Computer Applications Lab

Project: Wikimedia Metadata Analysis

Members:

1. Hazim Abdelmahdi 0191894
2. Layan Barham 0195716
3. Yussif Abdalla 2180142

The role of each student in the project:

PageViews Dataset → Hazim

GeoEditors Dataset → Layan

Unique Devices Dataset → Yussif

Done by: **Hazim Adel**

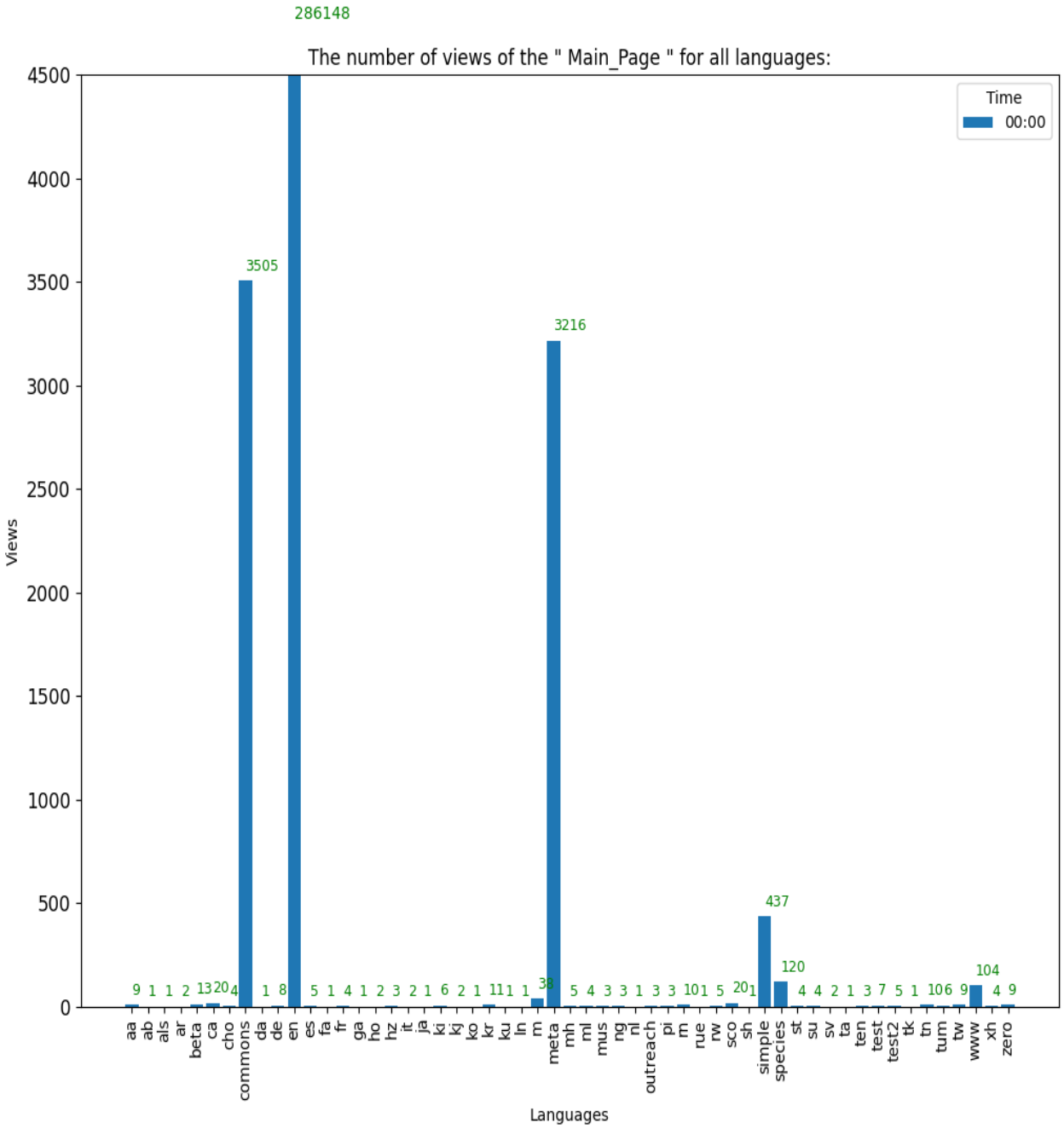
PageViews Dataset

I choose **2018/5/3** to do these plots and charts at time **00:00** and **12:00**.

1. First Plot:

This bar chart shows the relation between the (the views of the main page) and languages (which the data connected by the language without the extension that shows where the page located)

I limit the views to 4500 because English language has a lot of views and others are very small.



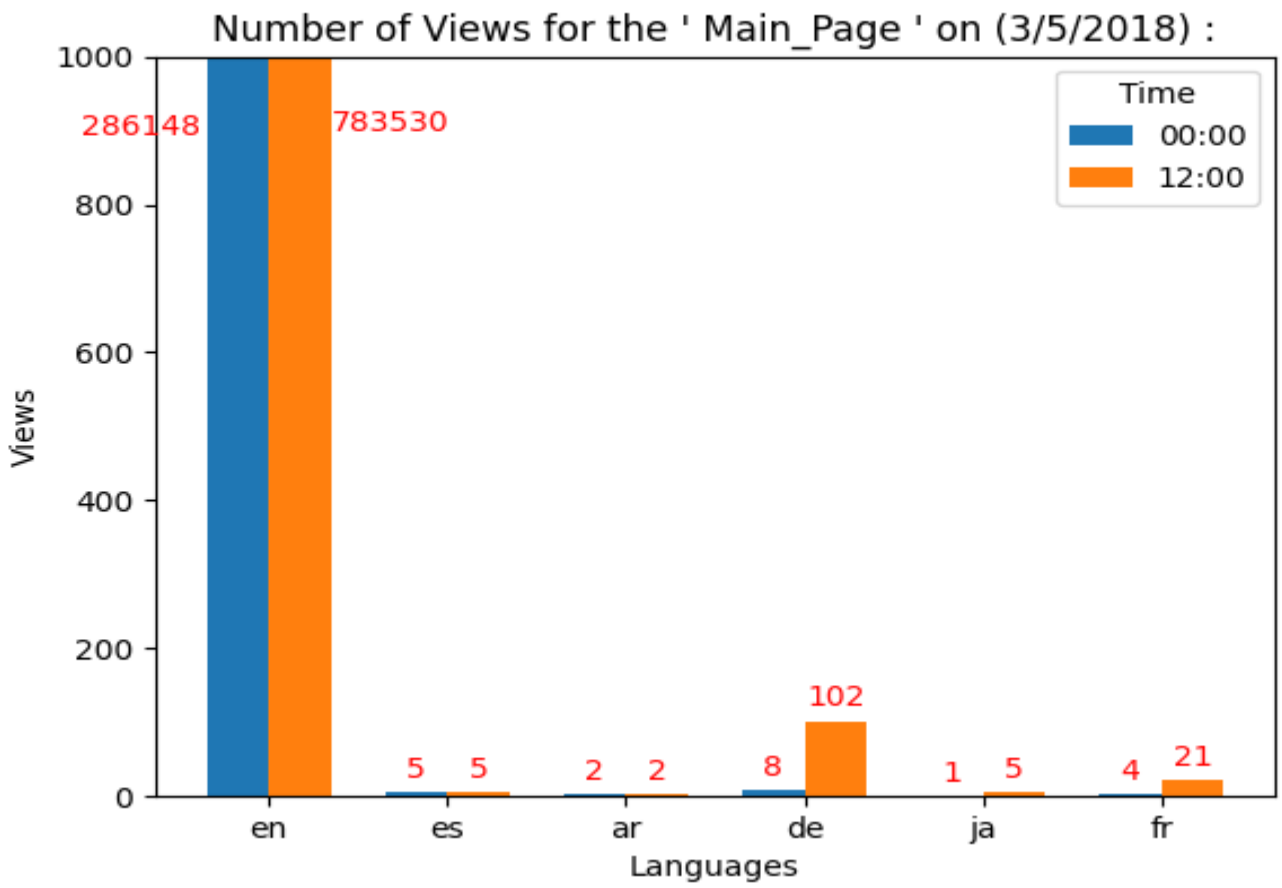
The code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import openpyxl

#read the text file which contains all languages
with open('python data\pageviews-20180503-000000.txt',encoding='utf-8') as f:
    contents = f.readlines()
    lst00 = []
    for i in range(len(contents)):
        lst00.append(contents[i].strip().split(' '))

#convert lst00 to dataframe
df00 =
pd.DataFrame(lst00,columns=['languageWithWebsite','article_name','views','page_size'])

#split all languages to view the first element of languages without a following
character are wikipedia projects
languages = df00.languageWithWebsite.str.split(".")
sublist =[]
sublist2 = []
for language in languages:
    sublist.append(language[0])
    try:
        sublist2.append(language[1])
    except:
        sublist2.append(np.NAN)

#Add sublist to the dataframe
df00['language'] = sublist
df00['website'] = sublist2
df00['website'].fillna('p',inplace=True)
views = []
size = []

#convert each element from str to int
for v in df00['views']:
    views.append(int(v))
for s in df00['page_size']:
    size.append(int(s))
df00['views'] = views
df00['page_size'] = size


