Report of Lab4

Yuxiang Chen 5110309783

November 4, 2012

Contents

1 The Purpose of Lab4 and My Preparation

It is definitely sure that we have to build our own search engine by ourselves. But how to write such a program that enables us to search certain information in a graphic user interface, e.g. in a web page(port)? The framework 'web.py' gives us the chance.
In this experiment, we need to learn how to use this framework to visit a certain server where we can do our searching jobs. To accomplish this experiment, I look up many things in the website introducing web in python, and find that 'web.py' is just a framework that makes it possible to connect our searching programs with the html lists. And there are also some other things to do, such as print out the url, main contents and the title of the web page in the form of a hyperlink.

2 The Main Part of the Experiment

To make it easy to understand, I split this experiment into three parts, which consists of setting up an index, searching for information you required and establishing the framework of the html page to be presented later.

2.1 Making an Index First

This part is mainly the same as the experiments we did earlier, except that we need to do some work to analyse and store the text in the neighborhood of the keyword requested. And we also need to exclude the useless messages in the content of the page, such as how the web site set up the html page. And I will present them in the following codes:

```python
import sys, os, lucene, threading, time,chardet,urllib2
from datetime import datetime
from BeautifulSoup import BeautifulSoup
from ctypes import *
import re
import string

def filter_tags(htmlstr):
    re_cdata=re.compile('//<!\[CDATA\[[^>]*//\]\]>',re.I)
    re_script=re.compile('<\s*script[^>]*>[^<]*<\s*/\s*script\s*>',re.I)
    re_style=re.compile('<\s*style[^>]*>[^<]*<\s*/\s*style\s*>',re.I)
    re_br=re.compile('<br\s*?/?>')
    re_h=re.compile('</?\w+[^>]*>')
    re_comment=re.compile('<!--[^>]*-->')
    s=re_cdata.sub('',htmlstr)
    s=re_script.sub('',s)
    s=re_style.sub('',s)
    s=re_br.sub('\n',s)
    s=re_h.sub('',s)
    s=re_comment.sub('',s)
    blank_line=re.compile('\n+')
    s=blank_line.sub('\n',s)
    s=replaceCharEntity(s)
    return s


def replaceCharEntity(htmlstr):
    CHAR_ENTITIES={'nbsp':'␣','160':'␣','lt':'<','60':'<','gt':'>','62':'>','amp
    re_charEntity=re.compile(r'&#?(?P<name>\w+);')
    sz=re_charEntity.search(htmlstr)
    while sz:
        entity=sz.group()
        key=sz.group('name')
        try:
            htmlstr=re_charEntity.sub(CHAR_ENTITIES[key],htmlstr,1)
            sz=re_charEntity.search(htmlstr)
        except KeyError:
            htmlstr=re_charEntity.sub('',htmlstr,1)
```

```
                sz=re_charEntity.search(htmlstr)
    return htmlstr


def repalce(s,re_exp,repl_string):
    return re_exp.sub(repl_string,s)


class Ticker(object):

    def __init__(self):
        self.tick = True

    def run(self):
        while self.tick:
            sys.stdout.write('.')
            sys.stdout.flush()
            time.sleep(1.0)

class IndexFiles(object):
    """Usage: python IndexFiles <doc_directory>"""

    def __init__(self, root, storeDir, analyzer):

        if not os.path.exists(storeDir):
            os.mkdir(storeDir)
        store = lucene.SimpleFSDirectory(lucene.File(storeDir))
        writer = lucene.IndexWriter(store, analyzer, True,
                                    lucene.IndexWriter.MaxFieldLength.LIMITED)
        writer.setMaxFieldLength(1048576)
        self.indexDocs(root, writer)
        ticker = Ticker()
        print 'optimizing index',
        threading.Thread(target=ticker.run).start()
        writer.optimize()
        writer.close()
        ticker.tick = False
        print 'done'

    def indexDocs(self, root, writer):
        for root, dirnames, filenames in os.walk(root):
            for filename in filenames:
                if filename.endswith('.txt'):
                    continue
                print "adding", filename
                try:
                    path = os.path.join(root, filename)
                    file = open(path)
                    buf = file.read()
                    contents=buf
                    result = chardet.detect(buf)['encoding']
                    if result=='GB2312':
                        contents = buf.decode('gbk').encode('utf8')
                    file.close()
                    soup=BeautifulSoup(contents)
                    url=mydict[filename]
                    title=str(soup.head.title.string).decode('utf8')
                    new_contents=filter_tags(contents)
                    new_contents=str(new_contents).strip().decode('utf8')
                    pos=new_contents.find('>')
                    new_contents=new_contents[pos+1:]
                    temp=new_contents.split()
                    newtext=' '.join(temp)
                    contents=''.join(soup.findAll(text=True))
                    doc = lucene.Document()
                    doc.add(lucene.Field("text", newtext,
                                         lucene.Field.Store.YES,
                                         lucene.Field.Index.NOT_ANALYZED))
                    doc.add(lucene.Field("url", url,
                                         lucene.Field.Store.YES,
                                         lucene.Field.Index.NOT_ANALYZED))
                    doc.add(lucene.Field("title", title,
                                         lucene.Field.Store.YES,
                                         lucene.Field.Index.NOT_ANALYZED))
                    if len(contents) > 0:
                        dll=cdll.LoadLibrary("F:\\ICTCLAS50_Windows_32_C\ICTCLAS
                        dll.ICTCLAS_Init(c_char_p("F:\\ICTCLAS50_Windows_32_C"))
                        strlen = len(c_char_p(contents).value)
                        t =c_buffer(strlen*6)
                        bSuccess = dll.ICTCLAS_ParagraphProcess(c_char_p(content
                        contents=t.value.decode('gbk').encode('utf8')
                        ##list=t.value.split()
                        ##print ' '.join(list)
                        dll.ICTCLAS_Exit()
                        doc.add(lucene.Field("contents", contents,
                                             lucene.Field.Store.NO,
                                             lucene.Field.Index.ANALYZED))
                    else:
                        print "warning: no content in %s" % filename
                    writer.addDocument(doc)
                except Exception, e:
                    print "Failed in indexDocs:", e
```

