**开源软件基础课程大作业**

**python\_holidays开源仓库分析**

**Embedded Music Player Based on STM32F407**

学 院（系）： 软件学院

专 业： 软件工程

学 生 姓 名： 邵一辰 侯硕伟

马洋 周鑫汐 杨晨

学 号：20212241028 20212241476

20212241346 20212241079

20212241242

指 导 教 师： 任志磊

评 阅 教 师：

完 成 日 期： 2024/1/31

大连理工大学

Dalian University of Technology

目录

[一、项目背景与意义 - 1 -](#_Toc157619955)

[二、项目设计与实现 - 1 -](#_Toc157619956)

[2.1 代码分析 - 1 -](#_Toc157619957)

[2.2 issue分析 - 5 -](#_Toc157619958)

[三、项目测试 - 9 -](#_Toc157619959)

[3.1 判断某一天是否为假期 - 9 -](#_Toc157619960)

[3.2 获得某个国家的假期 - 10 -](#_Toc157619961)

[3.3添加自定义假期 - 10 -](#_Toc157619962)

[四、项目分析 - 11 -](#_Toc157619963)

[五、项目总结、不足与改进 - 11 -](#_Toc157619964)

[六、成员贡献与感想 - 12 -](#_Toc157619965)

## 一、项目背景与意义

Python-Holidays是github上一个致力于解决假期问题的开源项目，旨在为无法承担全球日历假期数据库高额费用的用户需求提供可行的解决方案，截止目前已有1.3k的star数。该仓库包含多达140个国家的日历，与此同时还支持各类特殊假期，如校园假期（寒暑假、春假秋假）、宗教假期、各个州的假期等等。此外，该项目支持例如中国的妇女节、儿童节、青年节这类半天假期。

不同的行业对于日历和假期的准确计算与表示都有需求，金融行业需要根据假期决定各类金融产品的到期时间，计算利息、调节产品参数浮动等；交通运输行业需要根据假期调整线路运量，确定合适的价格；旅游行业需要根据假期预见性部署，及时实施限流等管制措施以防出现踩踏等危险事故。Python-Holidays仓库可以为各行各业提供关于日历与假期的帮助。

针对Python-Holidays仓库的代码部分进行分析，可以对该仓库对日期的判断有更为清晰的认知。对该仓库的issue部分进行分析，则可以发现该仓库存在的问题，例如部分政治目的强的节日难以用逻辑判断描述、不规律的调休引起的判断错误、对宗教节日收录不完全等，可以据此对该仓库提出相应的改进意见和建议。

## 二、项目设计与实现

### 2.1 代码分析

#### 2.1.1 软件包的结构

holidays包，包含constants, helpers, holiday\_base, observed\_holiday\_base, registry, utils模块，以及calendars, countries, financial, groups子软件包。

holiday\_base是所有国家假期类的基类，是此软件包的核心，主要负责节假日填充逻辑代码。其继承字典类型，支持下标访问，支持多种格式的日期直接查询，内置了日期类型转换函数\_\_keytransform\_\_。\_populate函数将填充指定一年的假期。

observed\_holiday\_base继承holiday\_base，增加了调休逻辑。文件内还定义了各个星期的调休规则。

registry是本模块的加载器，用于实现模块的动态加载。

utils包含了一些静态函数，产生特定参数的假期字典，也可以获取本模块对各国家语言的支持情况。

helper包含一些辅助函数，进行参数类型的转换。

constants定义了有关日期、星期的常量，以及各种节日类型名称。

calendars软件包包括了世界各种日历，日历内包含各个节日对应年份的日期，以及get方法。还包括gregorian模块定义了公历日历的常用函数，custom模块包含StaticHoliday类方便加载自定义的日历，一般是特例节假日。

groups中的模块将一个或多个日历结合成群组，使群组中的日历可以被加载进字典。还包括了international，可以加载国际假期。

countries是本软件包的主要组成部分，包含各个国家的假期字典。每个国家假期对象都继承自HolidayBase，此外还可选继承自相应的日历群组（来自groups包），可选继承自StaticHoliday。例如，China由ObservedHolidayBase, ChineseCalendarHolidays, InternationalHolidays, StaticHolidays继承而来，其中StaticHolidays加载了自定义的ChinaStaticHolidays类，填补了特殊节假日（抗日70周年假日等）。

