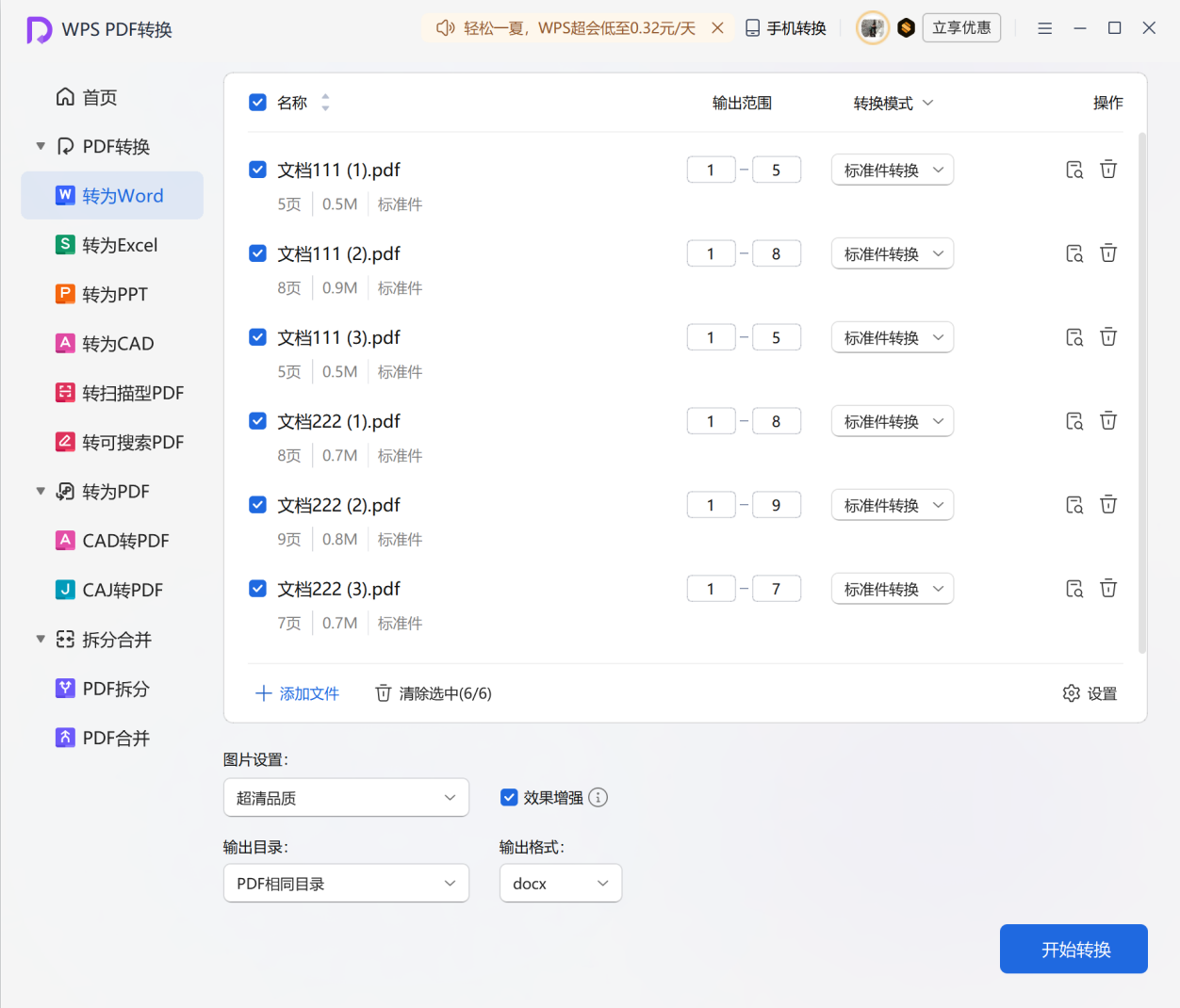
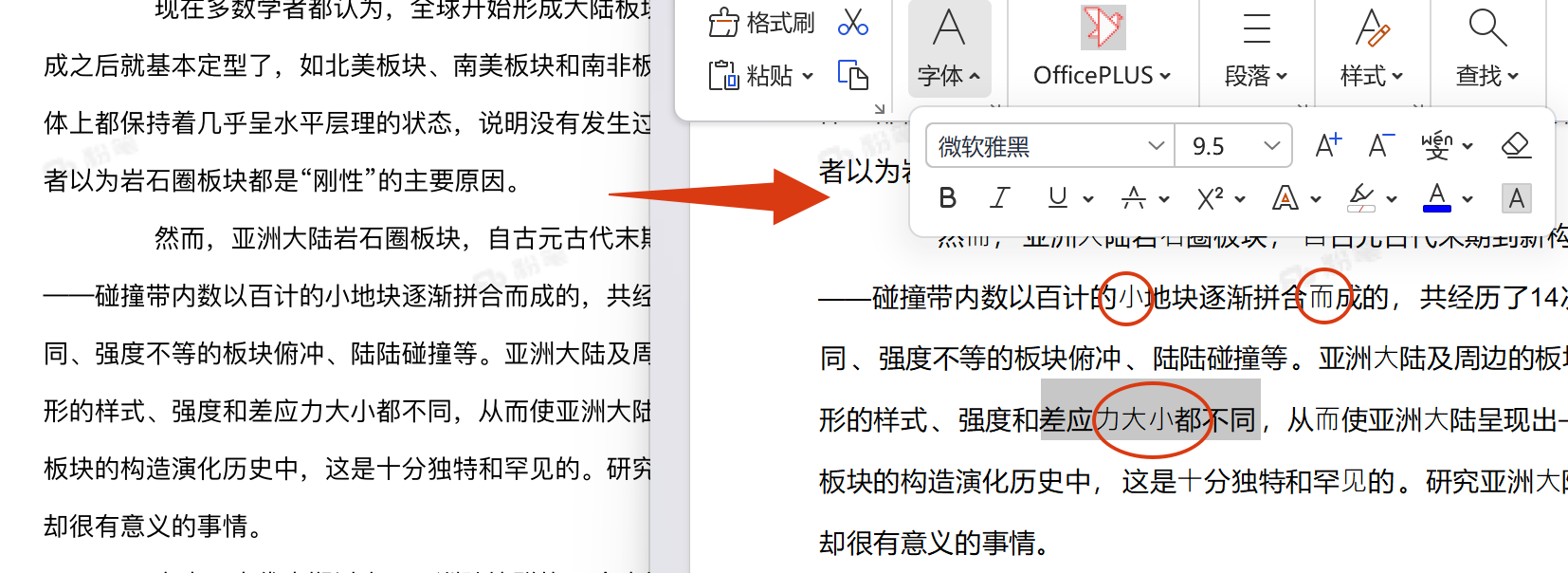
最近遇到个有点意思的问题，加上快一年没写代码，就想着磨下刀，于是写了这个工具并开源，《解决关于PDF转换Word后部分字的显示字体异常且无法切换字体的问题》，有需要的朋友自取哈~

