

**《数据结构与算法分析》**

**课程设计**

**题目：**

**学生姓名：**

**学 号：**

**专业班级:**

**指导教师：**

**学 院: 大数据与计算机学院**

**2019年 月**

1. 项目总体设计

软件名称：简单文本处理程序

1.1 需求分析

用户可通过程序打开硬盘上已有的文本文件，也通过程序新建一个文本文件，从键盘输入文本保存在该文件中。

（1） 统计功能：分别统计出文本文件中的各类字符的个数，包括英文字母个数、空格个数、汉字个数、标点符号个数、总字数等并显示统计信息；允许用户统计某一字符串在文章中出现的次数，并显示统计信息；

（2） 基本操作：包括文本的复制、粘贴、剪切、删除、查找、替换等功能。

（3） 加密和解密：用户可对指定文本文件进行加密和解密操作。

（4） 保存功能：以上操作完成后，用户可保存该文件。

1.2系统功能模块设计

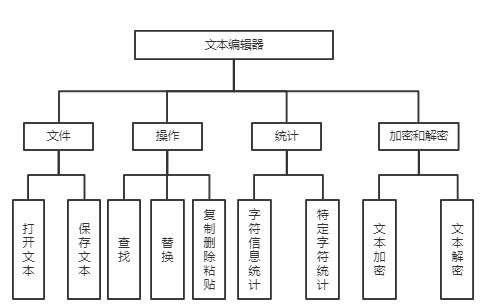


图1-1 系统功能模块图

2．详细实现步骤、流程图、代码：

项目文档的主体，要求思路清晰、描述准确、结构完整、重点突出，可以插入相应的代码和插图来支持主题。

2.1 阐述程序已实现的功能（必须是已经实现的、能够演示的功能）。

已实现功能包括:

(1). 打开和保存文本文件(\*.txt)

