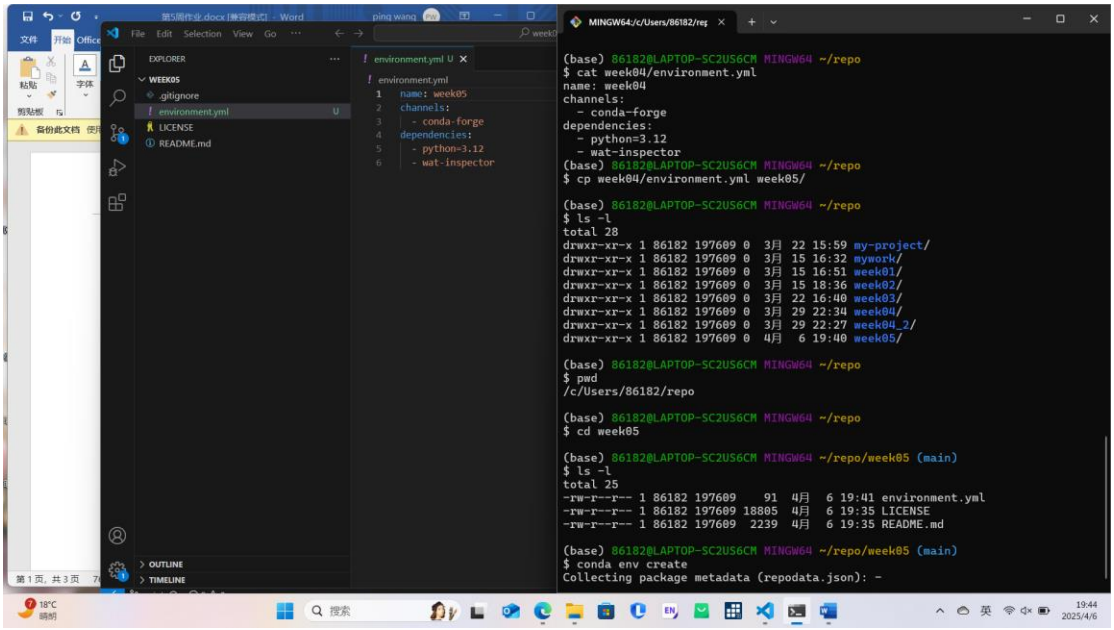
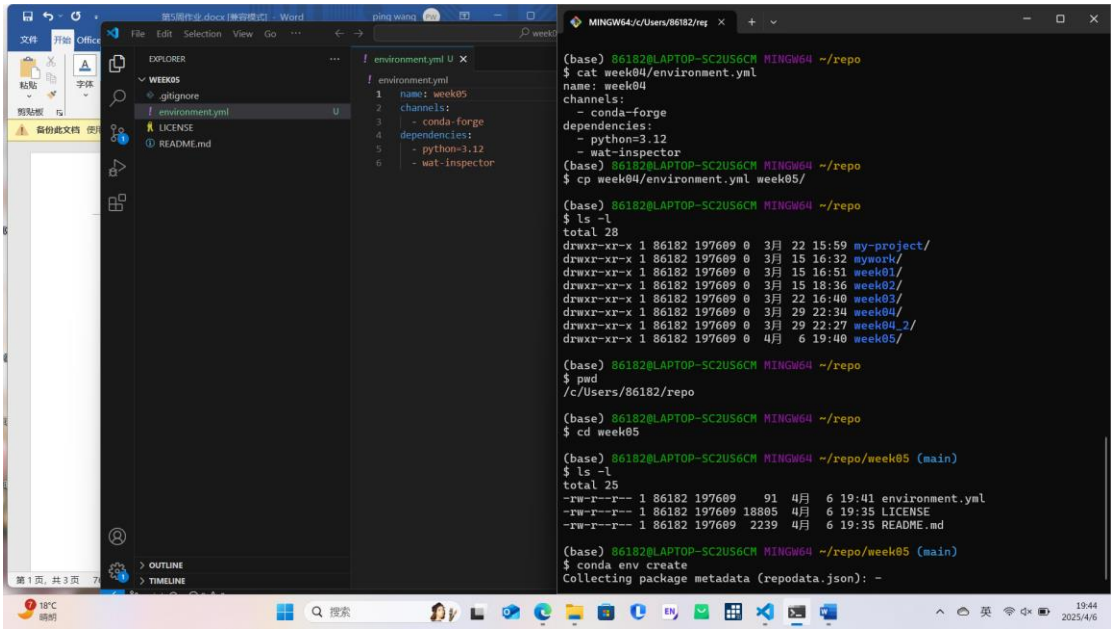


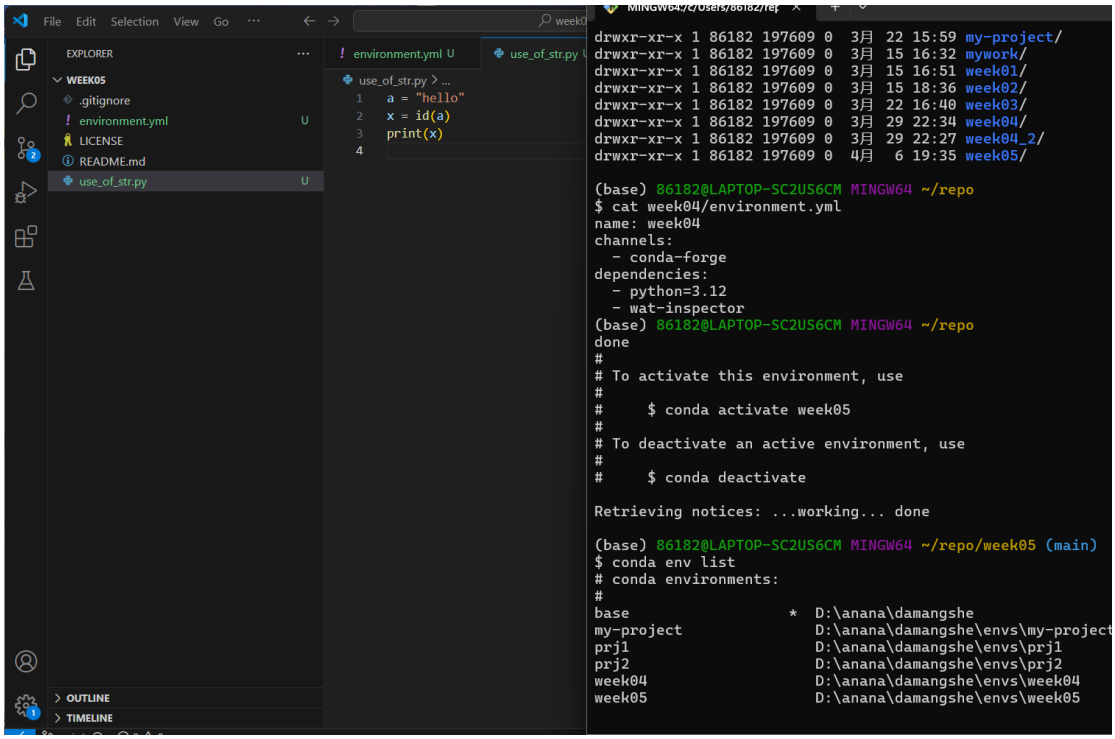
1. Fork 第 05 周打卡 仓库至你的名下，然后将你名下的这个仓库 Clone 到你的本地计算机
2. 用 VS Code 打开项目目录，新建一个 environment.yml 文件，指定安装 Python 3.12，然后运行 conda env create 命令创建 Conda 环境