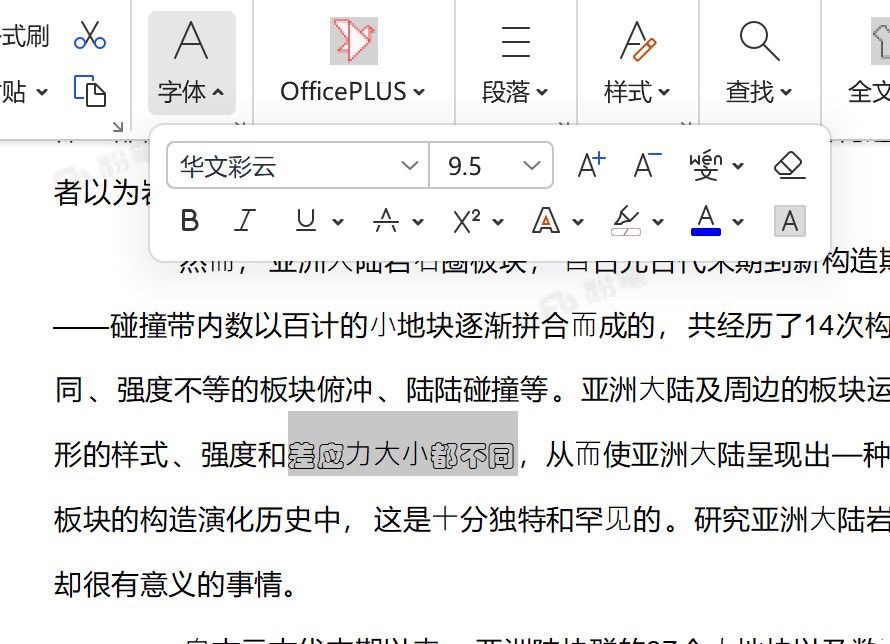
比如：使用WPS的PDF转换工具，使用**标准件转换**转换Word后：



部分字的字体虽然是微软雅黑，但并没有显示微软雅黑的样式：



然后这些字切换别的字体后，也无法显示对应的字体的样式：



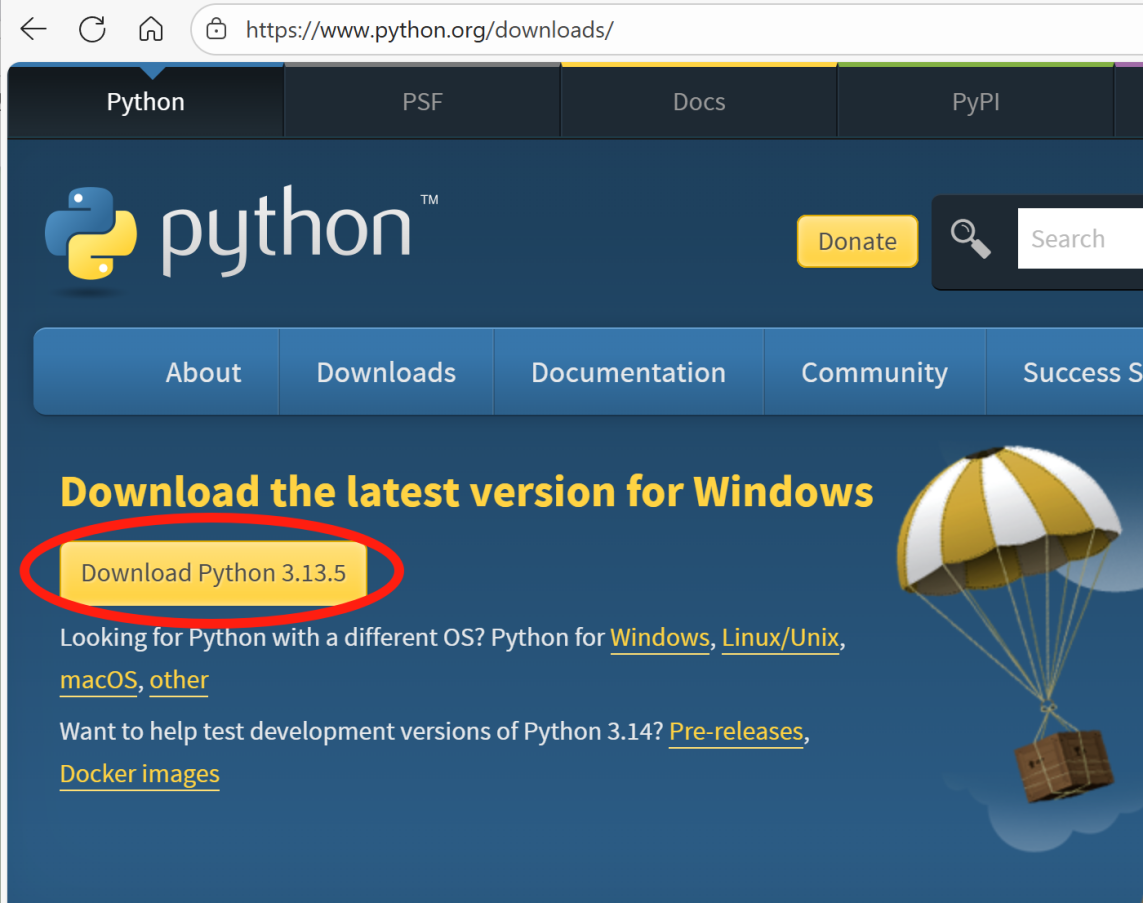
出现以上类似的问题，都可以用这个工具一键批量转换，把转换后有缺陷的Word文档修复正确。

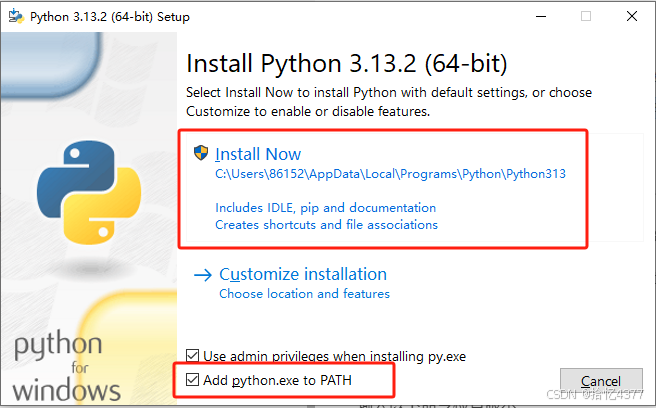
然后言归正传~

1. 首先，目前这个问题，我使用多个搜索引擎和问多个AI，花了半个多小时没能找到简单、安全、可靠、批量的解决方案(在开源这个工具前刚确认)
2. 其次，请面对太多的文字不要有畏难情绪~可以尝试看一下，并不会花很长时间。
3. 再者，请面对看似需要许多步骤的操作不要有畏难情绪~为了让使用者不用担心这个工具带有病毒，我选择了开源，也就是提供了源代码，所以确实需要多两步操作步骤。
4. 最后，请面对看似很复杂的代码，但其实就是“与机器的一段很简单的对话”不要有畏难情绪~跟着提供的步骤来，不会代码也能搞定。

无脑说明步骤写在前面，原理和Q&A放后面

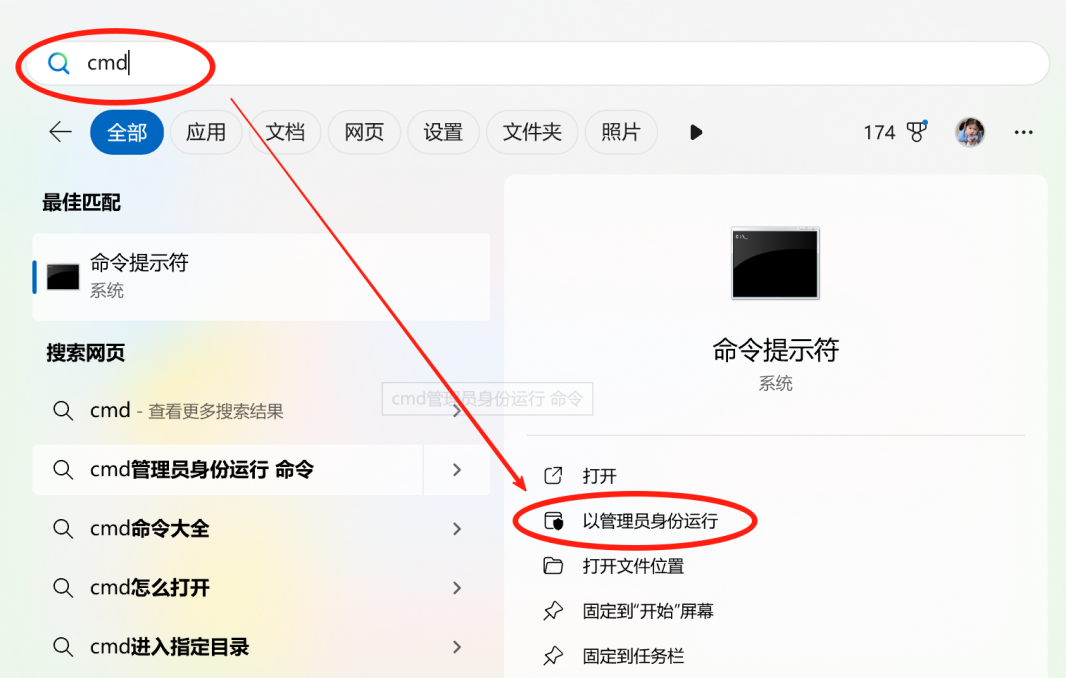
1. 无脑操作步骤：
2. 需要安装Python3，有写语法需要3.6后，所以能装多新就装多新，我装的是3.13，在官网下载安装即可：<https://www.python.org/downloads/> 。如图：

基本 运行下载的.exe，然后各种勾选和安装路径留它默认就行，**唯一要注意的就是一定要勾选​Add Python to PATH​（自动配置环境变量）**，如下图：