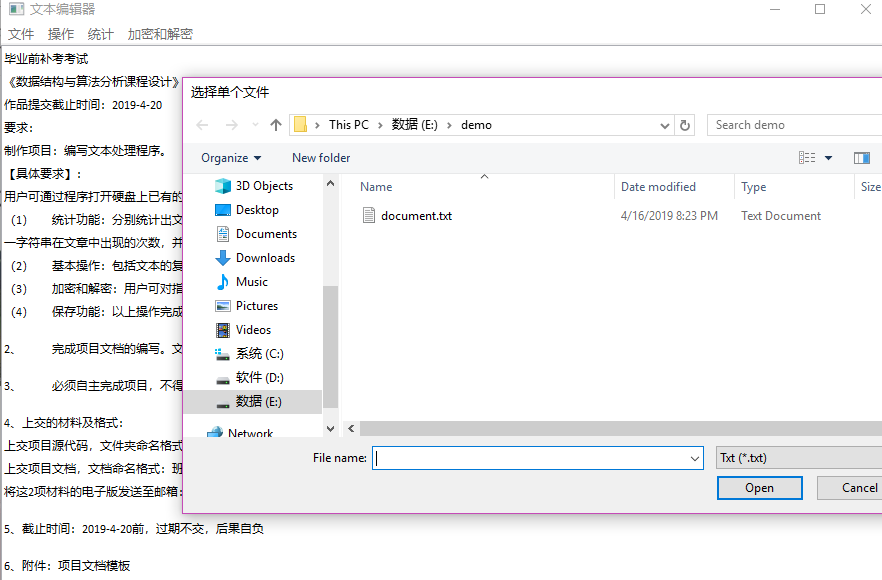


图2-1 打开文件

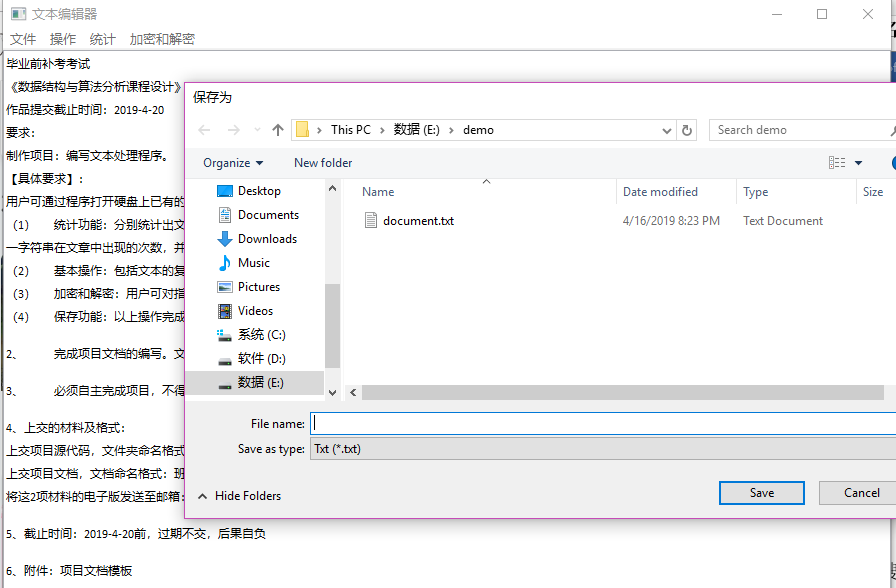


图2-2 保存文件

(2). 对文本内容进行搜索和替换

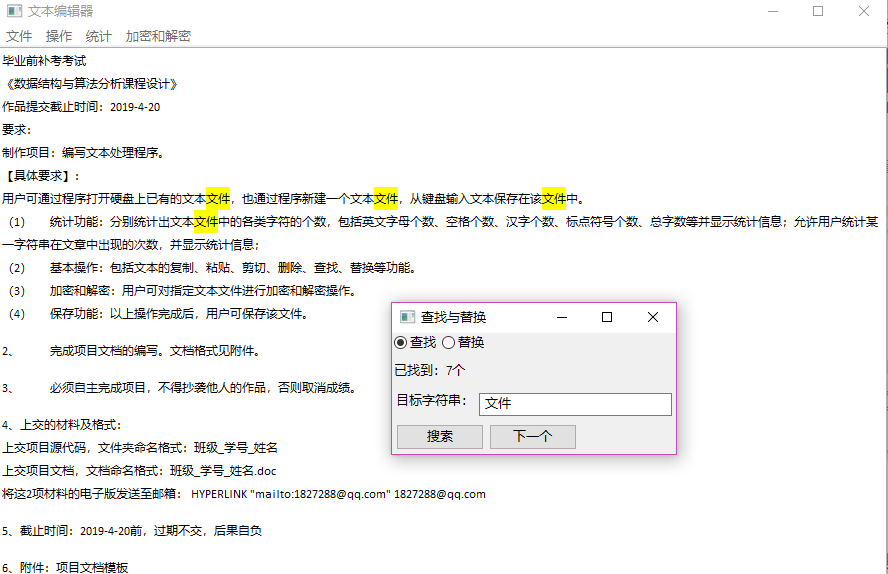


图2-3 搜索字符串

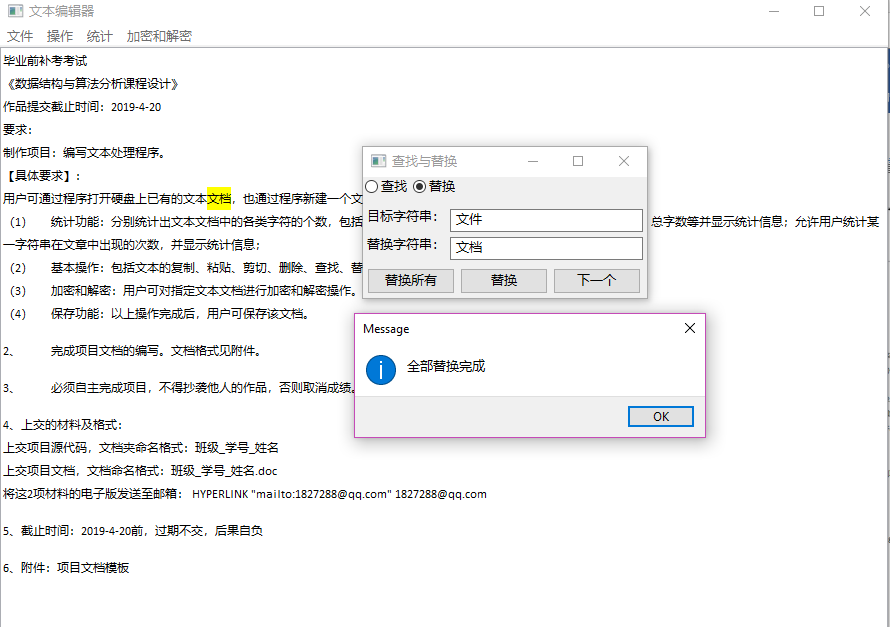


图2-4 字符串替换

(3). 右击鼠标对文本内容进行复制，粘贴，撤销，反撤销，剪切



图2-5 基本操作

(4). 统计文本出现的英文字母，汉字，标点符号，总字符数以及查询特定字符串在文本中出现的次数

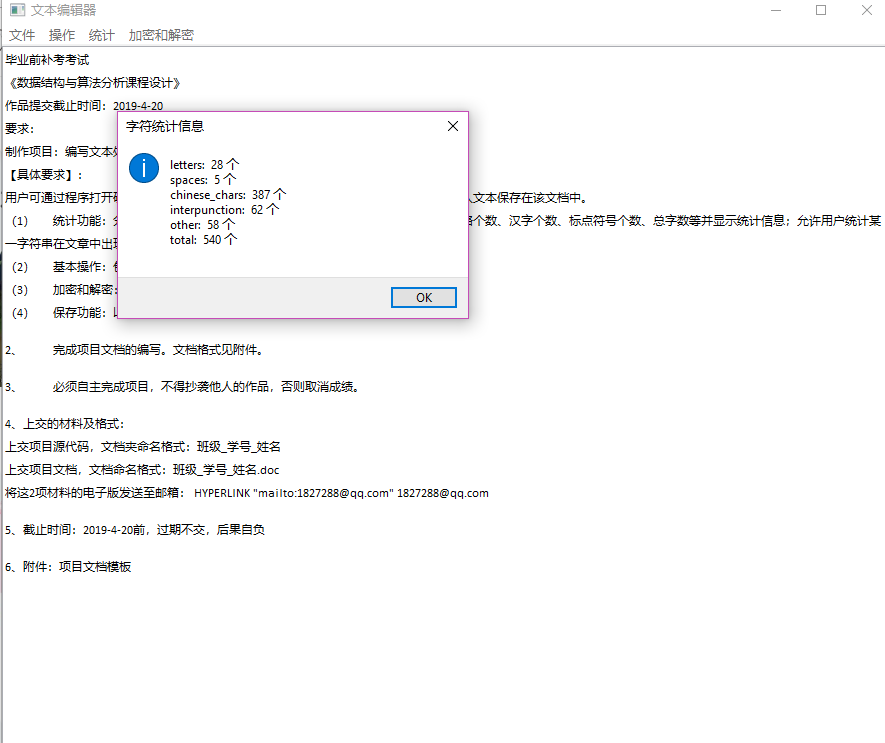


图2-6 统计

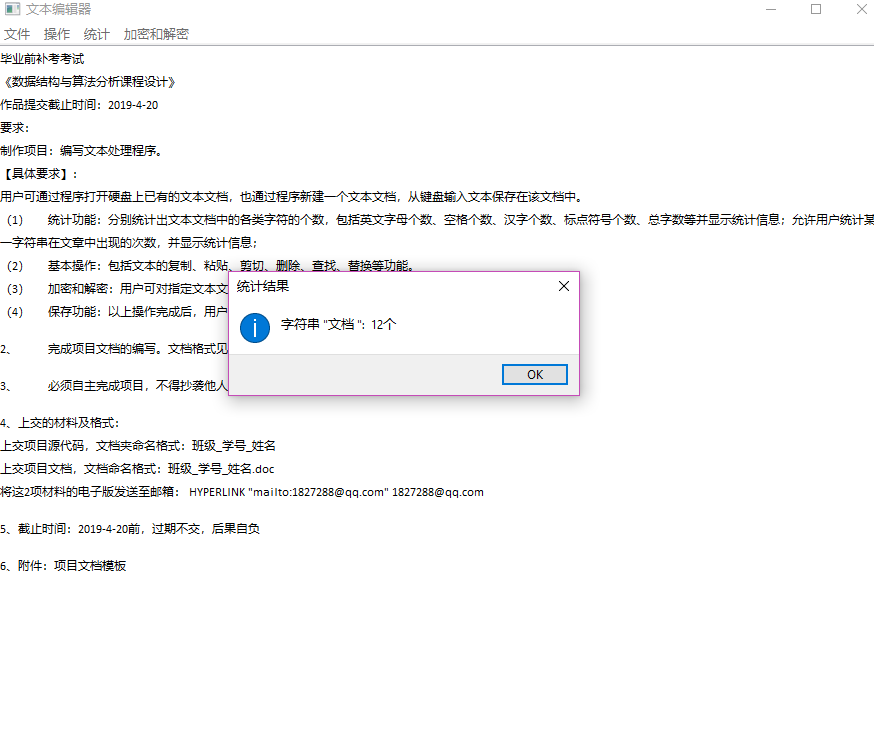


图2-7 统计特定字符串

(5). 对文本内容进行加密和解密

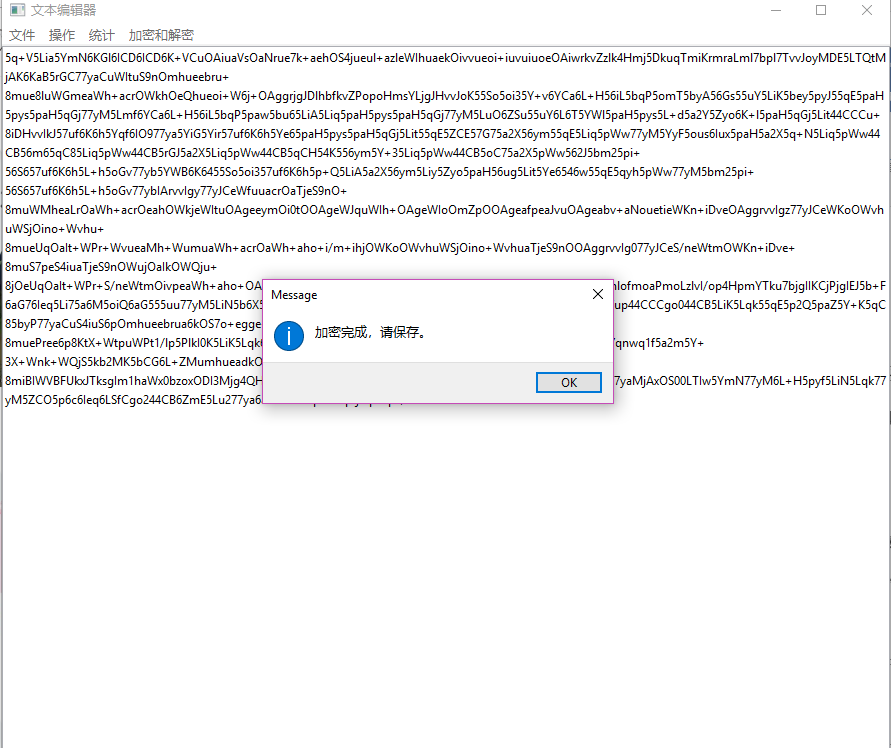


图2-8文本加密

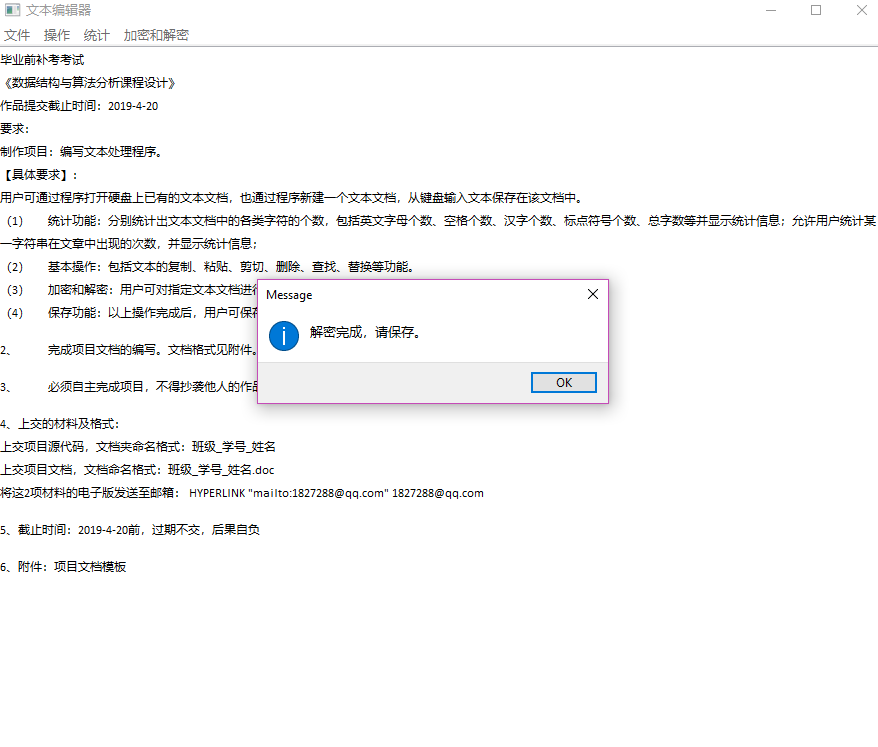


图2-9文本解密

2.2 阐述使用了哪些数据结构及技术来实现这些功能，是如何实现的。重点阐述核心功能的实现过程。

该文本处理程序采用Python语言以及其扩展包WxPython开发。义字符串(数组)形式保存和显示文本内容。

功能实现：

(1). 打开文件；以自读方式打开文件，以字符串数组方式保存文件内容并显示在wx.TextCtrl内。

1. **def** open\_file(self,evt):
2. file\_filter = "Txt (\*.txt)|\*.txt|" "All files (\*.\*)|\*.\*"