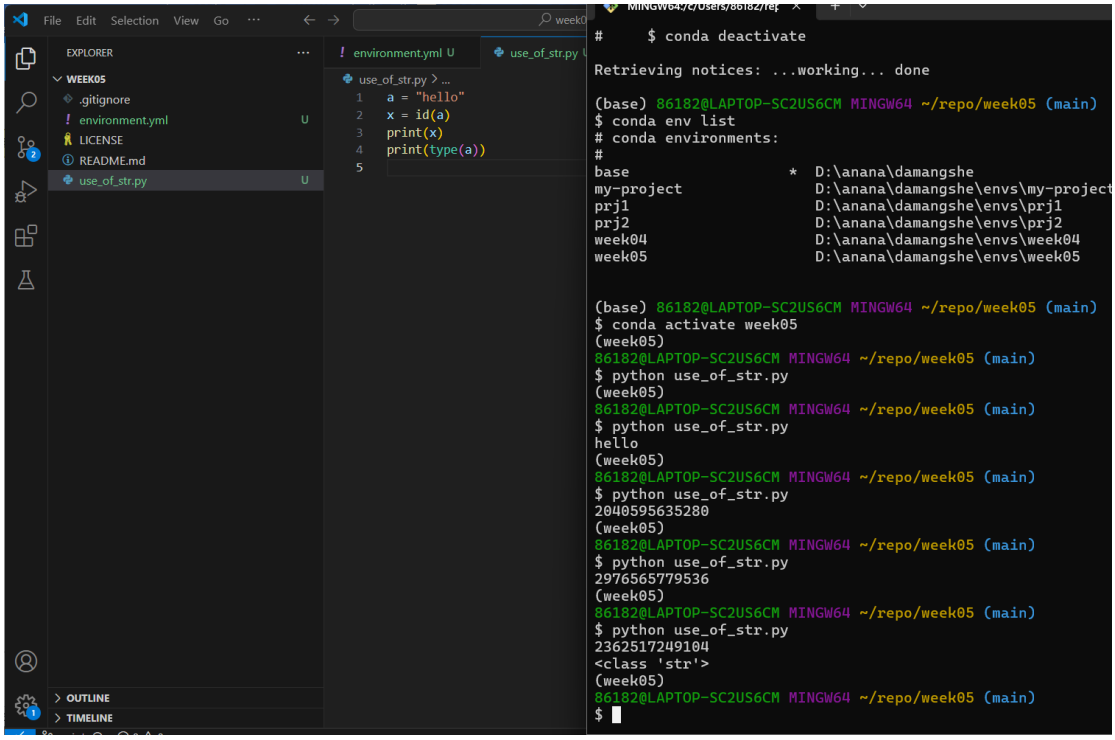
3. *逐个* 创建 use_of_{name}.py 文件，其中 {name} 替换为上述要求掌握的对象类型，例如 use_of_str.py:
    - 在全局作用域 (global scope) 内尝试键入 (活学活用) Python 代码，

亲手验证概念 (Proof of Concept, PoC)

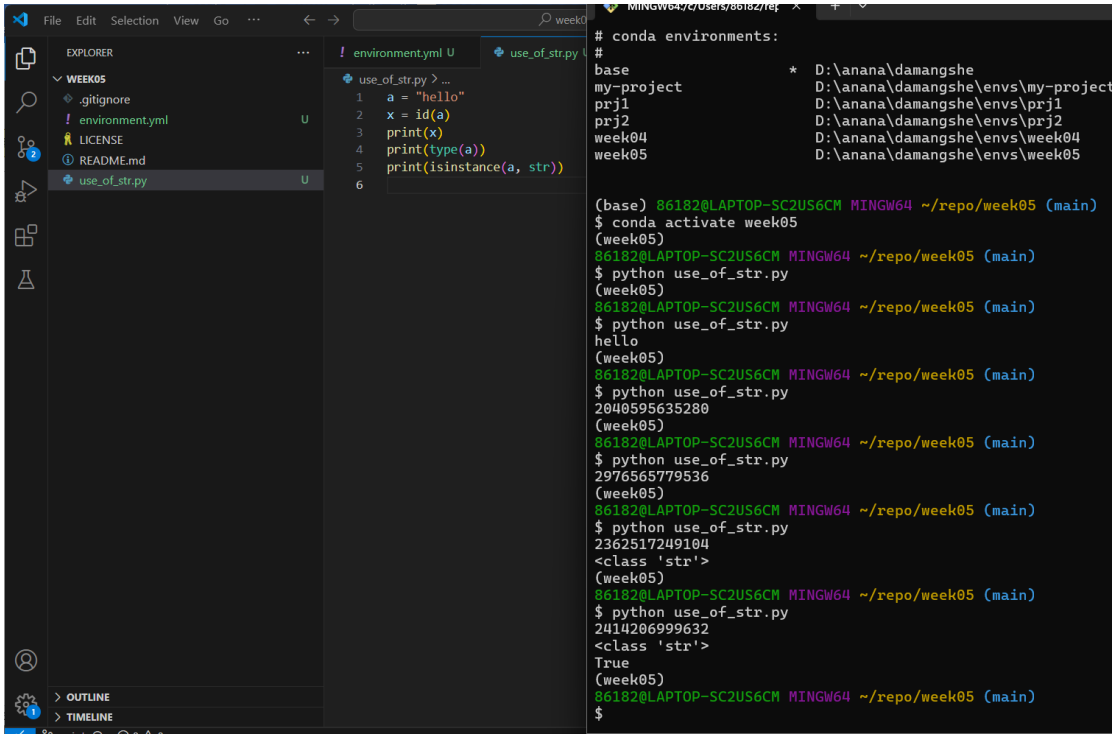
- 对于任何对象，都可以传给以下内置函数 (built-in function) 用于检视 (inspect):
  - id() -- 返回对象在虚拟内存中的地址（正整数），如果 id(a) == id(b)，那么 a is b (is 是个运算符)