#Part 1

temp00 = df00[df00['article_name']=='Main_Page']
x = np.unique(temp00.language)
y = temp00.groupby('language').sum().reset_index()['views']
fig,ax = plt.subplots(figsize=(12,10))
plt.bar(x,y)
plt.xticks( rotation=90, fontsize=10)
for i in range(len(x)):
    if y[i]>4500:
        plt.text(i, y[i]//60, y[i], color='green', fontsize=10)
    else:
        plt.text(i,y[i]+50,y[i],color = 'green',fontsize = 9)
plt.ylim([0,4500])
plt.title('The number of views of the \" Main_Page \" for all languages:')
plt.legend(['00:00'],title = 'Time')
plt.xlabel('Languages')
plt.ylabel('Views')
plt.yticks(fontsize = 12)
plt.show()
```

2. Second Plot:

In this bar chart I compare between the views of some languages at different time.

I put limit to the y-axis because English language has giant numbers of views unlike other languages.



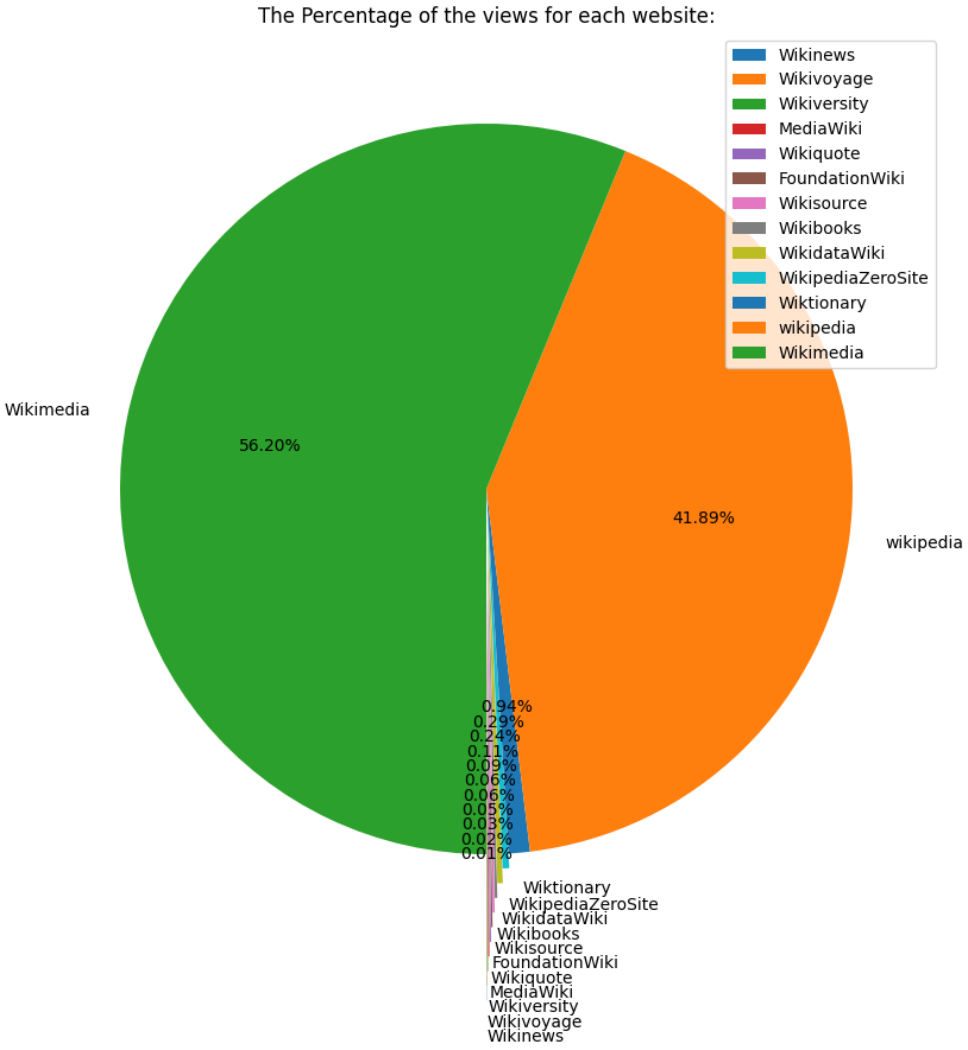
The code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import openpyxl

#read the text file which contains all languages
with open('python data\pageviews-20180503-000000.txt',encoding='utf-8') as f:
    contents = f.readlines()
    lst00 = []
    for i in range(len(contents)):
        lst00.append(contents[i].strip().split(' '))

#convert lst00 to dataframe
df00 = pd.DataFrame(lst00,columns=['languageWithWebsite','article_name','views','page_size'])

#split all languages to view the first element of languages without a following character are
wikipedia projects
languages = df00.languageWithWebsite.str.split(".")
sublist =[]
sublist2 = []
for language in languages:
    sublist.append(language[0])
    try:
        sublist2.append(language[1])
    except:
        sublist2.append(np.NAN)

#Add sublist to the dataframe
df00['language'] = sublist
df00['website'] = sublist2
df00['website'].fillna('p',inplace=True)
views = []
size = []
```

```
#convert each element from str to int
for v in df00['views']:
    views.append(int(v))
for s in df00['page_size']:
    size.append(int(s))
df00['views'] = views
df00['page_size'] = size

#Part 2
#read the text file which contains all languages
with open('python data\pageviews-20180503-120000.txt',encoding='utf-8') as f:
    contents = f.readlines()
    lst12 = []
    for i in range(len(contents)):
        lst12.append(contents[i].strip().split(' '))

#convert lst12 to dataframe
df12 = pd.DataFrame(lst12,columns=['languageWithWebsite','article_name','views','page_size'])

#split all languages to view the first element of languages without a following character are
wikipedia projects
languages = df12.languageWithWebsite.str.split(".")
sublist =[]
for language in languages:
    sublist.append(language[0])

#Add sublist to the dataframe
df12['language'] = sublist
views = []
size = []

#convert each element from str to int
for v in df12['views']:
    try:
        views.append(int(v))
    except:
        views.append(v)
for s in df12['page_size']:
    try:
        size.append(int(s))
    except:
        size.append(s)
df12['views'] = views
df12['page_size'] = size

temp00 = df00[df00['article_name']=='Main_Page']
temp12 = df12[df12['article_name']=='Main_Page']
group00 = temp00.groupby('language')['views'].sum()
group12 = temp12.groupby('language')['views'].sum()
l = ['en','es','ar','de','ja','fr']

x = np.arange(len(l))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots()

data1 = ax.bar(x - width/2,group00[l], width, label='00:00')
data2 = ax.bar(x + width/2,group12[l], width, label='12:00')

ax.bar_label(data1, padding=3,color = 'red')
ax.bar_label(data2, padding=3,color = 'red')

ax.text(0-width*3, group00['en']//320, group00['en'], color='red', fontsize=10)
ax.text(0+width, group12['en']//870, int(group12['en']), color='red', fontsize=10)

plt.ylim([0,1000])
plt.ylabel('Views')
plt.xlabel('Languages')
ax.set_title('Number of Views for the \' Main_Page \' on (3/5/2018) :')
ax.set_xticks(x)
ax.set_xticklabels(l)
ax.legend(title = 'Time')
plt.show()
```

3. Third Plot:

In this pie chart shows the percentage for different websites at **00:00 3/5/2018**



The code:

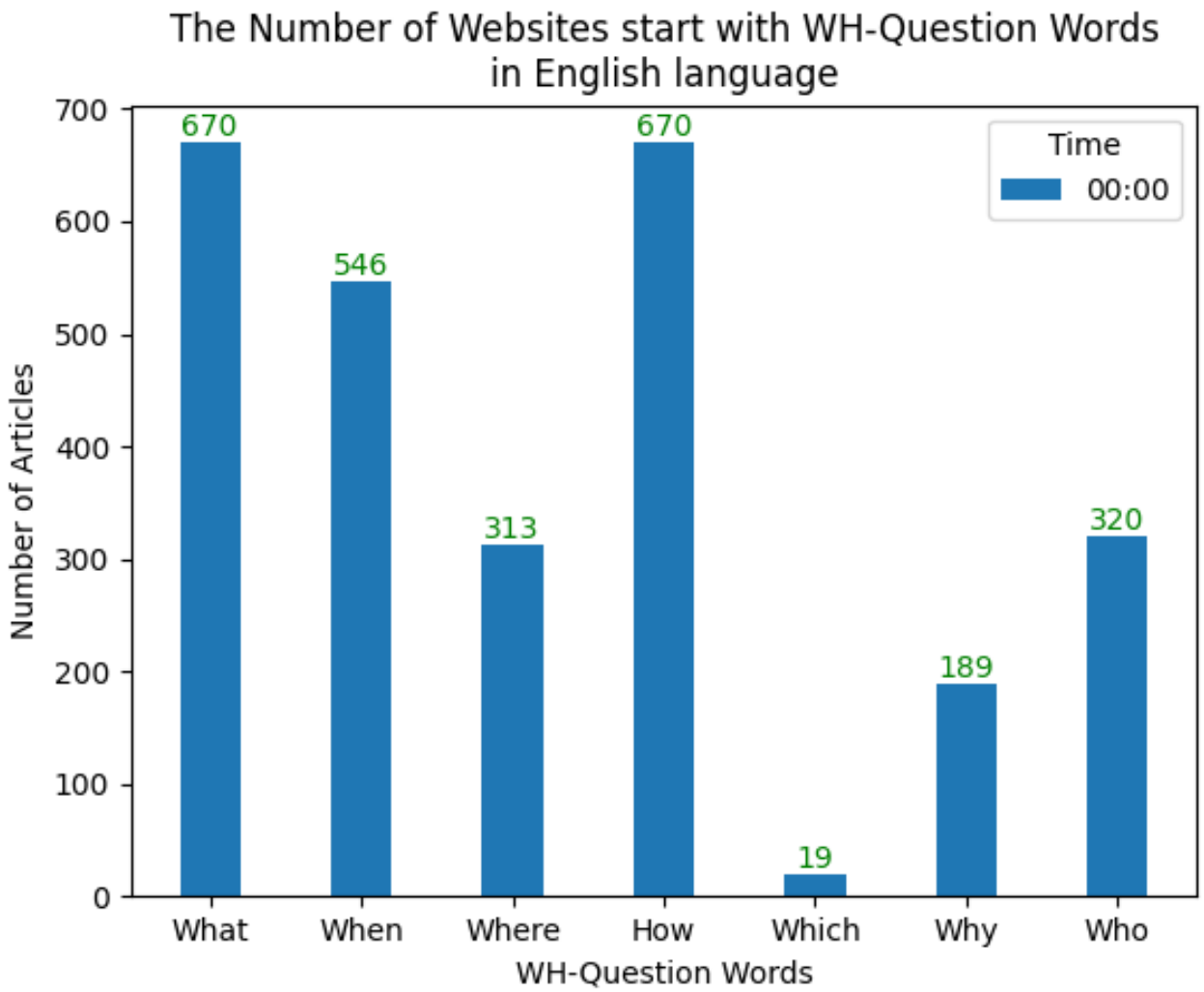
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import openpyxl