financial是金融假期包，其结构与countries相同。

#### 2.1.2 运行流程

本软件包宣称“as fast and flexible as possible”，多处运用到了即时构建（built on the fly）的思想。

为了直观地看到代码运行的流程，使用sys库中的settrace, gettrace进行语句追踪。将import和函数调用的语句分开读取，分析执行过程。

|  |
| --- |
| **from** sys **import** settrace, gettrace **import** linecache  f = open(**'trace\_record\_import.txt'**, **'w+'**, encoding=**'utf-8'**) proj\_prefix = **'D:\\Docs\\Course\_sync\\OpenSource\\Final\\demo\\python-holidays\\holidays\\'** last\_filename = **None   def** tracer(frame, event, arg):  old = gettrace()  settrace(**None**)  **global** last\_filename  **if** event == **'call'**:  filename = frame.f\_code.co\_filename  **if** filename.startswith(proj\_prefix):  lineno = frame.f\_lineno  func = frame.f\_code.co\_name  f.write(**f'call {**filename**}:{**lineno**}: {**func**}\n'**)   **if** event == **'line'**:  filename = frame.f\_code.co\_filename  *# if filename != last\_filename and filename.startswith(proj\_prefix):* **if** filename.startswith(proj\_prefix):  *# if True:* line\_no = frame.f\_lineno  line = linecache.getline(filename, line\_no).strip()  filename = frame.f\_code.co\_filename  f.write(**f'{**filename**}:{**line\_no**}: {**line**}\n'**)  last\_filename = filename  settrace(old)  **return** tracer   settrace(tracer) **import** holidays  f = open(**'trace\_record\_call.txt'**, **'w+'**, encoding=**'utf-8'**) holidays.CN()[**'2023-04-05'**] *# 清明节* |

得到两个追踪文件，从中可以看到执行过程。从加载软件包开始，import holidays并不将所有国家的假期全部加载进来，而是只加载必要的常量和工具类模块。使用registery中的EntityLoader进行动态加载模块。EntityLoader在初始化时只将模块名加载进变量空间，而不产生实体类，被调用时才会加载。以下代码来自EntityLoader的get\_entity和\_\_call\_\_。

|  |
| --- |
| **def** get\_entity(self) -> Optional[HolidayBase]:  *"""Return lazy-loaded entity."""* **if** self.entity **is None**:  self.entity = getattr(importlib.import\_module(self.module\_name), self.entity\_name)   **return** self.entity  **def** \_\_call\_\_(self, \*args, \*\*kwargs) -> HolidayBase:  *"""Create a new instance of a lazy-loaded entity."""* cls = self.get\_entity()  **return** cls(\*args, \*\*kwargs) *# type: ignore[misc, operator]* |

这样做使得不必要的国家假期不会加载到内存中，大大节省加载软件包的时间。

每个国家假期的填充同样是动态的。初始时holiday\_base只是一个空字典。每当查询到来，传入的关键字被\_\_ keytransform\_\_转换成date类型，接着，如果这个日期所在的年份没有被填充，则执行\_populate填充。一次只填充一年的假期。以下代码显示假期的动态填充。

|  |
| --- |
| *# Automatically expand for `expand=True` cases.* **if** self.expand **and** dt.year **not in** self.years:  self.years.add(dt.year)  self.\_populate(dt.year) |

这样进一步减少了调用时的执行时间。

#### 2.1.3 该软件包的难点

其一是维护全球一百多个国家的节假日需要大量当地程序员，并且要及时跟进政策更新。对于免费开源仓库来说，很难保证各个国家的开发者长期坚持贡献。

另外，本地化工作比较艰难。本仓库对世界多语言的支持仍然有很大缺陷，每个节日翻译成多国语言需要语言专业方面的工作，完美维护需要大量人力。节日翻译至今仍有很多新问题，例如observed holiday被直译成了观察日，实际上这里的observe是庆祝的意思，更本土化的说法叫调休。