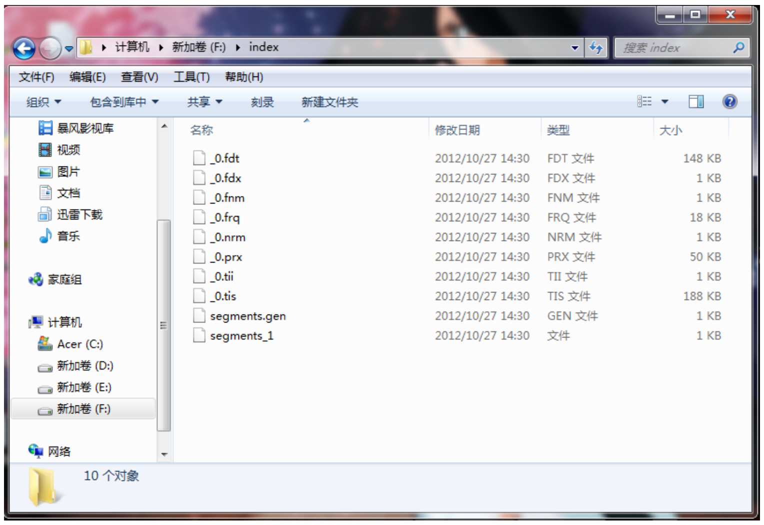
```

if __name__ == '__main__':
##     if len(sys.argv) < 2:
##         print IndexFiles.__doc__
##         sys.exit(1)
    lucene.initVM()
    print 'lucene', lucene.VERSION
    start = datetime.now()
    dic= open('F:\\html\index.txt')
    d = dic.readlines()
    dic.close()
    mydict = {}
    for word in d:
        value=''
        try:
            key = word.split(';')[0]
            value = word.split(';')[1]
            mydict[key] = value
        except:
            pass
    try:
##         IndexFiles(sys.argv[1], "index", lucene.SimpleAnalyzer(lucene.Version.
        IndexFiles('F:\\html', "F:\\index", lucene.SimpleAnalyzer(lucene.Version
        end = datetime.now()
        print end - start
    except Exception, e:
        print "Failed:␣", e
```

So after this part, most of the useless information in the web page is eliminated, leaving the messages we need in the 'text' variable. There's any output after the program being processed. So only the screenshot of the index files will be given.