3. file\_dialog = wx.FileDialog(self, message="选择单个文件", wildcard=file\_filter, style=wx.FD\_OPEN)
4. r = file\_dialog.ShowModal()
5. **if** r != wx.ID\_OK:
6. **return**
7. path = file\_dialog.GetPath()
8. with open(path, 'r') as f:
9. self.main\_text.SetValue(f.read())

(2). 保存文件；获取wx.TextCtrl中的内容，以二进制格式写入指定保存的文件中。

1. **def** save\_file(self, evt):
2. **global** main\_text
3. file\_filter = "Txt (\*.txt)|\*.txt|" "All file(\*.\*)|\*.\*"
4. file\_dialog = wx.FileDialog(self, message="保存为", wildcard=file\_filter, style=wx.FD\_SAVE | wx.FD\_OVERWRITE\_PROMPT)
5. r = file\_dialog.ShowModal()
6. **if** r == wx.ID\_OK:
7. with open(file\_dialog.GetPath(), 'w') as f:
8. f.write(self.main\_text.GetValue())

(3). 加密和解密；采用base64对文本内容进行加密和解密。

1. **def** encrypt\_file(self, evt):
2. content = self.main\_text.GetValue()
3. encry\_content = base64.b64encode(content.encode())
4. self.main\_text.SetValue(encry\_content.decode())
5. wx.MessageDialog(self,"加密完成，请保存。").ShowModal()

8. **def** decrypt\_file(self, evt):
9. content = self.main\_text.GetValue()
10. decry\_content = base64.b64decode(content)
11. self.main\_text.SetValue(decry\_content)
12. wx.MessageDialog(self, "解密完成，请保存。").ShowModal()

(4). 统计字符；用正则表达式对文本内容中的汉字，英文字母，标点符号，空格的进行匹配，以获取全部出现的次数。由于文本内容以字符串形式保存，所以计算字符串的长度等于文本内容的总字符数。

对特定字符串进行计算，从文本内容开头开始匹配，找到记录+1，直到文本末尾。

1. **def** statistics\_all(self, evt):
2. content = self.main\_text.GetValue()
3. statistics\_info = {'letters':0, 'spaces':0, 'chinese\_chars':0,
4. 'interpunction':0, 'other':0, 'total':len(content)
5. }
6. regex\_letters = re.compile(r'[a-zA-Z]')
7. regex\_spaces = re.compile(r' ')
8. # [\u4E00-\u9FA5]
9. regex\_chinese\_chars = re.compile(r'[\u2E80-\u9FFF]')
10. regex\_interpunction = re.compile(r'[。，、；：？！“”……——《》（）\[\]\.,]')
11. statistics\_info['letters'] = len(regex\_letters.findall(content))
12. statistics\_info['spaces'] = len(regex\_spaces.findall(content))
13. statistics\_info['chinese\_chars'] = len(regex\_chinese\_chars.findall(content))
14. statistics\_info['interpunction'] = len(regex\_interpunction.findall(content))
15. statistics\_info['other'] = statistics\_info['total'] - statistics\_info['letters']\
16. - statistics\_info['spaces'] - statistics\_info['chinese\_chars']\
17. - statistics\_info['interpunction']
18. infos\_str = ""
19. **for** key, value **in** statistics\_info.items():
20. infos\_str += key + ":  " + str(value) + " 个\n"
21. infos\_dialog = wx.MessageDialog(self, message=infos\_str,caption="字符统计信息")
22. infos\_dialog.ShowModal()