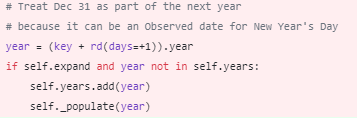
大多数节假日虽然日期固定、有确定的判断逻辑，但是每年仍有一些国家存在特殊庆祝的假期，以及政策调休的问题。这导致节假日判断问题不可能一劳永逸地解决。本开源仓库针对这个问题已经带来了很方便的方案，但仍需要永久维护。

#### 2.1.4 部分Bug分析

#3 <https://github.com/vacanza/python-holidays/issues/3> RuntimeError when creating custom class with Dec 31st holiday

Bug现象：查询某一年的12月31日导致RuntimeError（无限递归调用爆栈）。

相关代码：来自修改前的\_\_keytransform\_\_函数



分析：此代码块在模块基类HolidayBase中的\_populate\_函数中。这段代码的作者认为12月31号可以作为第二年新年的调休日，所以直接放到下一年进行拓展。也就是说，对第year的拓展包含了第year-1年12月31日的假期。这里对year的转换看似巧妙，实则隐含漏洞。由于HolidayBase类的\_\_getitem\_\_和\_\_setitem\_\_的参数都需要调用\_\_keytransform\_\_处理，在\_populate函数中对节假日的设定隐含地调用了\_\_keytransform。那么这里的year就会被不断+1到下一年，执行新的填充，没有终止条件。

解决方案：将12月31仍作为当年节假日，对调休的判断做相应调整，不能在\_\_keytransform中做太多无关动作。

#4 <https://github.com/vacanza/python-holidays/issues/4> Error when combining holiday objects for different provinces

Bug现象：对同一个国家的不同地区的假期进行相加操作会出错（基类重复）。

相关代码：

|  |
| --- |
| **def** createHolidaySum(class1, class2):  **class** HolidaySum(class1, class2):  **def** \_populate(self, year):  class2.\_populate(self, year)  class1.\_populate(self, year)  **return** HolidaySum |

分析：需要进行两个国家或地区的假期融合时，原作者的代码通过继承原来的两个假期类，自然融合成新的假期类。当class1和class2不同时，确实没什么问题。但是假期类是由国家划分的，而同一国家的不同地区假期可能不同，但其均为同一个类，不能被重复继承。

解决方案：放弃继承融合，使用有状态的HolidaySum。HolidaySum存储一个holidays列表，像做加法一样加入两边的所有假期。原来的融合只关注类，而更改后加法作用与具体对象。

|  |
| --- |
| **def** createHolidaySum(h1, h2):  **class** HolidaySum(HolidayBase):  **def** \_\_init\_\_(self, \*\*kwargs):  self.holidays = []  **if** getattr(h1, **'holidays'**, **False**):  **for** h **in** h1.holidays:  self.holidays.append(h)  **else**:  self.holidays.append(h1)  **if** getattr(h2, **'holidays'**, **False**):  **for** h **in** h2.holidays:  self.holidays.append(h)  **else**:  self.holidays.append(h2)  HolidayBase.\_\_init\_\_(self, \*\*kwargs)   **def** \_populate(self, year):  **for** h **in** self.holidays[::-1]:  h.\_populate(year)  self.update(h)  **return** HolidaySum |

#1644 <https://github.com/vacanza/python-holidays/issues/1644> Chinese Spring Festival holiday in 2020 error

Bug现象：2020年中国春节假期因为疫情延长3天，代码中缺少实现。

分析：此类假期属于延期情况，软件包的StaticHolidays已经为此做了铺垫，在自定义类ChinaStaticHolidays中增加相应假期即可，不需要其他结构方面的改动。

### 2.2 issue分析

#### 2.2.1 分析概述

我们从仓库外围切入，具体工作分为两个部分，数据爬取和热词分析。

数据爬取部分，我们采用requests+beautifulsoup方式，爬取了github中Python-Holidays仓库下所有的issue与comment，爬取结果一式两份存放于json和xlsx文件中。

