Pycharm中的debug使用

已修改 2022-08-25 已创建 2022-08-25

首先，还是用示例说话，我们书写一段简短的代码。

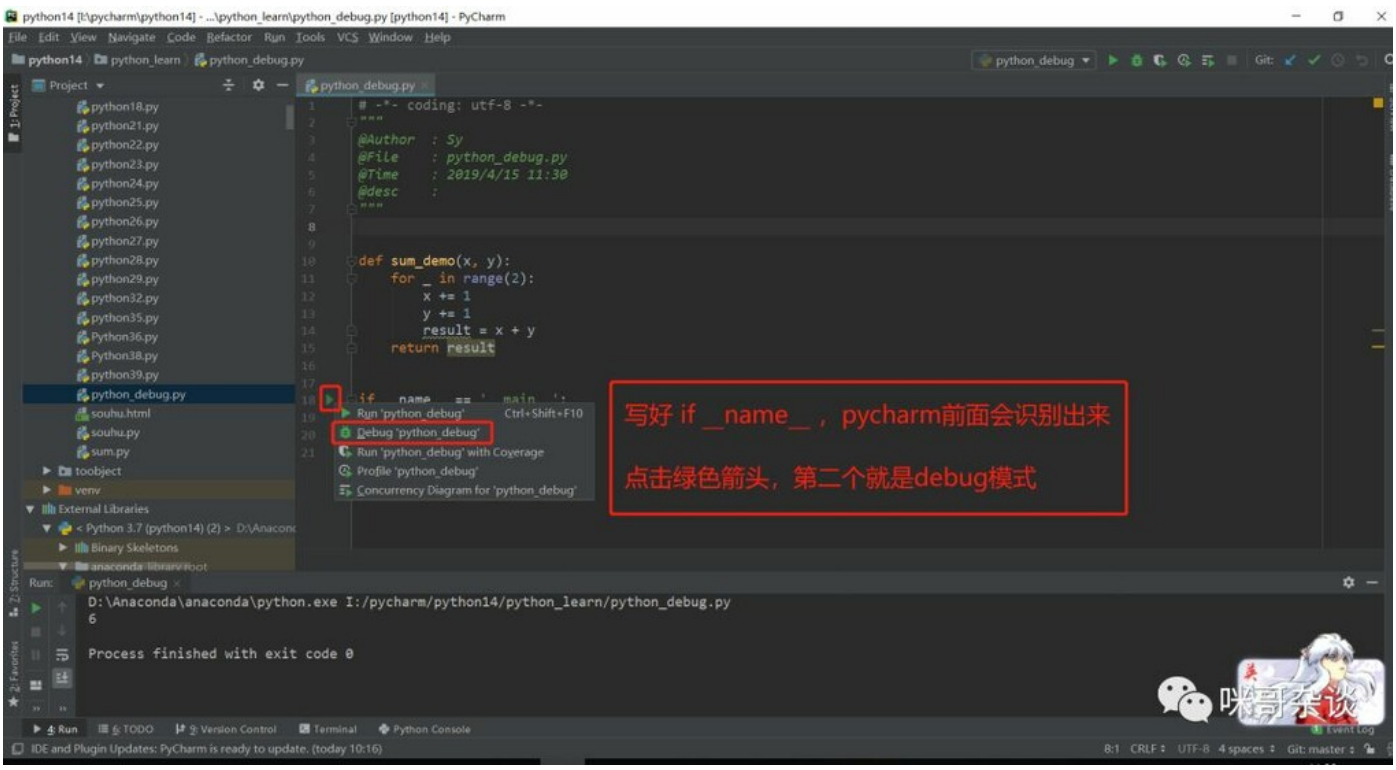
```
def sum_demo(x, y):
    for _ in range(2):
        x += 1
        y += 1
        result = x + y
    return result

if __name__ == '__main__':
    result = sum_demo(1, 1)
    print(result)
```

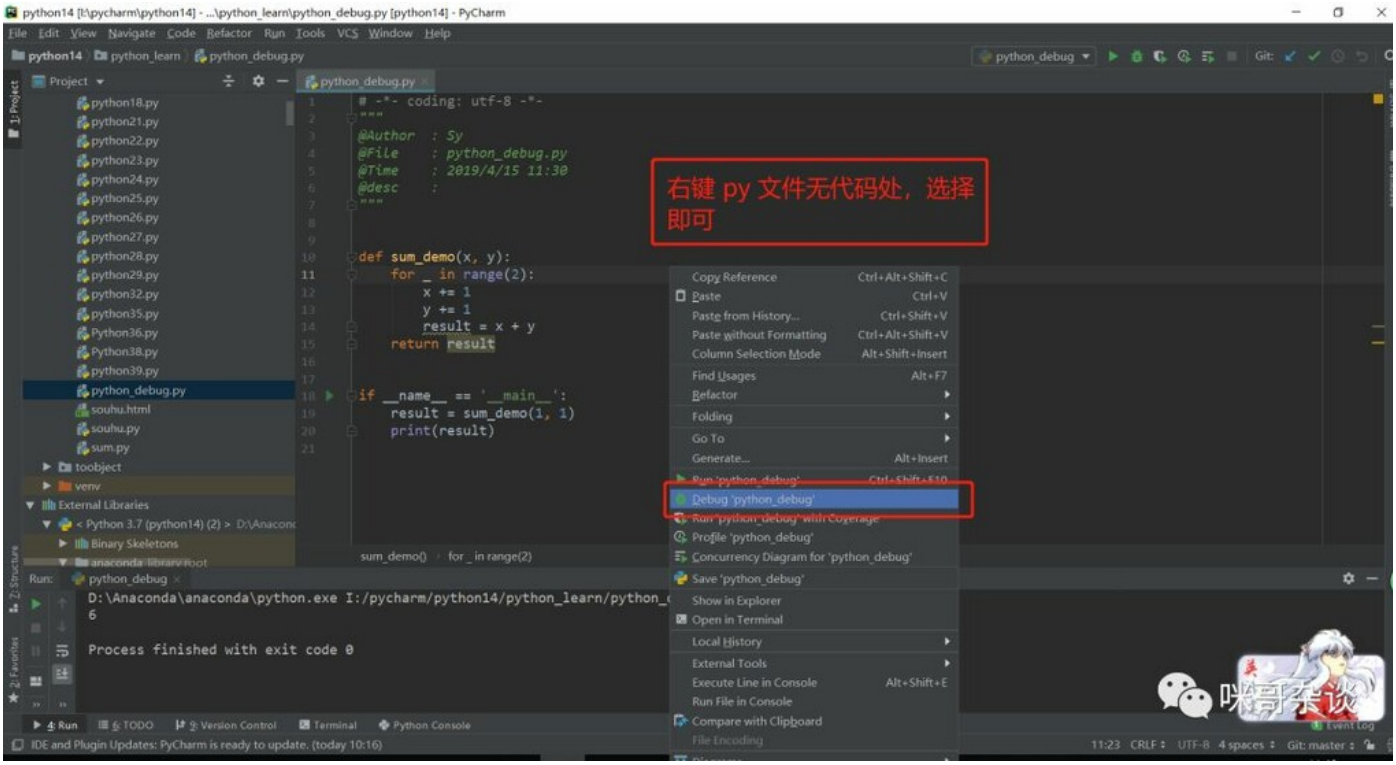
肉眼识别下，猜猜结果是多少呢？初学者可能没见过 for 循环中的下划线，在 Python 中是占位符的意思，因为单纯的循环两次而已，并不用到它的循环结果。最终 result 会输出 6 。

