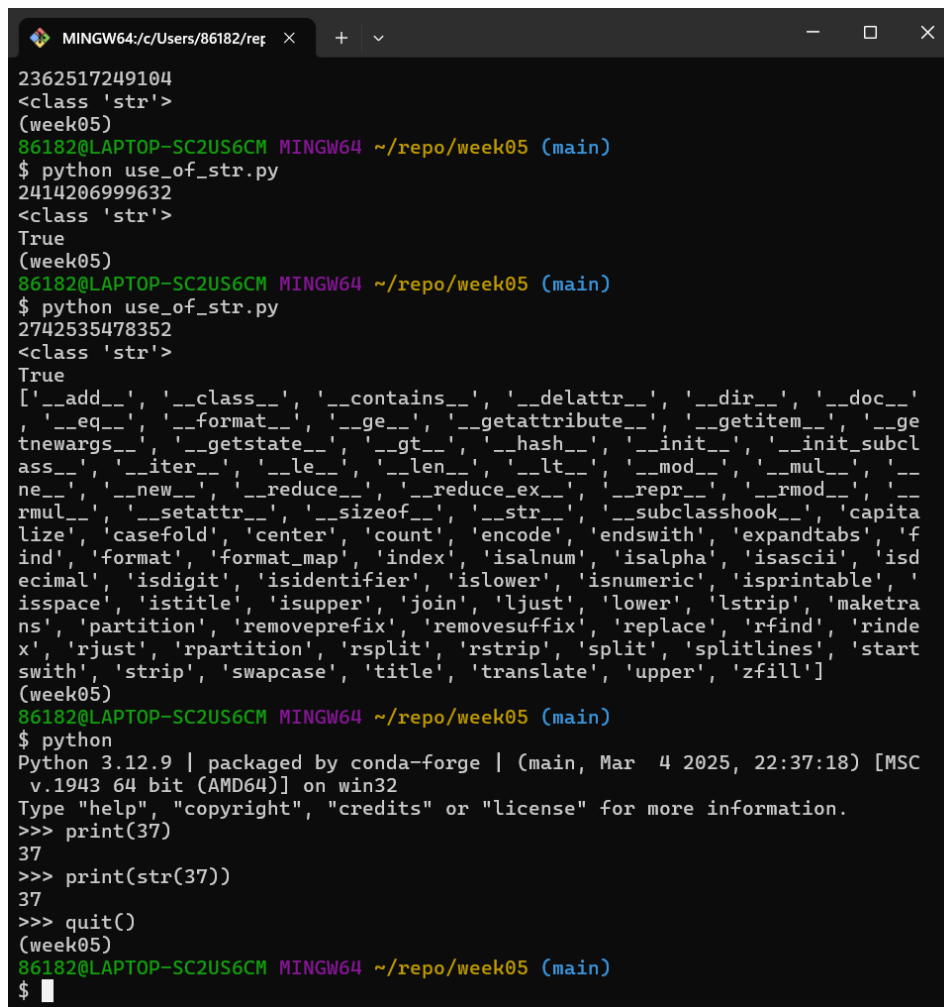
The image shows a VS Code editor window with a file explorer on the left showing a project named 'WEEK05' containing files like '.gitignore', 'environment.yml', 'LICENSE', 'README.md', and 'use_of_str.py'. The main editor displays the content of 'use_of_str.py':

```
1 a = "hello"
2 x = id(a)
3 print(x)
4 print(type(a))
5 print(isinstance(a, str))
6 print(dir(a))
7
```

The terminal on the right shows the execution of the script multiple times, displaying the memory address of the string object, the class type, and the list of attributes for the string object.

```
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2040595635280
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2976565779536
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2362517249104
<class 'str'>
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2414206999632
<class 'str'>
True
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2742535478352
<class 'str'>
True
['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewargs__', '__getstate__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__rmul__', '__setattr__', '__sizeof__', '__str__', '__subclasshook__', 'capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'removeprefix', 'removesuffix', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$
```