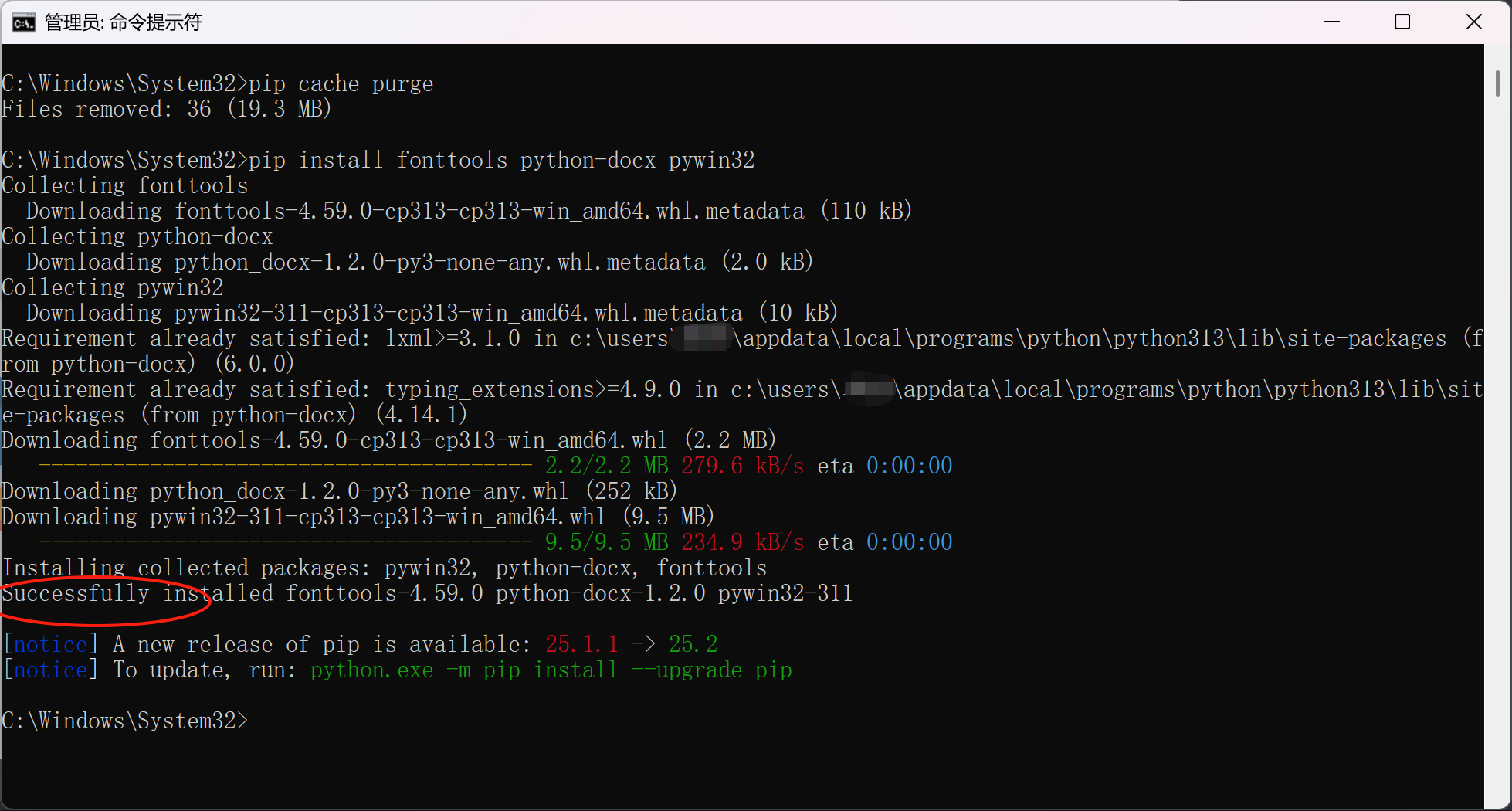


其他一路点就行，如果是Mac或者有兴趣想进一步了解，可以看这篇文章：<https://blog.csdn.net/qq_40998698/article/details/146391220>

1. 安装一些需要的库
   1. 按win键，输入cmd，以管理员身份运行

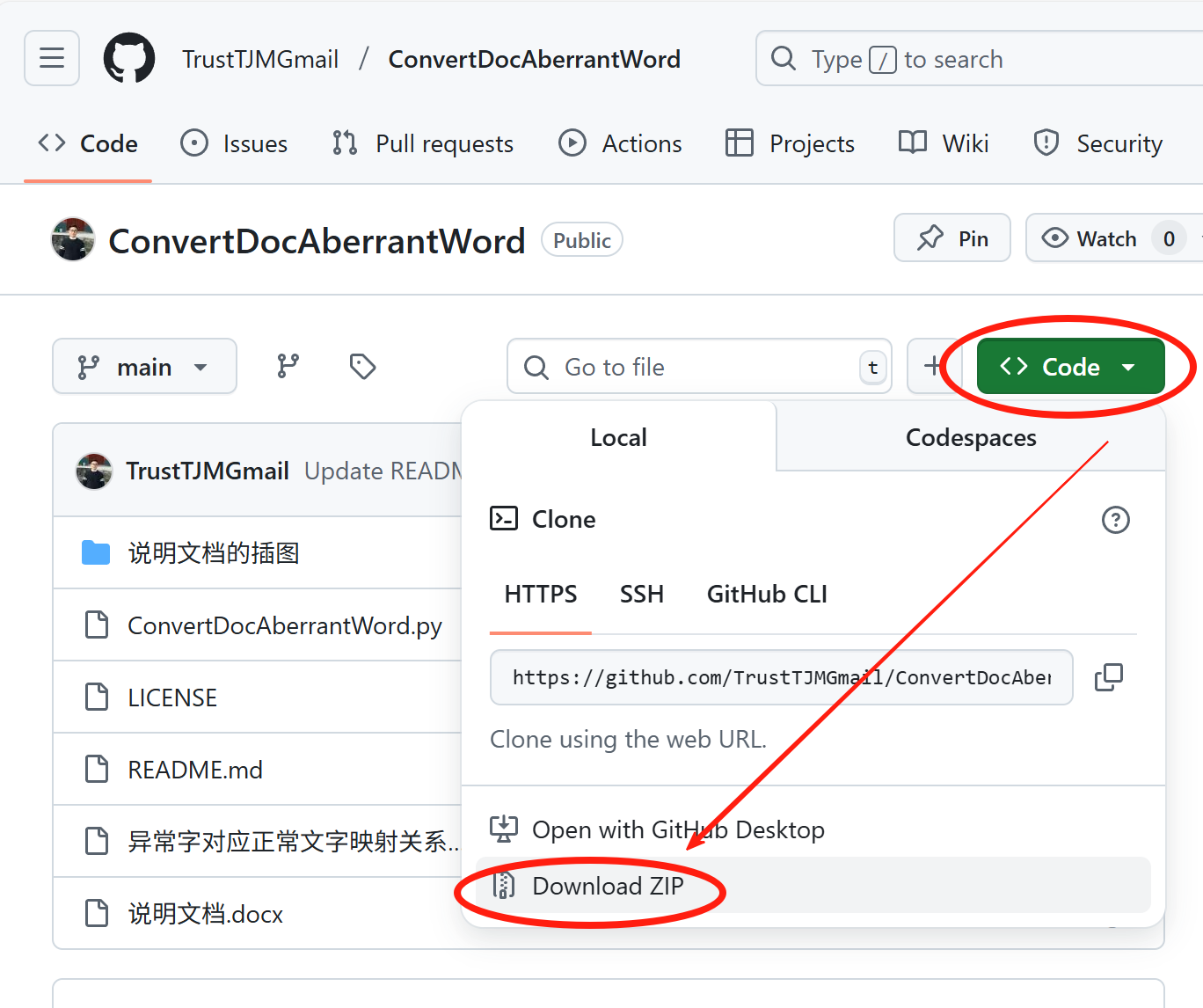


* 1. 安装字体工具，输入pip install fonttools python-docx pywin32，然后回车安装。温馨提示:复制可以用ctrl+c，但粘贴的快捷键不是ctrl+v，在命令行中的粘贴方式是，鼠标右键单击，或者键盘快捷键是Alt + 空格，显示菜单后点一下，如图：，然后按回车键确定，如下图显示Successfully字样说明安装成功