在 pycharm 中，如何开启 debug 调试，一共有三种进入的方法：

方法一：



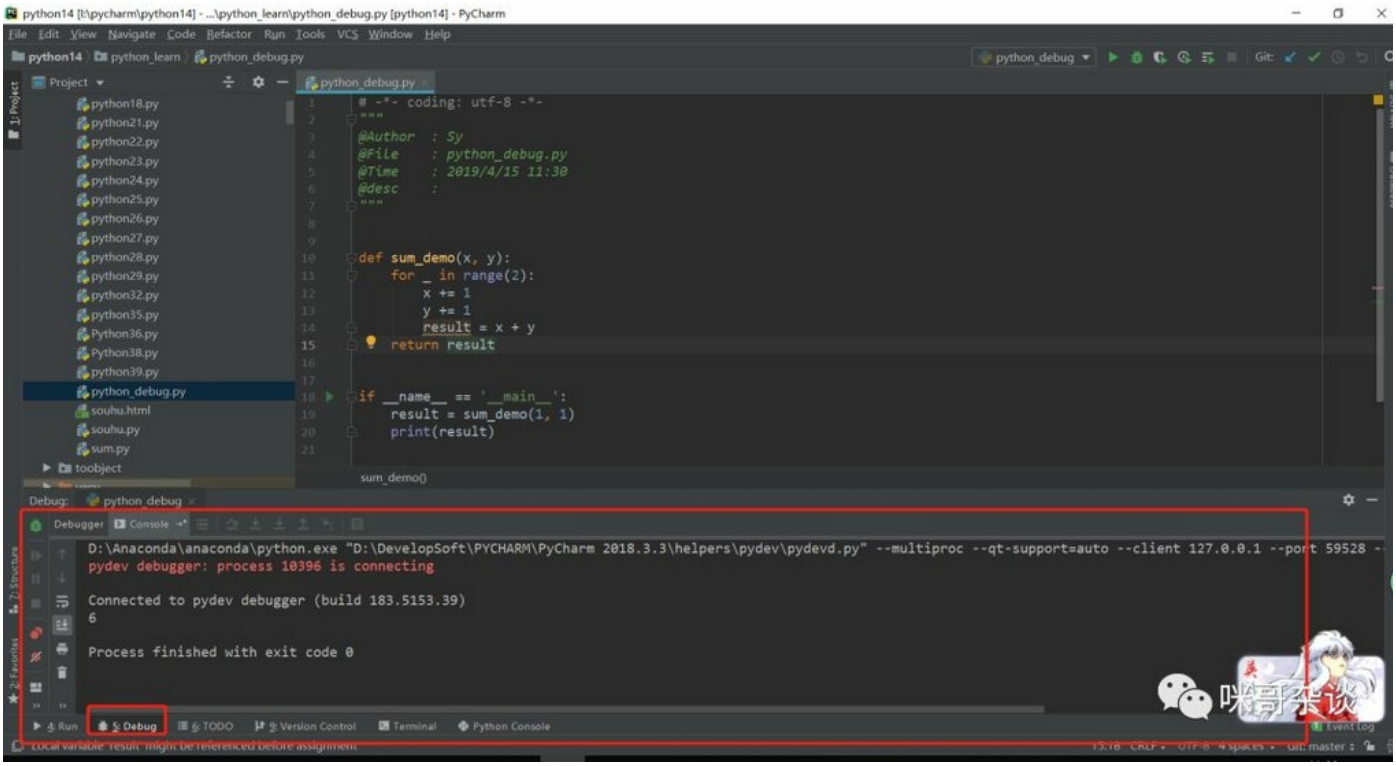
方法二：



方法三：

还有一种方法，就是 pycharm 导航栏处，有个run，点开以后即可看到 debug ，这里就不截图演示了。

单纯的进入 debug 模式，你会发现，与正常的 run 去运行程序没有差异。差异就是 pycharm 的控制台部分，从 run 跑到了 debug 显示。



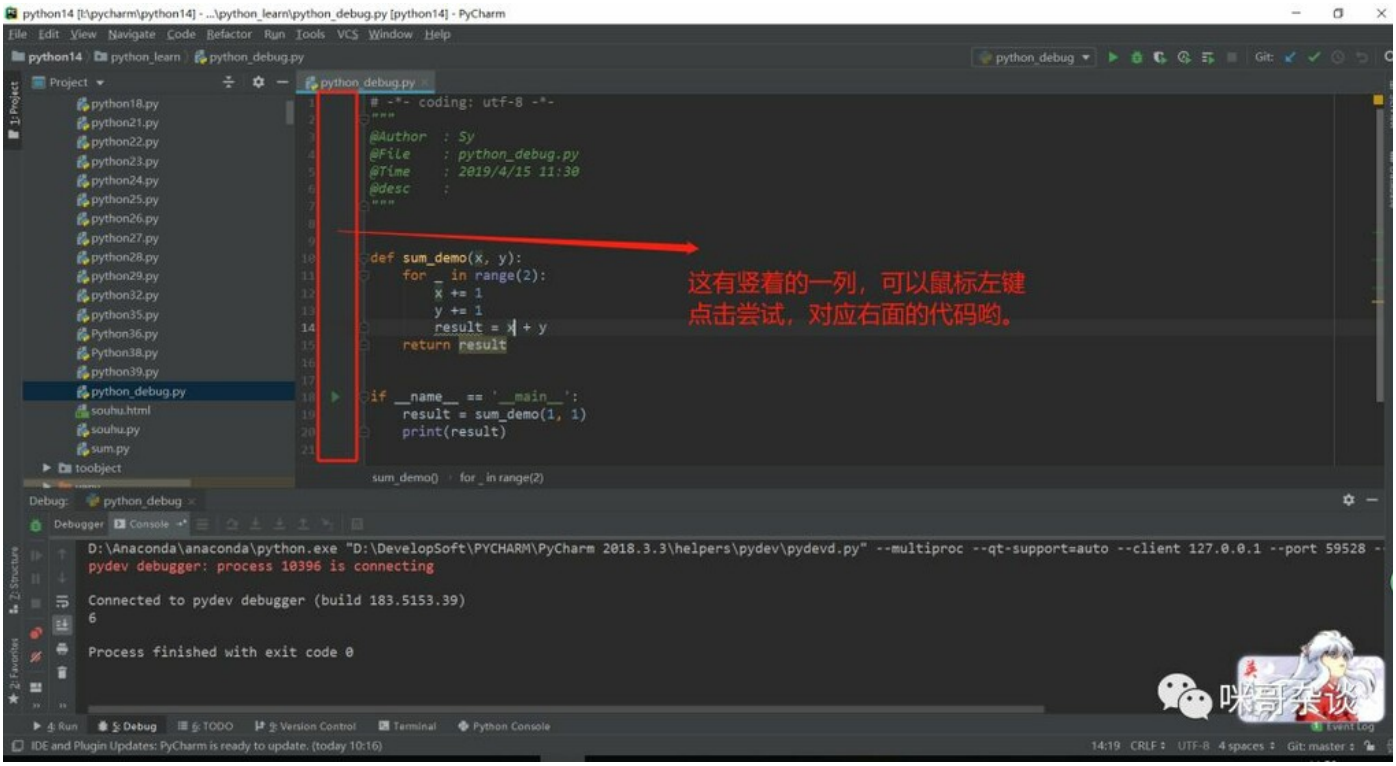
接下来要讲的，才是 debug 中的重中之重，即断点调试！

debug 的断点调试

断点调试，英文 breakpoint。用大白话来解释下，断点调试其实就是在程序自动运行的过程中，你在代码某一处打上了断点，当程序跑到你设置的断点位置处，则会中断下来，此时你可以看到之前运行过的所有程序变量。