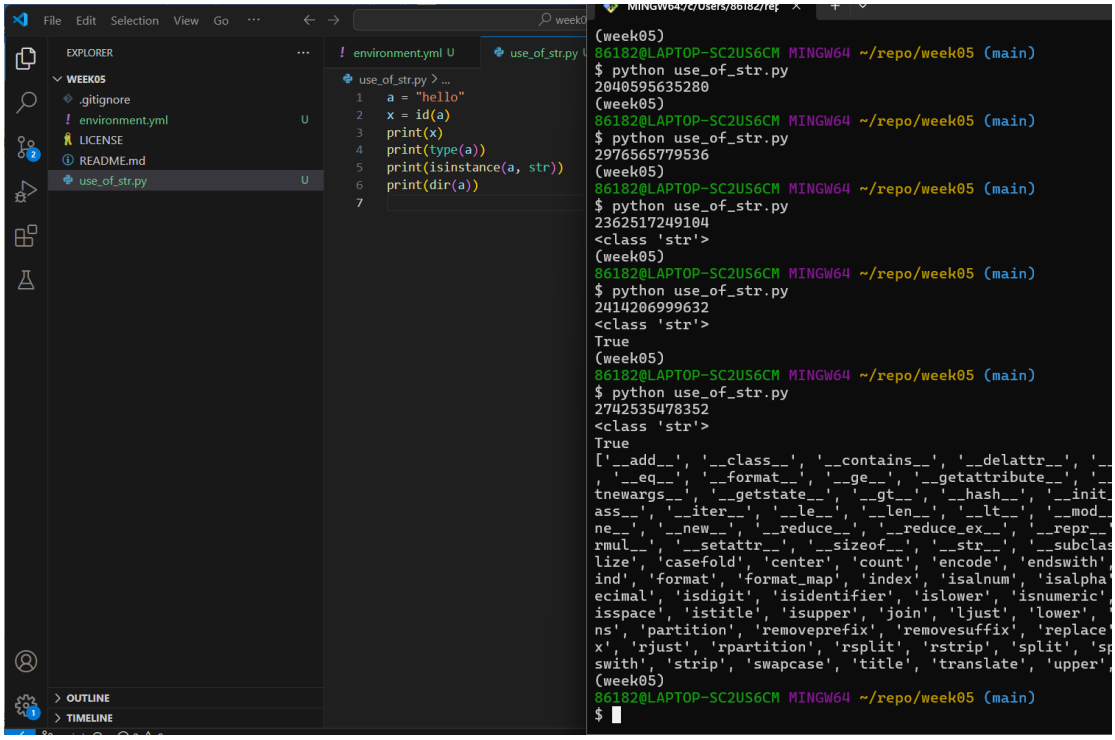
  - type() -- 返回对象的类型



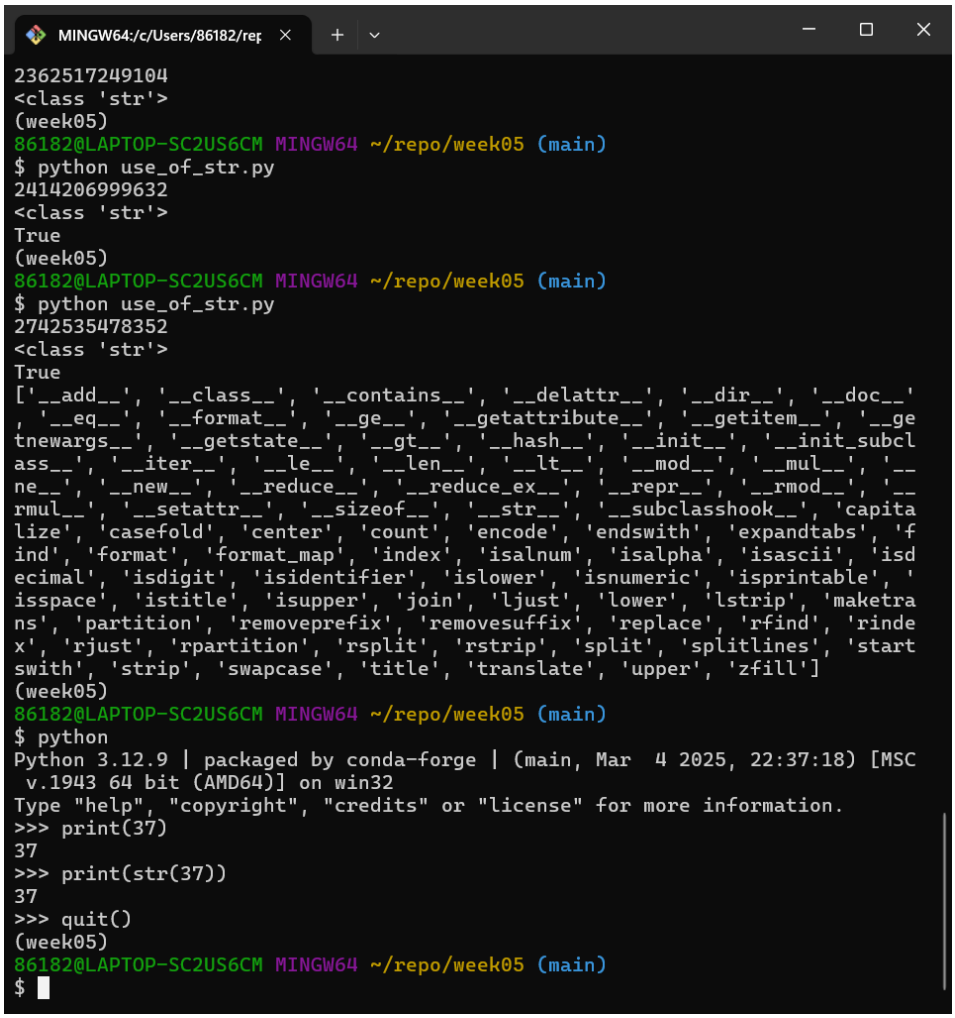
- isinstance() -- 判断对象是否属于某个 (或某些) 类型



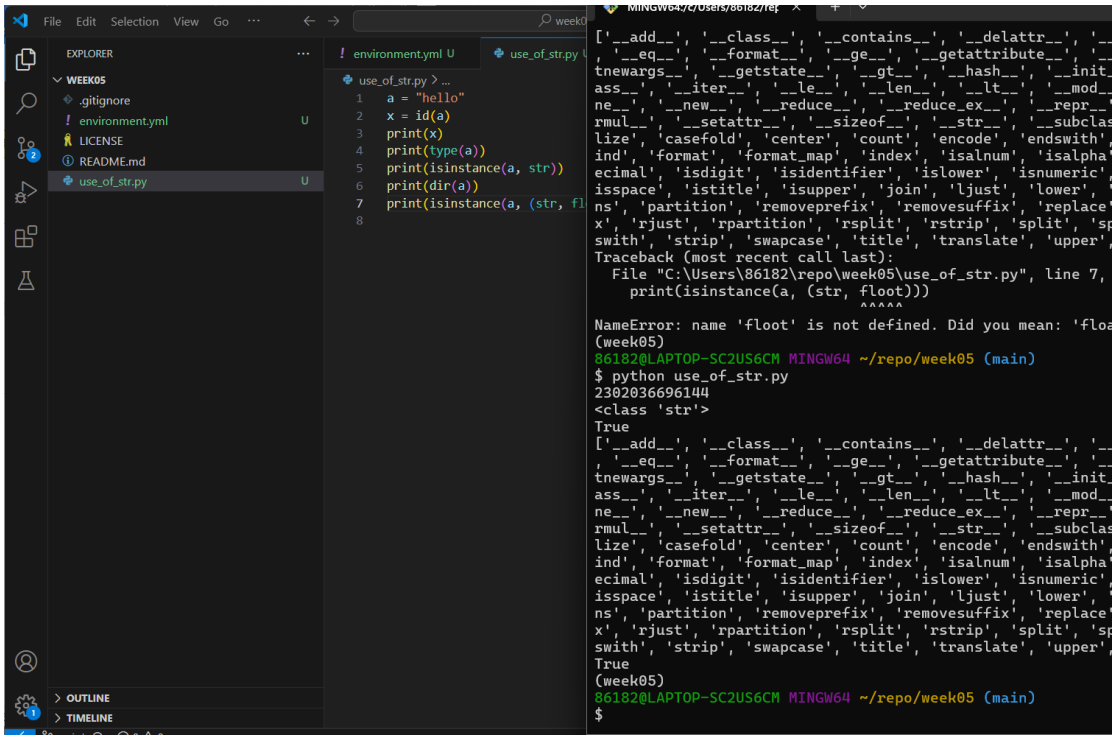
- dir() -- 返回对象所支持的属性 (attributes) 的名称列表