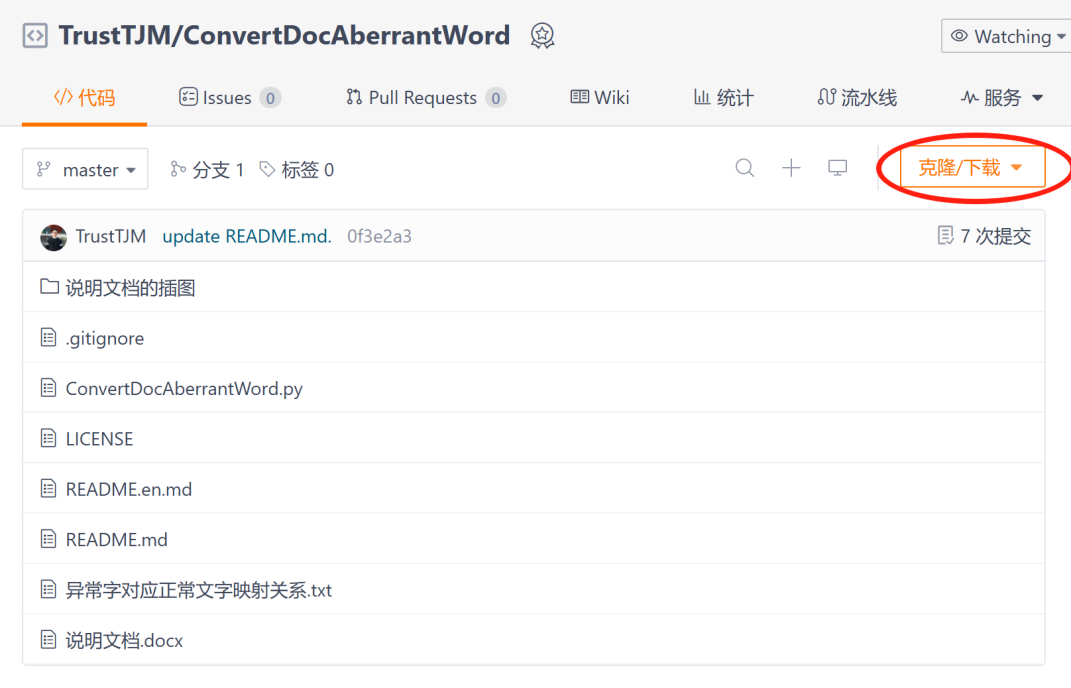
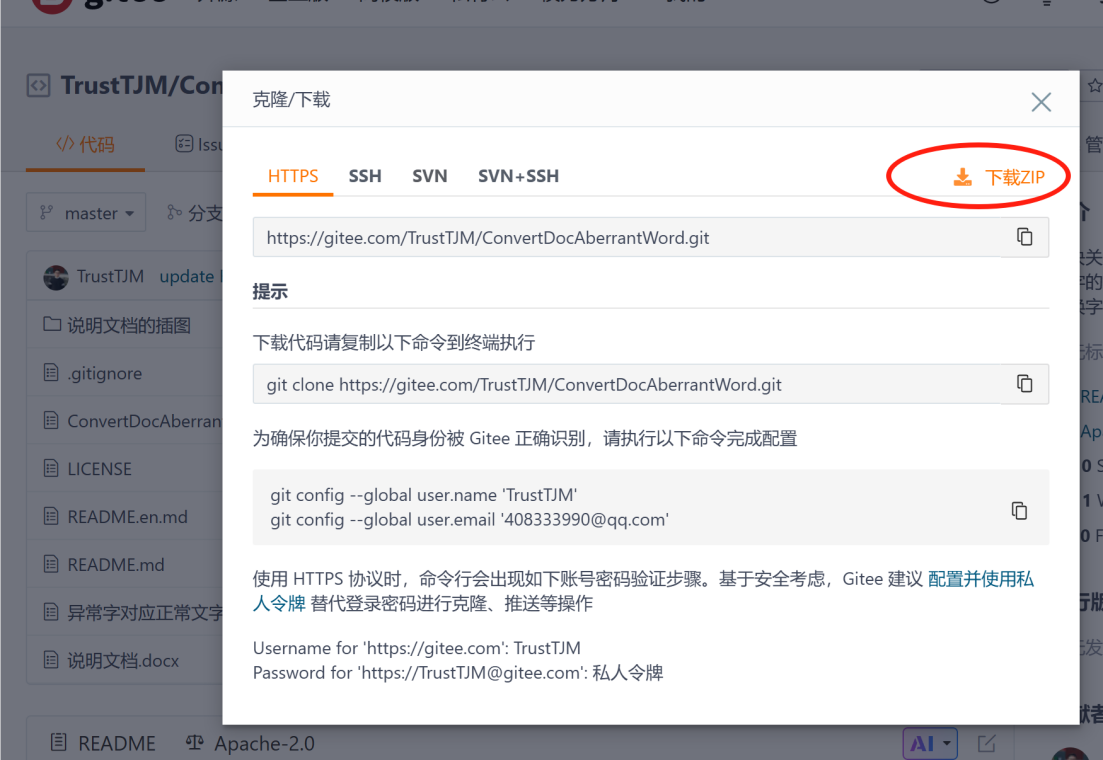


1. 提供两个下载源代码的地址，

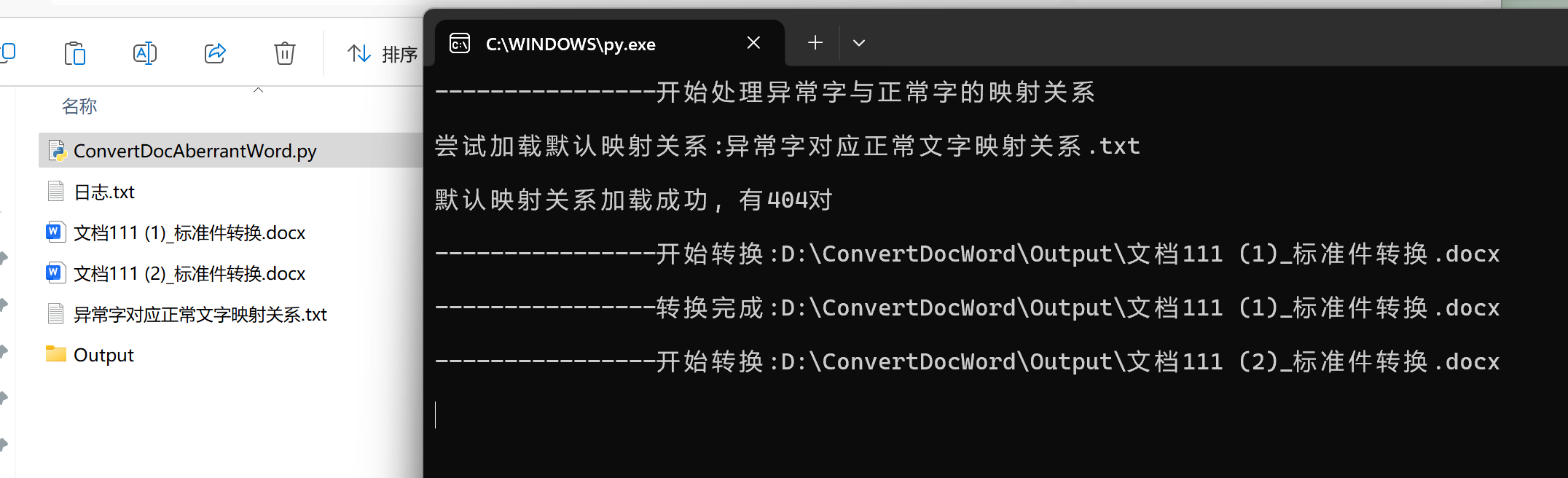
GitHub可以在<https://github.com/TrustTJMGmail/ConvertDocAberrantWord>下载， 如图：



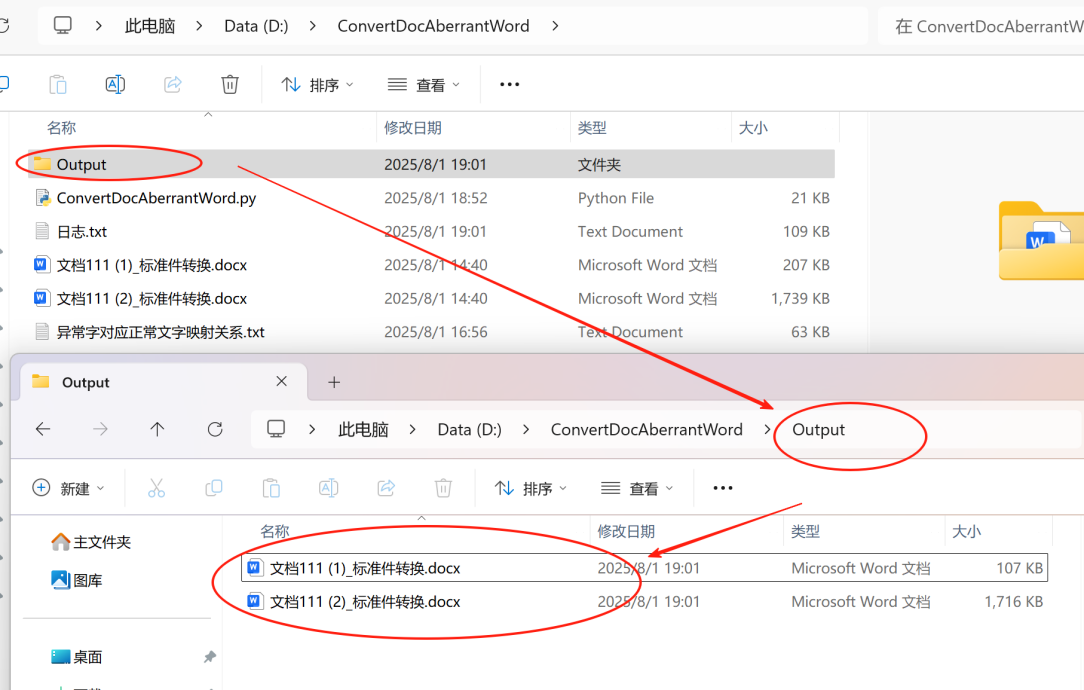
码云可以在：<https://gitee.com/TrustTJM/ConvertDocAberrantWord>，如图：

