基于Python的自动化录取系统的设计与实现

涂青云

江西财经职业学院

摘　要：本文介绍了一种基于Python的高职院校自动化录取和打印录取通知书的方法。该方法利用Python编程语言和相关库，实现了从数据采集、数据处理、通知书生成到打印的全部自动化流程。相较于传统的手动制作方式，该方法大大提高了效率，减少了错误率，为高校招生工作提供了便捷的工具。文章首先概述了系统的设计，包括数据采集、数据处理、通知书生成和打印等模块，然后详细阐述了每个模块的实现方法和具体的操作流程。本系统的设计和实现方法具有普遍的适用性，可以为其他高校或教育机构提供一种有效的自动录取和化打印录取通知书解决方案。

关键词：Python；自动化打印；录取通知书；数据采集；数据处理

【DOI】10.12252/j.issn.2096-6288.2022.10.097

一、引言

在当今信息化社会，自动化和智能化已经成为各行各业追求的目标。特别是在教育领域，随着招生规模的不断扩大和信息化建设的深入推进，高校招生工作的效率和准确性已经成为关注的焦点。录取通知书作为高校招生工作的重要环节，其制作和发放过程的质量和效率直接关系到考生的切身利益和高校的声誉。因此，如何实现录取通知书的自动化制作和打印，提高工作效率和准确性，是当前高校招生工作面临的一个重要问题。

针对这一问题，本文提出了一种基于Python的高职院校自动化打印录取通知书的方法。该方法利用Python编程语言和相关库，实现了从数据采集、数据处理、通知书生成到打印的全部自动化流程。相较于传统的手动制作方式，该方法不仅大大提高了效率，还减少了错误率，为高校招生工作提供了便捷的工具。同时，该方法还可以根据高校需求进行定制化开发，满足不同高校的需求。

本文的目的是通过实现基于Python的高职院校自动化打印录取通知书，提高招生工作的效率和准确性，为考生提供更好的服务。本文的研究成果不仅具有一定的理论价值，还具有一定的实践指导意义。

二、python在教育自动化办公方面的应用

（一）python在自动化办公方面的优势

（1）门槛低、易学易用：Python的语法简洁明了，易于理解，使得初学者可以快速上手。同时，Python的代码编写较为直观，易于阅读和维护，减少了代码维护的难度和成本。

（2）丰富的库和工具：Python拥有庞大的库和工具生态系统，可以满足各种不同的需求。例如，Python的pandas库可以用于数据处理和分析，matplotlib库可以用于数据可视化，numpy库可以用于数值计算等。这些库和工具的集成使得Python成为自动化办公的理想选择。

（3）自动化任务：Python可以轻松地实现各种自动化任务，例如定时任务、批量任务、自动化测试等。这些任务的自动化实现可以大大提高办公效率，减少人工由于粗心造成的错误。

（4）集成开发环境：Python拥有丰富的集成开发环境（IDE），例如Spyder、PyCharm、Jupyter Notebook等，这些IDE提供了强大的代码编辑、调试和管理功能，使得开发人员可以更加高效地进行自动化办公。同时，很多的开发环境都有社区版，完全可以满足日常的工作需要。

（5）跨平台性：Python可以运行在多种操作系统上，包括Windows、Linux和苹果操作系统等。这使得Python在自动化办公方面具有更好的跨平台兼容性，可以满足不同用户的需求。

（二）python在教育行业的应用

（1）自动化评分：Python可以轻松地读取和处理各种格式的文档，例如Word、PDF等。通过使用Python的库，可以自动化地读取和评分学生提交的作业或考试答卷，大大提高评分效率和准确性。

（2）数据分析：Python的pandas库和numpy库可以用于处理和分析大量的教育数据。通过自动化数据采集和处理，可以发现学生的学习特点和规律，为教师提供更准确的学生表现评估和教学策略优化。

（3）智能辅助教学：Python可以与机器学习和人工智能技术结合，实现智能辅助教学。例如，通过分析学生的学习行为和成绩，可以自动化地推荐学习资料、制定学习计划和预测学生未来的学习表现。

（4）自动化任务管理：Python可以自动化地管理各种教育任务，例如发送通知、管理日程、安排课程等。这样可以减轻教师的工作负担，提高工作效率。

（5）教育资源共享：Python可以用于开发教育资

源共享平台，实现教育资源的自动化管理和共享。学生和教师可以轻松地搜索、下载和使用各种教育资源，提高教育资源的利用效率。

三、自动化录取系统设计

自动化录取系统设计分为数据采集模块设计、数据处理模块设计、录取通知书生成模块设计和发送通知模块设计等4个方面。当高考结束后，每位同学都知道了自己的高考分数和省内排名，他们会结合自己的爱好，就业情况等消息来选择适合自己的学校和专业。同时，每个学校录取的新生较多，一般有几千人，这么多人的录取和打印及邮寄通知书工作，如果靠人工完成，不仅需要大量的工作人员，而且有可能造成不同的工作人员配合不默契出现错误情况。而出现的错误如果影响了某位学生录取情况，将会是重大的录取事故。因此，从各个方面将，采用基于python的自动化录取系统都十分必要。