2.2 The Main Searching Part of the Program

In this part, I will show you the main process of searching according to a user's query. And I try my best to make it as flexible as it could. E.g., if you search for a certain Chinese word, the program will search it by treating the word as a whole expression and then treating it as maybe two words by split them. So in this way, I try to return as much information as I can. And for other potential errors, I deal with the 'try...except' sentence.

```
import web
from web import form
import urllib2
```



名称	修改日期	类型	大小
_0.fdt	2012/10/27 14:30	FDT 文件	148 KB
_0.fdx	2012/10/27 14:30	FDX 文件	1 KB
_0.fnm	2012/10/27 14:30	FNM 文件	1 KB
_0.frq	2012/10/27 14:30	FRQ 文件	18 KB
_0.nrm	2012/10/27 14:30	NRM 文件	1 KB
_0.prx	2012/10/27 14:30	PRX 文件	50 KB
_0.tii	2012/10/27 14:30	TII 文件	1 KB
_0.tis	2012/10/27 14:30	TIS 文件	188 KB
segments.gen	2012/10/27 14:30	GEN 文件	1 KB
segments_1	2012/10/27 14:30	文件	1 KB

Figure 1: the files of the index

```
import os
from lucene import \
    QueryParser, IndexSearcher, SimpleAnalyzer, SimpleFSDirectory, File, \
    VERSION, initVM, Version
from ctypes import *

urls = (
    '/', 'index',
    '/s', 's'
)


render = web.template.render('templates') # your templates

login = form.Form(
    form.Textbox('keyword'),
    form.Button('Search'),
)

def func(command, searcher, analyzer):
    if len(command) > 0:
        dll=cdll.LoadLibrary("F:\\ICTCLAS50_Windows_32_C\ICTCLAS50.dll")
        dll.ICTCLAS_Init(c_char_p("F:\\ICTCLAS50_Windows_32_C"))
        strlen = len(c_char_p(command).value)
        t =c_buffer(strlen*6)
        bSuccess = dll.ICTCLAS_ParagraphProcess(c_char_p(command),c_int(strlen),
        command=t.value.decode('gbk').encode('utf8')
        ##list=t.value.split()
        ##print ' '.join(list)
        dll.ICTCLAS_Exit()
        command=command.decode('utf8')
    query = QueryParser(Version.LUCENE_CURRENT, "contents",analyzer).parse(comma
    scoreDocs = searcher.search(query, 50).scoreDocs
    total=len(scoreDocs)
    qtitle=[]
    qurl=[]
    qnew_text1=[]
    qnew_text2=[]
    q_query=[]
    new_query=str(query).replace ('contents:','').decode('utf8')
    if total==0:
        return command,qurl,qtitle,qnew_text1,qnew_text2,q_query,total
    for scoreDoc in scoreDocs:
        doc = searcher.doc(scoreDoc.doc)
        text=doc.get("text")
        new_text=str(text).decode('utf8')
        temp_query=new_query.replace('␣','')
        num=new_text.find(temp_query)
        query_len=len(temp_query)
        splited_query=new_query.split('␣')
        splitlen=len(splited_query)
        if (num!=-1):
            try:
                new_text1=str(new_text[num-30:num]).strip().decode('utf8','ignor
                new_text2=str(new_text[num+query_len:num+30+query_len]).strip().
            except:
                try:
                    new_text1=str(new_text[num-30:num]).strip().decode('utf8','i
                    new_text2=""
                except:
                    try:
                        new_text1=""
                        new_text2=str(new_text[num+query_len:num+30+query_len]).
                    except:
                        new_text1=""
                        new_text2=""
            q_query.append(temp_query)
        else:
            for i in range(splitlen):
                parted_query=splited_query[i].decode('utf8')
                num=new_text.find(parted_query)
                if num==-1:
                    continue
                query_len=len(parted_query)
                try:
                    new_text1=str(new_text[num-30:num]).strip().decode('utf8','i
                    new_text2=str(new_text[num+query_len:num+30+query_len]).stri
                except:
                    try:
                        new_text1=str(new_text[num-30:num]).strip().decode('utf8
                        new_text2=""
                    except:
                        try:
                            new_text1=""
                            new_text2=str(new_text[num+query_len:num+30+query_le
                        except:
                            new_text1=""
                            new_text2=""
                q_query.append(parted_query)
                break
            if num==-1:
                total=total-1
                continue
        title=doc.get("title")
        url=doc.get("url")
        qurl.append(url)
        qtitle.append(title)
        qnew_text1.append(new_text1)
        qnew_text2.append(new_text2)
    return command,qurl,qtitle,qnew_text1,qnew_text2,q_query,total

class index:
    def GET(self):
        f = login()
        return render.formtest(f)

class s:
    def GET(self):
        vm_env = initVM()
        form1 = login()
        user_data = web.input()
        vm_env.attachCurrentThread()
        STORE_DIR = "F:\\index"
        directory = SimpleFSDirectory(File(STORE_DIR))
        searcher = IndexSearcher(directory, True)
        analyzer = SimpleAnalyzer(Version.LUCENE_CURRENT)
        a,b,c,d,e,f,g= func(user_data.keyword,searcher,analyzer)
        searcher.close()
```

```
            return render.result(form1,a,b,c,d,e,f,g)

if __name__ == "__main__":
    while True:
        app = web.application(urls, globals())
        app.run()
```