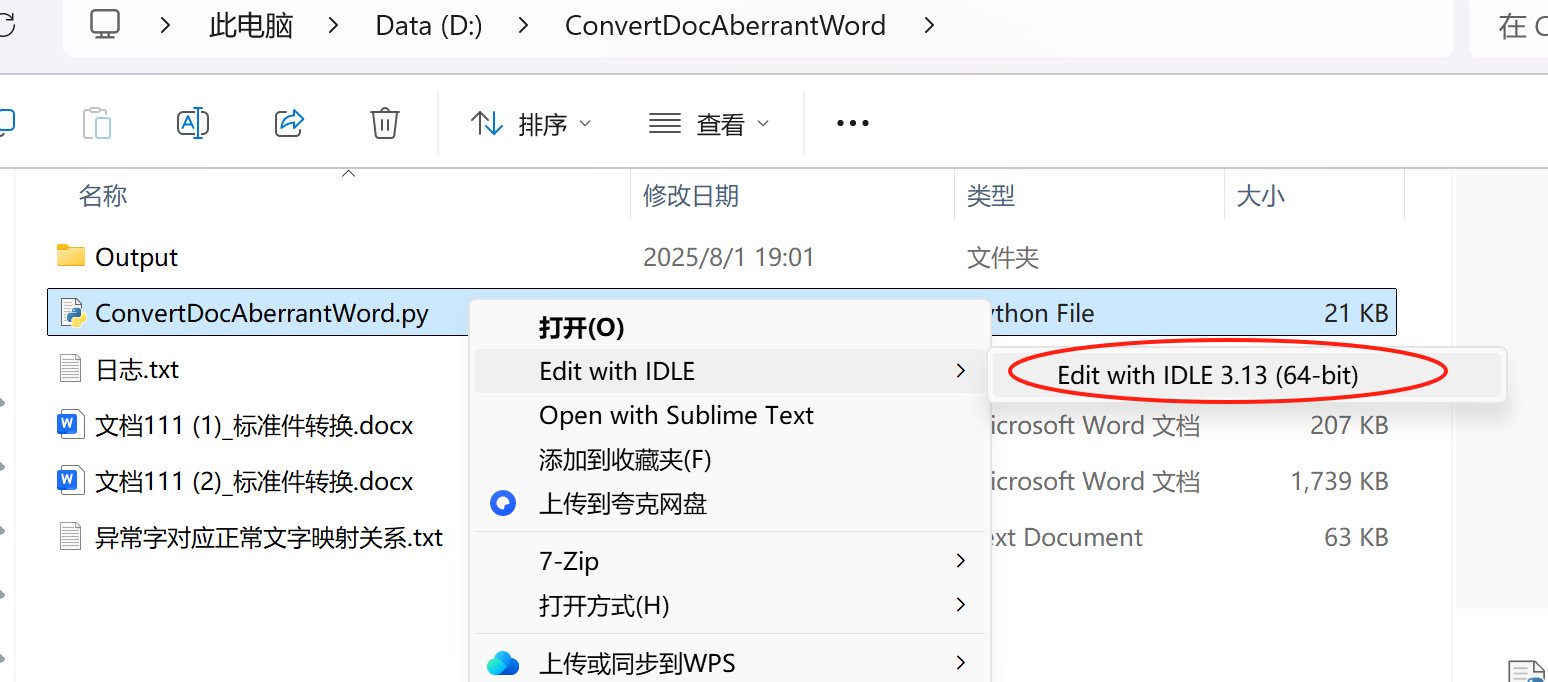
1. 直接下载ZIP，然后解压即可。如果Word文档不多，直接复制到代码的同一个目录下，然后双击运行ConvertDocAberrantWord.py，如果弹出一个对话框，如图：

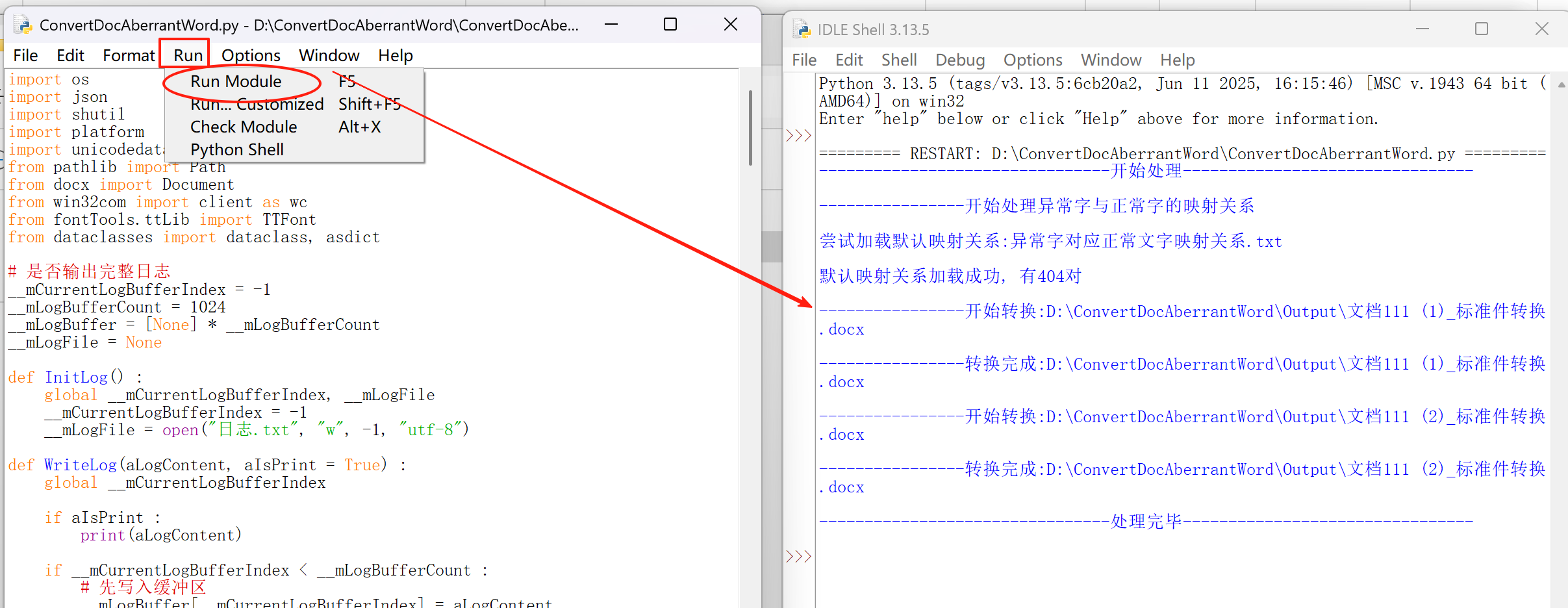


等它自动消失即可，然后看看Output文件夹中是不是生成新的Word文档，如图：



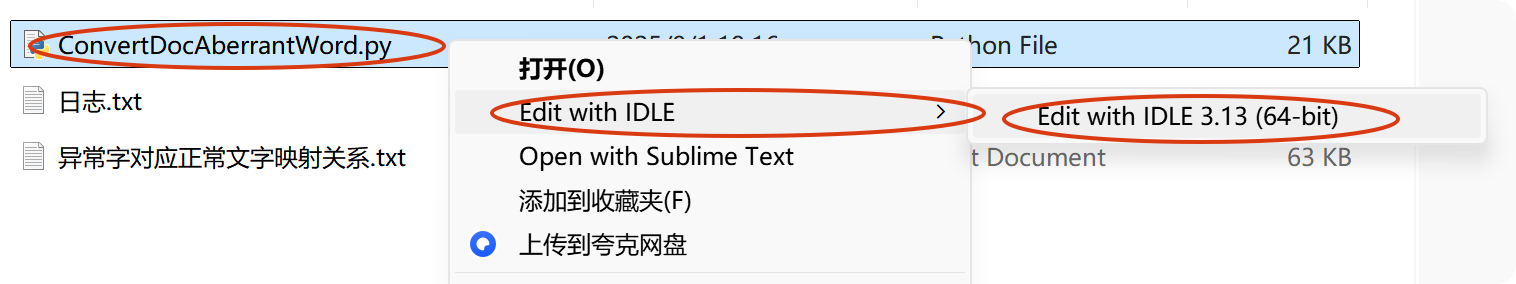
情况1：假如没有弹出命令行对话框，而是弹出文本编辑器，那么说明安装环境有冲突，这时候需要手动运行，右键代码文件->选择Edit with IDLE->Edit with IDLE 3.13(64-bit)，如图：