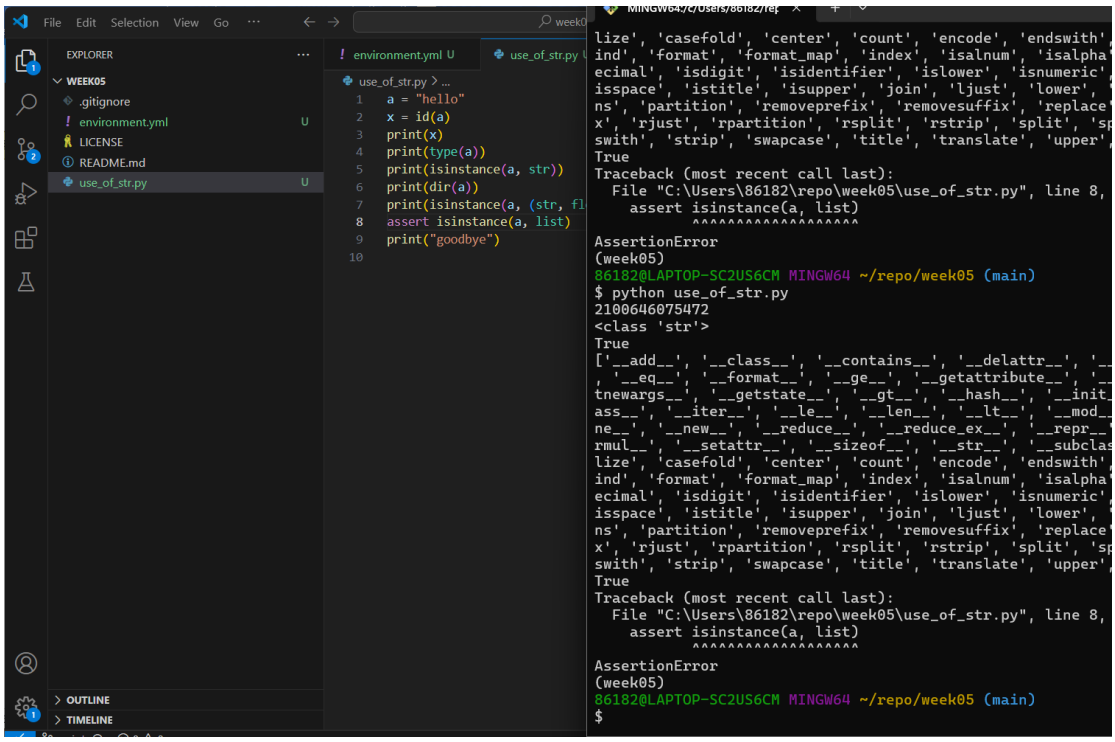
- str() -- 返回对象 print 时要显示在终端的字符串

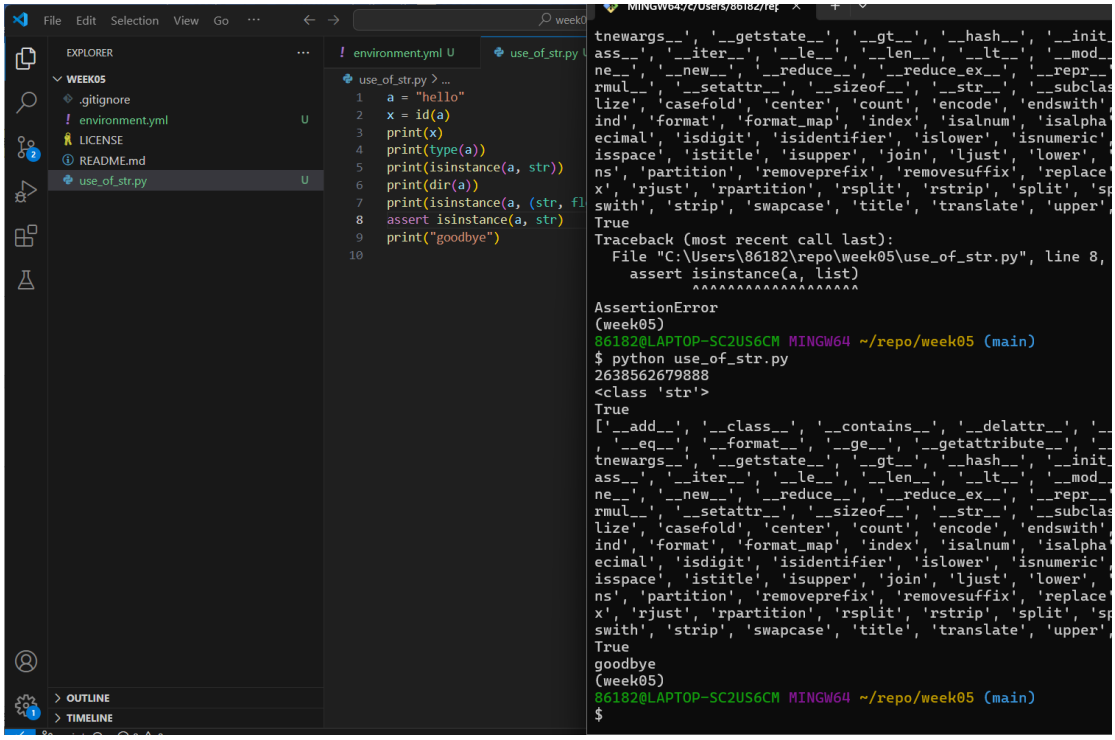


- 可以调用 print() 函数将表达式（expression）输出到终端，查看结果是否符合预期

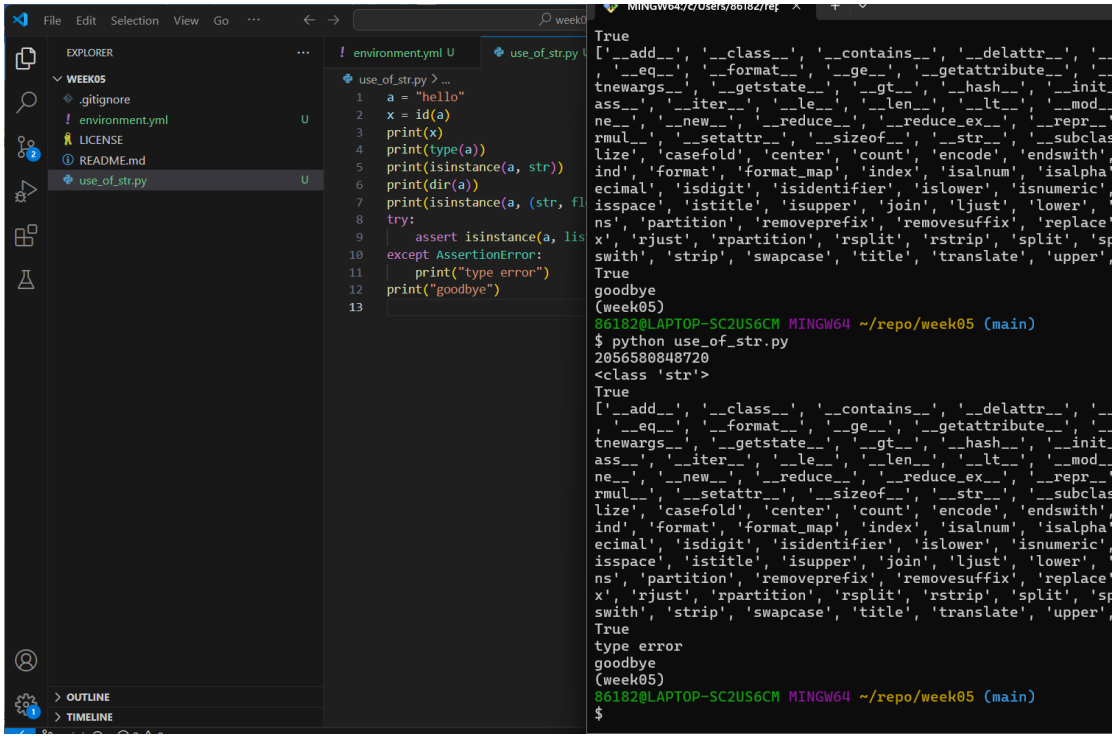


- 可以利用 assert 语句查验某个表达式（expression）为真，否则报错 (AssertionError) 退出

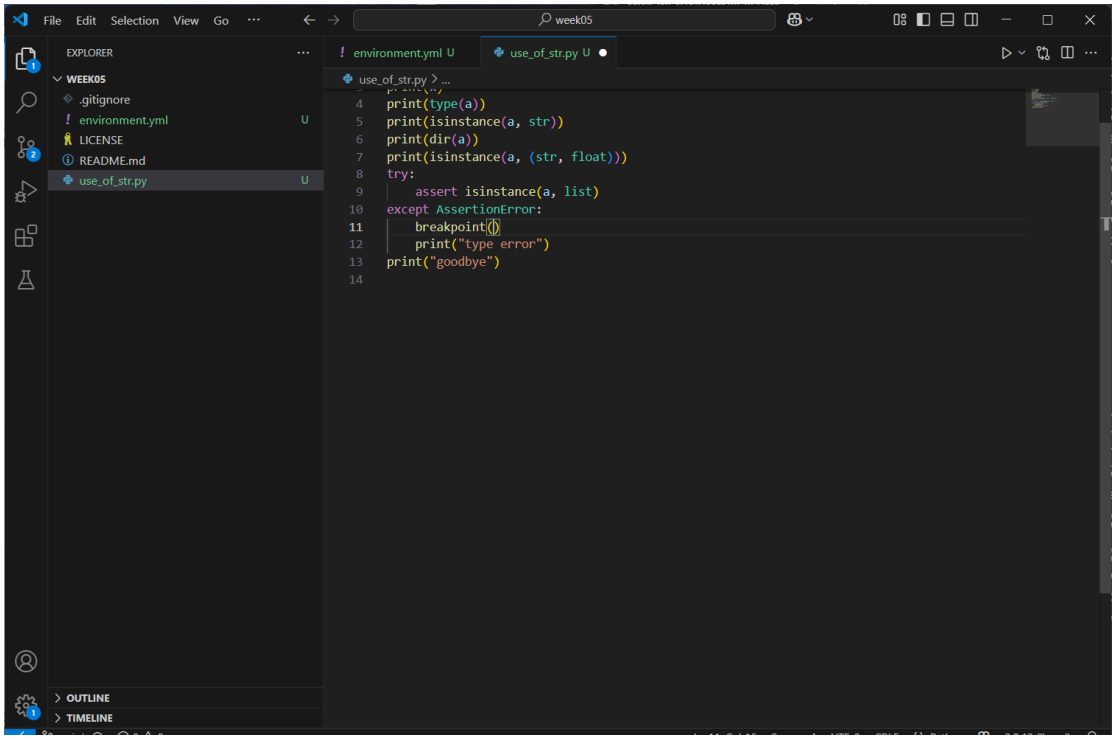




- 可以利用 try 语句拦截报错，避免退出，将流程（flow）转入 except 语句