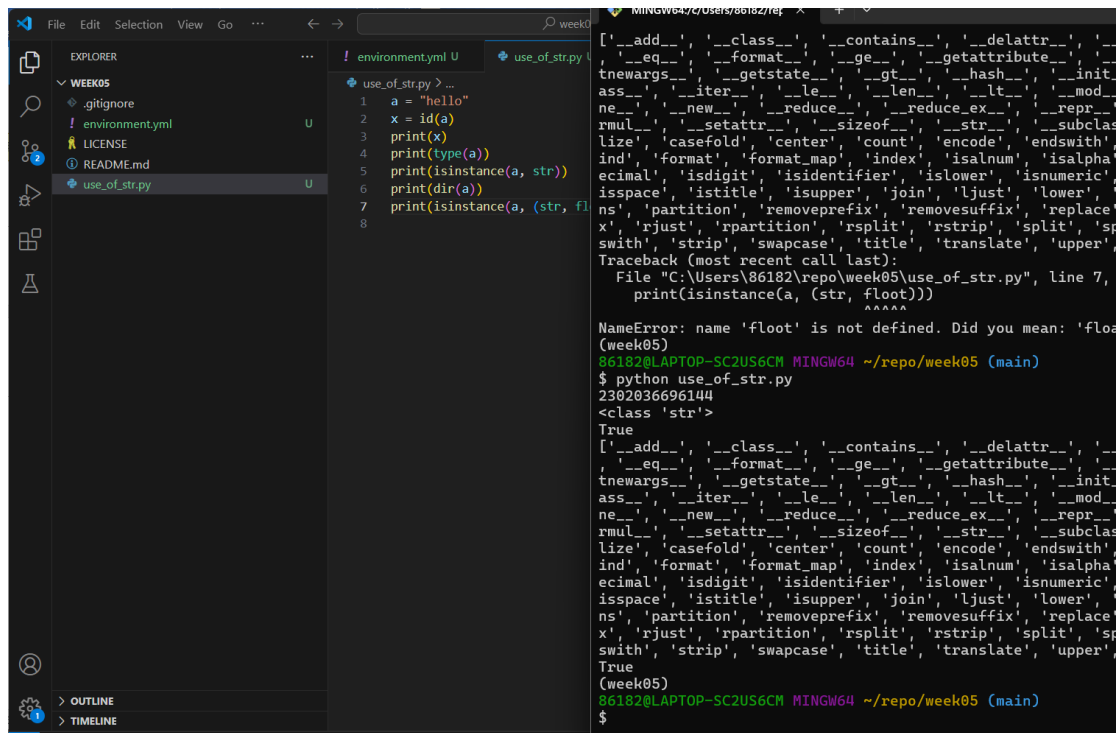
- `str()` -- 返回对象 `print` 时要显示在终端的字符串



The image shows a terminal window with the following output:

```
2362517249104
<class 'str'>
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2414206999632
<class 'str'>
True
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2742535478352
<class 'str'>
True
['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewargs__', '__getstate__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__rmul__', '__setattr__', '__sizeof__', '__str__', '__subclasshook__', 'capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'removeprefix', 'removesuffix', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python
Python 3.12.9 | packaged by conda-forge | (main, Mar 4 2025, 22:37:18) [MSC
v.1943 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print(37)
37
>>> print(str(37))
37
>>> quit()
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$
```

- 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



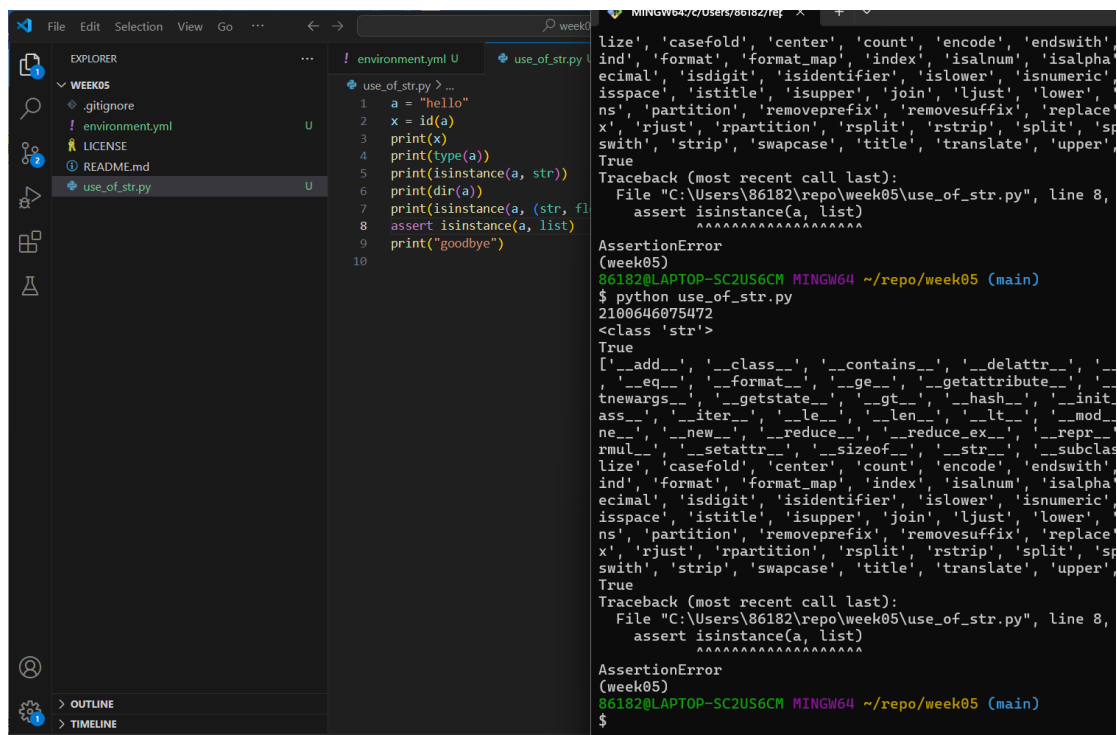
The screenshot shows a VS Code editor with a file named `use_of_str.py`. The code in the file is:

```
1 a = "hello"
2 x = id(a)
3 print(x)
4 print(type(a))
5 print(isinstance(a, str))
6 print(dir(a))
7 print(isinstance(a, (str, float)))
8
```

The terminal output shows the execution of the script, resulting in a `NameError` because `float` is not defined. The error message is:

```
NameError: name 'float' is not defined. Did you mean: 'float'?
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2302036696144
<class 'str'>
True
['_add_', '_class_', '_contains_', '_delattr_', '_eq_', '_format_', '_ge_', '_getattr_', '_gt_', '_hash_', '_init_', '_iter_', '_le_', '_len_', '_lt_', '_mod_', '_new_', '_reduce_', '_reduce_ex_', '_repr_', '_rmul_', '_setattr_', '_sizeof_', '_str_', '_subclass_', '_lize_', '_casefold_', '_center_', '_count_', '_encode_', '_endswith_', '_ind_', '_format_', '_format_map_', '_index_', '_isalnum_', '_isalpha_', '_isdecimal_', '_isdigit_', '_isidentifier_', '_islower_', '_isnumeric_', '_isspace_', '_istitle_', '_isupper_', '_join_', '_ljust_', '_lower_', '_ns_', '_partition_', '_removeprefix_', '_removesuffix_', '_replace_', '_x_', '_rjust_', '_rpartition_', '_rsplit_', '_rstrip_', '_split_', '_split_', '_strip_', '_swapcase_', '_title_', '_translate_', '_upper_', '_True'
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$
```

- 可以利用 `assert` 语句查验某个表达式 (expression) 为真，否则报错 (`AssertionError`) 退出



The screenshot shows a VS Code editor with a file named `use_of_str.py`. The code in the file is:

```
1 a = "hello"
2 x = id(a)
3 print(x)
4 print(type(a))
5 print(isinstance(a, str))
6 print(dir(a))
7 print(isinstance(a, (str, float)))
8 assert isinstance(a, list)
9 print("goodbye")
10
```