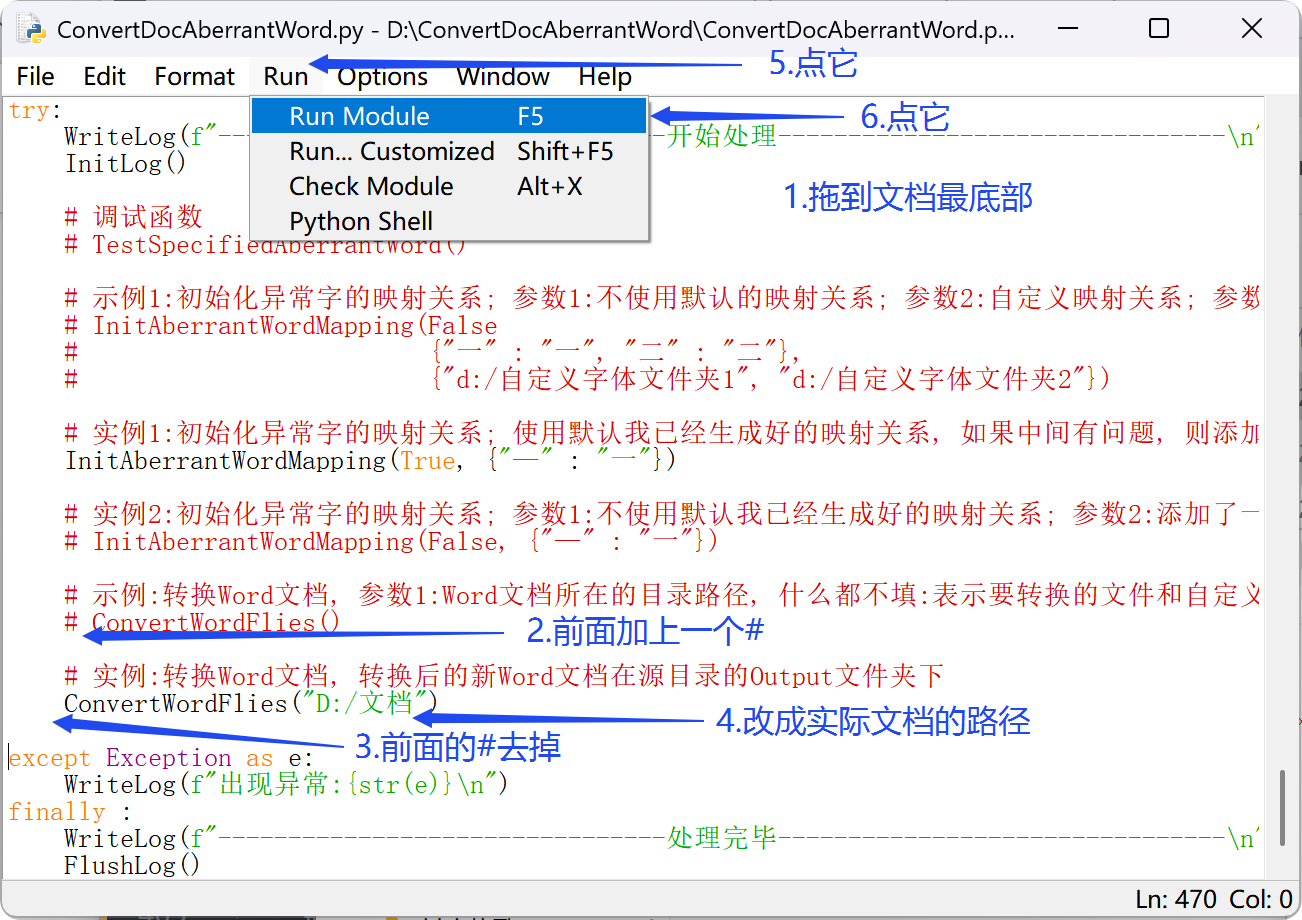


如果这一步也不行，比如没有Edit with IDLE选项，说明安装Python软件有问题，建议卸载Python然后重装。

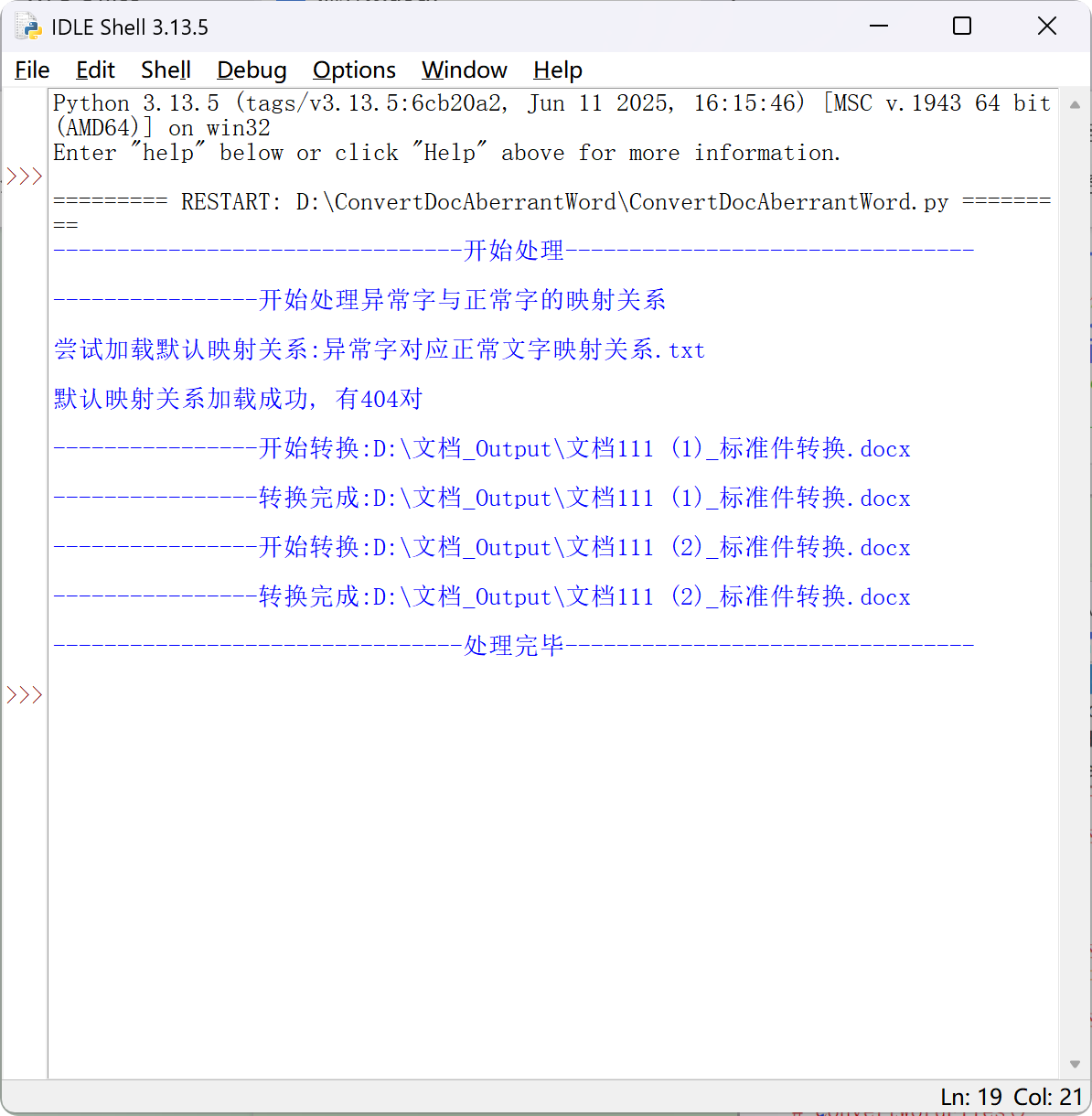
1. 如果Word文档多或者是不想复制Word文档，那么可以简单的修改几个英文字母就行，
   1. 右键代码文件->选择Edit with IDLE->Edit with IDLE 3.13(64-bit)：