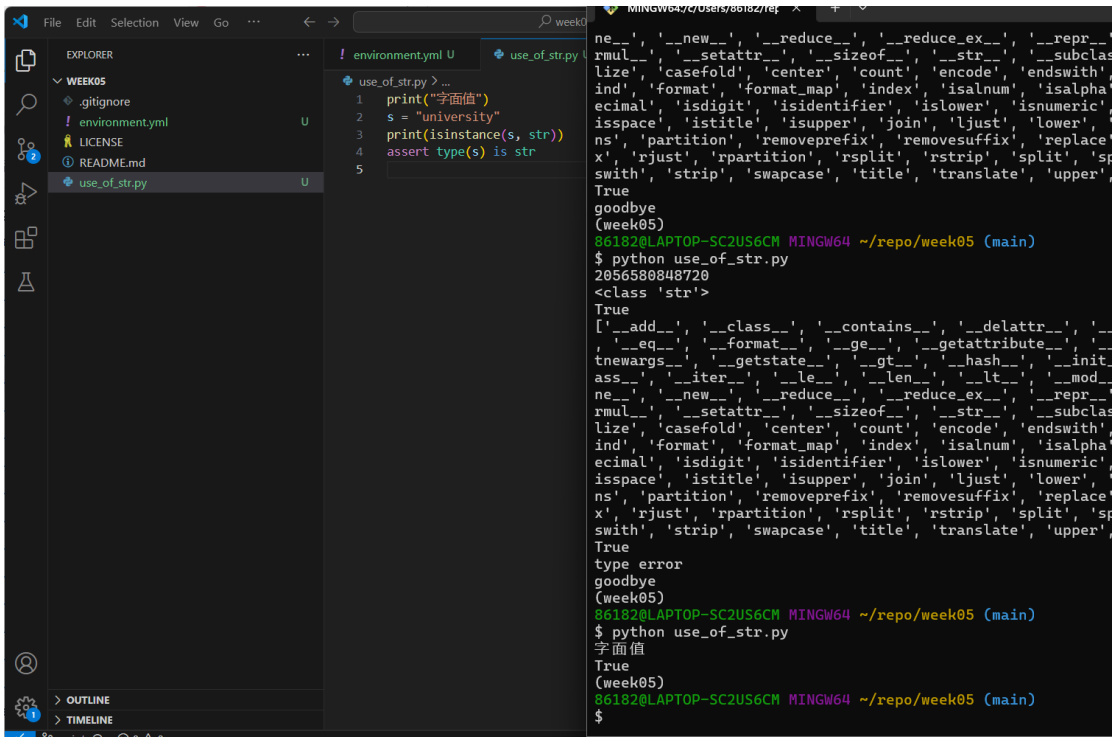


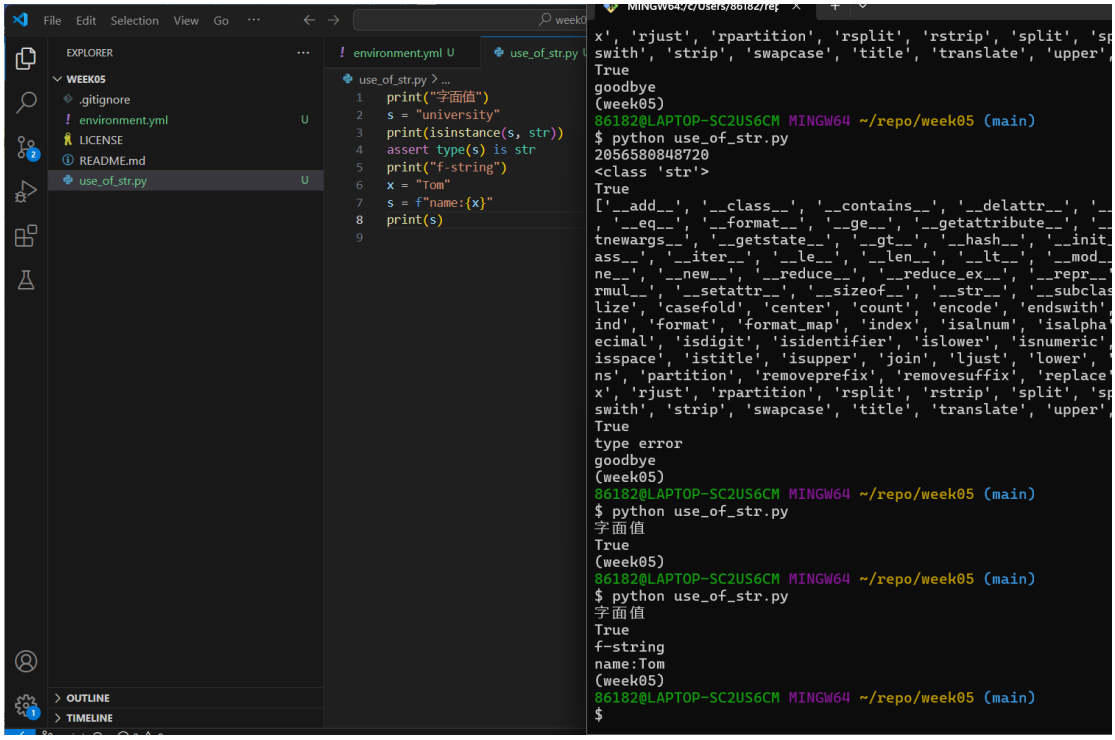
- 可以调用 breakpoint() 函数暂停程序运行，进入 pdb 调试（debug）模式

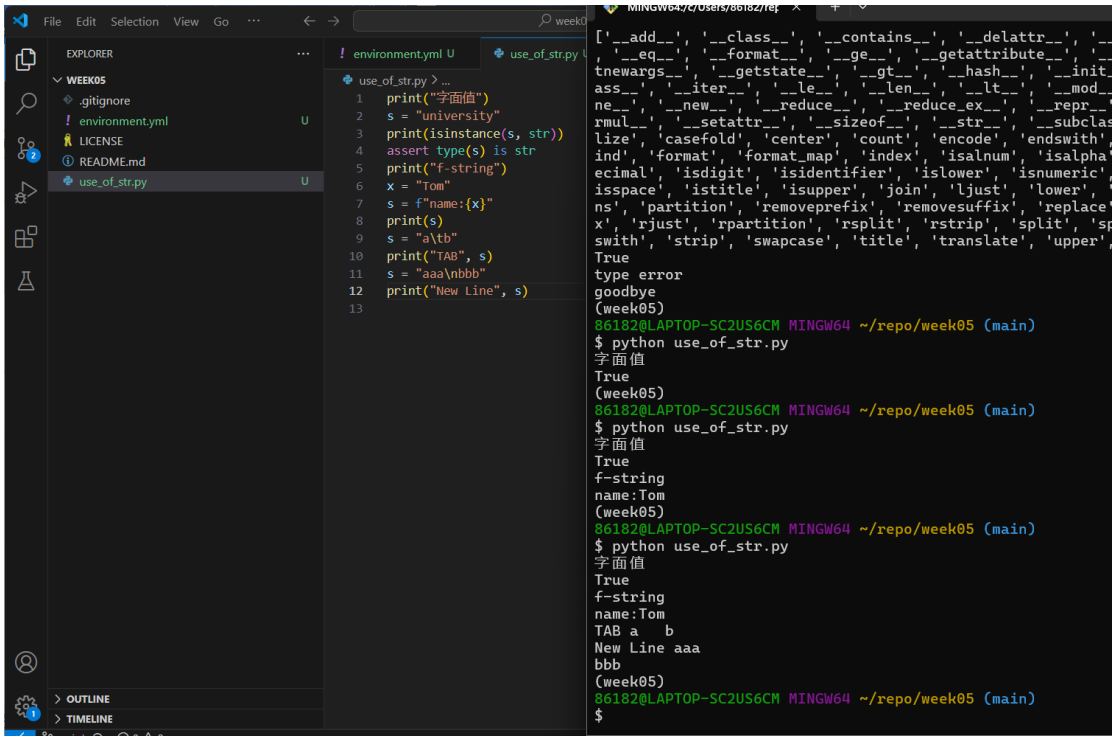


4. 对于 *每一个* 上述要求掌握的对象类型（将来遇到新的对象类型也应该如此），我们首先应该熟悉如何通过 **表达式** (expression) 得到他们的 **实例** (instance)，一般包括以下途径:

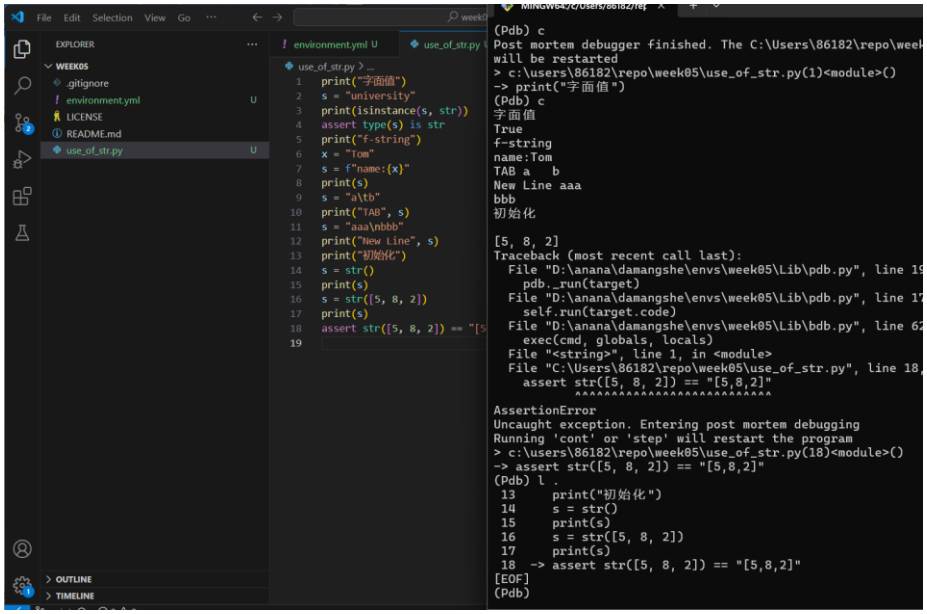
    - 字面值 (literal) (包括 f-string 语法)