25. **def** statistics\_spec(self, evt):
26. text\_dialog = wx.TextEntryDialog(self, message="输入字符串", caption="统计特等字符串数目")
27. r = text\_dialog.ShowModal()
28. text\_input = text\_dialog.GetValue()
29. **if** r == wx.ID\_OK:
30. message = "请输入字符串"
31. **if** text\_input:
32. count = 0
33. **for** i **in** range(self.main\_text.GetNumberOfLines()):
34. old\_pos, new\_pos = 0, 0
35. **while** new\_pos != -1:
36. new\_pos = self.main\_text.GetLineText(i).find(text\_input, old\_pos)
37. **if** new\_pos == -1:
38. **break**
39. old\_pos = new\_pos + len(text\_input)
40. count += 1
41. message = '字符串 "' + text\_input + ' ":  ' + str(count) + "个"
42. message\_box = wx.MessageDialog(text\_dialog, message,caption="统计结果")
43. message\_box.ShowModal()
44. text\_dialog.ShowModal()

(5). 查找和替换；类似地查找和替换字符串也可以用正则表达式实现。

1. **def** query(self, evt):
2. self.next\_btn.Enable()
3. content = self.text\_content.GetValue()
4. key = self.query\_text.GetValue()
5. **if** **not** key:
6. wx.MessageDialog(parent=self,message="输入查找字符串").ShowModal()
7. **return**
8. **if** **not** content:
9. wx.MessageDialog(parent=self,message="请先打开文件").ShowModal()
10. **return**
12. self.text\_content.SetStyle(0, len(content), wx.TextAttr("black", "white"))
13. count = 0
14. **for** i **in** range(self.text\_content.GetNumberOfLines()):
15. old\_pos, new\_pos = 0, 0
16. **while** new\_pos != -1:
17. new\_pos = self.text\_content.GetLineText(i).find(key, old\_pos)
18. **if** new\_pos == -1:
19. **break**
20. old\_pos = new\_pos + len(key)
21. count += 1
23. self.label\_count.SetLabel("已找到："+ str(count) + "个")
24. **def** replace\_all(self, evt):
25. # replace all or alone
27. key = self.query\_text.GetValue()
28. replace\_str = self.replace\_text.GetValue()
29. content = self.text\_content.GetValue()
30. **if** **not** key:
31. wx.MessageDialog(parent=self,message="输入查找字符串").ShowModal()
32. **return**
33. **if** **not** content:
34. wx.MessageDialog(parent=self,message="请先打开文件").ShowModal()
35. **return**
37. self.text\_content.SetValue(content.replace(key, replace\_str, -1))
38. index = self.text\_content.GetValue().find(replace\_str, 0)
39. self.text\_content.SetFocus()
40. self.text\_content.Bind(wx.EVT\_SET\_FOCUS, self.highlightText(index, index + len(replace\_str)))
41. self.SetFocus()
43. message = wx.MessageDialog(self, message="全部替换完成")
44. message.ShowModal()

2.3 程序的调试和运行情况的简述。

在win10 64位系统下运行情况良好，但只局限于正常操作，若打开的文件不是\*.txt类型的文本文件，虽然能打开，但是不会显示内容，如打开\*.docx文件。

1. 项目总结

该软件在用户正常操作情况下能基本完成需求指定的功能，但存在容错性不足。如：

(1). 在已打开文件的情况下再次打开文件，没有提示保存当前文件，而是直接打开新文件。(2). 重复点击‘查找和替换’/‘统计’会出现多个子窗口。

(3). 对已加密的文件需要手动保存，这会令用户误改内容而导致源文本内容丢失

(4). 加密方式相对简单，没有密钥进行解密，安全性大大降低。

参考文献

Noel Rappin, Robin Dunn. 《wxPython in Action》. 2006年.

Eric Matthes (作者) 袁国忠 (译者).《Python编程：从入门到实践》. 2018年