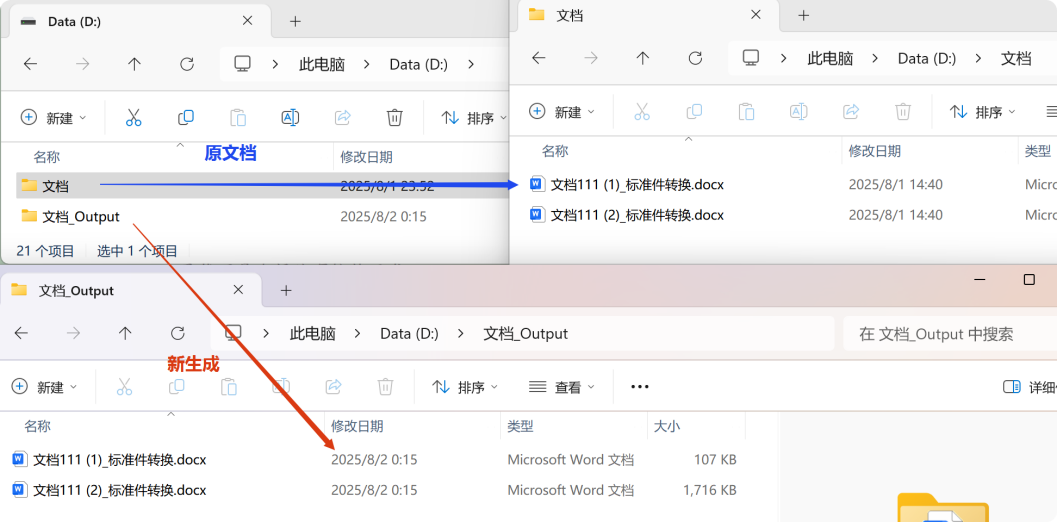
* 1. 拖到底部改几个单词：



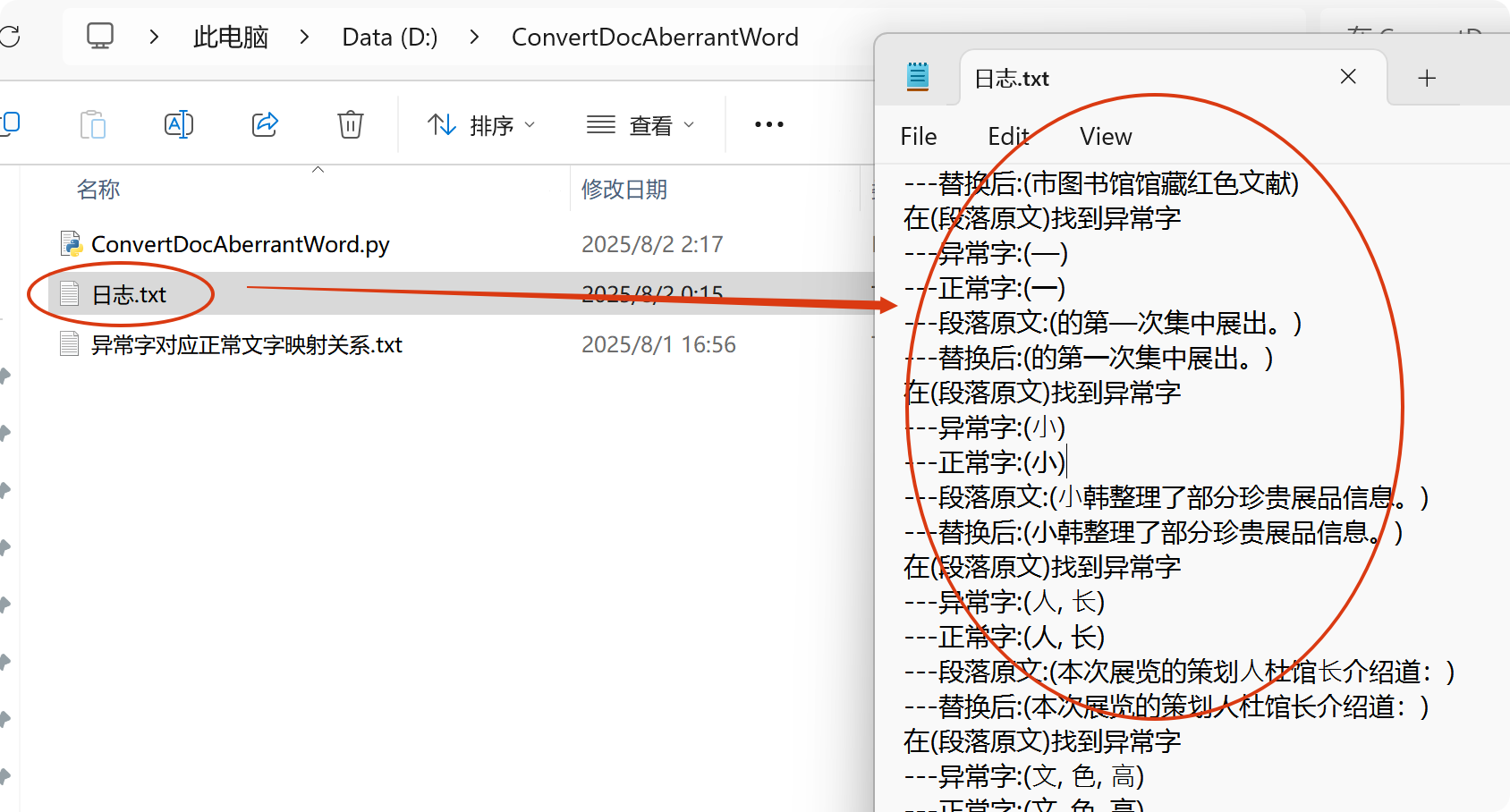
* 1. 等这个弹窗显示处理完毕：



* 1. 查看对应生成的文件：



1. 如果有兴趣或者想查看替换处理细节，可以查看日志：

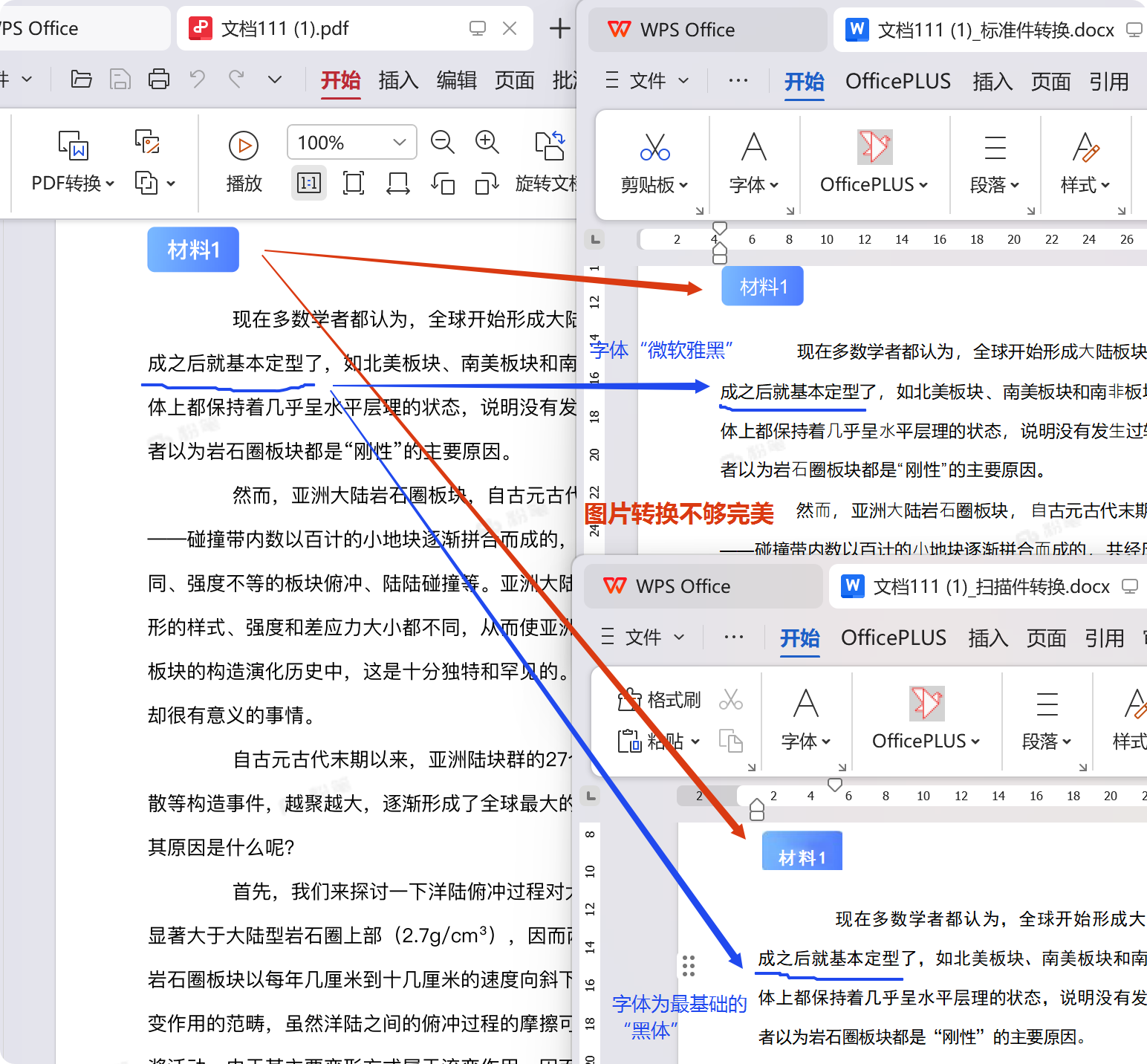


1. 原理：

简单来说，文字的显示是根据这个文字片段中的编码，挨个读取二进制数据，根据编码格式的不同，一个汉字占2到4个字节，读取到对应的字节数据后，转换对应文字编码，然后在文字片段中的字体中找到与之相对应的字形，然后进行后续显示。当然这样解释可能还是有点绕，**那么再简单一点来说，比如:(⾕)和(谷)就不是一个字，**它们只是看着像，因为它们的**字形一样**，但它们的**编码不一样，**所以在字体与字体间缺少对应关系，只保留了最基础的兼容性显示，所以Word中切换字体才不会有效果，所以必须这个看着很像的字换成我们日常的字才行。但这项操作是没有现成的，因为这类“看着像的字”为了兼容性，仍然是被认为是有效的字（Q&A第6点会展开说明），所以想要批量解决，得建立“看着像的字”和“现在日常用字”的映射关系，然后在利用映射关系在文档中批量替换。

1. Q&A：
2. Q：文中只提到工具只针对“标准件转换”的处理，那“扫描件转换”的文档可以用这个工具解决吗？

A：“扫描件转换”的文档如果出现类似问题，当然也可以使用，只不过在我自己的测试中，使用“扫描件转换”的文档一般不会出现类似问题，所以在没有特别指明。然后“标准件转换”和“扫描件转换”的详细区别有兴趣可以问AI，这里简单提一下，对于WPS的PDF转换工具来说，我个人觉得这两者的转换都有各自的不够好的地方，“标准件转换”容易出现部分字转换异常，而“扫描件转换”部分图片转换的异常，字体转换也不一致，所以根据具体情况具体取舍就行，“扫描件转换”如图：