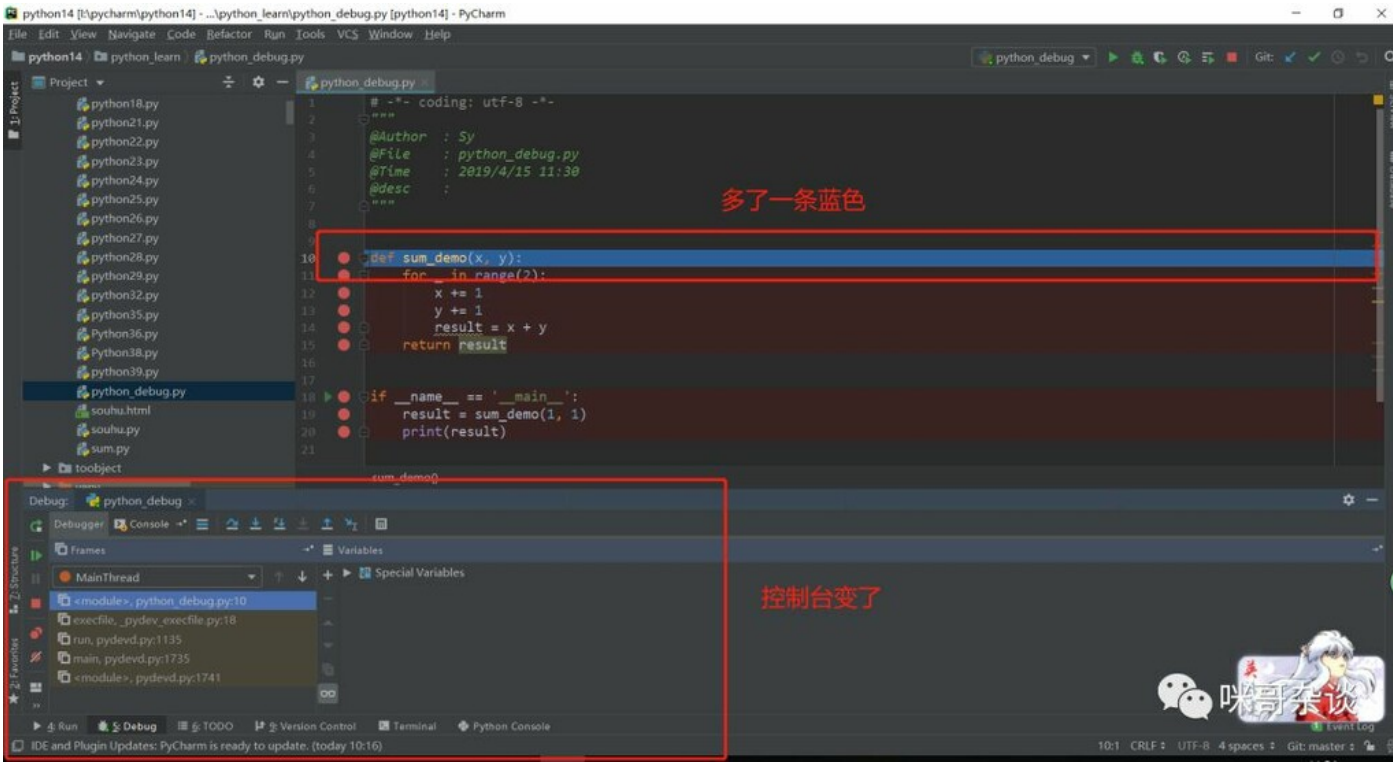
来继续刚才的演示，pycharm 中如何设置断点。

点击前：

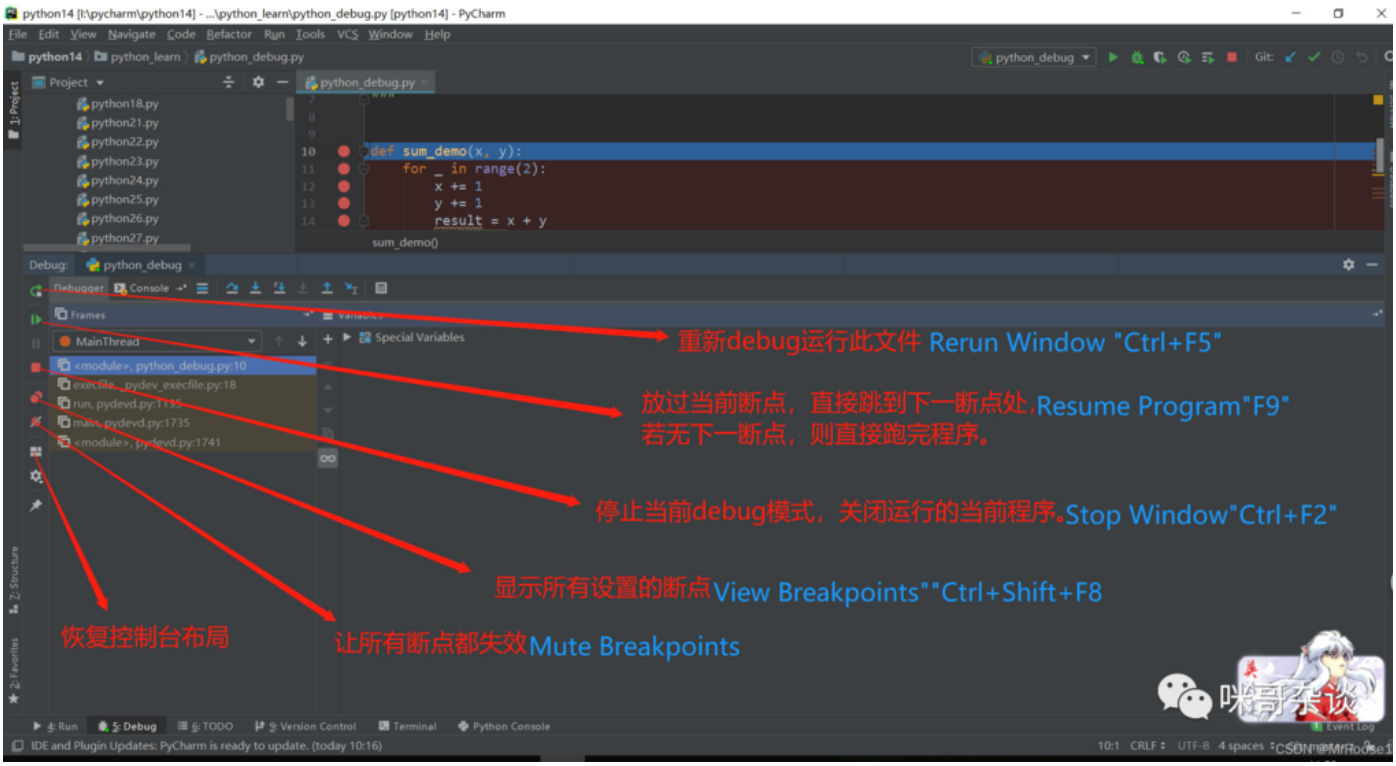


点击后，皮一下，每行代码都设置上断点：

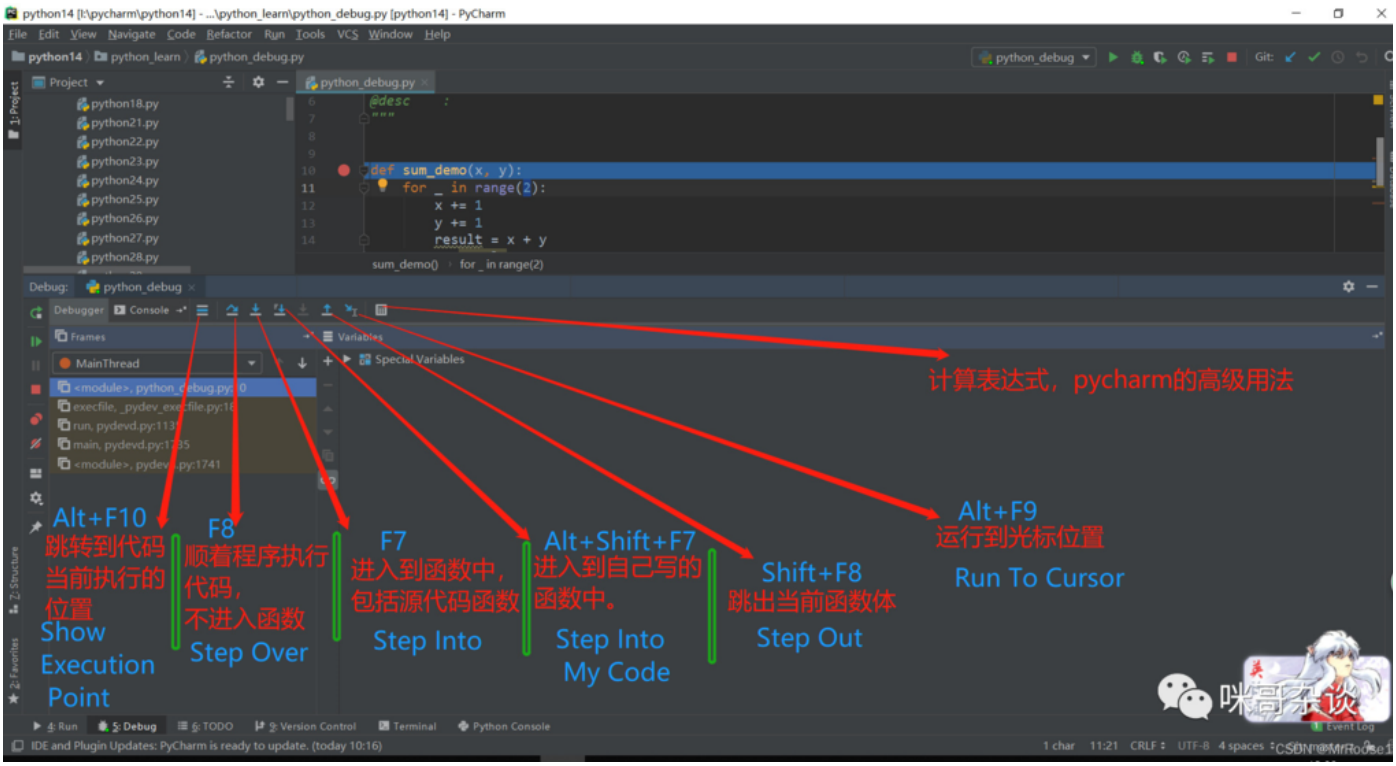
设置完断点后，开启 debug 调试模式运行下，看到结果：



看到了这么多新摆设，是不是有点怕了！不怕，咱们先来从控制台每个按钮讲起：



如果要是忘记中文意思的话，没有关系，鼠标指到按钮处，悬浮一会儿，会有英文提示的。继续再来说横排按钮：



其中，横排最重要，经常用到的按钮，来解释一下，自己鼠标悬浮去看英文即可：