The terminal output shows the execution of the script, resulting in an `AssertionError` because `a` is a string, not a list. The error message is:

```
AssertionError
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2100646075472
<class 'str'>
True
['_add_', '_class_', '_contains_', '_delattr_', '_eq_', '_format_', '_ge_', '_getattr_', '_gt_', '_hash_', '_init_', '_iter_', '_le_', '_len_', '_lt_', '_mod_', '_new_', '_reduce_', '_reduce_ex_', '_repr_', '_rmul_', '_setattr_', '_sizeof_', '_str_', '_subclass_', '_lize_', '_casefold_', '_center_', '_count_', '_encode_', '_endswith_', '_ind_', '_format_', '_format_map_', '_index_', '_isalnum_', '_isalpha_', '_isdecimal_', '_isdigit_', '_isidentifier_', '_islower_', '_isnumeric_', '_isspace_', '_istitle_', '_isupper_', '_join_', '_ljust_', '_lower_', '_ns_', '_partition_', '_removeprefix_', '_removesuffix_', '_replace_', '_x_', '_rjust_', '_rpartition_', '_rsplit_', '_rstrip_', '_split_', '_split_', '_strip_', '_swapcase_', '_title_', '_translate_', '_upper_', '_True'
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$
```

```
File Edit Selection View Go ... use_of_str.py
EXPLORER
WEEK05
.gitignore
environment.yml
LICENSE
README.md
use_of_str.py
environment.yml U
use_of_str.py U
1 a = "hello"
2 x = id(a)
3 print(x)
4 print(type(a))
5 print(isinstance(a, str))
6 print(dir(a))
7 print(isinstance(a, (str, list)))
8 assert isinstance(a, list)
9 print("goodbye")
10

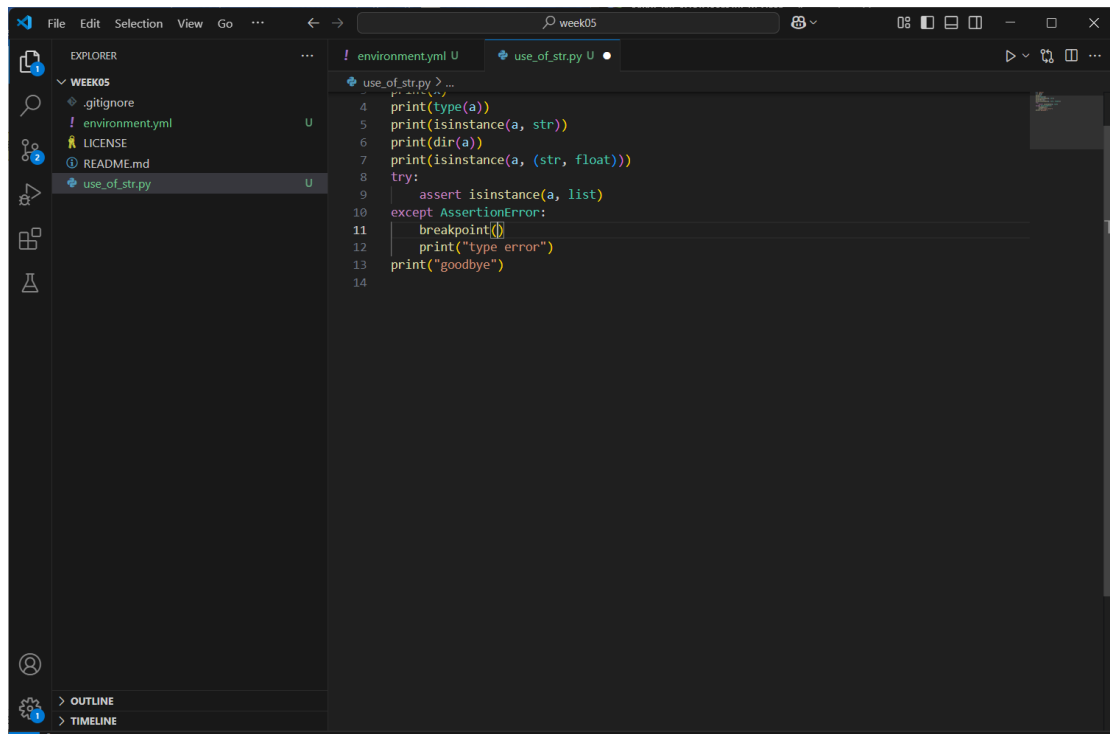
Traceback (most recent call last):
  File "C:\Users\86182\repo\week05\use_of_str.py", line 8, in <module>
    assert isinstance(a, list)
AssertionError
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2638562679888
<class 'str'>
True
['__add__', '__class__', '__contains__', '__delattr__', '__dict__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getstate__', '__gt__', '__hash__', '__init__', '__iter__', '__le__', '__len__', '__lt__', '__mod__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__rmul__', '__setattr__', '__sizeof__', '__str__', '__subclasshook__', 'casefold', 'center', 'count', 'encode', 'endswith', 'find', 'format', 'format_map', 'index', 'isalnum', 'isalpha', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'partition', 'removeprefix', 'removesuffix', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper']
True
goodbye
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$
```