热词分析部分，我们基于爬取到的数据，提取出其中issue标题与comment内容等文本信息，引入nltk库对文本进行分词、标注与筛选，最后统计词频，统计结果存放于result.txt文件中。

#### 2.2.2 分析代码

1） 数据爬取(fetch.py)

|  |
| --- |
| proxies = {      'http': 'http://127.0.0.1:10809',      'https': 'http://127.0.0.1:10809'  }  headers = {"User-Agent": "Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/72.0.3626.119 Safari/537.36"}  def get\_response(url):      turns = []      res = requests.get(url=url,proxies=proxies,headers=headers)      soup = BeautifulSoup(res.text,'html.parser')      turns\_ = soup.find\_all('tr', class\_="d-block")      for turn in turns\_:          dialog = turn.text.replace('\xa0'\*4, '\n\n').strip()          if dialog == "The text was updated successfully, but these errors were encountered:":              continue          if not dialog:              continue          turns.append(dialog)      return turns  def get\_links(url\_base, pages, tag="open"):      if not url\_base.endswith("/"):          url\_base = url\_base + '/'      datas = []      for page in range(pages+1):          url = url\_base + "issues?page={page}&q=is%3Aissue+is%3A{tag}".format(page=page+1, tag=tag)          print(url)          res = requests.get(url=url,proxies=proxies,headers=headers)          soup = BeautifulSoup(res.text,'html.parser')          issue\_groups = soup.find\_all('a', id=re.compile(r'issue\_\d+\_link'))          response\_groups = soup.find\_all('a', class\_="Link--muted")          response\_dict = dict()          for g in response\_groups:              try:                  link = "https://github.com" + g.get("href")                  count = int(g.span.string)                  response\_dict[link] = count              except:                  pass          if not issue\_groups:              break          for group in issue\_groups:              link = "https://github.com" + group.get("href")              title = group.string              if not title:                  continue              count = response\_dict.get(link, 0)              turns = get\_response(link)              data = {"url\_base":url\_base, "source\_url":url, "link":link, "title":title, "count":count, "turns":turns}              datas.append(data)      return datas  def request\_issue(project\_url):      pages = 100      print("started crawling....", project\_url)      print("started crawling open issues...", project\_url)      open\_datas = get\_links(url\_base=project\_url, pages=pages, tag="open")      print("started crawling closed issues...", project\_url)      closed\_datas = get\_links(url\_base=project\_url, pages=pages, tag="closed")      datas = open\_datas + closed\_datas      print("saved issues...", project\_url)      out = open('vacanza\_python-holidays\_issue.json', 'w+',encoding='utf-8')      out.write(json.dumps(datas, ensure\_ascii=False))      out.close()      return |

2) 热词分析(summary.py)

|  |
| --- |
| def get\_title\_turns():      with open('vacanza\_python-holidays\_issue.json', 'r',encoding='utf-8') as load\_f:          content = json.load(load\_f)      fp = open('vacanza\_python-holidays\_issue.txt', mode="w", encoding="utf-8")      for data in content:          title = data["title"]          turns = data["turns"]          data\_str = str(title) + '\n'          for turn in turns:              data\_str = data\_str + turn + '\n'          data\_str += '\n'          fp.write(data\_str)      fp.close()  def summary():      with open('vacanza\_python-holidays\_issue.txt','r',encoding='utf-8') as f:          t = f.read()      stopwords\_list = stopwords.words('english')      # 分词      w = word\_tokenize(t)      # 去除标点符号以及停止词      w = [word for word in w if word.isalnum() and word not in stopwords\_list]      # 词性标注      tw = pos\_tag(w)      # 提取名词      nounwords = [name for name,value in tw if value in ['NN','NNS','NNP','NNPS']]      # 词频统计      d = {}      for i in w:          if i in d:              d[i] += 1          else:              d[i] = 1      freq\_noun = {}      for i in d.keys():          if i in nounwords:              freq\_noun[i] = d[i]      # 排序      dict = sorted(freq\_noun.items(),key = lambda d: d[1],reverse=True)      with open('result.txt','w',encoding='utf-8') as f:          [f.write(str(item)+'\n') for item in dict]          f.close() |