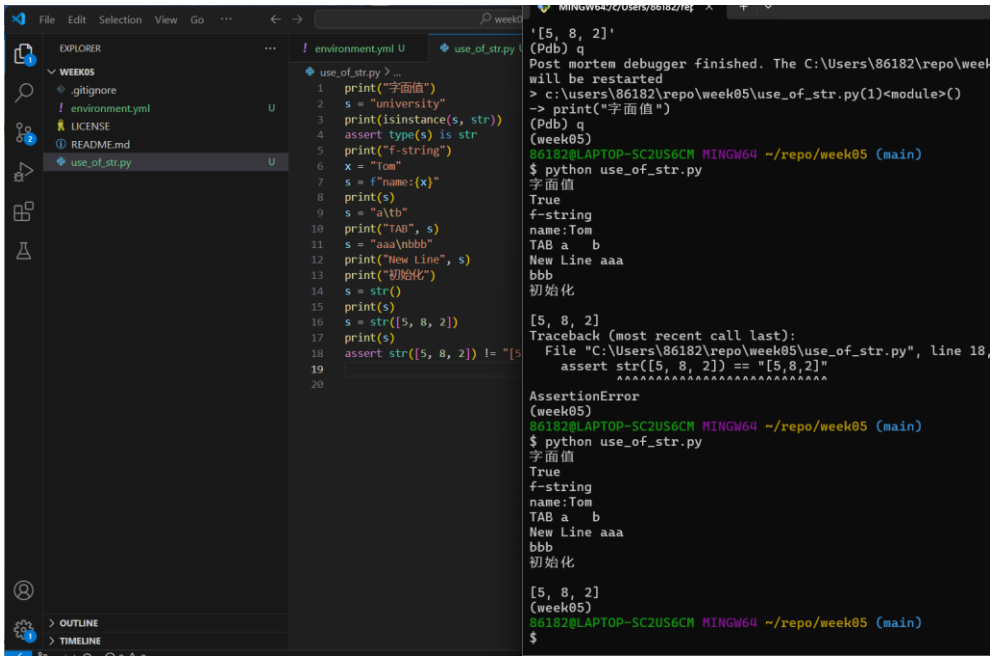




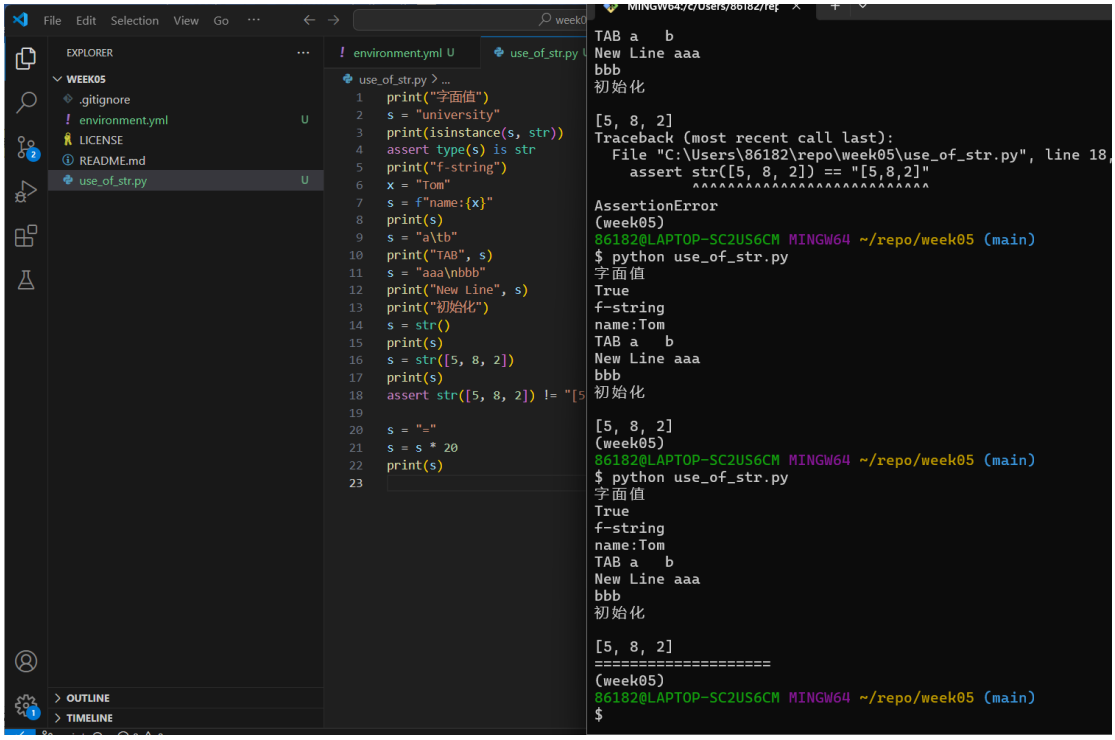
- 推导式 (comprehension)（仅限 list、dict、set

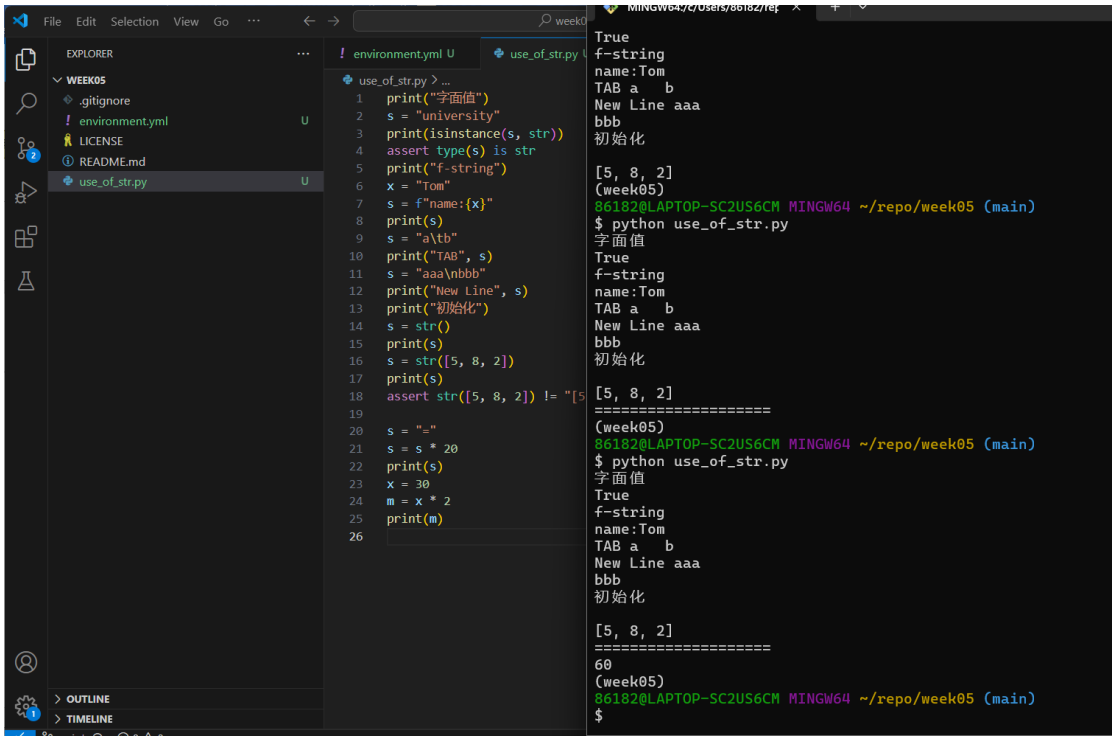


- 初始化 (init)

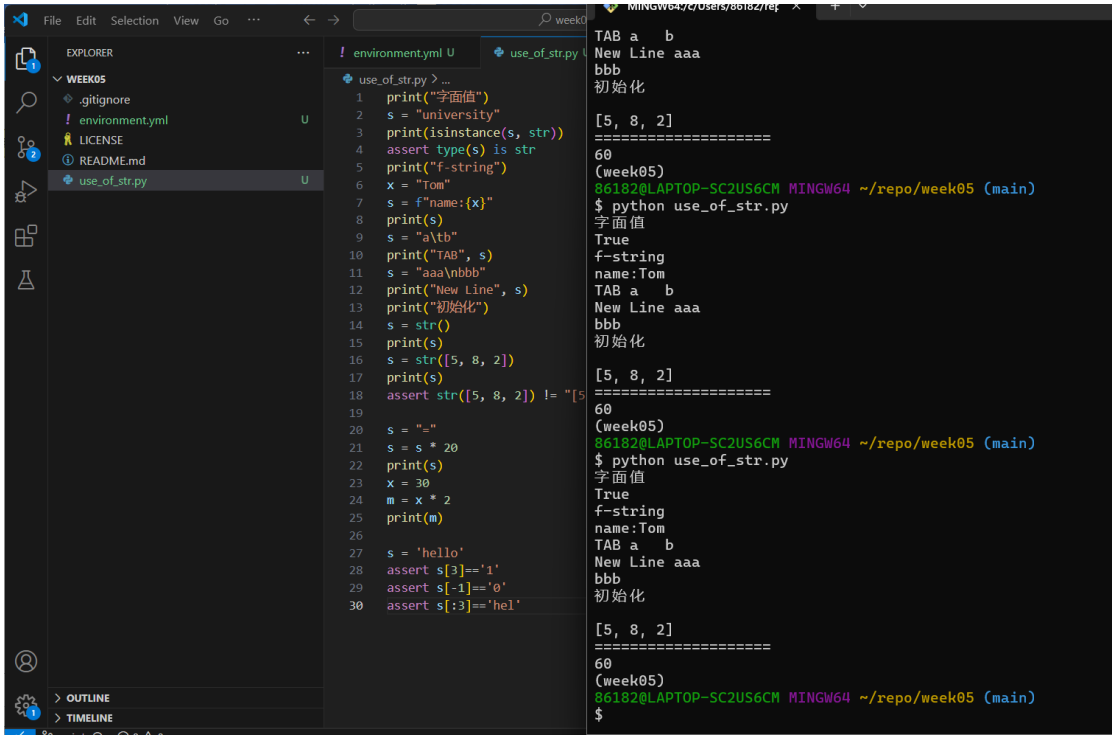


- 运算值 (operator)