- 可以利用 try 语句拦截报错，避免退出，将流程 (flow) 转入 except 语句

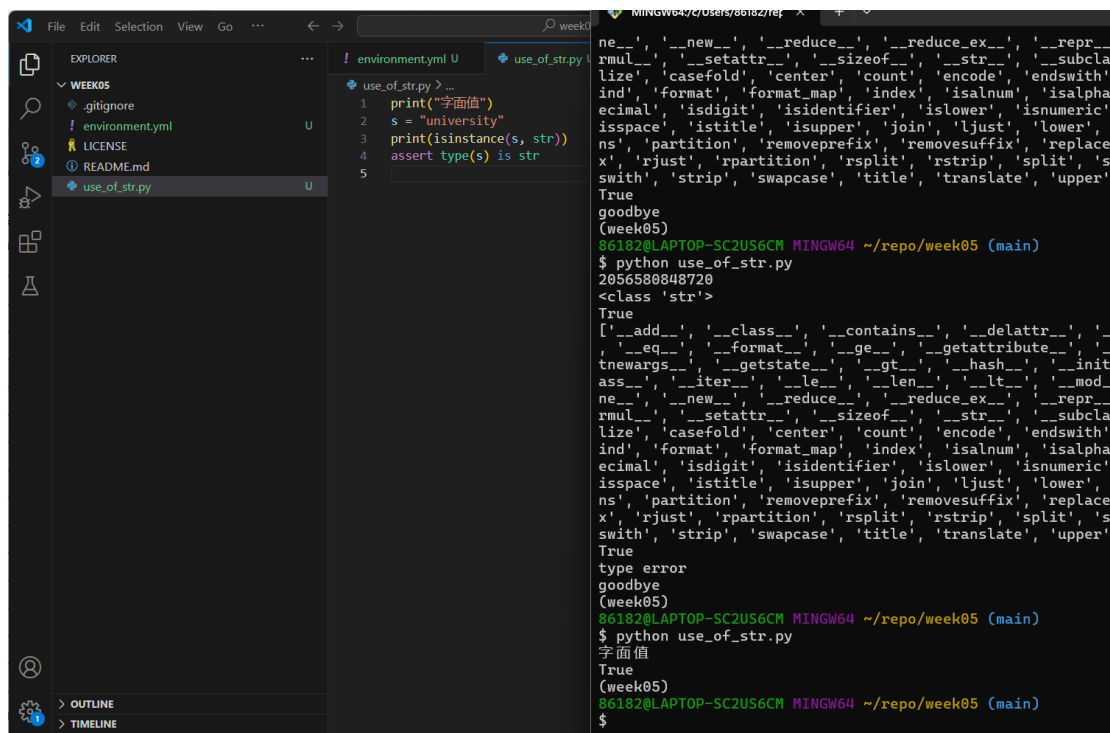
```
File Edit Selection View Go ... use_of_str.py
EXPLORER
WEEK05
.gitignore
environment.yml
LICENSE
README.md
use_of_str.py
environment.yml U
use_of_str.py U
1 a = "hello"
2 x = id(a)
3 print(x)
4 print(type(a))
5 print(isinstance(a, str))
6 print(dir(a))
7 print(isinstance(a, (str, list)))
8 try:
9     assert isinstance(a, list)
10 except AssertionError:
11     print("type error")
12 print("goodbye")
13

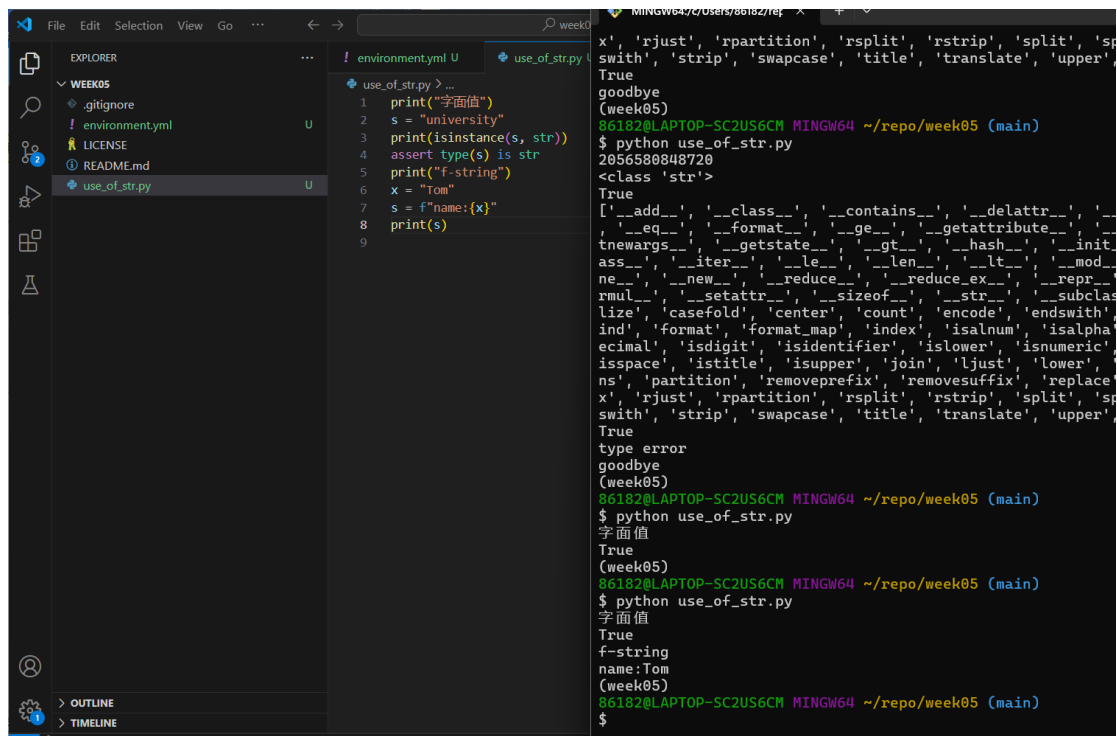
True
['__add__', '__class__', '__contains__', '__delattr__', '__dict__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getstate__', '__gt__', '__hash__', '__init__', '__iter__', '__le__', '__len__', '__lt__', '__mod__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__rmul__', '__setattr__', '__sizeof__', '__str__', '__subclasshook__', 'casefold', 'center', 'count', 'encode', 'endswith', 'find', 'format', 'format_map', 'index', 'isalnum', 'isalpha', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'partition', 'removeprefix', 'removesuffix', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper']
True
type error
goodbye
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2056580848720
<class 'str'>
True
type error
goodbye
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$
```