#### 2.2.3 数据分析

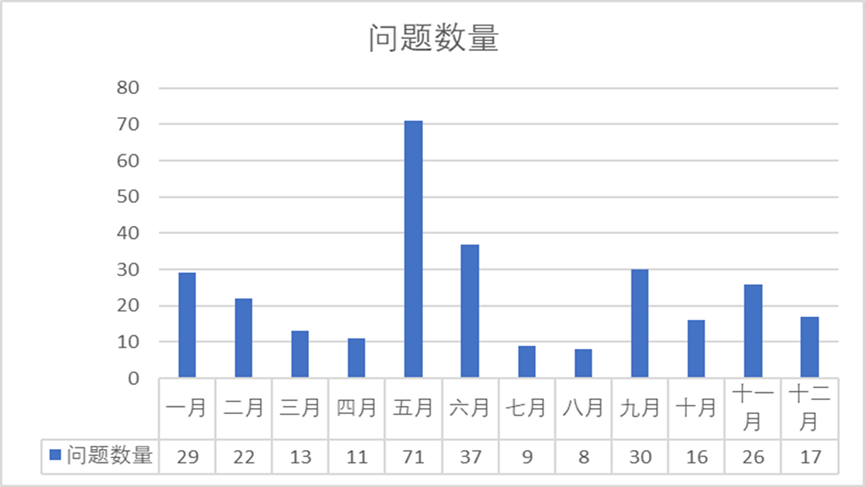


图2.1 月份-问题数量柱状图

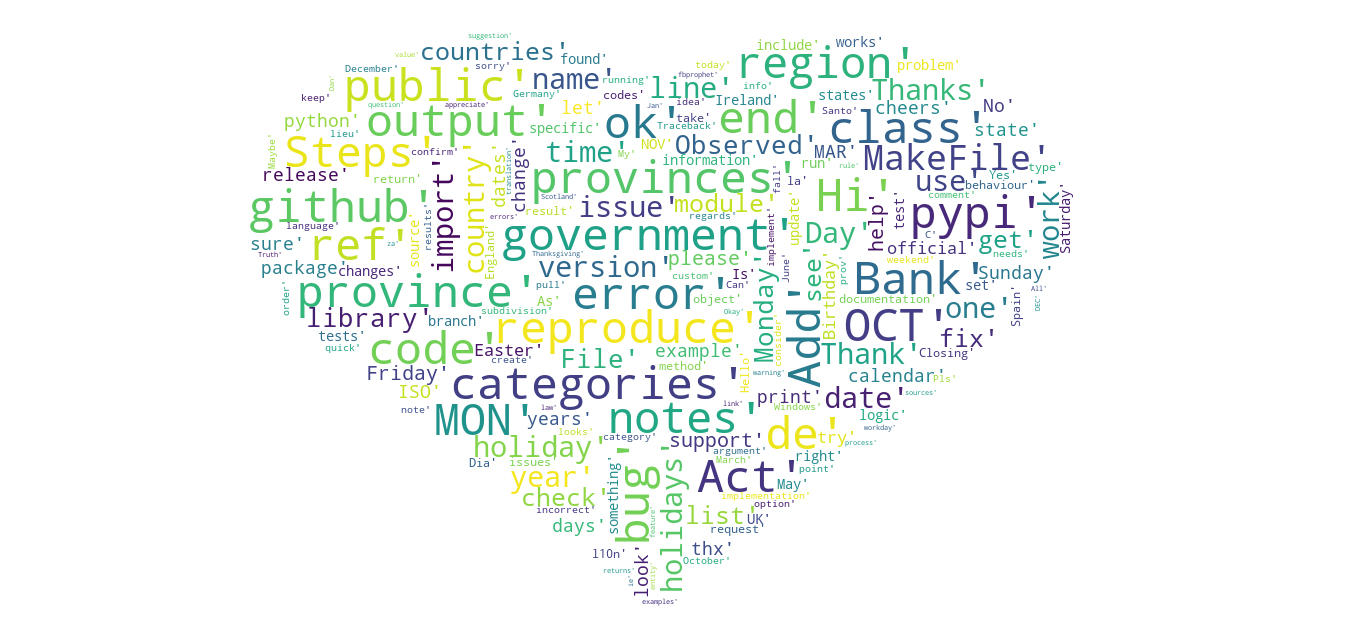


图2.2 词云图

由统计得出的数据可知，五月上传问题的数目最多，八月上传问题的数目最少。说明在五月发现相关问题的次数最多，八月发现相关问题的次数最少。再不考虑问题提前发现的前提下，近似认为项目出现bug是在上传问题当月。此外，本开源仓库的主题是调查各国各年各月休假状况的，同时，大部分国家位于北半球，所以，要想弄清仓库为什么会有相关bug或者issue，首先要调查各国当月的假期情况。根据调查数据得知，五月有国际劳动节这种国际化的节日，并且包含了母亲节、日韩人口相对多的儿童节等节日；而八月仅仅有瑞士马来西亚这种人口相对少的国家的国庆日等节日，因此出问题相对较少。同时，根据此调查结果，我们探索了本开源仓库的内部代码架构，发现单独依靠逻辑判断是没有办法维护好假期数据的，因为复杂度很高且具有一定不可预测性，很多假期都需要手动列出。因此，上传问题跟，不可预测的假期有着比较强的关联，例如我国除夕突然不放假的规定，使得代码容易出现错误。

## 三、项目测试

### 3.1 判断某一天是否为假期

**3.1.1 导入库和模块**

 导入了holidays库以及用于处理日期的date类

import holidays

from datetime import date

**3.1.2 创建国家假期对象**

 使用CountryHoliday类创建一个代表中国（国家代码为'CN'）的假期对象cn\_holidays。这个对象包含了中国的所有假期信息。

cn\_holidays = holidays.CountryHoliday('CN')

**3.1.3 检查特定日期是否是假期**

使用in关键字检查特定日期（2024年1月1日）是否在中国的假期列表中。如果是假期，就打印相应的消息，包括该日期对应的假期名称，否则打印非假期的消息。

if date(2024, 1, 1) in cn\_holidays:

print("2024.1.1("+cn\_holidays.get('2024-01-01')+") is a holiday!")

else:

print("2024.1.1 is not a holiday :(")

### 3.2 获得某个国家的假期

**3.2.1 导入库和模板**

import holidays

