在完成一项工作之前，可以先去GitHub上搜一搜，找到合适的修改，这样既可以学习到东西，又可以避免重复造轮子。

实际上，在日常的工作中，我们很多需求，无论是常见的、还是不常见的，Python 都为我们提供了一些独特的解决方案，既不需要自己造轮子，也不需要引入新的依赖（引入新的依赖势必会增加项目的复杂度）。

**装饰器**

直接对于 dict 的操作从实际上来讲可维护性是很差的。

这是为什么？

在改进这段代码之前，我们首先要明确 Python 中一个很重要的概念，即：函数/方法是：First Class Member 。用不精确的话来讲，就是函数/方法可以作为参数被传递、被使用。

1. import typing
2. def execute(func: typing.Callable, \*args, \*\*kwargs) -> typing.Any:
3. return func(\*args, \*\*kwargs)
4. def print\_func(data: int) -> None:
5. print(data)
6. execute(print, 2)

实际上装饰器是一个语法糖，具体可以参见 PEP318 Decorators for Functions and Methods。

简而言之，实际上是 Python 替我们做了一个替换过程。