- 可以调用 breakpoint() 函数暂停程序运行，进入 pdb 调试 (debug) 模式



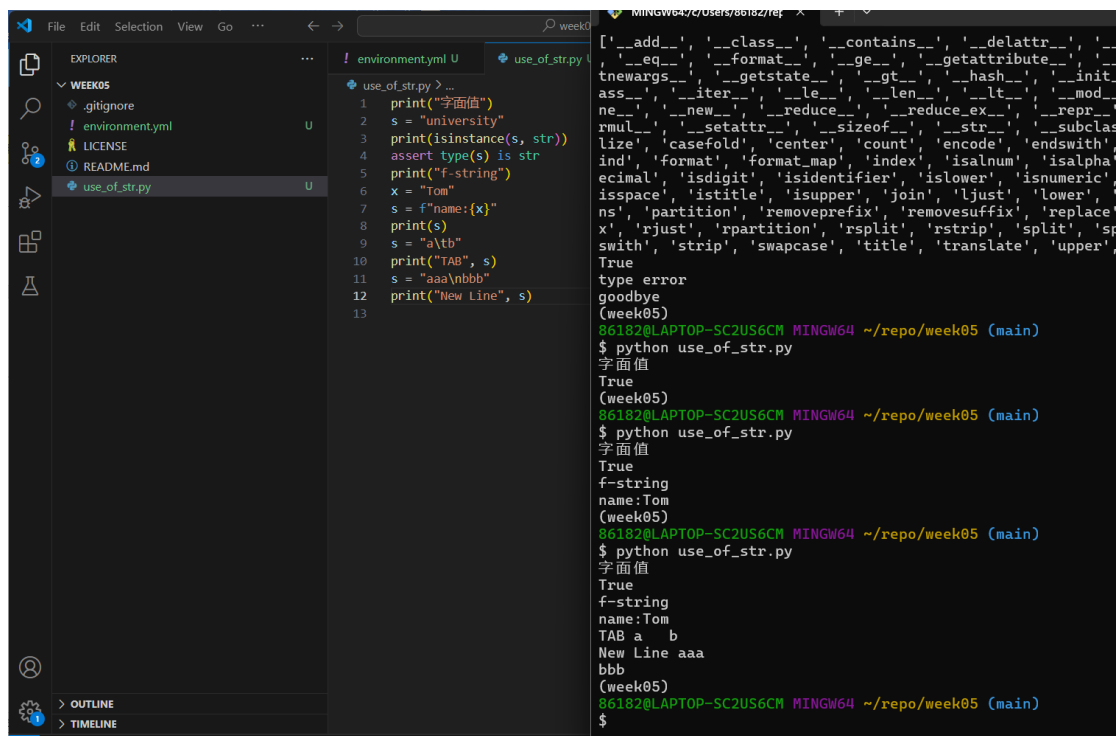
4. 对于 每一个上述要求掌握的对象类型 (将来遇到新的对象类型也应该如此), 我们首先应该熟悉如何通过 **表达式** (expression) 得到他们的 **实例** (instance), 一般包括以下途径:
- 字面值 (literal) (包括 f-string 语法)





```
1 print("字面值")
2 s = "university"
3 print(isinstance(s, str))
4 assert type(s) is str
5 print("f-string")
6 x = "Tom"
7 s = f"name:{x}"
8 print(s)
9
```

```
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
2056580848720
<class 'str'>
True
type error
goodbye
(word05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
(word05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
(word05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$
```



```
1 print("字面值")
2 s = "university"
3 print(isinstance(s, str))
4 assert type(s) is str
5 print("f-string")
6 x = "Tom"
7 s = f"name:{x}"
8 print(s)
9 s = "a\tb"
10 print("TAB", s)
11 s = "aaa\nbbb"
12 print("New Line", s)
13
```

```
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
(word05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
(word05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
(word05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$
```

。 推 导 式 (comprehension) (仅 限 list 、 dict 、 set

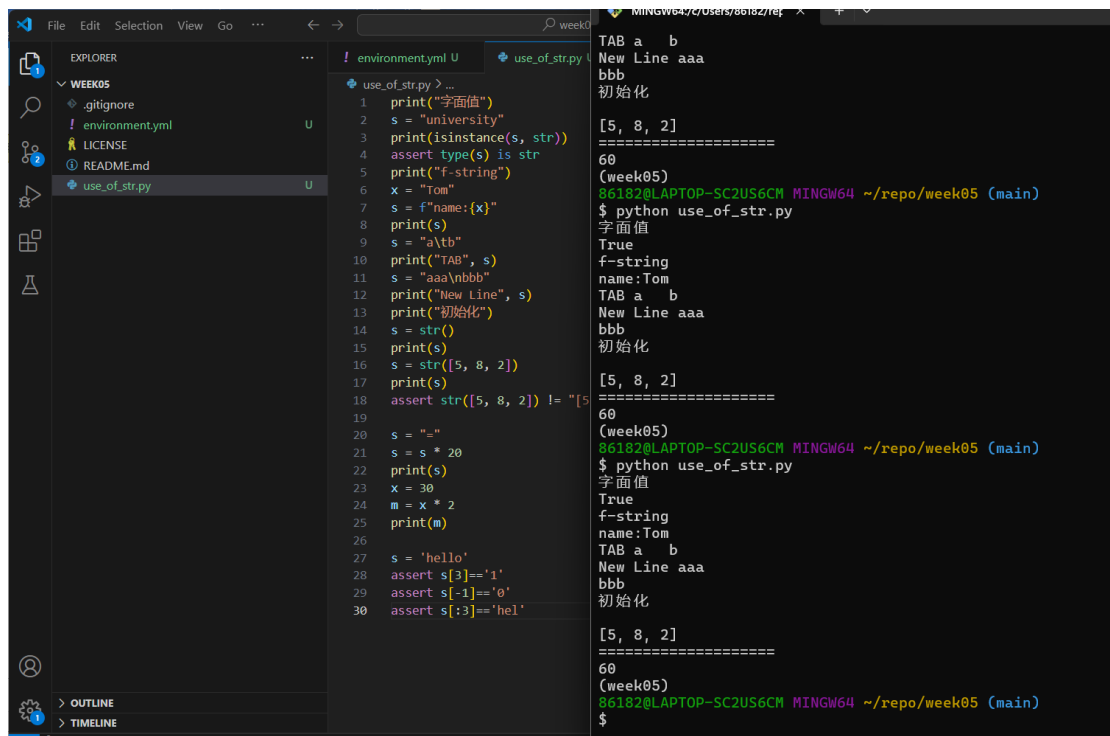

```
1 print("字面值")
2 s = "university"
3 print(isinstance(s, str))
4 assert type(s) is str
5 print("f-string")
6 x = "Tom"
7 s = f"name:{x}"
8 print(s)
9 s = "a\tb"
10 print("TAB", s)
11 s = "aaa\nbbb"
12 print("New Line", s)
13 print("初始化")
14 s = str()
15 print(s)
16 s = str([5, 8, 2])
17 print(s)
18 assert str([5, 8, 2]) != "[5"
19
20 s = "="
21 s = s * 20
22 print(s)
23
```

```
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
Traceback (most recent call last):
  File "C:\Users\86182\repo\week05\use_of_str.py", line 18, in <module>
    assert str([5, 8, 2]) == "[5,8,2]"
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
AssertionError
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$
```

```
1 print("字面值")
2 s = "university"
3 print(isinstance(s, str))
4 assert type(s) is str
5 print("f-string")
6 x = "Tom"
7 s = f"name:{x}"
8 print(s)
9 s = "a\tb"
10 print("TAB", s)
11 s = "aaa\nbbb"
12 print("New Line", s)
13 print("初始化")
14 s = str()
15 print(s)
16 s = str([5, 8, 2])
17 print(s)
18 assert str([5, 8, 2]) != "[5"
19
20 s = "="
21 s = s * 20
22 print(s)
23 x = 30
24 m = x * 2
25 print(m)
26
```

```
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
60
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$
```