Of course, the web pages to be presented afterwards are also included in the above part, but how to set up the web page will be presented in the next part. And in order to make the pages a little more beautiful, I revised some of the variables when necessary. And the other parts are nearly the same as the 'search' program in lab3.
The pictures will be given after I finished introducing the 'html' part.

2.3 The Part of Establishing the Web Page

In this part, I wrote two html files for the searching web page, and as required, it can present the url, some contents near the keywords, and title in the form of a hyperlink of the result. Here are the codes of the two pages:
The first is the formtest.html:

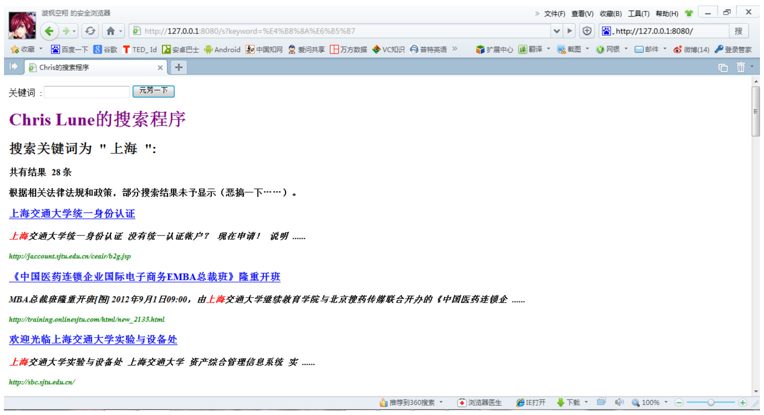
```
$def with(form)
<head>
<title>Chris        </title>
<form name="input" form action="/s" method="GET">          :
<input type="keyword" name="keyword" />
<input type="submit" value="         "/>
</form>
```

There are some parts written in Chinese which can't be showed in the pdf file, you can look for it in my code files. And the following is the result.html:

```
$def with (form,name,qurl,qtitle,qnew_text1,qnew_text2,q_query,total)

<title>Chris        </title>
<form name="input" form action="/s" method="GET">          :
<input type="keyword" name="keyword" />
<input type="submit" value="         "/>
</form>
<head><h1><strong></strong><font color="purple">Chris  L u n e        </h1><strong
                <h2><font color="black">                "␣$name␣":</h2>
                <h4><font color="black">           $total   </h4>
                        <h4><b><font color="black">
$if qtitle and qurl and len(qnew_text1) and len(qnew_text2):
        $for i in range(total):
        <h3><a href="␣$qurl[i]␣"><font color="blue">$qtitle[i]</a></h3>
        <div><i><font color="black">$qnew_text1[i].strip()<font color="red">$q_q
        <h5><div><font color="green"><i>$qurl[i]</i></div></h5>
```

```
$else:
    <em>There's no result!</em>!
```