自动化录取系统的主要任务是从教育部门采集填报志愿的考生信息，从各高校采集各专业的招生信息，包括省份、专业、录取人数等，然后将两部分信息比对，进行择优录取，减少人工工作量和人工失误。具体步骤如下：

（1）遵循录取规则：高校会根据自己的招生计划和预设的录取规则，对所有报考的考生进行排序和筛选。这个过程通常会综合考虑考生的高考成绩、专业志愿以及其他可能的加分项等。

（2）进行预录取：经过排序和筛选后，高校会进行预录取，即初步确定哪些考生符合录取条件。这个过程通常是按照一定的比例进行的，以确保有足够的考生供最后的选择。

（3）录取确认：预录取后，高校会与相关的招生部门进行录取确认，确保每一位被录取的考生都符合相关的政策和规定。这个过程通常会包括对考生档案的再次审核，以及与省级招办进行确认等。确认录取后，高校会锁定考生的电子档案，其他学校就不能再次录取该考生。

（4）发布录取结果：完成录取确认后，高校会通过官方网站、招生简章等渠道向社会公布录取结果，包括被录取的考生的姓名、准考证号等信息。同时，高校也会向被录取的考生寄发正式的录取通知书。

（一）数据采集模块设计

数据采用模块的主要工作任务是通过python的第三方库requests库实现考生电子档案的自动化采集。在高考录取过程中，高校会首先下载并审阅所有报考该校的考生的电子档案。这些档案中包含了考生的高考成绩、专业志愿、是否服从调剂等重要信息。自动化系统之前采取的方法是人工逐个下载，逐个分析是否符合所有报考条件。具体步骤如下：

（1）定义爬虫策略，包括要采集的数据类型、数据来源和采集方式等；

（2）使用requests库发送HTTP请求，获取需要采集的网页内容；

（3）使用BeautifulSoup库解析网页内容，提取所需数据；

（4）将采集的数据存储到数据库或文件中，以备后续处理。

（二）数据处理模块设计

数据处理模块的主要任务是对采集到的数据进行处理，包括数据清洗、转换和分析等操作。本系统采用Python的第三方库Pandas来进行数据清洗、转换和分析操作。具体实现步骤如下：

（1）对采集到的数据进行清洗，去除无效和错误数据；

（2）对清洗后的数据进行转换，将不同来源和格式的数据转换成统一的格式；

（3）对转换后的数据进行统计分析，提取重要信息和特征；

（4）将处理后的数据存储到数据库或文件中，以备后续通知书生成和发送通知等操作。

（三）通知书生成模块设计

通知书生成模块的主要任务是利用模板引擎根据处理后的数据自动生成录取通知书。本系统采用Jinja2模板引擎来生成录取通知书。具体实现步骤如下：

（1）定义录取通知书模板，包括所需的变量和格式；

（2）将处理后的数据与模板进行绑定，将变量替换成实际数据；

（3）使用Jinja2模板引擎将绑定后的数据进行渲染，生成录取通知书；

（4）将生成的录取通知书存储到指定目录或文件中。

（四）发送通知模块设计

发送通知模块的主要任务是通过邮件、短信等方式自动向考生发送录取通知书。本系统采用Python的第三方库smtplib来发送邮件通知，使用Twilio库来实现短信通知。具体实现步骤如下：

（1）定义发送策略，包括发送方式、发送时间、发送对象等；

（2）使用第三方库smtplib发送邮件通知，通过邮件服务器将录取通知书发送给考生；

（3）使用第三方库Twilio发送短信通知，通过短

信网关将录取通知书发送给考生；

（4）记录发送结果和状态，以便后续查询和管理。

四、自动化录取系统实现

（一）自动化打印录取通知书的流程

（1）数据采集：通过与招生系统对接或者爬取相关网页数据的方式实现数据采集。在数据采集过程中，需要确保数据的准确性和完整性，同时避免重复采集和无效采集。

（2）数据处理：对采集的数据进行处理，如数据清洗、格式转换等。数据处理过程中需要注意数据的规范性和一致性，避免因数据格式不规范导致后续处理出现问题。

（3）通知书模板设计：根据学校特色和需求设计通知书模板，包括通知书格式、排版、字体、颜色等。同时需要考虑通知书模板的灵活性和可维护性，方便后续更新和维护。

（4）通知书生成：将处理后的数据与通知书模板进行绑定，使用Python中的第三方库PDF库（如PDFMiner或PyPDF2等）将绑定后的数据进行渲染，生成PDF格式的录取通知书。在通知书生成过程中需要注意PDF文件的格式和质量，确保打印效果良好。