from datetime import date

**3.2.2 创建假期对象**

cn\_holidays = holidays.CountryHoliday('CN')

**3.2.3 遍历2024年的假期并打印**

holidays.China(years=2024) 返回一个代表中国2024年的假期对象

items() 方法返回一个包含日期和假期名称的元组列表

sorted() 函数用于按日期排序输出

遍历排序后的元组列表，并使用 print 打印每个假期的日期和名称

for date, name in sorted(holidays.China(years=2024)。items()):

print(date, name)

### 3.3添加自定义假期

自定义一个新的假期类，继承自china.CN，通过覆盖\_populate方法来添加额外的自定义假期。

**3.3.1 导入库和模块**

from datetime import date

from holidays.countries import china

**3.3.2 自定义假期类 ‘MyHolidays’**

 创建了一个名为MyHolidays的类，继承自china.CN，这表示它包含了中国的假期信息

 覆盖了\_populate方法，首先调用了父类的方法（super().\_populate(year)），然后在父类的假期基础上，添加了一个自定义假期，即在每年的7月13日定义了一个名为"My Holiday"的假期

class MyHolidays(china.CN):

def \_populate(self, year):

super().\_populate(year)

self[date(year, 7, 13)] = "My Holiday"

**3.3.3 创建日历实例并查询假期**

 创建了MyHolidays类的实例 holidays

 使用get方法查询特定日期的假期信息。在例子中，分别查询了2023年7月13日和10月1日的假期信息

# 创建一个日历实例

holidays = MyHolidays()

# 查询特定日期的假期信息

print(holidays.get('2023-07-13')) # output: My Holiday

print(holidays.get('2023-10-01')) # output: 国庆节 (China National Day)