So after these preparations, the result of the searching program is shown in the following screenshots:



Chris Lune的搜索程序

搜索关键词为 " 上海 ":

共有结果 28 条

根据相关法律法规和政策，部分搜索结果未予显示（恶搞一下……）。

上海交通大学统一身份认证

上海交通大学统一身份认证 没有统一认证帐户？ 现在申请！ 说明 ......

http://jaccount.sjtu.edu.cn/ceair/b2g.jsp

《中国医药连锁企业国际电子商务EMBA总裁班》隆重开班

MBA总裁班隆重开班[图] 2012年9月1日09:00，由上海交通大学继续教育学院与北京搜药传媒联合开办的《中国医药连锁企 ......

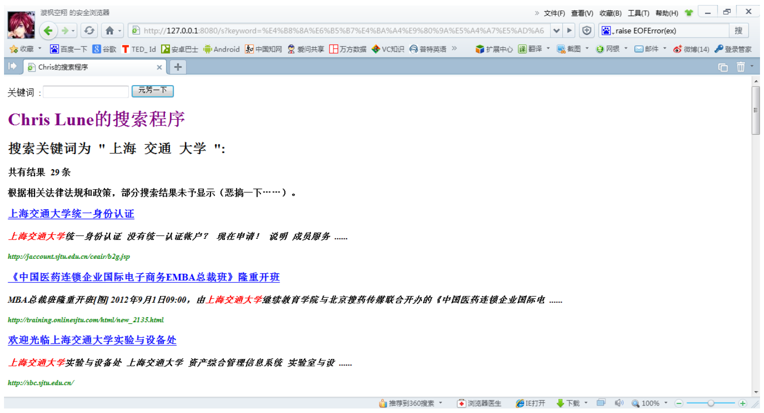
http://training.onlinesjtu.com/html/new_2135.html

欢迎光临上海交通大学实验与设备处

上海交通大学实验与设备处 上海交通大学 资产综合管理信息系统 实 ......

http://sbc.sjtu.edu.cn/

Figure 2: the outcome 1


旋枫空翔 的安全浏览器
http://127.0.0.1:8080/s?keyword=%E4%B8%8A%E6%B5%B7%E4%BA%A4%E9%80%9A%E5%A4%A7%E5%AD%A6
Chris的搜索程序
关键词：
元芳一下

Chris Lune的搜索程序

搜索关键词为 " 上海 交通 大学 ":

共有结果 29 条

根据相关法律法规和政策，部分搜索结果未予显示（恶搞一下……）。

上海交通大学统一身份认证

上海交通大学统一身份认证 没有统一认证帐户？ 现在申请！ 说明 成员服务 ......

http://jaccount.sjtu.edu.cn/ceair/b2g.jsp

《中国医药连锁企业国际电子商务EMBA总裁班》隆重开班

MBA总裁班隆重开班[图] 2012年9月1日09:00，由上海交通大学继续教育学院与北京搜药传媒联合开办的《中国医药连锁企业国际电 ......

http://training.onlinesjtu.com/html/new_2135.html

欢迎光临上海交通大学实验与设备处

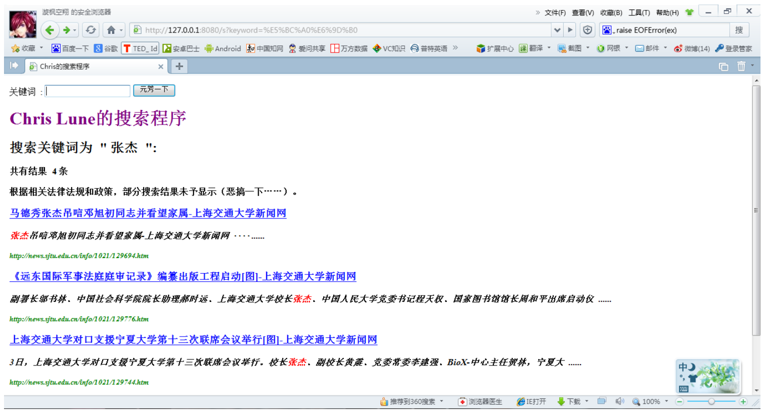
上海交通大学实验与设备处 上海交通大学 资产综合管理信息系统 实验室与设 ......