step over（F8快捷键）：在单步执行时，在函数内遇到子函数时不会进入子函数内单步执行，而是将子函数整个执行完再停止，也就是把子函数整个作为一步。在不存在子函数的情况下是和step into效果一样的。简单的说就是，**程序代码越过子函数，但子函数会执行，且不进入。**

step into（F7快捷键）：在单步执行时，遇到子函数就进入并且继续单步执行，有的会跳到源代码里面去执行。

step into my code（Alt+Shift+F7快捷键）：在单步执行时，遇到子函数就进入并且继续单步执行，不会进入到源码中。

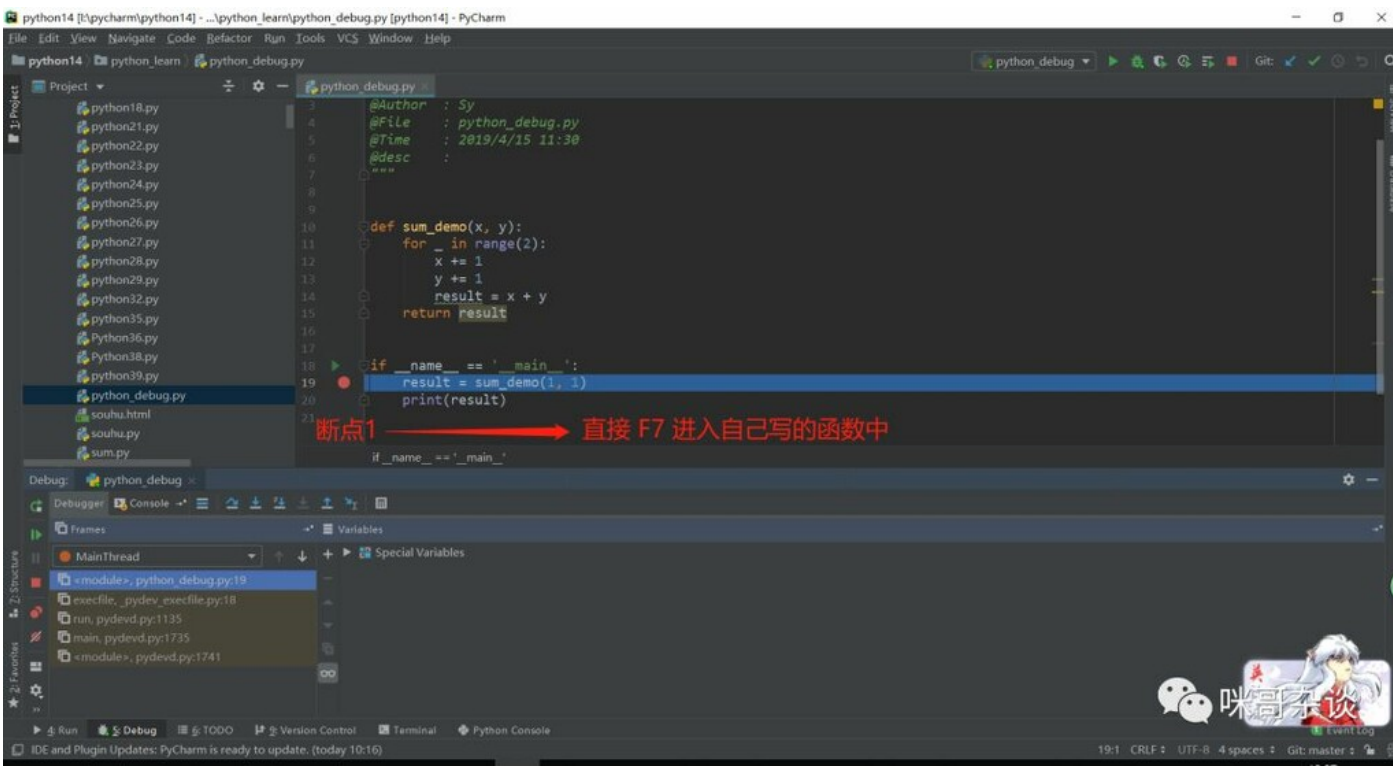
step out（Shift+F8快捷键）：假如进入了一个函数体中，你看了两行代码，不想看了，跳出当前函数体内，返回到调用此函数的地方，即使用此功能即可。