#read the text file which contains all languages
with open('python data\pageviews-20180503-000000.txt',encoding='utf-8') as f:
    contents = f.readlines()
    lst00 = []
    for i in range(len(contents)):
        lst00.append(contents[i].strip().split(' '))

#convert lst00 to dataframe
df00 = pd.DataFrame(lst00,columns=['languageWithWebsite','article_name','views','page_size'])

#split all languages to view the first element of languages without a following character are
wikipedia projects
languages = df00.languageWithWebsite.str.split(".")
sublist =[]
sublist2 = []
for language in languages:
    sublist.append(language[0])
    try:
        sublist2.append(language[1])
    except:
        sublist2.append(np.NAN)
```

```
#Add sublist to the dataframe
df00['language'] = sublist
df00['website'] = sublist2
df00['website'].fillna('p',inplace=True)
views = []
size = []

#convert each element from str to int
for v in df00['views']:
    views.append(int(v))
for s in df00['page_size']:
    size.append(int(s))
df00['views'] = views
df00['page_size'] = size

#Part 3
fig,ax = plt.subplots(figsize=(12,10))
temp = df00.groupby('website').sum()['views']
l =
['Wikinews','Wikivoyage','Wikiversity','MediaWiki','Wikiquote','FoundationWiki','Wikisource','Wiki
books','WikidataWiki','WikipediaZeroSite','Wiktionary','wikipedia','Wikimedia']
ex = [0.4,0.36,0.32,0.28,0.24,0.2,0.16,0.12,0.08,0.04,0,0,0]
plt.pie(sorted(temp),labels=l,autopct='%1.2f%%',startangle=270,explode=ex)
plt.legend()
plt.title('The Percentage of the views for each website:')
plt.show()
```

4. Fourth Plot:

This chart shows number of articles start with WH-Question words on **(3/5/2018 – 00:00)**

For English language**.**

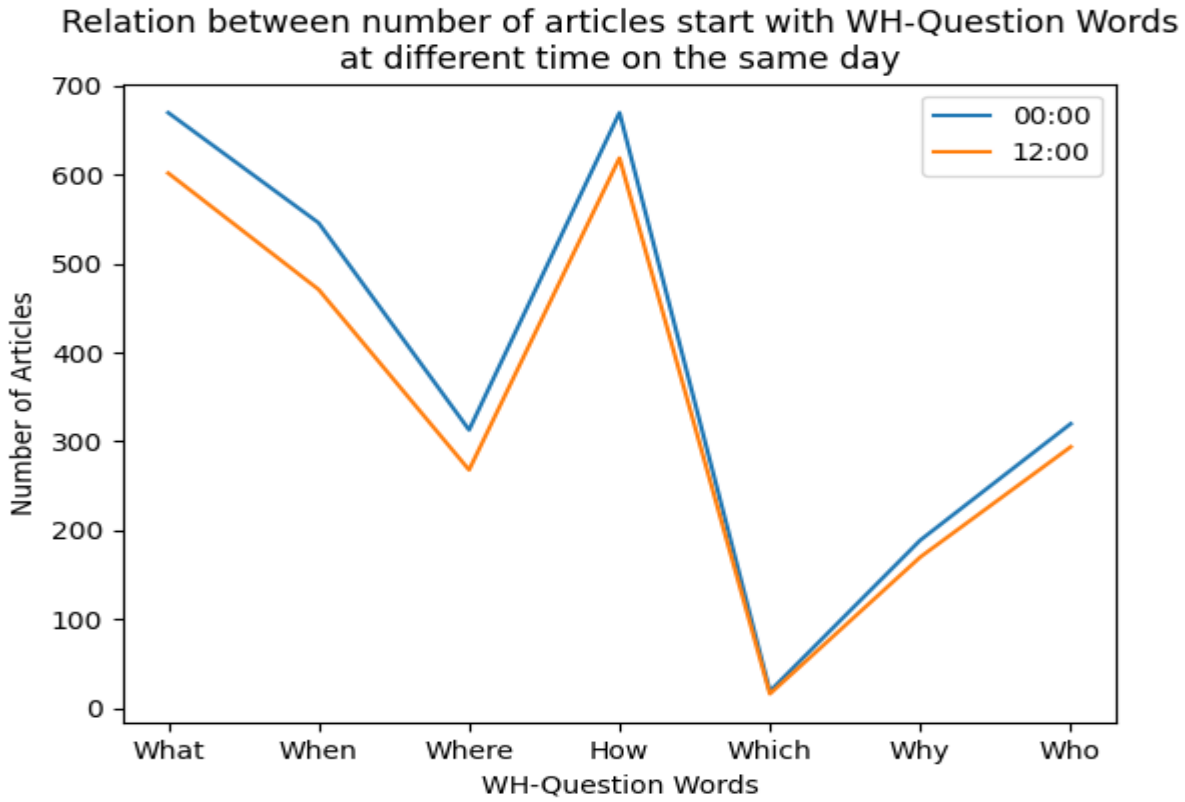


The code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import openpyxl

#read the text file which contains all languages
with open('python data\pageviews-20180503-000000.txt',encoding='utf-8') as f:
    contents = f.readlines()
    lst00 = []
    for i in range(len(contents)):
        lst00.append(contents[i].strip().split(' '))

#convert lst00 to dataframe
df00 =
pd.DataFrame(lst00,columns=['languageWithWebsite','article_name','views','page_size'])

#split all languages to view the first element of languages without a following
character are wikipedia projects
languages = df00.languageWithWebsite.str.split(".")
sublist =[]
sublist2 = []
for language in languages:
    sublist.append(language[0])
    try:
        sublist2.append(language[1])
    except:
        sublist2.append(np.NAN)

#Add sublist to the dataframe
df00['language'] = sublist
df00['website'] = sublist2
df00['website'].fillna('p',inplace=True)
views = []
size = []

#convert each element from str to int
for v in df00['views']:
    views.append(int(v))
for s in df00['page_size']:
    size.append(int(s))
df00['views'] = views
df00['page_size'] = size

#Part 4
en = df00[df00['language']=='en']
words = en['article_name'].str.split('_')
slist = []
for word in words:
    slist.append(word[0])
labels = ['What','When','Where','How','Which','Why','Who']
counts = pd.DataFrame(slist).value_counts()[labels]

fig, ax = plt.subplots()

data = ax.bar(labels,counts,width=0.4)
plt.title('The Number of Websites start with WH-Question Words\nin English language')
plt.xlabel('WH-Question Words')
plt.ylabel('Number of Articles')
plt.legend(['00:00'],title = 'Time')
plt.bar_label(data,color = 'green')
plt.show()
```

5. Fifth plot:

This plot display how the number of articles start with WH-Question words changing when the time change in the same day (**3/5/2018**).



The code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import openpyxl

#read the text file which contains all languages
with open('python data\pageviews-20180503-000000.txt',encoding='utf-8') as f:
    contents = f.readlines()
    lst00 = []
    for i in range(len(contents)):
        lst00.append(contents[i].strip().split(' '))

#convert lst00 to dataframe
df00 =
pd.DataFrame(lst00,columns=['languageWithWebsite','article_name','views','page_size'])

#split all languages to view the first element of languages without a following
character are wikipedia projects
languages = df00.languageWithWebsite.str.split(".")
sublist =[]
sublist2 = []
for language in languages:
    sublist.append(language[0])
    try:
        sublist2.append(language[1])
    except:
        sublist2.append(np.NAN)

#Add sublist to the dataframe
df00['language'] = sublist
df00['website'] = sublist2
df00['website'].fillna('p',inplace=True)
views = []
size = []
```

```
#convert each element from str to int
for v in df00['views']:
    views.append(int(v))
for s in df00['page_size']:
    size.append(int(s))
df00['views'] = views
df00['page_size'] = size

#read the text file which contains all languages
with open('python data\pageviews-20180503-120000.txt',encoding='utf-8') as f:
    contents = f.readlines()
    lst12 = []
    for i in range(len(contents)):
        lst12.append(contents[i].strip().split(' '))

#convert lst12 to dataframe
df12 =
pd.DataFrame(lst12,columns=['languageWithWebsite','article_name','views','page_size'])

#split all languages to view the first element of languages without a following
character are wikipedia projects
languages = df12.languageWithWebsite.str.split(".")
sublist =[]
for language in languages:
    sublist.append(language[0])