http://sbc.sjtu.edu.cn/

Figure 3: the outcome 2

3 The Problems I Met in the Experiment and My Solution

As for the first part, the main problem I meet is how to eliminate the useless information near the key words, and after looking for some solutions on the



Chris Lune的搜索程序

搜索关键词为 " 张杰 ":

共有结果 4条

根据相关法律法规和政策，部分搜索结果未予显示（恶搞一下……）。

马德秀张杰吊唁邓旭初同志并看望家属-上海交通大学新闻网

张杰吊唁邓旭初同志并看望家属-上海交通大学新闻网 ····......

http://news.sjtu.edu.cn/info/1021/129694.htm

《远东国际军事法庭庭审记录》编纂出版工程启动[图]-上海交通大学新闻网

副署长邬书林、中国社会科学院院长助理郝时远、上海交通大学校长张杰、中国人民大学党委书记程天权、国家图书馆馆长周和平出席启动仪 ......

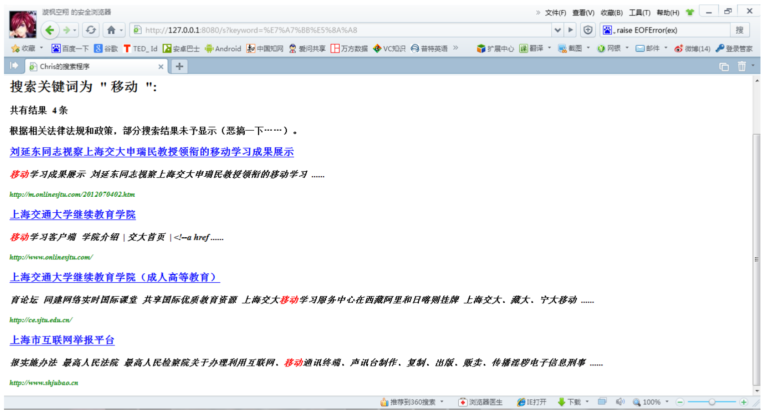
http://news.sjtu.edu.cn/info/1021/129776.htm

上海交通大学对口支援宁夏大学第十三次联席会议举行[图]-上海交通大学新闻网

3日，上海交通大学对口支援宁夏大学第十三次联席会议举行。校长张杰、副校长黄震、党委常委李建强、BioX-中心主任贺林，宁夏大 ......

http://news.sjtu.edu.cn/info/1021/129744.htm

Figure 4: the outcome 3



搜索关键词为 " 移动 ":

共有结果 4条

根据相关法律法规和政策，部分搜索结果未予显示（恶搞一下……）。

刘延东同志视察上海交大申瑞民教授领衔的移动学习成果展示

移动学习成果展示 刘延东同志视察上海交大申瑞民教授领衔的移动学习 ......

http://m.onlinesjtu.com/2012070402.htm

上海交通大学继续教育学院

移动学习客户端 学院介绍 | 交大首页 | <!--a href ......

http://www.onlinesjtu.com/

上海交通大学继续教育学院（成人高等教育）

育论坛 同建网络实时国际课堂 共享国际优质教育资源 上海交大移动学习服务中心在西藏阿里和日喀则挂牌 上海交大、藏大、宁大移动 ......

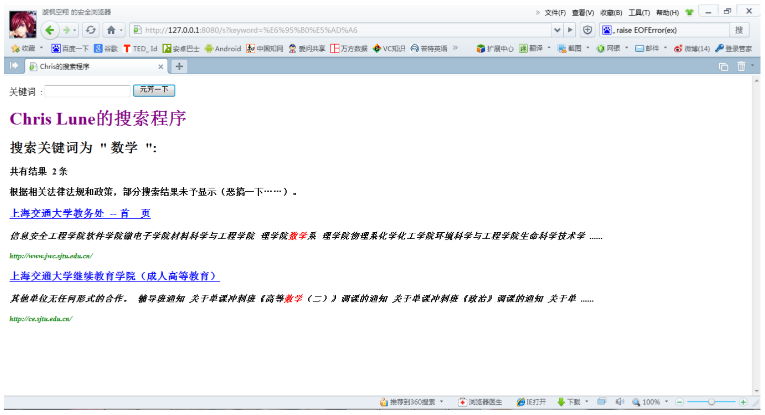
http://ce.sjtu.edu.cn/

上海市互联网举报平台

报实施办法 最高人民法院 最高人民检察院关于办理利用互联网、移动通讯终端、声讯台制作、复制、出版、贩卖、传播淫秽电子信息刑事 ......