Resume program(F9快捷键)：继续恢复程序，直接运行到下一断点处。
以上四个功能，就是最常用的功能，一般操作步骤就是，**设置好断点，debug运行，然后 F8 单步调试，遇到想进入的函数 F7 进去，想出来在 shift + F8，跳过不想看的地方，直接设置下一个断点，然后 F9 过去。**

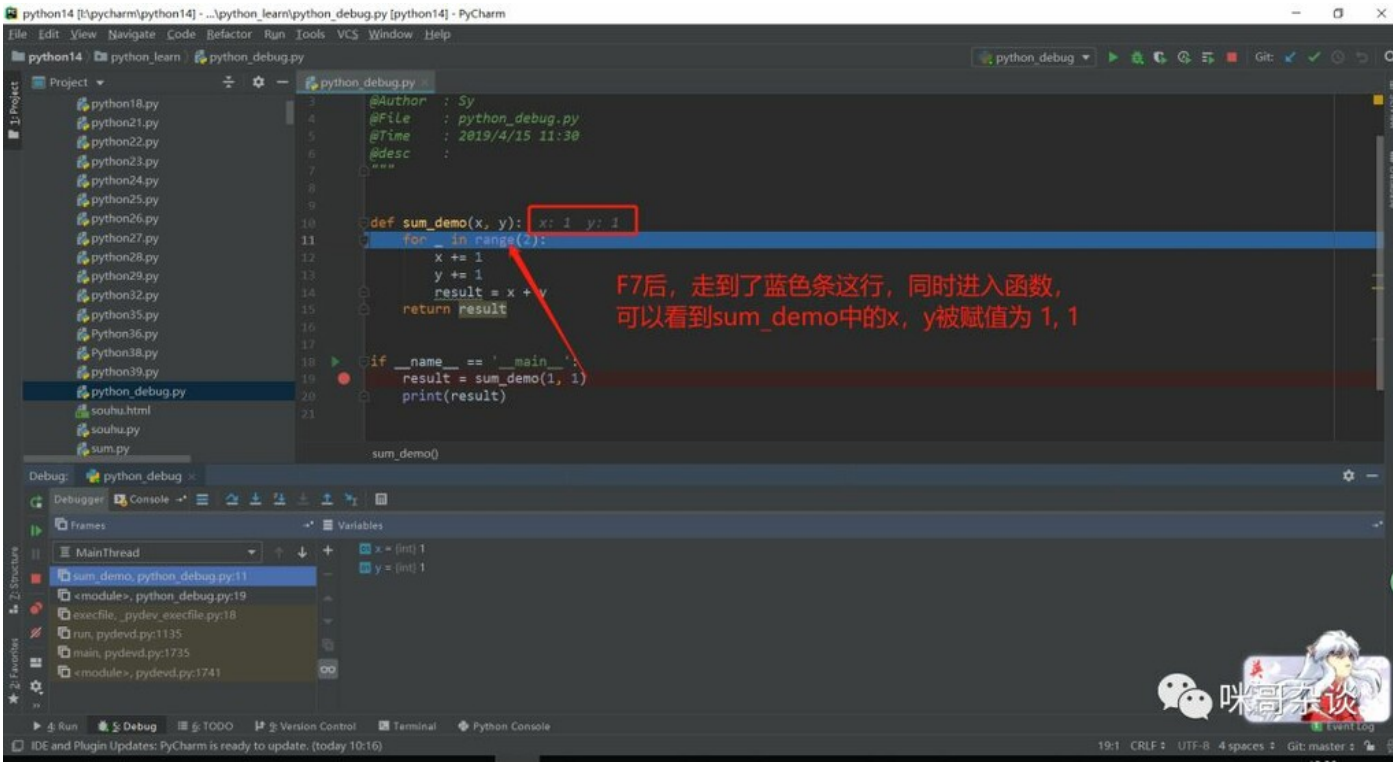