## 四、项目分析

我们引入nltk库进行热词分析，在停止词选择，词性标注方面带来了巨大的便利，通过停止词和名词筛选过滤掉大量的非名词及常见无意义的名词。但随之的问题是nltk库的停止词与本项目所需的不尽相符，对于标注出的名词，筛选NN/NNS（名词）与NNP/NNPS（专有名词）则范围太广，筛选NNP/NNPS则范围太窄，与预期效果之间存在一定的差异。此外，nltk对与非英文的支持不够完善，部分中国及其他国家语言词汇未能成功标注，使热词分析结果存在一定的偏差。

## 五、项目总结、不足与改进

本次完成的项目整体符合标准，初步具备分析所需要的完整功能，但存在广度上和深度上的不足。我们小组完成的项目整体表现力不足，仅使用了常规的词云图和柱状图表示最终结果。数据收集的不足和数据本身质量的问题也影响了我们对结果的呈现，例如issue中存在大量语气用词和语法助词，在初步统计时并未做到详细区分，进而导致初次生成的词云图存在大量冗余信息，影响图片的直观性。此外，我们统计的信息只是英文文字相关信息，并未对不同语言和不同格式的信息（如图片、代码段等）进行统计，统计结果不是很完善。

这些问题的产生，一方面是因为时间较为紧凑，假期内我们小组的各位组员都有自己的时间安排，难以做到时间统一，减慢了项目的完成进度；一方面则是小组成员的编码开发经验不足，对于协作开发的使用不够熟练，仅掌握了开源软件基础课程所设置的网站的使用，该网站在校外无法访问，小组成员前期更多使用QQ这一即时通讯软件进行协作开发，降低了开发效率；最后则是运用知识解决实际问题的能力欠缺，存在自己造轮子、函数调用不当等问题，减缓了项目进度。

我们小组由于技术上和时间上的不成熟，未能将结果更好地呈现出来。未来可以从自身能力素养和多元数据统计两方面进行改进，提升自身学习运用知识的能力，结合实际程序、项目有针对性地练习；同时爬取统计更多种类的数据，运用合适的工具对其进行有效分析，实现结果的丰富性与完备性，减少因收集单种语言中单中类型的数据可能引发的歧义。

## 六、成员贡献与感想

（1）20212241028邵一辰（组长）

工作占比：20%

主要工作内容：总体进度安排、文档统合

感想：

开源软件基础课程大作业是一个很好的机会，让我们在实践中学习开源软件的知识和技能。本次大作业中我担任组长，主要负责对各个成员的进度进行统一安排和协调，并在最后完成对代码部分和文档部分的整理与统合。由于时间安排出了些问题，小组启动项目的时间较晚，导致整体上进度较为紧凑，部分环节也存在疏忽和遗漏。更重要的是，我意识到了在做项目的过程中，各司其职、环环相扣是必不可少的，甚至可以影响一个项目本身的成败。这时我也明白了为什么要在开源软件课程上讲述版本控制相关的知识，使用QQ群等聊天工具进行代码交流，不仅效率低下，而且会产生大量无用文件和冗余信息。相应地，使用版本控制工具可以提高开发的效率，信息明确且修改便捷。

各位小组成员互相学习、互相配合，也是我们项目完成的关键。每一位小组成员都有自己擅长的领域，有人擅长编程代码，有人擅长总结归纳，有人擅长统筹安排，有人擅长绘图分析。在我们小组每一个人的共同努力下，这个项目也算是顺利完成了。我也从每个小组成员身上学到许多：如何对整个项目的代码进行分析，如何爬取所需数据，如何生成直观清晰的统计图，如何测试某个开源项目等等。不仅将课堂所学知识运用到了生活实际上，也拓展了不少相关的课外知识，这对我未来的发展大有帮助。

（2）20212241476侯硕伟

工作占比：20%

主要工作内容：仓库分析

感想：

我在开源软件基础课程的大作业中负责对Python-Holidays仓库代码进行分析的工作。在对Python-Holidays仓库代码进行分析的过程中，我不仅学习了如何阅读和理解他人编写的代码，还学到了如何在代码中发现潜在的问题并提出改进的建议。通过与团队成员的讨论和合作，我学会了如何有效地沟通和协作，以及如何将自己的想法和建议清晰地表达出来。