#Add sublist to the dataframe
df12['language'] = sublist
views = []
size = []

#convert each element from str to int
for v in df12['views']:
    try:
        views.append(int(v))
    except:
        views.append(v)
for s in df12['page_size']:
    try:
        size.append(int(s))
    except:
        size.append(s)
df12['views'] = views
df12['page_size'] = size


en = df00[df00['language']=='en']
words = en['article_name'].str.split('_')
slist = []
for word in words:
    slist.append(word[0])
labels = ['What','When','Where','How','Which','Why','Who']
counts = pd.DataFrame(slist).value_counts()[labels]

en2 = df12[df12['language']=='en']
words2 = en2['article_name'].str.split('_')
slist2 = []
for word in words2:
    slist2.append(word[0])
counts2 = pd.DataFrame(slist2).value_counts()[labels]
plt.plot(labels,counts)
plt.plot(labels,counts2)
plt.legend(labels = ['00:00','12:00'])
plt.title('Relation between number of articles start with WH-Question Words\nat
different time on the same day')
plt.xlabel('WH-Question Words')
plt.ylabel('Number of Articles')
plt.show()
```

Done by: Layan Barham

GeoEditors Dataset

File used: https://dumps.wikimedia.org/other/geoeditors/geoeditors-monthly-2021-01.tsv

Description of dataset:

The file will have the following columns:

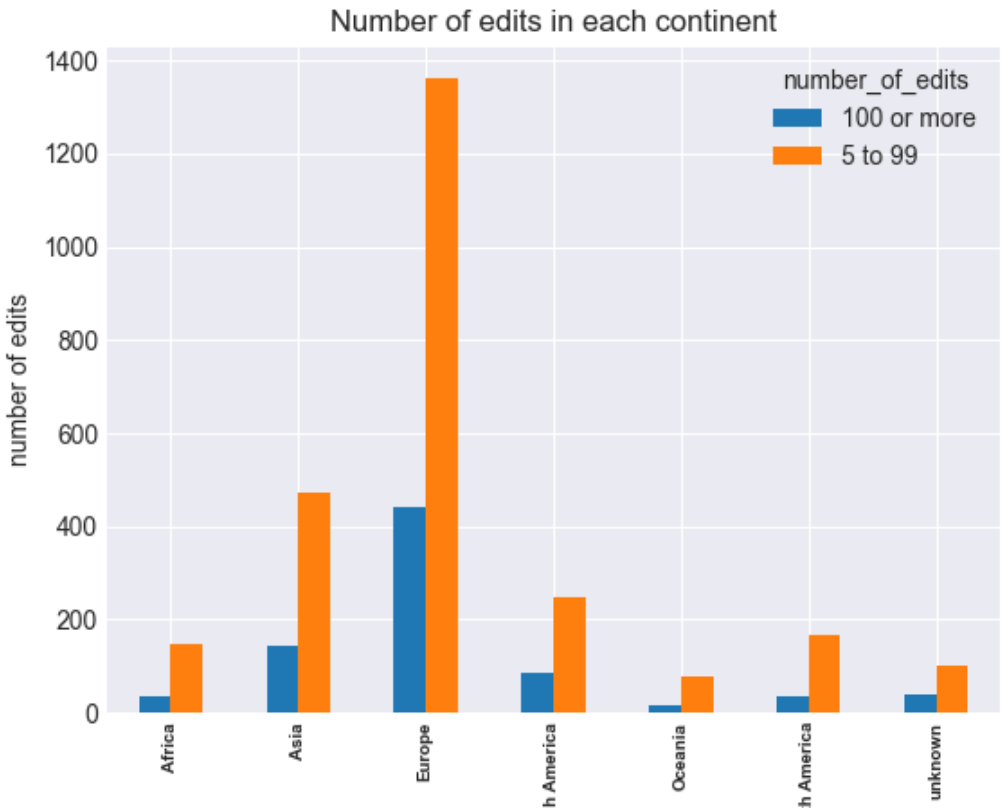
- wiki db: the code name for the wiki, "enwiki" for English Wikipedia, at this time the dataset is available just for Wikipedias
- country: the name of the country with editors of this wiki
- activity level: how many edits this group of editors has made in the past month (either 5 to 99 or more than 100)
- lower bound: at least this many editors in this group
- upper bound: at most this many editors in this group

Number of columns: 5

Number of rows: 3373

Plots :

The figure show number of edits in each continent in January 2021

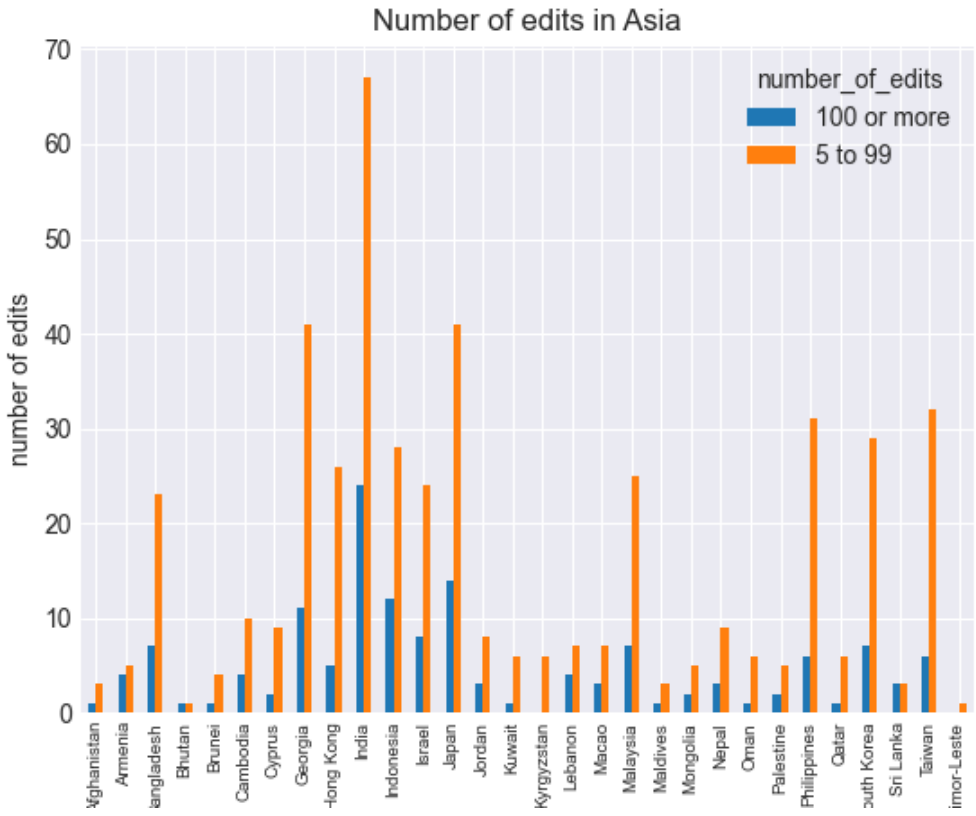


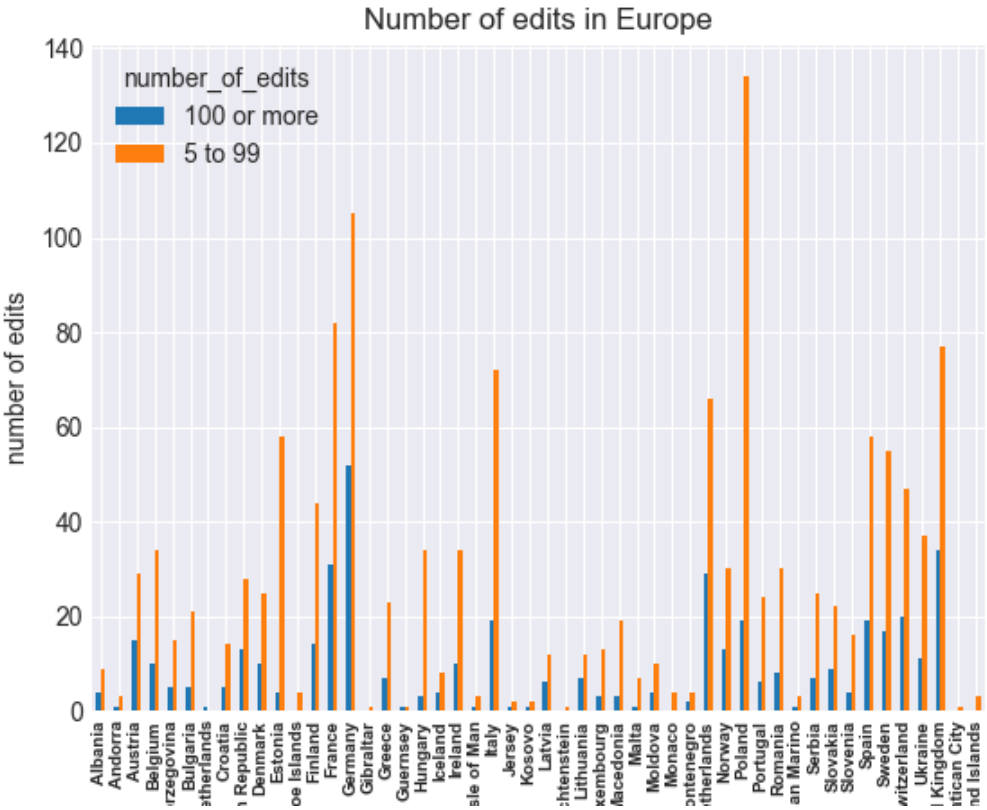
Code:

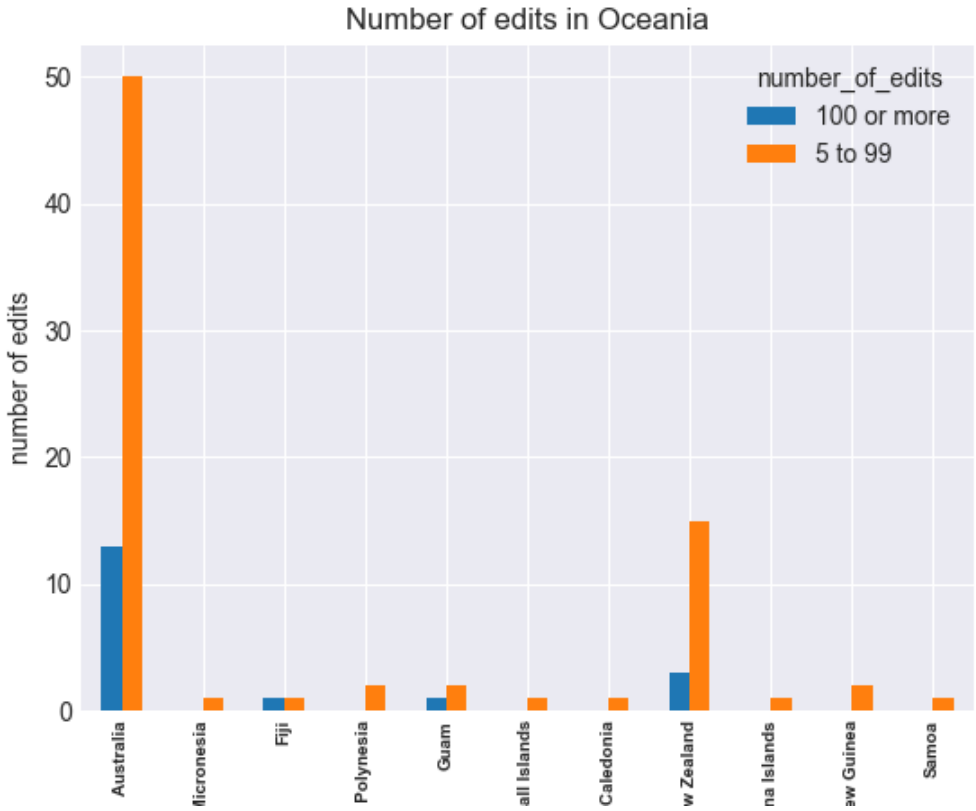
```
import pandas as pd
import matplotlib.pyplot as plt
import pycountry_convert as pc