（5）通知书打印：将生成的PDF文件通过打印机进行打印。在打印过程中需要注意打印质量和效率，同时避免纸张浪费和环境污染。为了提高打印效率和精度，可以考虑使用专业的打印管理软件或者云打印服务。

（6）通知书封装：将打印好的录取通知书进行封装，以防止邮件在传递过程中受到损坏或者丢失。封装方式可以因学校需求而异，如使用信封或者专门的通知书包装袋等。

（7）发送通知：通过邮件、短信等方式向考生发送录取通知书，提醒考生查收。在发送通知过程中需要注意信息的准确性和及时性，避免因通知不及时导致考生错过重要信息。

```
import os
from docx import Document
# 读取录取数据文件（假设为 CSV 格式）
import pandas as pd
data = pd.read_csv('admitted_students.csv')
# 定义通知书模板路径和输出路径
template_path = 'template.docx'
output_path = 'output/'
# 循环遍历录取数据，生成每位学生的通知书
for index, row in data.iterrows():
    # 复制通知书模板并命名
    doc = Document(template_path)
    doc.save(output_path + row['student_id'] + '.docx')
    # 打开新生成的通知书文档并编辑
    doc = Document(output_path + row['student_id'] + '.docx')
    doc.add_paragraph('亲爱的 ' + row['student_name'] + ' 同学：')
    doc.add_paragraph('恭喜你被我校录取为 XXXX 级新生！')
    doc.add_paragraph('专业：' + row['major'])
    doc.add_paragraph('学费：' + row['tuition'] + ' 元/年')
    doc.add_paragraph('请于 XXXX 年 XX 月 XX 日前到校报到。')
    doc.save(output_path + row['student_id'] + '.docx')
    # 打印通知书
    os.system('lpr -P printer_name ' + output_path + row['student_id'] + '.docx')
```

图 1

（二）python代码实现

为了实现这个功能，首先需要创建一个包含所有必要信息的字典，然后使用该字典生成录取通知书。图1为自动生成的电子版通知书的核心python代码。

下面为针对图1代码的5点说明，实际上每位同学收到的录取通知书除了姓名和录取的专业不同之外，还可以进行个性化设计，反应不同省份不同地市的特色，例如可以将江西九江的学生通知书设计成庐山和陶渊明的水印。

（1）读取录取数据文件：使用pandas库读取CSV格式的录取数据文件，获取每个学生的信息。

（2）循环遍历录取数据：使用for循环遍历录取数据，依次生成每位学生的通知书。

（3）复制通知书模板并命名：使用python-docx库复制通知书模板，并将新生成的通知书命名为学生的学号。

（4）编辑新生成的通知书文档：使用python-docx库打开新生成的通知书文档，并根据学生信息编辑通知书内容。

（5）打印通知书：使用os库调用系统命令打印通知书。需要根据实际情况指定打印机名称。

结束语

本文介绍了一种基于Python的自动化高考录取通知书的设计与实现方法。该系统可实现从数据采集、数据处理、通知书生成、发送通知等一系列自动化操作，为高校招生工作提供了极大的便利。相比传统的手工制作方式，该系统大大提高了工作效率和准确性，降低了招生工作的成本和错误率。同时，该系统还具有可扩展性和可定制性，可以根据不同高校的需求进行功能扩展和定制化开发。因此，本文的研究成果具有一定的理论和实践指导意义。

参考文献

[1]赵晓霞.Python自动化测试框架设计与实现[J].软件工程与应用，2021，10（6）：45-52.

[2]李婷婷，王磊.基于pytest的Web应用自动化测试研究[J].计算机工程与应用，2021，57（12）：23-30.

[3]张志强.基于Selenium的自动化测试工具设计与实现[D].北京：北京邮电大学，2021.

[4]王华，李燕.Python在自动化测试中的应用与实践[J].软件导刊，2021，16（4）：45-52.

[5]刘明，王丽.基于Pandas的测试数据处理与分析[J].计算机工程与应用，2021，57（7）：34-40.

[6]陈小军.基于pytest的自动化测试框架设计与应用[D].上海：上海交通大学，2021.