**编译原理实践第2次课**

**（基于正则表达式的LaTex编译器）**

1. **什么是LaTex？**

LaTeX是一种基于ΤΕΧ的排版系统，由美国计算机学家莱斯利·兰伯特（Leslie Lamport）在20世纪80年代初期开发，利用这种格式，即使使用者没有排版和程序设计的知识也可以充分发挥由TeX所提供的强大功能，能在几天，甚至几小时内生成很多具有书籍质量的印刷品。对于生成复杂表格和数学公式，这一点表现得尤为突出。因此它非常适用于生成高印刷质量的科技和数学类文档。这个系统同样适用于生成从简单的信件到完整书籍的所有其他种类的文档。

1. **从Latex到PDF**

将LaTex源文件解析的结果构建为pyfPDF能够识别格式比如HTML，并通过pyfPDF生成pdf（example.pdf）

正则表达式

Latex源文件

HTML文件

PDF

FPDF

example.tex的标记很多，本次实验不要求完全实现，尽量能解析多的标记即可，但至少要完结解析如下标记：

* document
* title
* abstract
* section
* subsection
* itemize/item
* tabular
* emph
* textbf

**完成实验报告并提交**

1. **pyFPDF**

一种将网页转换为pdf的工具

项目主页：https://github.com/reingart/pyfpdf

安装方法：

1. 在线安装：pip install fpdf
2. 本地安装：python setup.py install

教程：https://github.com/reingart/pyfpdf/blob/master/docs/Tutorial.md

4、实验代码：

#! /usr/bin/env python

#coding=utf-8

import re

from html2pdf import html2pdf

# parse by regex

def re\_find(p,text):

m=p.findall(text)

if len(m)==1:

return m[0]

else:

return ''

def re\_findall(p,text):

return p.findall(text)

def clear\_text(text):

lines=[]

for line in text.split('\n'):

line=line.strip()

if len(line)>0:

lines.append(line)

return ' '.join(lines)

# 1. read tex file

content=open('example.tex','rb').read()

# 2. extract document part

p\_doc=re.compile(r'\\begin{document}(.+?)\\end{document}',re.S)

document=re\_find(p\_doc,content)

# 3. extract head

p\_title=re.compile(r'\\title{(.+?)}',re.S)

title=re\_find(p\_title,document)

# 4. extract abstract

p\_abs=re.compile(r'\\begin{abstract}(.+?)\\end{abstract}',re.S)

abstract=re\_find(p\_abs,document)

abstract=clear\_text(abstract)

# 5. sections

p\_sec=re.compile(r'\\section{(.+?)}(.+?)\\section',re.S) # only for Section 1

section\_title,section\_content=re\_find(p\_sec,document)

section\_content=clear\_text(section\_content)

p\_sec2=re.compile(r'\\section{([S].+?)}(.+?)\\subsection',re.S) # only for Section 2

section\_title2,section\_content2=re\_find(p\_sec2,document)

section\_content2=clear\_text(section\_content2)

p\_subsec=re.compile(r'\\subsection{(.+?)}(.+?)\\subsubsection',re.S) # only for SubSection

subsection\_title,subsection\_content=re\_find(p\_subsec,document)

subsection\_content=clear\_text(subsection\_content)

p\_subsubsec=re.compile(r'\\subsubsection{(.+?)}(.+?)\\subsubsection',re.S) # only for SubSubSection

subsubsection\_title,subsubsection\_content=re\_find(p\_subsubsec,document)

subsubsection\_content=clear\_text(subsubsection\_content)

p\_subsubsecion\_content=re.compile(r'([-].+?)\\',re.S)

subsubsection\_content=re\_findall(p\_subsubsecion\_content,subsubsection\_content)

p\_subsubsec2=re.compile(r'\\subsubsection{([A][u].+?)}(.+?)\\subsection',re.S)

subsubsection\_title2,subsubsection\_content2=re\_find(p\_subsubsec2,document)

subsubsection\_content2=clear\_text(subsubsection\_content2)

p\_subsubsecion\_content2=re.compile(r'[}](.+?)\\[bei]',re.S)

subsubsection\_content2=re\_findall(p\_subsubsecion\_content2,subsubsection\_content2)

# 6. generate html text

html\_text=''

# title

html\_text+='<h1>%s</h1>\n\n' %title

# abstract

html\_text+='<p>%s</p>\n\n' %abstract

# section -- 1

html\_text+='<h2>%s</h2>\n\n' %section\_title

html\_text+='<p>%s</p>\n\n' %section\_content

html\_text+='<h2>%s</h2>\n\n' %section\_title2

html\_text+='<p>%s</p>\n\n' %section\_content2

html\_text+='<h2>%s</h2>\n\n' %subsection\_title

html\_text+='<p>%s</p>\n\n' %subsection\_content

html\_text+='<h2>%s</h2>\n\n' %subsubsection\_title

for each\_content in subsubsection\_content:

html\_text+='<p>%s</p>\n\n' %each\_content

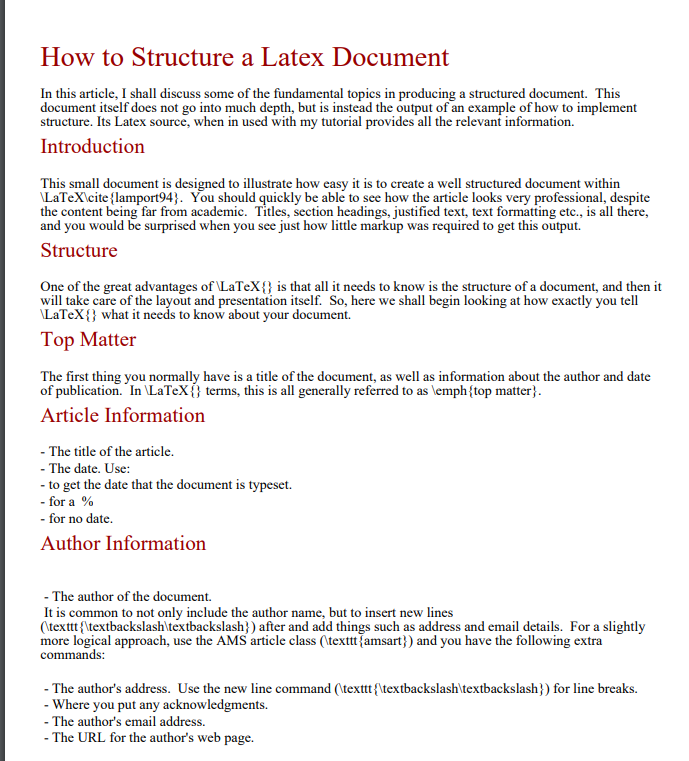
html\_text+='<h2>%s</h2>\n\n' %subsubsection\_title2

for each\_content2 in subsubsection\_content2:

html\_text+='<p>%s</p>\n\n' %each\_content2

html2pdf(html\_text, '2.pdf')

运行结果



5、实验总结

通过本次实验，我学会了利用正则表达式匹配latex的标签，并且将标签进行对应的替换然后再利用python的fPDF将html转化为PDF文件并实现输出，借此我进一步掌握了正则表达式。