geoeditor = pd.read_csv('C:\\Users\\GTS\\Downloads\\geoeditors.csv')

def country_to_continent(country_name):
    try:
        country_alpha2 =
pc.country_name_to_country_alpha2(country_name)
        country_continent_code =
pc.country_alpha2_to_continent_code(country_alpha2)
        country_continent_name =
pc.convert_continent_code_to_continent_name(country_continent_code)
        return country_continent_name
    except:
        if country_name == 'unknown': return 'unknown'
        if country_name == 'Caribbean Netherlands': return "Europe"
        if country_name == 'Kosovo': return "Europe"
        if country_name == 'Vatican City': return "Europe"
        if country_name == 'Timor-Leste': return "Asia"
        print(country_name)
        return ''

plt.style.use('seaborn-darkgrid')
```

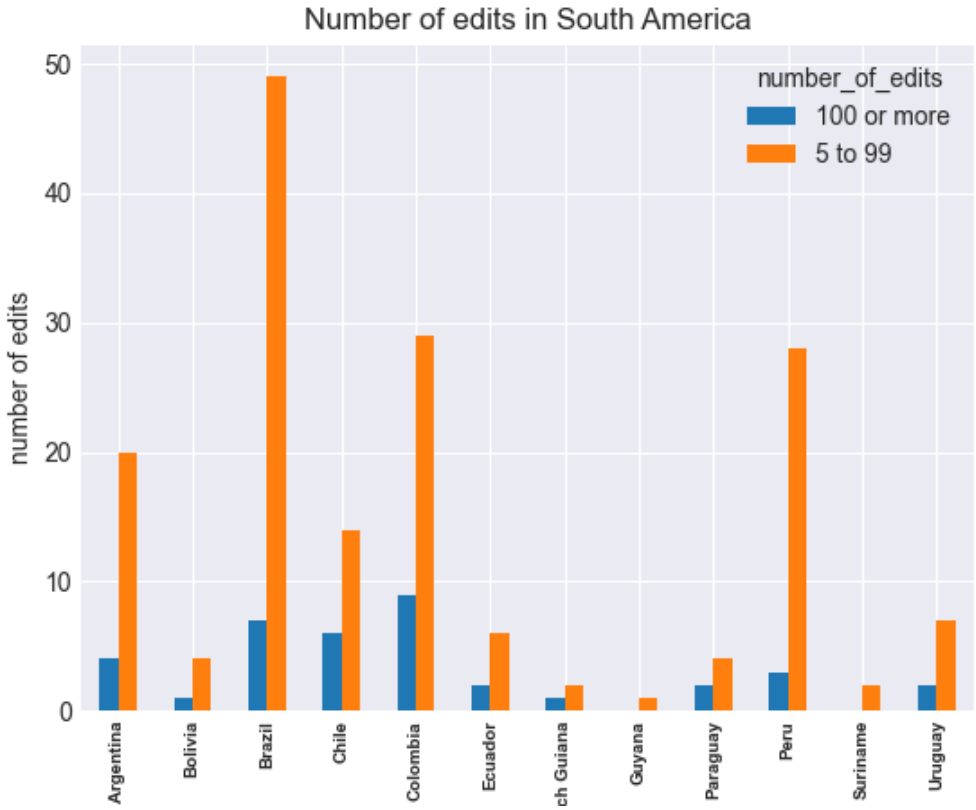
```
geoeditor['continent'] =
geoeditor['country'].apply(country_to_continent)