。 索引值 (subscription)

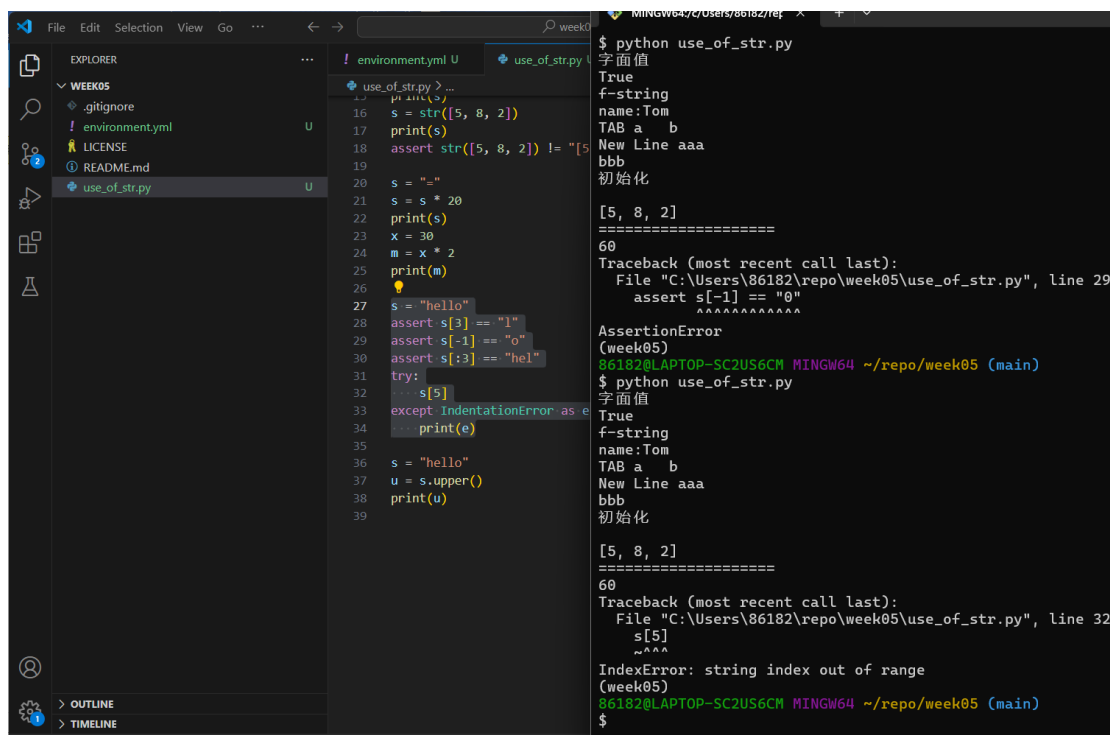


```
File Edit Selection View Go ... week05
EXPLORER
WEEK05
.gitignore
! environment.yml U
LICENSE
README.md
use_of_str.py U

! environment.yml U
use_of_str.py > ...
1 print("字面值")
2 s = "university"
3 print(isinstance(s, str))
4 assert type(s) is str
5 print("f-string")
6 x = "Tom"
7 s = f"name:{x}"
8 print(s)
9 s = "a\tb"
10 print("TAB", s)
11 s = "aaa\nbbb"
12 print("New Line", s)
13 print("初始化")
14 s = str()
15 print(s)
16 s = str([5, 8, 2])
17 print(s)
18 assert str([5, 8, 2]) != "[5
19
20 s = "="
21 s = s * 20
22 print(s)
23 x = 30
24 m = x * 2
25 print(m)
26
27 s = "hello"
28 assert s[3] == "l"
29 assert s[-1] == "o"
30 assert s[:3] == "hel"
31
```

```
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
60
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
60
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
60
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$
```