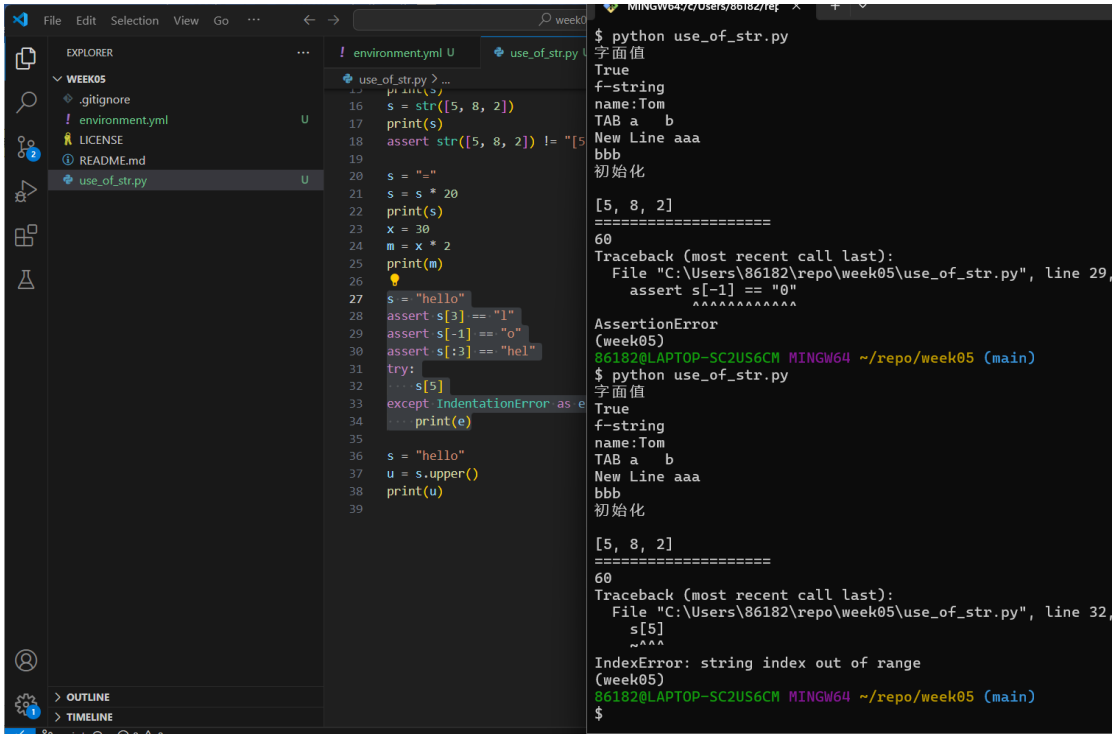


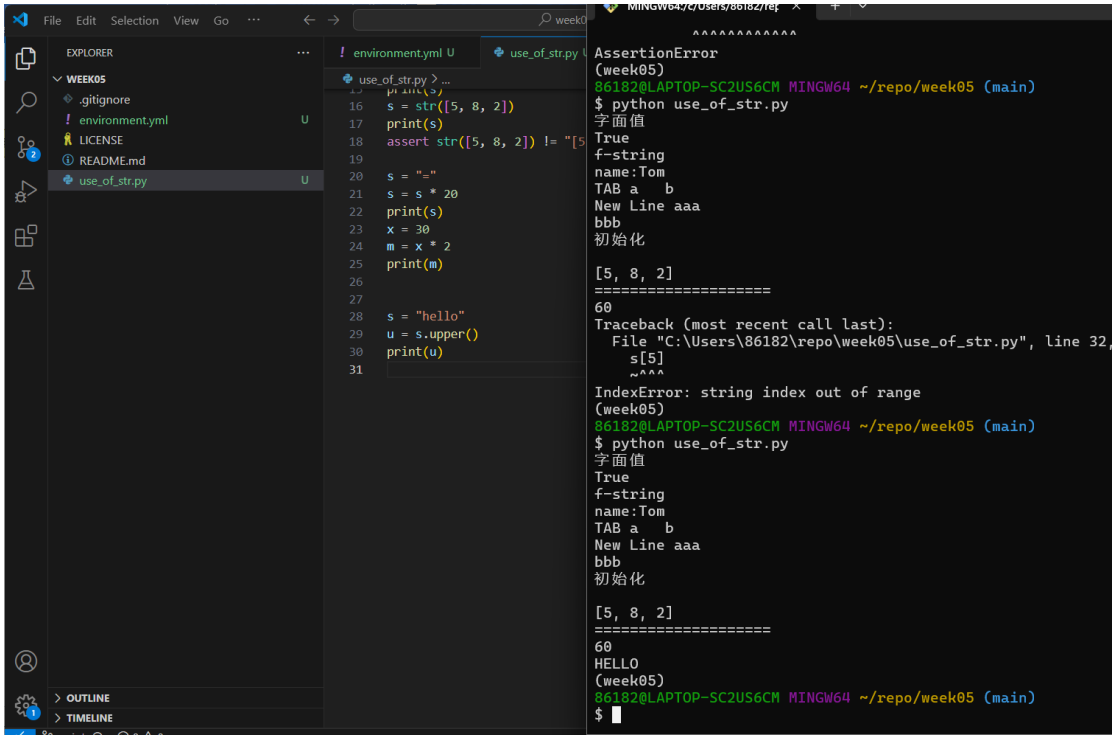


- 索引值 (subscription)

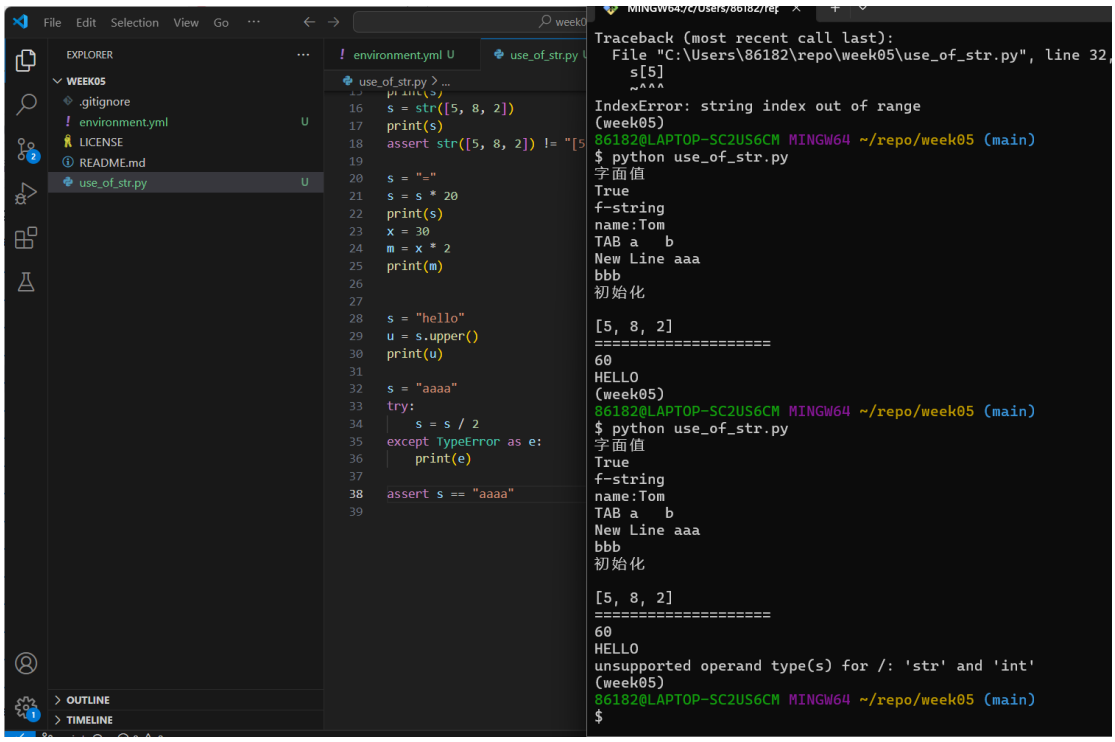


- 返回值 (return value of function/method call)