示例演示

上面的基础概念明白了以后，直接用图片示例演示下：

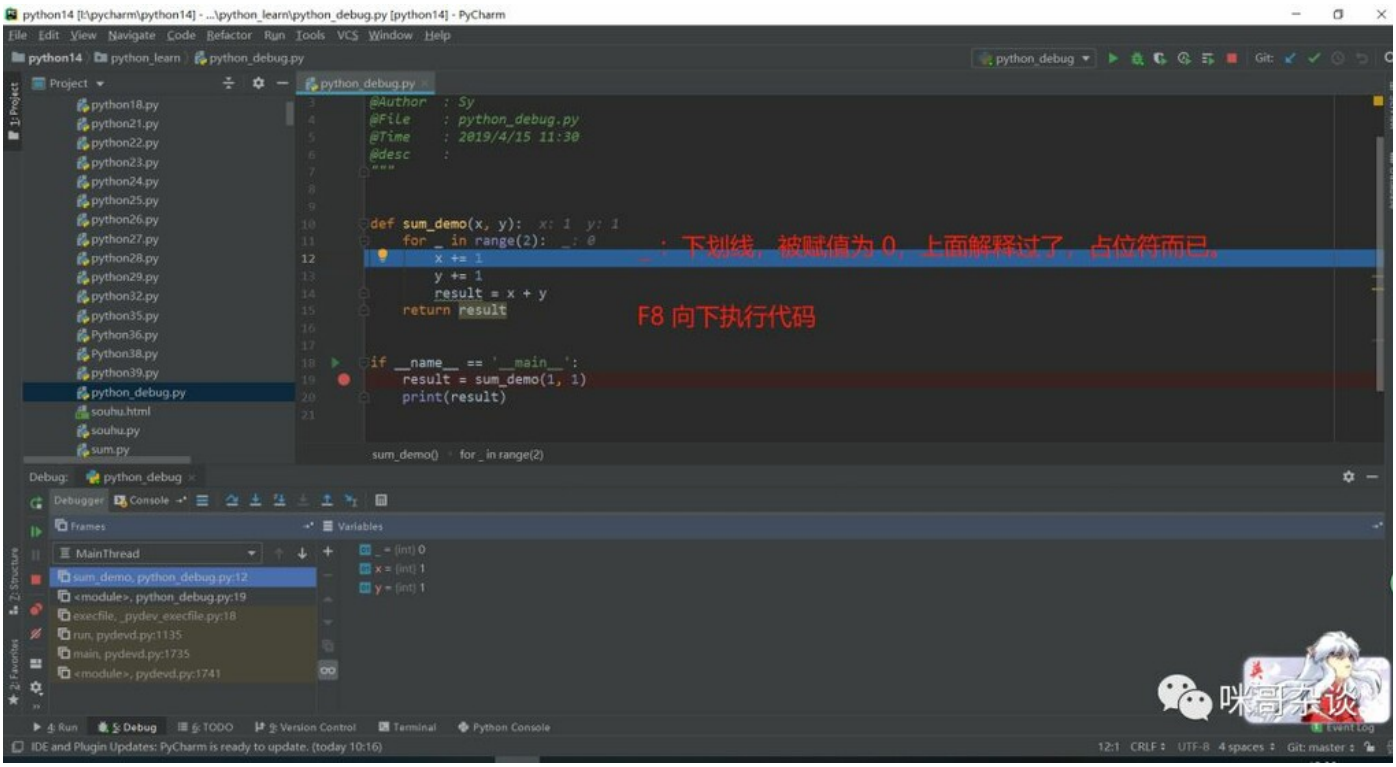
1. 设置初步断点



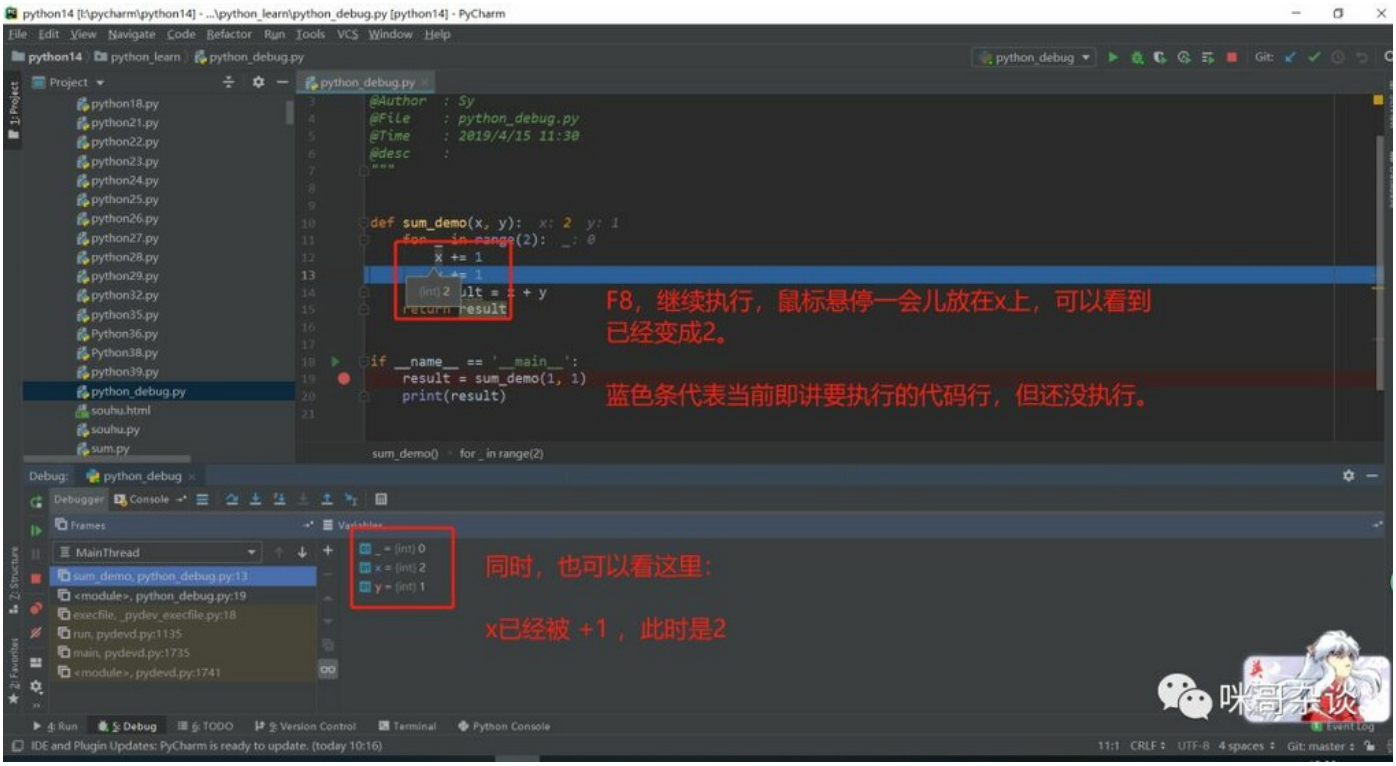
2. F7 进入函数



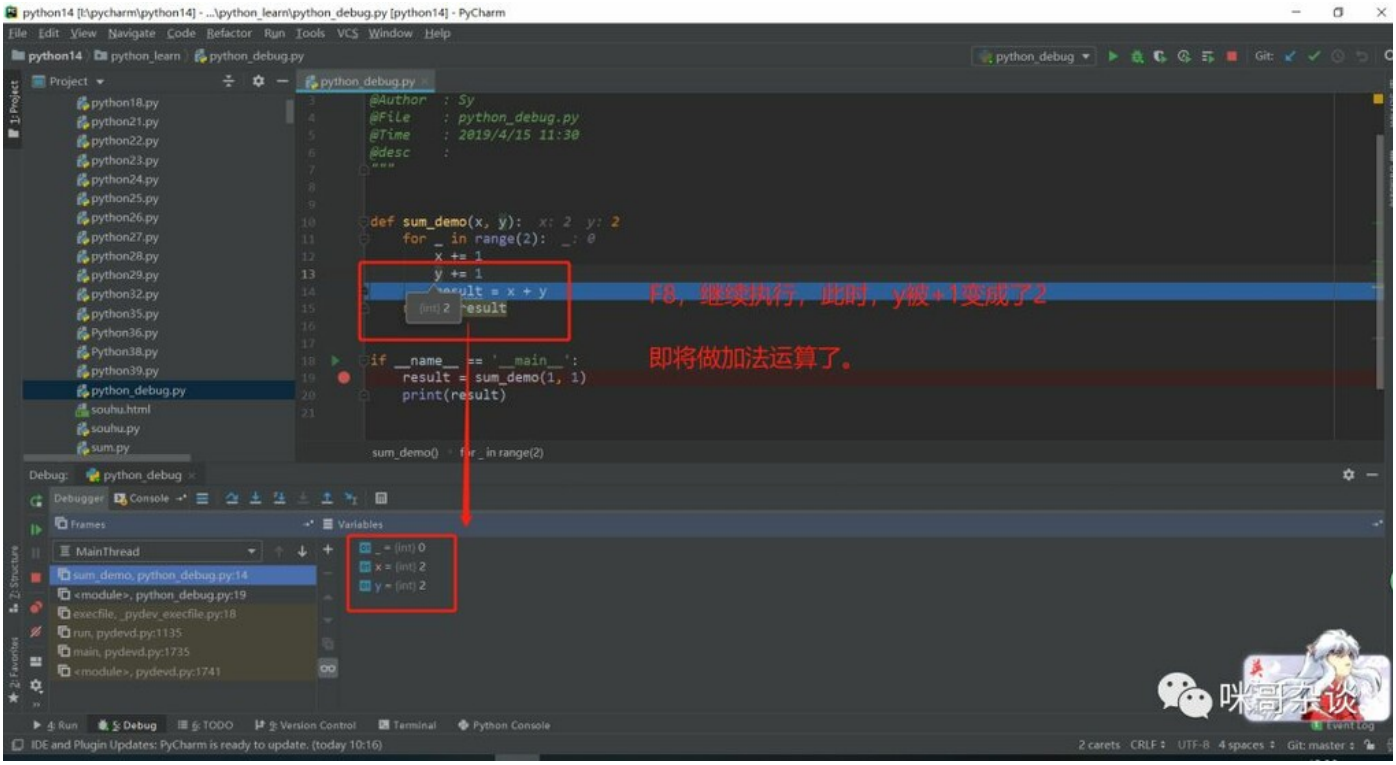