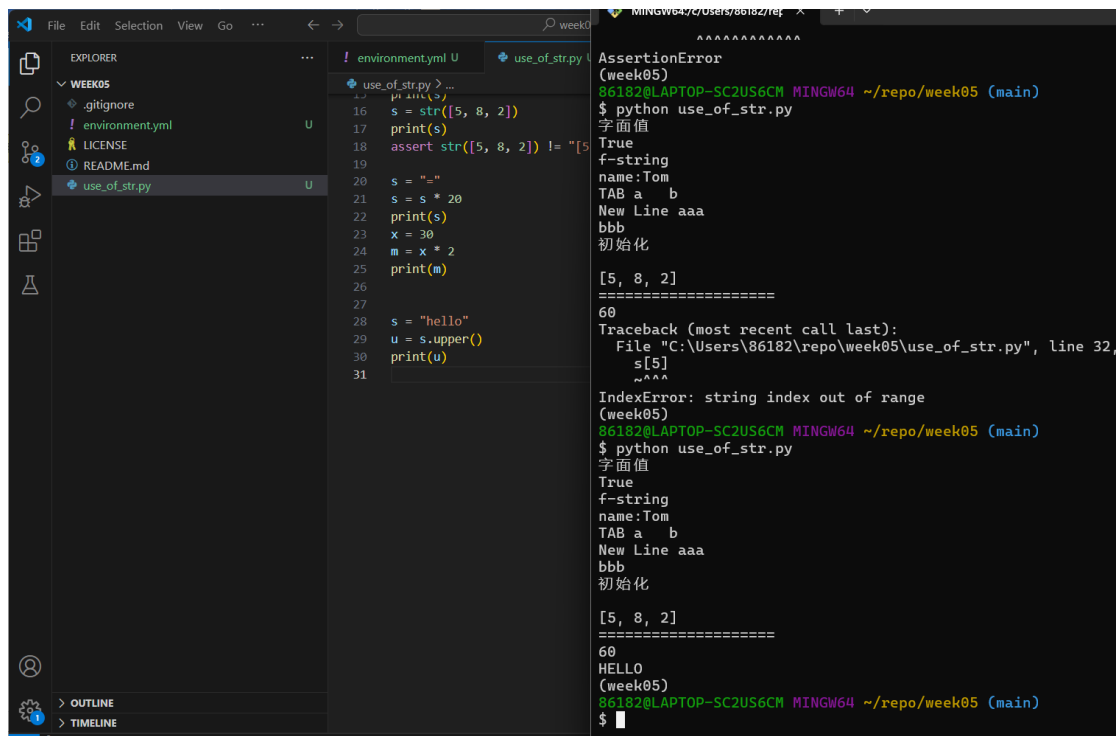
- 返回值 (return value of function/method call)



```
File Edit Selection View Go ... week05
EXPLORER
WEEK05
.gitignore
! environment.yml U
LICENSE
README.md
use_of_str.py U

! environment.yml U
use_of_str.py > ...
16 s = str([5, 8, 2])
17 print(s)
18 assert str([5, 8, 2]) != "[5
19
20 s = "="
21 s = s * 20
22 print(s)
23 x = 30
24 m = x * 2
25 print(m)
26
27 s = "hello"
28 assert s[3] == "l"
29 assert s[-1] == "o"
30 assert s[:3] == "hel"
31 try:
32     s[5]
33 except IndexError as e:
34     print(e)
35
36 s = "hello"
37 u = s.upper()
38 print(u)
39
```

```
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
60
Traceback (most recent call last):
  File "C:\Users\86182\repo\week05\use_of_str.py", line 29, in <module>
    assert s[-1] == "o"
    ^^^^^^^^^^^^^^^^^
AssertionError
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
60
Traceback (most recent call last):
  File "C:\Users\86182\repo\week05\use_of_str.py", line 32, in <module>
    s[5]
    ~^^^
IndexError: string index out of range
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$
```



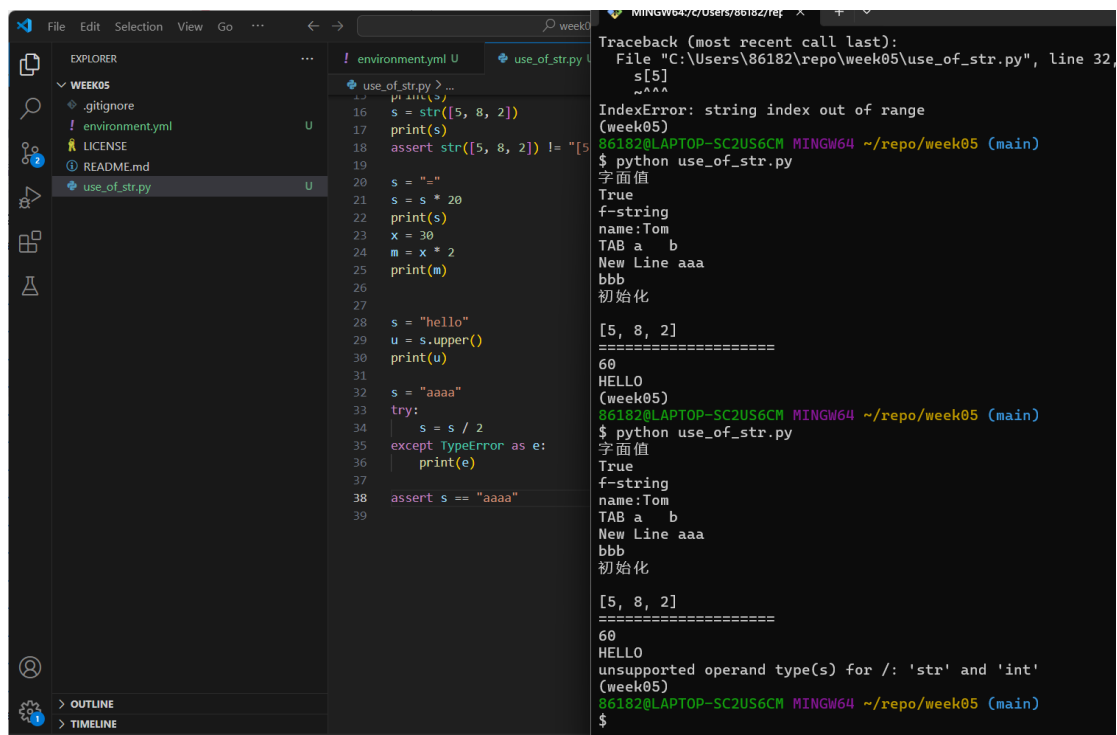
```
File Edit Selection View Go ... week05
EXPLORER
WEEK05
.gitignore
! environment.yml
LICENSE
README.md
use_of_str.py
! environment.yml U
use_of_str.py U
16 s = str([5, 8, 2])
17 print(s)
18 assert str([5, 8, 2]) != "[5
19
20 s = "="
21 s = s * 20
22 print(s)
23 x = 30
24 m = x * 2
25 print(m)
26
27
28 s = "hello"
29 u = s.upper()
30 print(u)
31

MINGW64/C:/Users/86182/repo/week05
AssertionError
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
60
Traceback (most recent call last):
  File "C:\Users\86182\repo\week05\use_of_str.py", line 32, in
    s[5]
    ~^^^
IndexError: string index out of range
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
60
HELLO
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$
```