filtered_df = geoeditor[['continent', 'number_of_edits']].copy()
pd.crosstab(filtered_df['continent'],filtered_df['number_of_edits']).pl
ot.bar()
plt.xticks(fontsize=7,weight='bold')
plt.title('Number of edits in each continent')
plt.xlabel('continents')
plt.ylabel('number of edits')
plt.show()
```

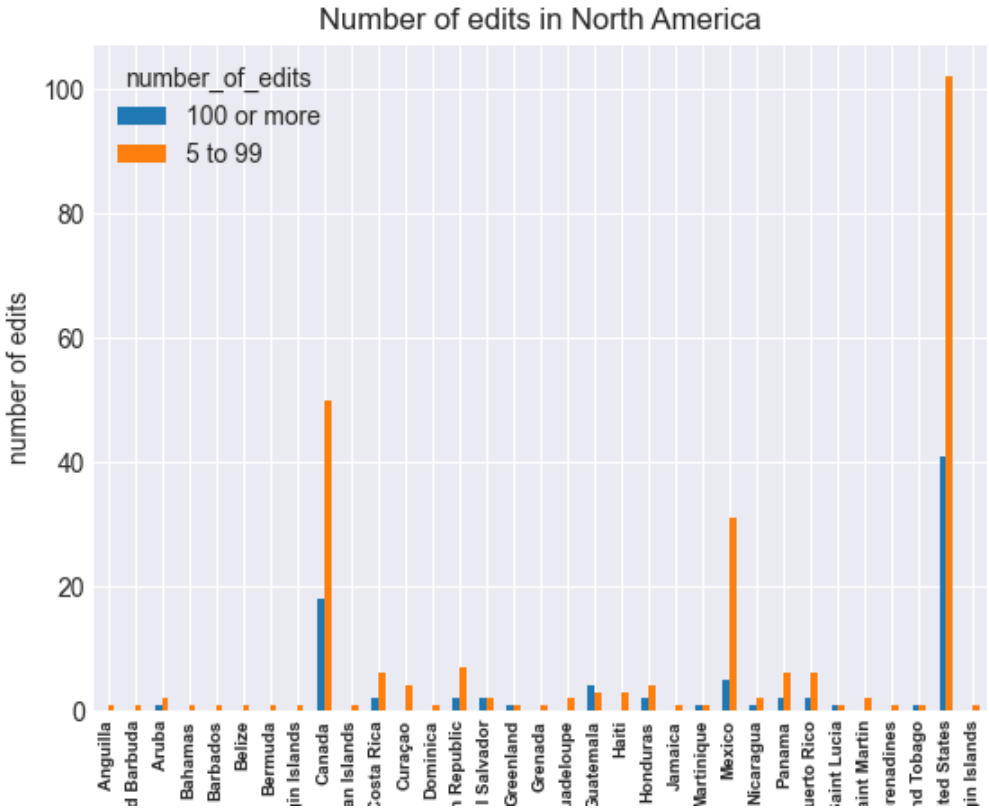
the figures show number of edits in each continent

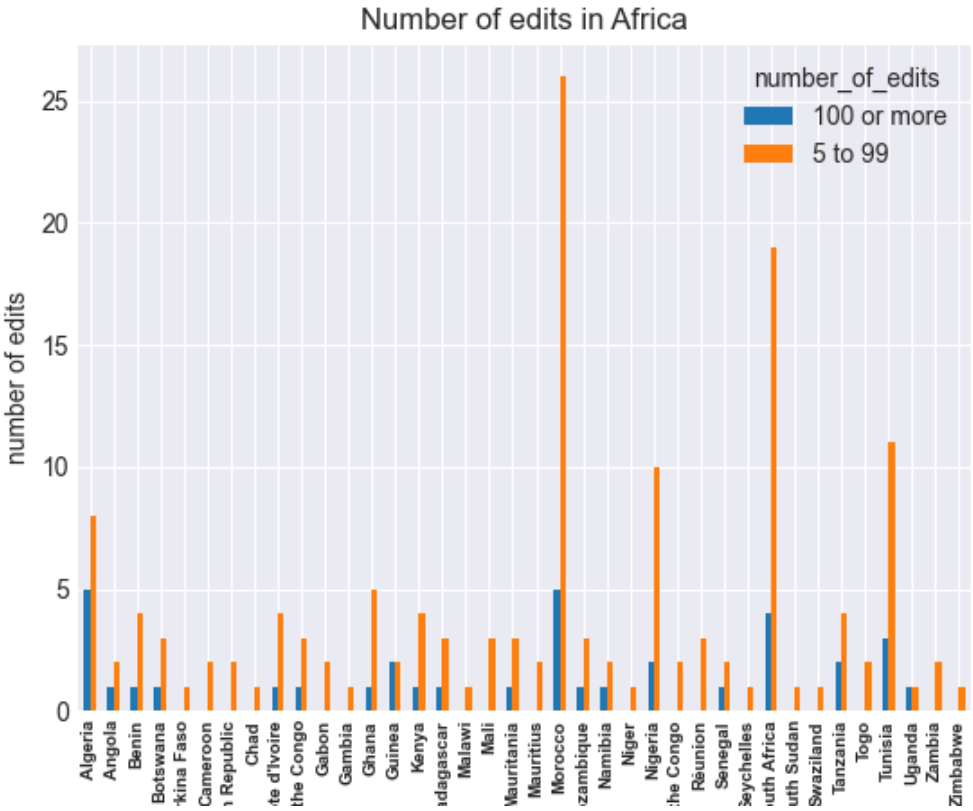


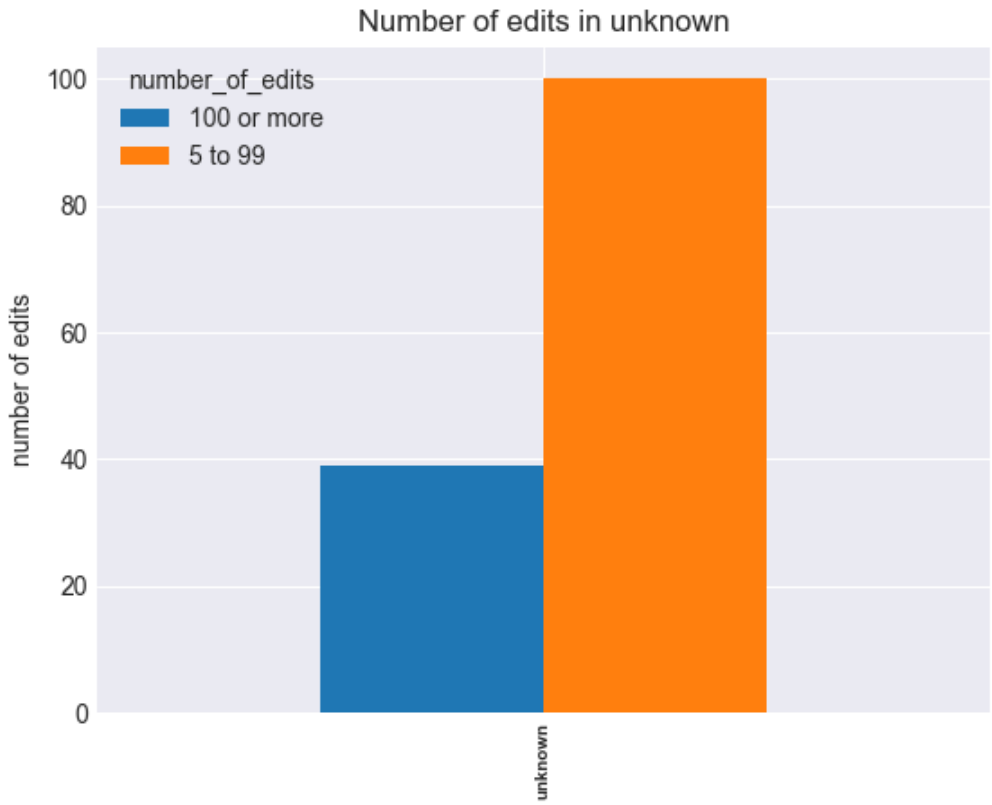

Number of edits in Europe
number_of_edits
100 or more
5 to 99
number of edits
0
20
40
60
80
100
120
140
Albania
Andorra
Austria
Belgium
Bulgaria
Croatia
Denmark
Estonia
Finland
France
Germany
Gibraltar
Greece
Guernsey
Hungary
Iceland
Ireland
Italy
Jersey
Kosovo
Latvia
Lithuania
Malta
Moldova
Monaco
Norway
Poland
Portugal
Romania
Serbia
Slovakia
Slovenia
Spain
Sweden
Ukraine


Number of edits in Oceania
number_of_edits
100 or more
5 to 99
number of edits
0
10
20
30
40
50
Australia
Fiji
Polynesia
Guam
Caledonia
Samoa


Number of edits in South America
number_of_edits
100 or more
5 to 99
number of edits
50
40
30
20
10
0
Argentina
Bolivia
Brazil
Chile
Colombia
Ecuador
ch Guiana
Guyana
Paraguay
Peru
Suriname
Uruguay


Number of edits in North America
number_of_edits
100 or more
5 to 99
number of edits
100
80
60
40
20
0
Anguilla
Aruba
Bahamas
Barbados
Belize
Bermuda
Canada
Haiti
Honduras
Jamaica
Mexico
Nicaragua
Panama


Number of edits in Africa
number_of_edits
100 or more
5 to 99
number of edits
0
5
10
15
20
25
Algeria
Angola
Benin
Botswana
Cameroon
Chad
Gabon
Gambia
Ghana
Guinea
Kenya
Malawi
Mali
Mauritius
Morocco
Namibia
Niger
Nigeria
Réunion
Senegal
Swaziland
Tanzania
Togo
Tunisia
Uganda
Zambia
Zimbabwe



Code:

```
import pandas as pd
import matplotlib.pyplot as plt
import pycountry_convert as pc

geoeditor = pd.read_csv('C:\\Users\\GTS\\Downloads\\geoeditors.csv')

def country_to_continent(country_name):
    try:
        country_alpha2 =
pc.country_name_to_country_alpha2(country_name)
        country_continent_code =
pc.country_alpha2_to_continent_code(country_alpha2)
        country_continent_name =
pc.convert_continent_code_to_continent_name(country_continent_code)
        return country_continent_name
    except:
        if country_name == 'unknown': return 'unknown'
        if country_name == 'Caribbean Netherlands': return "Europe"
        if country_name == 'Kosovo': return "Europe"
        if country_name == 'Vatican City': return "Europe"
        if country_name == 'Timor-Leste': return "Asia"
        print(country_name)
        return ''

plt.style.use('seaborn-darkgrid')
```

```
geoeditor['continent'] =
geoeditor['country'].apply(country_to_continent)


asia_df=geoeditor[geoeditor['continent'] == 'Asia']
pd.crosstab(asia_df['country'],asia_df['number_of_edits']).plot.bar()
plt.xticks(fontsize=7)
plt.title('Number of edits in Asia')
plt.xlabel('countries')
plt.ylabel('number of edits')
plt.show()

europe_df=geoeditor[geoeditor['continent'] == 'Europe']
pd.crosstab(europe_df['country'],europe_df['number_of_edits']).plot.bar
()
plt.xticks(fontsize=7,weight='bold')
plt.title('Number of edits in Europe')
plt.xlabel('countries')
plt.ylabel('number of edits')
plt.show()

oceania_df=geoeditor[geoeditor['continent'] == 'Oceania']
pd.crosstab(oceania_df['country'],oceania_df['number_of_edits']).plot.b
ar()
plt.xticks(fontsize=7,weight='bold')
plt.title('Number of edits in Oceania')
plt.xlabel('countries')
plt.ylabel('number of edits')
plt.show()

s_america_df=geoeditor[geoeditor['continent'] == 'South America']
pd.crosstab(s_america_df['country'],s_america_df['number_of_edits']).pl
ot.bar()
plt.xticks(fontsize=7,weight='bold')
plt.title('Number of edits in South America')
plt.xlabel('countries')
plt.ylabel('number of edits')
plt.show()

n_america_df=geoeditor[geoeditor['continent'] == 'North America']
pd.crosstab(n_america_df['country'],n_america_df['number_of_edits']).pl
ot.bar()
plt.xticks(fontsize=7,weight='bold')
plt.title('Number of edits in North America')
plt.xlabel('countries')
plt.ylabel('number of edits')
plt.show()

afrcia_df=geoeditor[geoeditor['continent'] == 'Africa']
pd.crosstab(afrcia_df['country'],afrcia_df['number_of_edits']).plot.bar
()
plt.xticks(fontsize=7,weight='bold')
plt.title('Number of edits in Africa')
plt.xlabel('countries')
plt.ylabel('number of edits')
plt.show()
```