3. F8 单步调试，往下执行代码



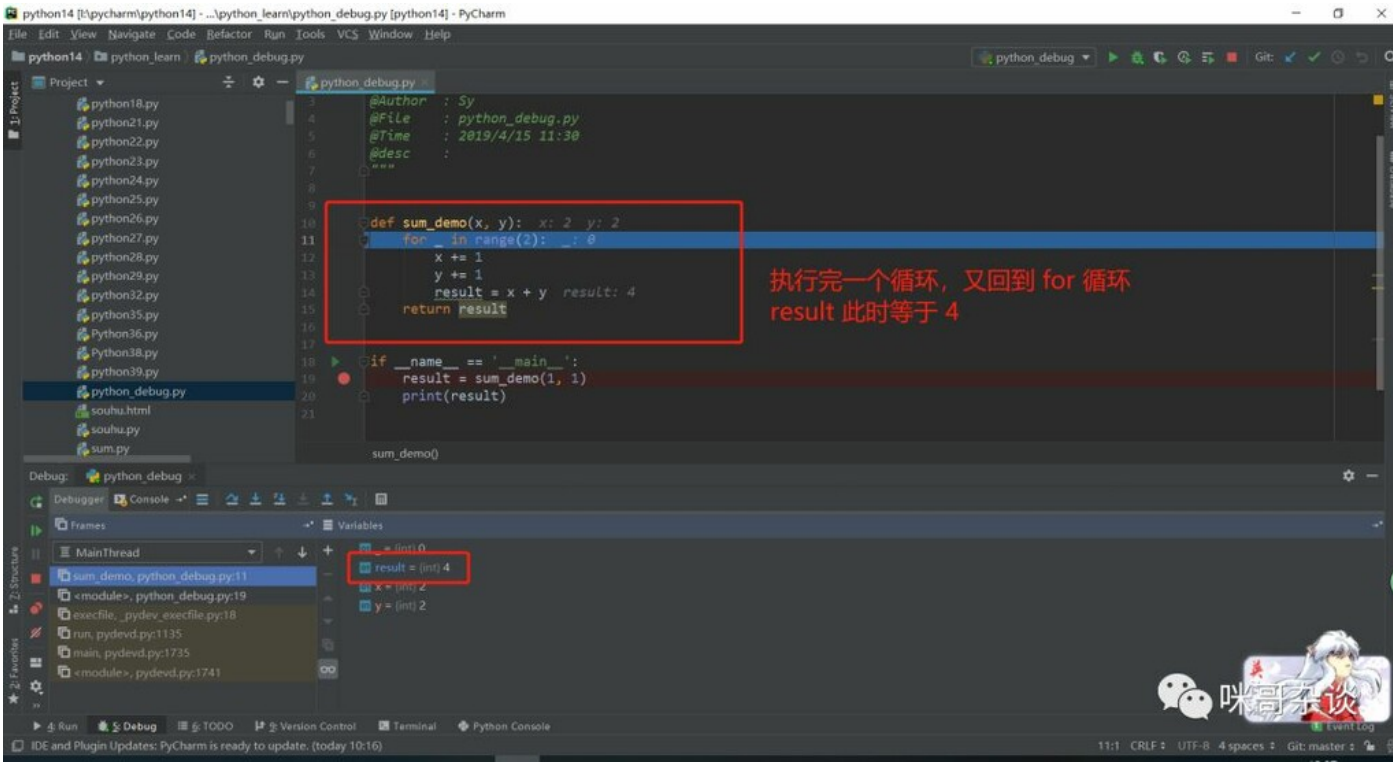
继续 F8 单步调试，往下执行代码：

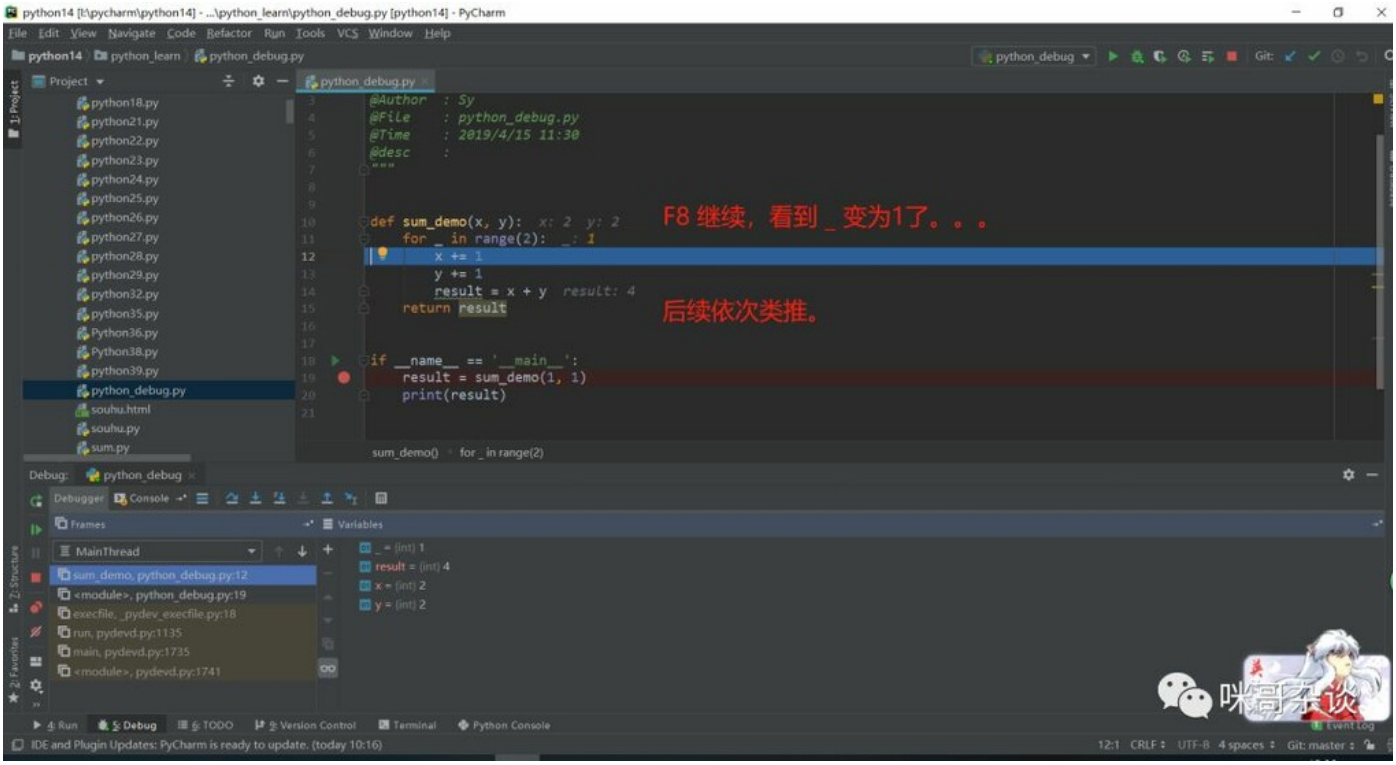


继续 F8 单步调试：

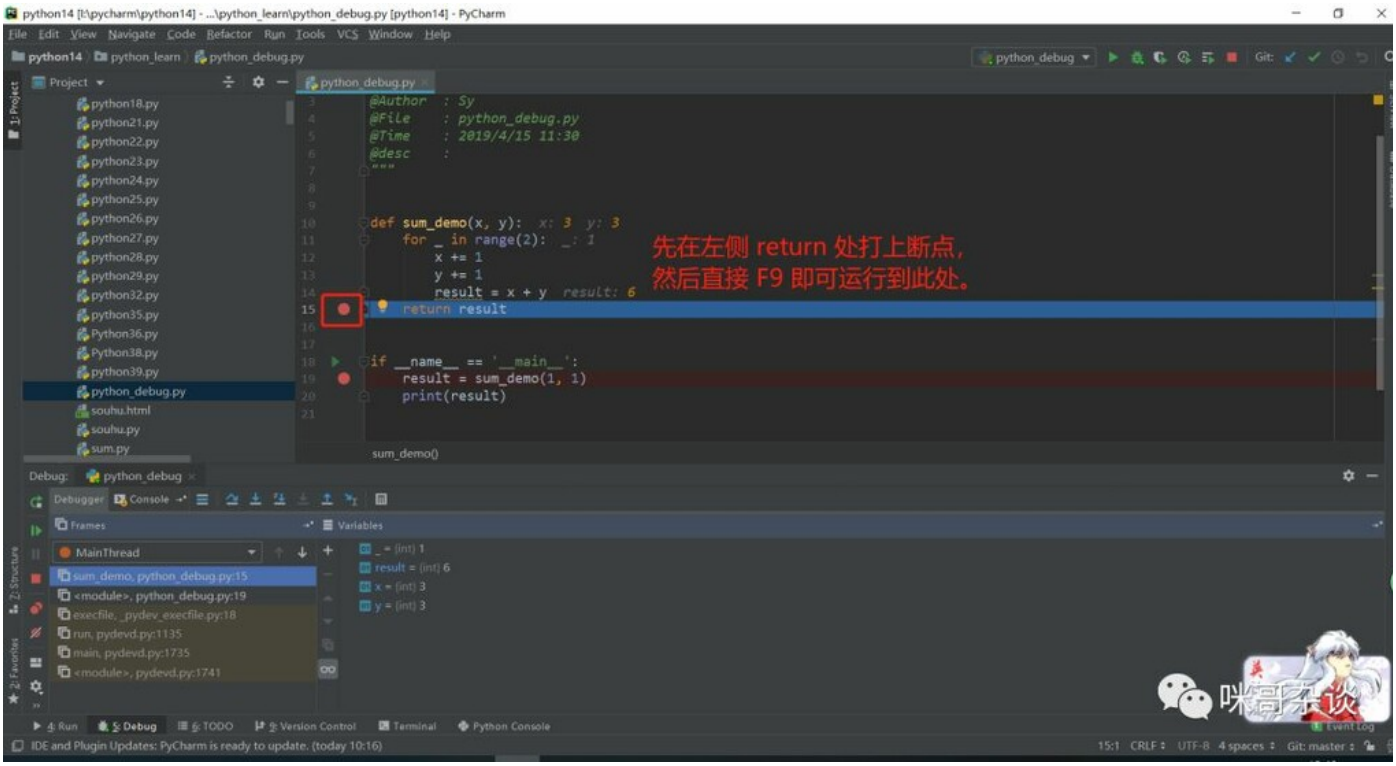