http://www.shjubao.cn

Figure 5: the outcome 4



Chris Lune的搜索程序

搜索关键词为 " 数学 ":

共有结果 2条

根据相关法律法规和政策，部分搜索结果未予显示（恶搞一下……）。

上海交通大学教务处 -- 首 页

信息安全工程学院软件学院微电子学院材料科学与工程学院 理学院数学系 理学院物理系化学化工学院环境科学与工程学院生命科学技术学 ......

http://www.jwc.sjtu.edu.cn/

上海交通大学继续教育学院（成人高等教育）

其他单位无任何形式的合作。 辅导班通知 关于单课冲刺班《高等数学（二）》调课的通知 关于单课冲刺班《政治》调课的通知 关于单 ......

http://ce.sjtu.edu.cn/

Figure 6: the outcome 5

Internet, I find the following method to solve it:

```
import re
def filter_tags(htmlstr):
    re_cdata=re.compile('//<!\[CDATA\[[^>]*//\]\]>',re.I)
    re_script=re.compile('<\s*script[^>]*>[^<]*<\s*/\s*script\s*>',re.I)
    re_style=re.compile('<\s*style[^>]*>[^<]*<\s*/\s*style\s*>',re.I)
    re_br=re.compile('<br\s*?/?>')
    re_h=re.compile('</?\w+[^>]*>')
    re_comment=re.compile('<!--[^>]*-->')
    s=re_cdata.sub('',htmlstr)
    s=re_script.sub('',s)
    s=re_style.sub('',s)
    s=re_br.sub('\n',s)
    s=re_h.sub('',s)
    s=re_comment.sub('',s)
    blank_line=re.compile('\n+')
    s=blank_line.sub('\n',s)
    s=replaceCharEntity(s)
    return s


def replaceCharEntity(htmlstr):
    CHAR_ENTITIES={'nbsp':' ','160':' ','lt':'<','60':'<','gt':'>','62':'>','amp
    re_charEntity=re.compile(r'&#?(?P<name>\w+);')
    sz=re_charEntity.search(htmlstr)
    while sz:
        entity=sz.group()
        key=sz.group('name')
        try:
            htmlstr=re_charEntity.sub(CHAR_ENTITIES[key],htmlstr,1)
            sz=re_charEntity.search(htmlstr)
        except KeyError:
            htmlstr=re_charEntity.sub('',htmlstr,1)
            sz=re_charEntity.search(htmlstr)
    return htmlstr
```

Then in this way, we can change some marks into the punctuations they present, and also delete some other messages such as the tag of the page, etc.
And in the second searching part, there are to many errors to deal with, and I write all solutions to the conditions I have met in the 'try...except' clause. Also, in the last part, I firstly have some trouble with giving out all results using a circling clause, then I learn how to solve it from the reference web site. And below are the codes:

```
$for i in range(total):
```

```
            <h3><a href=" $qurl[i] "><font color="blue">$qtitle[i]</a></h3>
            <div><i><font color="black">$qnew_text1[i].strip()<font color="red">$q_q
            <h5><div><font color="green"><i>$qurl[i]</i></div></h5>
```

And after solving these problems ,I basically finish the searching program.

4 Some of My Thoughts

In lab4, it's easy to find that we are able to write some part of our own searching engine by ourselves. Although at first I'm a little unfamiliar with how to write some clause in the html file, I manage to solve it by learning from the Internet. And now I feel some kind of satisfied by searching for some queries using my own searching files. And now that we have finished the searching program of texts in the page, I think it's also not hard to search and return some images in the same way.
But there may still exist some mistakes which I haven't found in my program yet, so I'll still testing it in the following days, trying to make it as perfect as I want.