# -\*- coding: utf-8 -\*-

"""

将输入的文献相关参数整理规范化，通过NCBI提供的接口找到输入信息的相

关文献ID，并获取。利用文献ID获取相关文献的摘要，并将其中的有关心信

息通过正则匹配得到。根据疾病名称在CDK中找到疾病的同义词信息

@author: ycy

"""

import urllib.request

import urllib

import re

import time

import xlsxwriter as xlsx

import xlrd

from bs4 import BeautifulSoup

class info\_sort:

def sort(self,info):#对输入的相关参数进行整理，避免空格，若出现空格则以+号取代

para = info.split()

info\_sorted = ''

for i in range(len(para)):

if i == len(para) - 1:

info\_sorted = info\_sorted + para[i]

break

info\_sorted = info\_sorted + para[i] + '+'

return info\_sorted

class Literature:

#通过相关参数在pubmed中获取相应的文献信息

def \_\_init\_\_(self,url,search,summary,info\_sorted,tool):

#info\_sorted是经过整理得到的相关参数

#tool网页的正则匹配工具

#listing建立excel表格，将获取文献信息导入

self.url = url

self.search = search

self.summary = summary

self.info\_sorted = info\_sorted

self.tool = tool

self.Esearch = url + search +str(info\_sorted)

def get\_uid(self):

#访问pubmed的api，匹配其中的UID,返回匹配的UID个数

try:

request = urllib.request.Request(self.Esearch)

reponse = urllib.request.urlopen(request)

content = reponse.read().decode('utf-8')

except urllib.error.URLError as Error:

print(Error)

patterns = re.compile('<Id>(.\*?)</Id>')

self.Id\_s = Id\_s = re.findall(patterns,content)

print('文献ID已成功匹配\n')

length = len(Id\_s)

return length

def info\_ready(self,i):

Esummary\_b = self.url + self.summary

Esummary = Esummary\_b + str(self.Id\_s[int(i)])

try:

request = urllib.request.Request(Esummary)

reponse = urllib.request.urlopen(request)

content = reponse.read().decode('utf-8')

except urllib.error.URLError as Error:

print(Error)

return content

def get\_info(self,element):

value = self.tool.replace(element)

return value

class Tool:

#利用正则表达式，匹配文献网页的相关元素

def \_\_init\_\_(self):

self.Id\_re = re.compile('<Id>(.\*?)</Id>')

self.Pubdate\_re = re.compile('<Item Name="PubDate" Type="Date">(.\*?)</Item>')

self.Source\_re = re.compile('<Item Name="Source" Type="String">(.\*?)</Item>')

self.Author\_re = re.compile('<Item Name="Author" Type="String">(.\*?)</Item>')

self.Title\_re = re.compile('<Item Name="Title" Type="String">(.\*?)</Item>')

self.PubType\_re = re.compile('<Item Name="PubType" Type="String">(.\*?)</Item>')

self.FullJournalName\_re = re.compile('<Item Name="FullJournalName" Type="String">(.\*?)</Item>')

def replace(self,content):

self.Id = Id = re.findall(self.Id\_re,content)

self.Pubdate = Pubdate = re.findall(self.Pubdate\_re,content)

self.Source = Source = re.findall(self.Source\_re,content)

self.Author = Author = re.findall(self.Author\_re,content)

self.Title = Title = re.findall(self.Title\_re,content)

self.PubType = PubType = re.findall(self.PubType\_re,content)

self.FullJournalName = FullJournalName = re.findall(self.FullJournalName\_re,content)

self.list\_value = [self.Id,self.Pubdate,self.Source,self.Source,self.Author,\

self.Title,self.PubType,self.FullJournalName]

return self.list\_value

class List:

#创建Excel表格，并导入相关的文献信息

def create(self,path,Literature,num):

self.path = path

self.wb = xlsx.Workbook(str(self.path))

self.ws = self.wb.add\_worksheet('Literature\_info')

list\_create = ['Id','Pubdate','Source','Source','Author',\

'Title','PubType','FullJournalName']

for i in range(len(list\_create)):

self.ws.write(0,i,list\_create[i])

for i in range(num):

element = Litera.info\_ready(i)

values = Litera.get\_info(element)

for k in range(len(values)):#此处信息录入表格，anthor含有多人，须更改

self.ws.write(i+1,k,str(values[k][0]))

print('已成功录入%d条文献信息'%(i+1))

time.sleep(0.5)

self.wb.close()

class NTB:

def synonym(self):

disease = input('输入疾病名称：\n')

url1 = 'http://ctdbase.org/basicQuery.go?bqCat=disease&bq='#CTD疾病库搜索网址

url2 = 'http://ctdbase.org/detail.go?type=disease&acc='#疾病同义词信息网址

url = url1 + str(disease)

try:

request = urllib.request.Request(url)

reponse = urllib.request.urlopen(request)

content = reponse.read().decode('utf-8')

except urllib.error.URLError as Error:

print(Error)

soup = BeautifulSoup(content,'lxml')

t = soup.find\_all('span',class\_="match")

pattern1 = re.compile('<a href=.\*?>(.\*?)</a>')#匹配多行信息中的名字的模式

pattern2 = re.compile('<a href="(.\*?)">.\*?</a>')#匹配相应的网址的模式

pattern3 = re.compile('/detail.go.\*?acc=(.\*)')#匹配网址信息的acc部分的模式

name = re.findall(pattern1,str(t))

url = re.findall(pattern2,str(t))

for i in range(len(name)):

if name[i] == disease:

acc = re.findall(pattern3,str(url[i]))

break;

url\_s = url2 + str(acc[0])

try:

request = urllib.request.Request(url\_s)

reponse = urllib.request.urlopen(request)

content = reponse.read().decode('utf-8')

except urllib.error.URLError as Error:

print(Error)

pattern4 = re.compile('<a href=".\*?" title="Keyword query" rel="nofollow">(.\*?)</a>')

print(re.findall(pattern4,content))

def MainLoop():

info =input('请输入相关的文献信息;\n')

sorting = info\_sort()

url = 'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/'

search = 'esearch.fcgi?db=pubmed&term='

summary = 'esummary.fcgi?db=pubmed&id='

path = input('input the file path:\n')

listing = List()

tool = Tool()

Litera = Literature(url,search,summary,sorting.sort(info),tool)

Id\_num = Litera.get\_uid()

listing.create(path,Litera,Id\_num)

ntb = NTB()

ntb.synonyms()

mainloop = MainLoop()