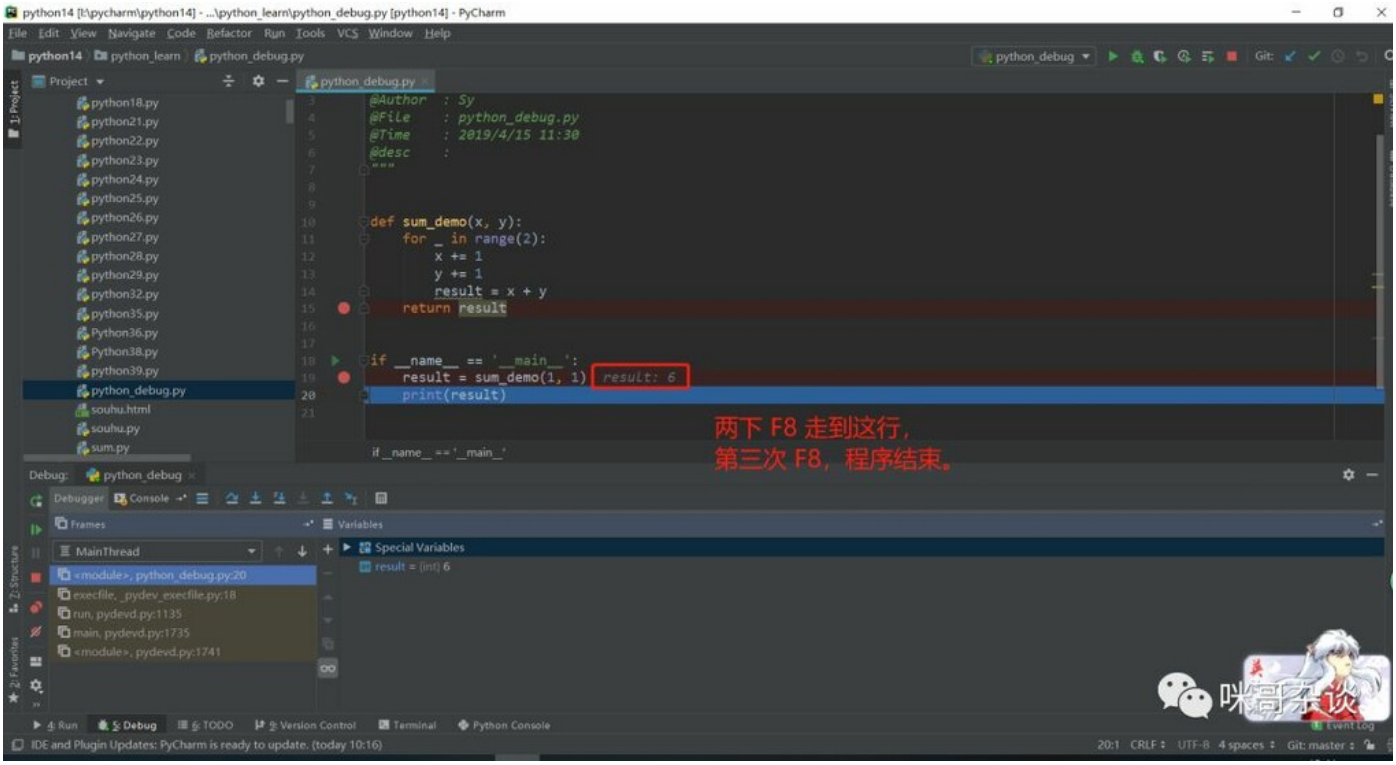
继续 F8 单步调试：





4. 看够了循环，想直接看最终 result 加完的结果，结果处打断点，直接 F9


先在左侧 return 处打上断点，
然后直接 F9 即可运行到此处。


两下 F8 走到这行，
第三次 F8，程序结束。