```
unknown_df=geoeditor[geoeditor['continent'] == 'unknown']
pd.crosstab(unknown_df['country'],unknown_df['number_of_edits']).plot.b
ar()
plt.xticks(fontsize=7,weight='bold')
plt.title('Number of edits in unknown')
plt.xlabel('unknown')
plt.ylabel('number of edits')
plt.show()
```

Arabic Wikipedia

figure shows continent using Arabic Wikipedia

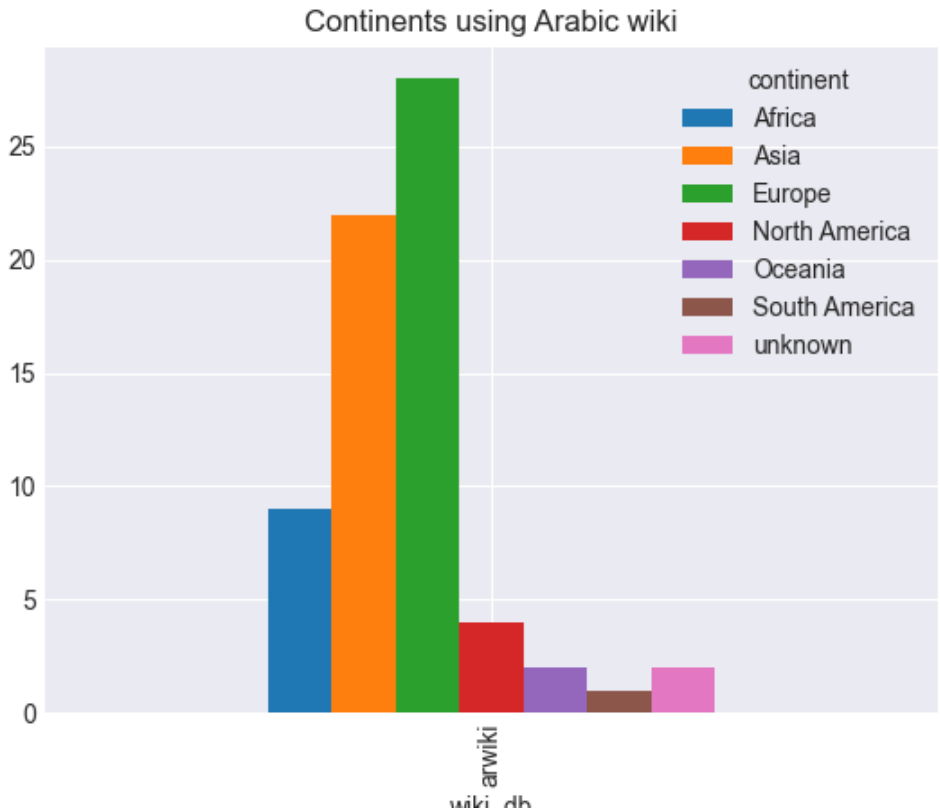


figure shows countries using Arabic Wikipedia

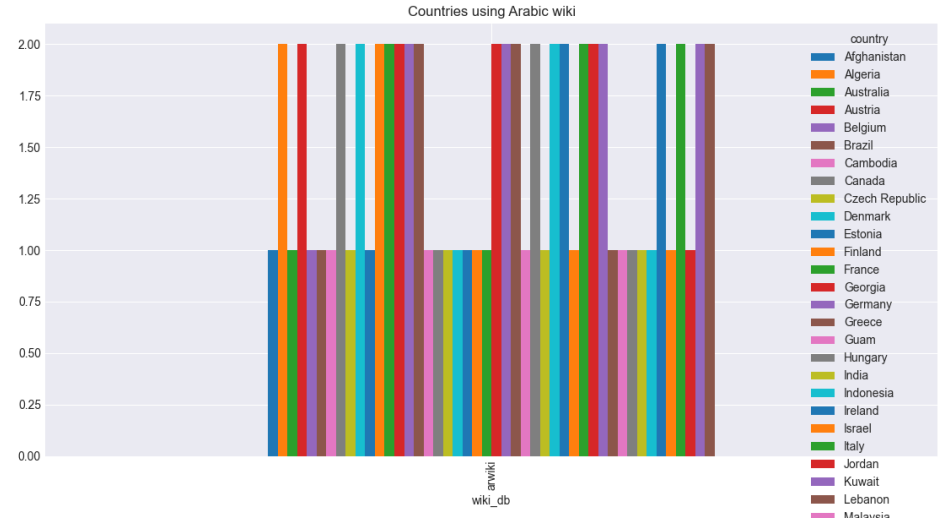


Figure shows lower bound of editors in a group from continent using Arabic Wikipedia

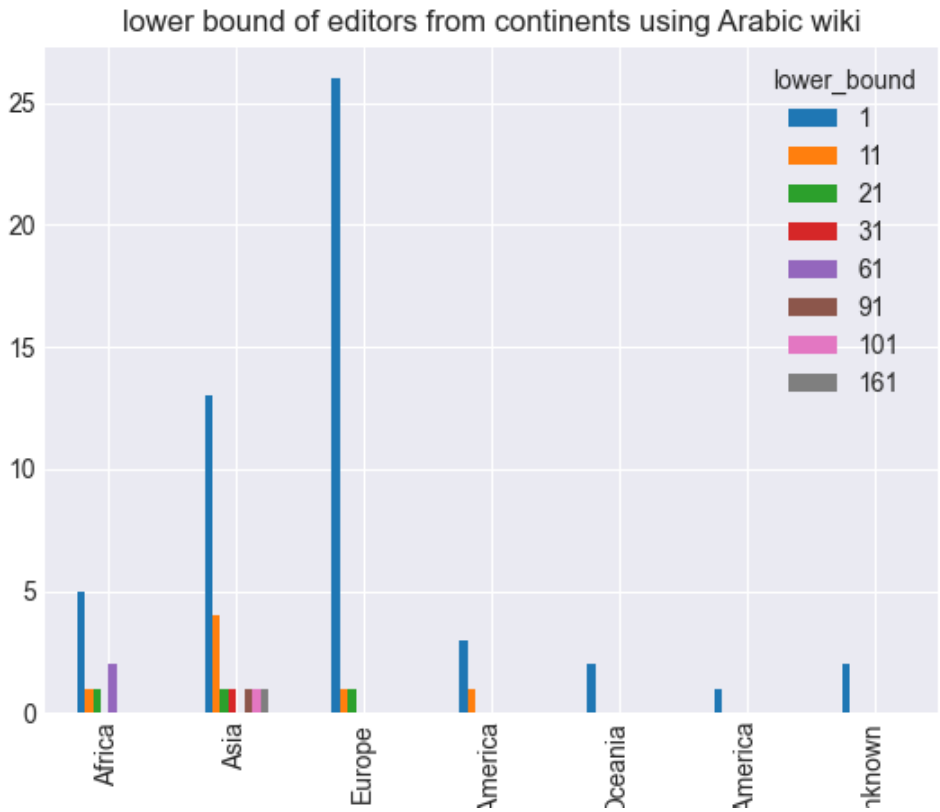
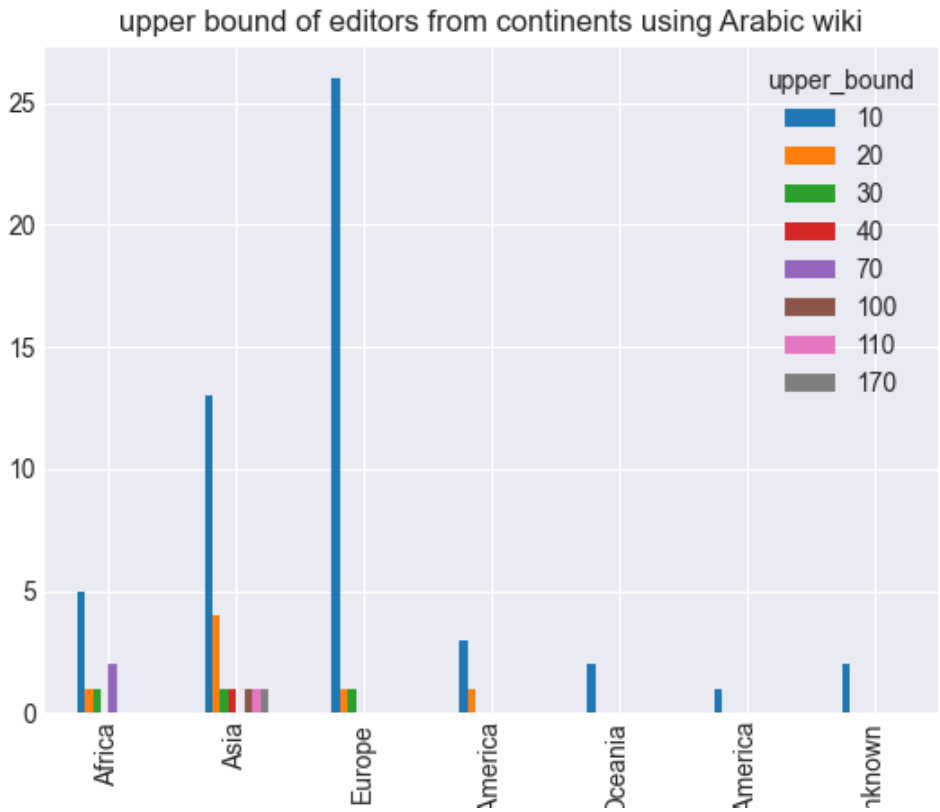


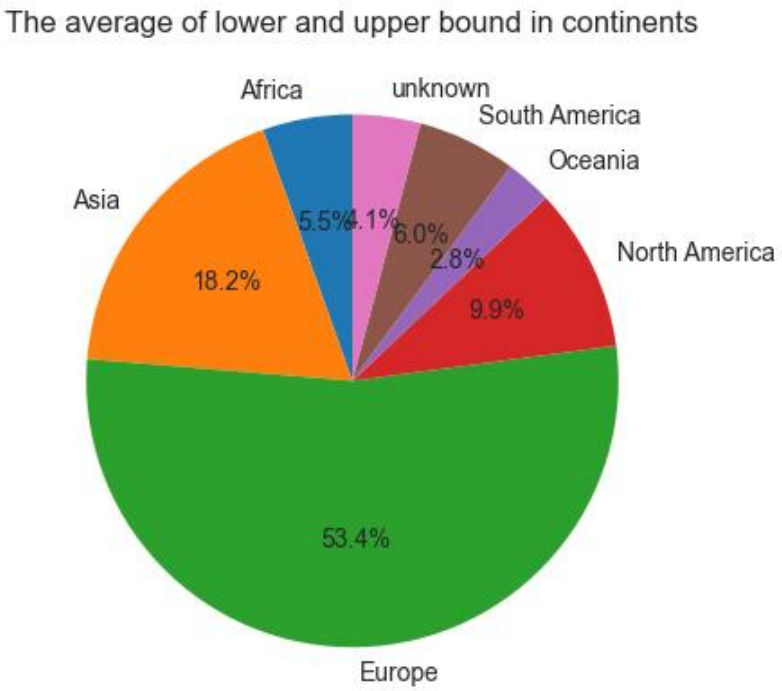
Figure shows upper bound of editors in a group from continent using Arabic Wikipedia