1. Q：为什么各个软件的PDF转换工具推出那么久，这个问题没有解决，反而需要我们用户自己解决？

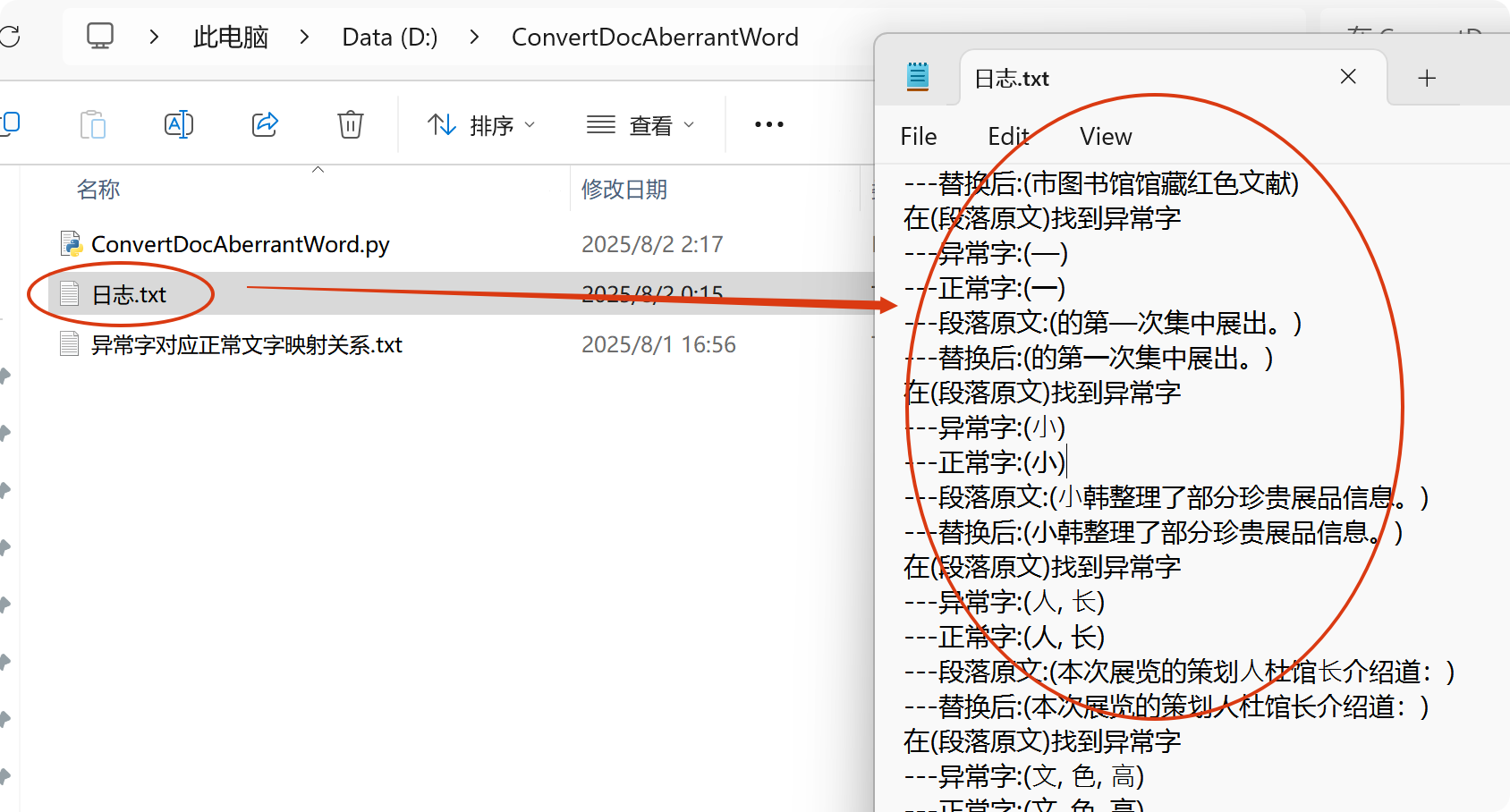
A：虽然我不是专门写软件的，但也写了10多年的代码，我个人觉得，这里面有很多的复合原因：（1）文档转换是一个复杂的系统工程，不单单只是文字、还有图片、控件等等，并且目前市面上的文字有成百上千个，里面涉及的适配问题的复杂度很高；（2）边际效应，或者说性价比问题。对于正式开发来说，除了做公益，大多情况还是要追求利润。对于市面上存量并且与日俱增的这么多字体，嗯你懂的；（3）正式开发的标准要求比较高，在下面Q&A的第3点会详细说明（4）正式开发的流程复杂，而越大的公司越复杂。通俗来说“代码不是程序员想写就可以写的”、“产品经理/策划不是想要一个功能就能做的”，一个功能，从它被提出，到它落地，可能要经过很多次“流产”，当然也不是说这样不好，只是侧重点不一样，站在的位置不同，考虑的角度不同。（5）版权问题，字体数量很多，很多都是商业字体，那么版权也是个很麻烦的事情，不买就不让用，买了还得经过财务等部门，那流程更多了；（6）而从前面几点就可以知道，KPI/OKR也是一种问题。

1. Q：那单从技术问题来说，解决这个问题真的很难吗？

A：俗话说的好“三千块有三千块的做法，三万块有三万的做法”，做为一个自认为写代码还凑合的码农，我可以很负责任的说，要求不同，实现的方式就不同，难度也就不同。“解决单个问题”、“解决单类问题”、“解决多类问题”、“解决系统性问题”，里面的要求和工作量同比增加。以日常的打球为例，业余朋友只要找块空地就可以愉快的穿着人字拖打野球，而专业人员需要地板/地胶、球鞋等等。那回到这个问题，我现在是无业游民，遇到一个必须要写代码才能解决的问题，那么我当然是想要尽快解决我遇到的问题，然后在性价比的时间内稍微注意性能和扩展功能就够了；假如我又开始吃码农这碗饭了，并且就负责wps的PDF转换工具，那么我觉得可以做的东西就很多了,比如：设计一个兼容性够健壮的适配框架、支持各种灵活方便的自定义配置功能、有着相对智能的定期获取更新与自检机制、有着大部分语言的语义分析和词法分析功能以便精准找到映射关系、支持多线程加速、支持部署到云端的分布式框架等等，所以得看具体的开发需求来具体分析。

1. Q：那这个工具代码量并不多？能搞定问题吗？会不会有误替换的情况？如果有会打算优化吗？

A：当然不能搞定所有问题，但对于开头图片提到的那些都能搞定。唯一存在误替换风险的是这个符号（—）,PDF转Word的汉字中我们平常写的（一）会转换成这个符号（—），而这个符号在标准的文字和文档中一般不会出现，即：这个符号既不是减号也不是破折号，所以风险低，为了追求保底可以查一下日志，搜一下有没有出现，日志里包含了所有的处理流程，肉眼确认一下，如图：



至于优化，暂时不考虑，因为我想到比较好的的解决方式就是说语义分析了，分析文字段落的上下文，看是否真的需要替换为符号、还是汉字、还是数字等等，但这个不多写个一两千行代码是搞不定的，所以时间有限、性价比不高，暂时不搞。

1. Q：为什么要选择Python来写？因为安装并使用Python的步骤还是太复杂了，能不能做个exe或者微信小程序来处理？

A：选择Python一是想让使用者能直接能看到源代码，这样就不用担心我是否会夹带病毒之类的，然后能执行开源代码，Python应该是最方便的语言之一了；二是想开源，因为知识、技术相互流通、交流才会有进步；三是我的电脑没带在身边，用我妹子的电脑，不想给她电脑装那么多环境。最后然后当然可以做exe和小程序，但不是现在，以后如果相关PDF转换还没有优化好，然后我又有空会做。

1. Q：为什么很多字体中会有两个长得很像的字？

A：具体情况可以问AI，这里简单说一下，主要原因是“解决兼容性问题”，可以很简单的理解，文字与字体的发展经历了很多年，总会面临这样或那样的问题，比如：对于一个字体来说，可能前几十年它长得成这样，但某天觉得不好看了、侵权了、文字数量需要扩容了等等。那么需要新增一个很像的字，来适应新的需求。那么可能会有人有疑问，那么不能改掉原先的字吗？答案：一般来说是不行的。因为使用这个字体的文档和用户数量可能太久太多，里面可能有大占比的东西是不能或不想被修改的。比如合同、法律文书等，同一份电子版，在不同的时间点打印出来文字却不一样，那么对于敏感性强的文档是不允许的。

1. Q：为什么这工具代码看起来有点别扭？

A：因为我不是从事软件开发，Python也不是我的主语言，以前经常同时用几个语言开发，所以有自己习惯的命名方式和书写结构，以便书写一致性。

最后记录一下，自己写这个工具的一些体会，虽然写一个工具对于任何一个程序员来说，其实并不算什么，但技术这个东西，确实旧不用，就会容易钝，所以主要想写下来给自己提个醒。因为原本是一个下午就能搞定的事，但却花了两三天。提醒一下自己：就是别偷懒。因为在分析是要针对字体进行处理后,却还抱着偷懒和侥幸心理，先不当它是一个适配问题去设计代码，而是用想加多一点判断就能解决所有的字体。但不出所料，字体之间就是不一样，有的包含一种字形对应两种编码，有的不包含；有些部分字体虽然包含两种编码，但他们的编码分布规则不一样；字体的集合规则不一样，有A交B、B包C等等。所以最后调试来调试去，也理所当然的返工了。