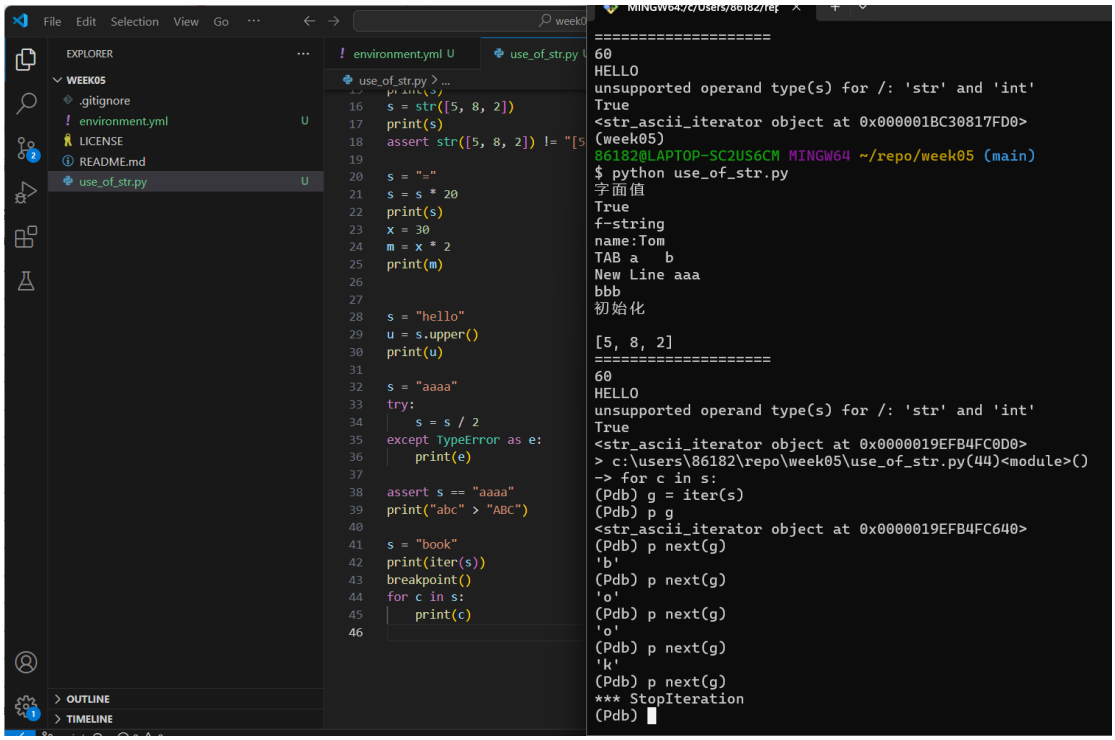




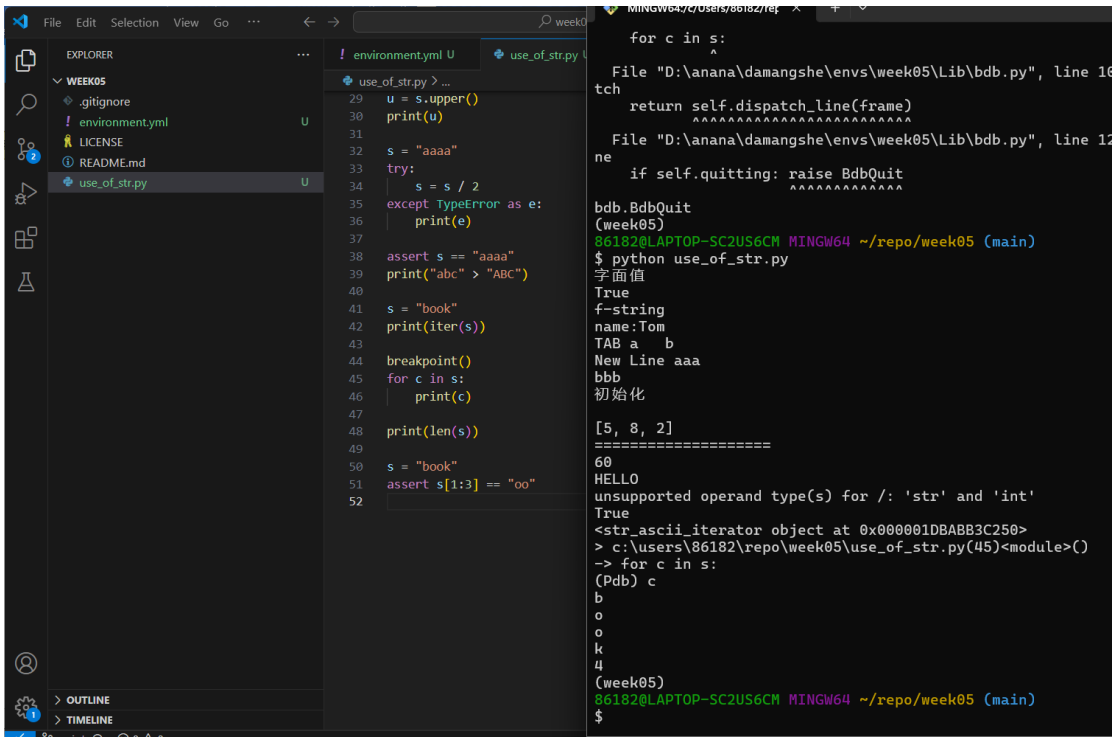
5. 对于 *每一个* 上述要求掌握的对象类型（将来遇到新的对象类型也应该如此），我们也要尝试验证其以下几个方面的 **属性** (attributes):
    - 对数学运算符 (+、-、*、/、//、%、@) 有没有支持
    - 如何判断相等 (==)



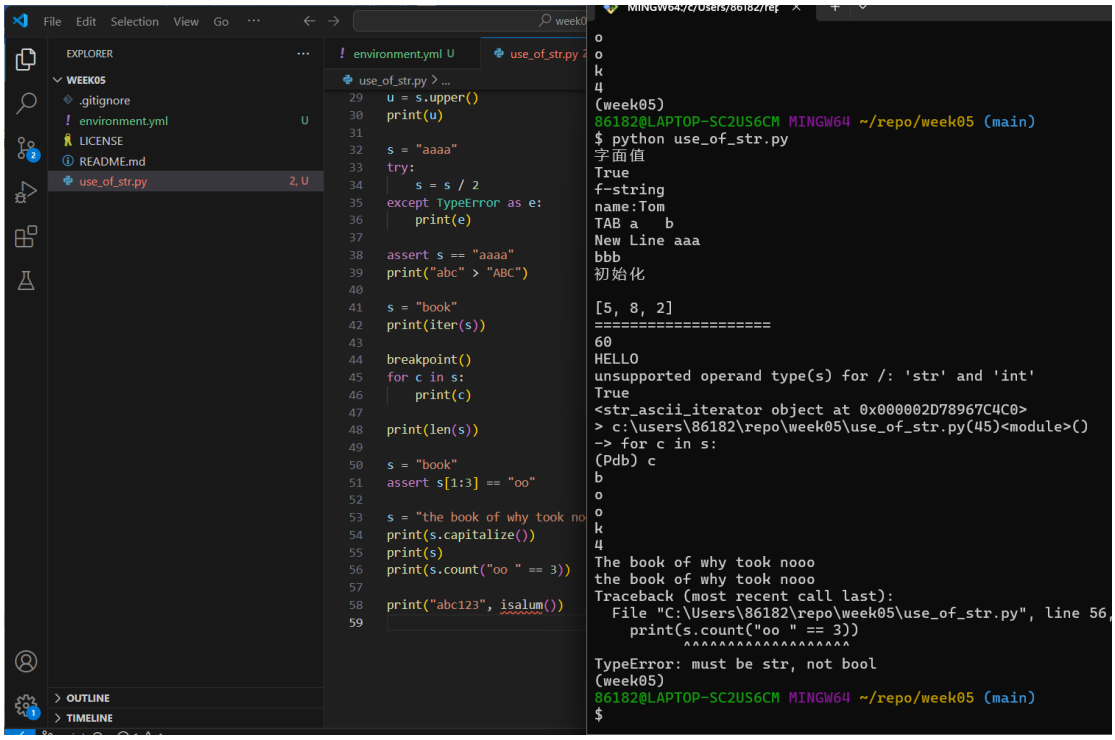
- 对于比较运算符 (>、<、>=、<=) 有没有支持
- 什么值被当作 True，什么值被当作 False
- 是否可迭代 (iterable)，如何做迭代 (for 循环)



- 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