此外，通过参与开源项目的贡献，我也更加深刻地理解了开源软件开发的核心价值和意义。开源软件不仅是一种技术手段，更是一种共同合作和分享知识的精神。通过开源软件项目，我们可以与全球范围内的开发者共同合作，共同解决各种技术难题，推动科技进步，造福人类社会。

总的来说，通过参与开源软件基础课程的大作业，我不仅提升了自己的编程能力和团队协作能力，还深刻体会到了开源软件开发的价值和意义。这对我的学习和职业发展都产生了积极的影响。

（3）20212241346马洋

工作占比：20%

主要工作内容：项目测试

感想：

开源软件基础课程的大作业，让我们在实践中学习开源软件的知识和技能。在完成这个作业的过程中，我学到了很多关于开源软件的技巧和方法，也学到了如何解决问题和思考问题的能力。在动手写代码之前，我们应该先规划好程序的框架，这样可以避免后期的反复修改和调试。在设计程序时，我们刚开始是根据老师提供的资料，用画图以及循环将框架设计出来，但是接下去便无从下手了。关于测试使用的函数需要我们自己设计，经过自己的反复尝试以及与与同学的讨论，测试函数顺利完成。

总之，这次设计程序让我受益良多，我学会了如何思考问题、解决问题，也让我知道团队精神得难能可贵。

（4）20212241079周鑫汐：

工作占比：20%

主要工作内容：issue爬取与分析

感想：

在GitHub开源仓库Issue爬取与分析的过程中，我深刻领略到了社区合作、问题解决和数据分析的优势所在。这个项目不仅提升了我的技术水平，还让我更好地理解了开源社区的运作方式以及问题管理的复杂性。

首先，我学到了如何使用Python中强大的爬虫库，如Beautiful Soup和Requests。这使我能够从网页上抓取大量的原始数据。在此过程中遇到了处理分页、过滤和排序等问题，这些都锻炼了我的网络编程能力。

基于爬取到的数据，我进行了热词分析，利用自然语言处理技术从文本数据中提取关键信息。这一部分的工作让我接触到了一些常用的文本挖掘技术，如分词、词频统计、TF-IDF等。通过分析热词洞察到数据中的关键信息，理解用户关注点。

（5）20212241242杨晨

工作占比：20%

主要工作内容：代码整合与绘图

感想：

本次开源项目中我参与编写了爬数据和数据可视化相关工作，在组长和优秀组员们的带领下，我们选择了python-holiday这一仓库。我们通过分析代码、爬取issue相关的数据，一步一步分析了这个开源仓库。并且，我切身体会了一个开源项目究竟是如何在这么多人的努力下进行的。在这段时间的工作之中，自身对知识的探索、组内的互相帮助，使我受益良多。首先，我认识到了开源项目是一个令人兴奋而充满挑战的经历。它是一种独特的学习机会，让我深入了解软件开发、团队协作和开放式社区的运作。通过与其他开发者合作，我学到了如何提高代码质量、编写清晰文档以及有效地使用版本控制工具。

同时我在开源项目中体验到了与来自不同擅长领域、有着不同技能水平的同学合作的重要性。这种多样性促使我们从不同的角度思考问题，找到更好的解决方案。同时，通过与他人合作，我学到了沟通和协调的重要性，以确保整个项目能够有序推进。

开源文化的核心价值：共享和协作。这两个词语在这次大作业的经历之中让我印象更为深刻。与其他开发者分享我的工作，接受他们的反馈和建议，让项目得以不断改进。这种协同努力的感觉令人愉悦，同时也让我更加珍惜这次宝贵的经历。

然而，开源项目也带来了挑战。项目的维护需要时间和精力。对于学生的我，去解决复杂的技术问题可能需要深入研究和大量的努力。但正是这些挑战让我不断成长，提高了解决问题和团队合作的能力。

总的来说，这次大作业的合作是一次丰富而深刻的体验。它不仅帮助我提高了技术水平，还培养了我在团队合作和社区互动方面的能力。