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 (==)



```
File Edit Selection View Go ... week05
EXPLORER
WEEK05
.gitignore
! environment.yml
LICENSE
README.md
use_of_str.py
! environment.yml U
use_of_str.py U
16 s = str([5, 8, 2])
17 print(s)
18 assert str([5, 8, 2]) != "[5
19
20 s = "="
21 s = s * 20
22 print(s)
23 x = 30
24 m = x * 2
25 print(m)
26
27
28 s = "hello"
29 u = s.upper()
30 print(u)
31
32 s = "aaaa"
33 try:
34     s = s / 2
35 except TypeError as e:
36     print(e)
37
38 assert s == "aaaa"
39

MINGW64/C:/Users/86182/repo/week05
Traceback (most recent call last):
  File "C:\Users\86182\repo\week05\use_of_str.py", line 32, in
    s[5]
    ~^^^
IndexError: string index out of range
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
60
HELLO
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
60
HELLO
unsupported operand type(s) for /: 'str' and 'int'
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$
```

- The screenshot shows a Windows IDE with a dark theme. On the left is the Explorer pane showing a project structure with files like .gitignore, environment.yml, LICENSE, README.md, and use_of_str.py. The main editor displays the content of use_of_str.py, which is a Python script. The script defines a list s, prints it, asserts its length, and then iterates over it. The output pane on the right shows the execution of the script, which results in a ValueError: unsupported operand type(s) for /: 'str' and 'int'. The error message is repeated twice, corresponding to the two iterations of the loop.

```

16 s = str([5, 8, 2])
17 print(s)
18 assert str([5, 8, 2]) != "[5
19
20 s = ""
21 s = s * 20
22 print(s)
23 x = 30
24 m = x * 2
25 print(m)
26
27
28 s = "hello"
29 u = s.upper()
30 print(u)
31
32 s = "aaaa"
33 try:
34     s = s / 2
35 except TypeError as e:
36     print(e)
37
38 assert s == "aaaa"
39 print("abc" > "ABC")
40
41 s = "book"
42 print(iter(s))
43 breakpoint()
44 for c in s:
45     print(c)
46
=====
60
HELLO
unsupported operand type(s) for /: 'str' and 'int'
True
<str_ascii_iterator object at 0x000001BC30817FD0>
(week05)
861820\APTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化

[5, 8, 2]
=====
60
HELLO
unsupported operand type(s) for /: 'str' and 'int'
True
<str_ascii_iterator object at 0x0000019EFB4FC0D0>
> c:\users\86182\repo\week05\use_of_str.py(44)<module>()
-> for c in s:
(Pdb) g = iter(s)
(Pdb) p g
<str_ascii_iterator object at 0x0000019EFB4FC640>
(Pdb) p next(g)
'b'
(Pdb) p next(g)
'o'
(Pdb) p next(g)
'o'
(Pdb) p next(g)
'o'
(Pdb) p next(g)
'k'
(Pdb) p next(g)
*** StopIteration
(Pdb)

```

- The screenshot shows a VS Code editor with a Python file named `use_of_str.py`. The code defines a class `BdbQuit` with a `dispatch_line` method and a `quitting` attribute. It then uses `pdb.set_trace()` to debug a loop that prints the length of a string `s` and its characters. The terminal output shows the script running and the debugger pausing at the breakpoint, displaying the current state of variables like `s` and `c`.

```

File "D:\anana\damangshe\envs\week05\Lib\bdb.py", line 16
    return self.dispatch_line(frame)
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
File "D:\anana\damangshe\envs\week05\Lib\bdb.py", line 17
    if self.quitting: raise BdbQuit
    ^^^^^^^^^^^^^^^^^
bdb.BdbQuit
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字符串值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化

[5, 8, 2]
=====
60
HELLO
unsupported operand type(s) for /: 'str' and 'int'
True
<str_ascii_iterator object at 0x000001DBABB3C250>
> c:\users\86182\repo\week05\use_of_str.py(45)<module>()
-> for c in s:
(Pdb) c
b
o
o
k
4
(week05)
86182@LAPTOP-SC2US6CM MINGW64 ~/repo/week05 (main)
$

```

- 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)

```
29 u = s.upper()
30 print(u)
31
32 s = "aaaa"
33 try:
34     s = s / 2
35 except TypeError as e:
36     print(e)
37
38 assert s == "aaaa"
39 print("abc" > "ABC")
40
41 s = "book"
42 print(iter(s))
43
44 breakpoint()
45 for c in s:
46     print(c)
47
48 print(len(s))
49
50 s = "book"
51 assert s[1:3] == "oo"
52
53 s = "the book of why took nooo"
54 print(s.capitalize())
55 print(s)
56 print(s.count("oo" == 3))
57
58 print("abc123", isalnum())
59
```

```
o
k
4
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$ python use_of_str.py
字面值
True
f-string
name:Tom
TAB a b
New Line aaa
bbb
初始化
[5, 8, 2]
=====
60
HELLO
unsupported operand type(s) for /: 'str' and 'int'
True
<str_ascii_iterator object at 0x000002D78967C4C0>
> c:\users\86182\repo\week05\use_of_str.py(45)<module>()
-> for c in s:
(Pdb) c
b
o
o
k
4
The book of why took nooo
the book of why took nooo
Traceback (most recent call last):
  File "C:\Users\86182\repo\week05\use_of_str.py", line 56,
    print(s.count("oo" == 3))
    ^^^^^^^^^^^^^^^^^^^^^^^^^
TypeError: must be str, not bool
(week05)
86182@LAPTOP-SC2U56CM MINGW64 ~/repo/week05 (main)
$
```

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