```
import pandas as pd
import matplotlib.pyplot as plt
import pycountry_convert as pc

geoeditor = pd.read_csv('C:\\Users\\GTS\\Downloads\\geoeditors.csv')

def country_to_continent(country_name):
    try:
        country_alpha2 =
pc.country_name_to_country_alpha2(country_name)
        country_continent_code =
pc.country_alpha2_to_continent_code(country_alpha2)
        country_continent_name =
pc.convert_continent_code_to_continent_name(country_continent_code)
        return country_continent_name
    except:
        if country_name == 'unknown': return 'unknown'
        if country_name == 'Caribbean Netherlands': return "Europe"
        if country_name == 'Kosovo': return "Europe"
        if country_name == 'Vatican City': return "Europe"
        if country_name == 'Timor-Leste': return "Asia"
        print(country_name)
        return ''

plt.style.use('seaborn-darkgrid')

geoeditor['continent'] =
```

```
geoeditor['country'].apply(country_to_continent)

arwiki_df=geoeditor[geoeditor['wiki_db'] == 'arwiki']
pd.crosstab(arwiki_df['wiki_db'],arwiki_df['continent']).plot.bar()
plt.title('Continents using Arabic wiki')
plt.show()
pd.crosstab(arwiki_df['wiki_db'],arwiki_df['country']).plot.bar()
plt.title('Countries using Arabic wiki')
plt.show()
pd.crosstab(arwiki_df['continent'],arwiki_df['lower_bound']).plot.bar()
plt.title('lower bound of editors from continents using Arabic wiki')
plt.show()
pd.crosstab(arwiki_df['continent'],arwiki_df['upper_bound']).plot.bar()
plt.title('upper bound of editors from continents using Arabic wiki')
plt.show()
```

The figure shows the average of lower and upper bound of editors in continents

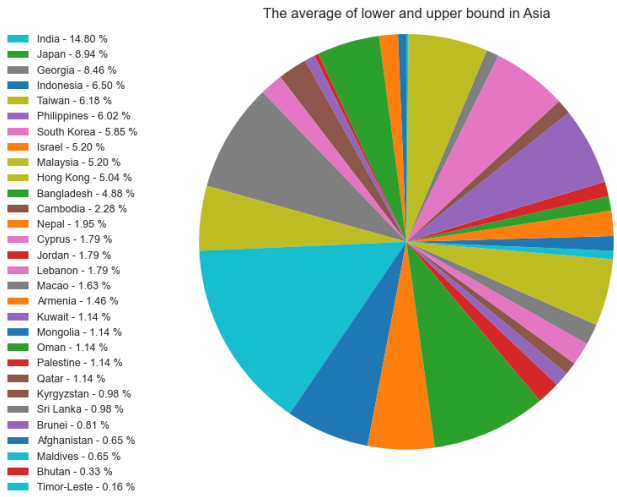


Code:

```
import pandas as pd
import matplotlib.pyplot as plt
import pycountry_convert as pc

geoeditor = pd.read_csv('C:\\Users\\GTS\\Downloads\\geoeditors.csv')

def country_to_continent(country_name):
```

```
    try:
        country_alpha2 =
pc.country_name_to_country_alpha2(country_name)
        country_continent_code =
pc.country_alpha2_to_continent_code(country_alpha2)
        country_continent_name =
pc.convert_continent_code_to_continent_name(country_continent_code)
        return country_continent_name
    except:
        if country_name == 'unknown': return 'unknown'
        if country_name == 'Caribbean Netherlands': return "Europe"
        if country_name == 'Kosovo': return "Europe"
        if country_name == 'Vatican City': return "Europe"
        if country_name == 'Timor-Leste': return "Asia"
        print(country_name)
        return ''

plt.style.use('seaborn-darkgrid')

geoeditor['continent'] =
geoeditor['country'].apply(country_to_continent)

col = geoeditor.loc[: , "lower_bound":"upper_bound"]
geoeditor['bound_mean'] = col.mean(axis=1)
continent = geoeditor.groupby("continent")['bound_mean'].count()
mylabels = ["Africa","Asia", "Europe", "North America",
"Oceania","South America","unknown"]
plt.pie(continent, labels = mylabels, startangle =
90,autopct='%1.1f%%')
plt.title('The average of lower and upper bound in continents')
plt.show()
```

The figure shows the average of lower and upper bound of editors in Asia



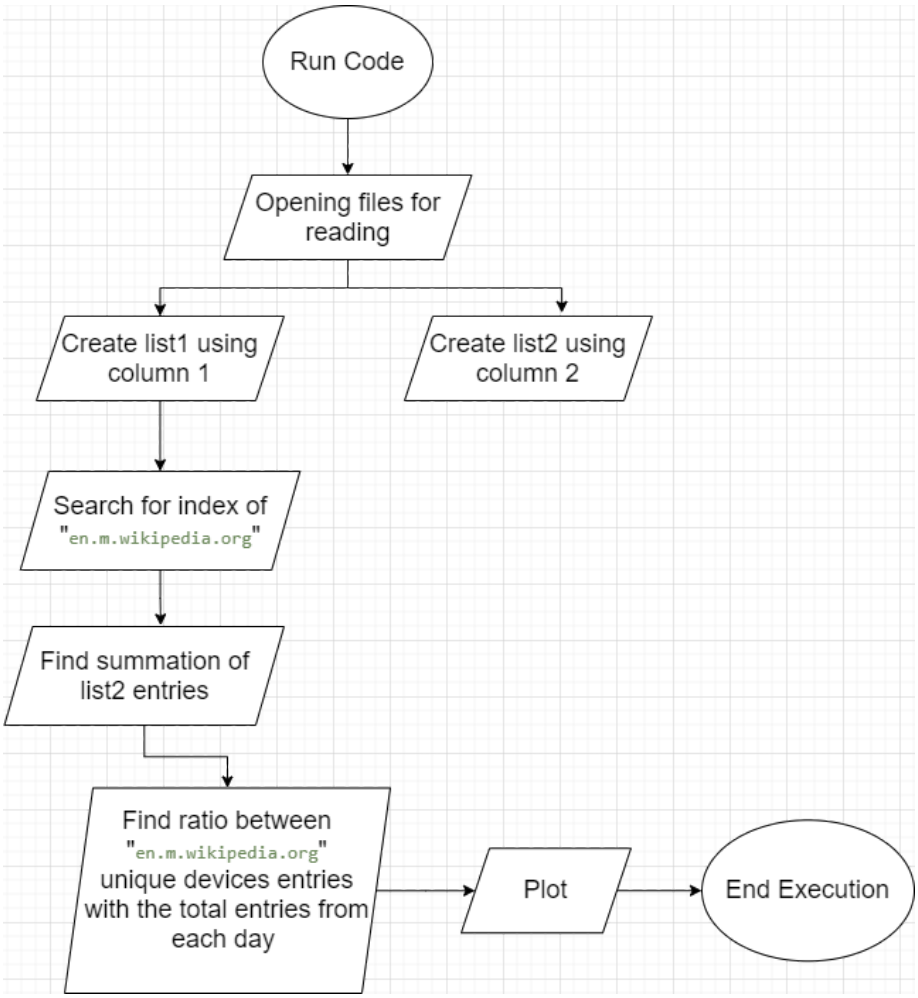
Code:

```
import pandas as pd
import matplotlib.pyplot as plt
import pycountry_convert as pc

geoeditor = pd.read_csv('C:\\Users\\GTS\\Downloads\\geoeditors.csv')

def country_to_continent(country_name):
    try:
        country_alpha2 =
pc.country_name_to_country_alpha2(country_name)
        country_continent_code =
pc.country_alpha2_to_continent_code(country_alpha2)
        country_continent_name =
pc.convert_continent_code_to_continent_name(country_continent_code)
        return country_continent_name
    except:
        if country_name == 'unknown': return 'unknown'
        if country_name == 'Caribbean Netherlands': return "Europe"
        if country_name == 'Kosovo': return "Europe"
        if country_name == 'Vatican City': return "Europe"
        if country_name == 'Timor-Leste': return "Asia"
        print(country_name)
        return ''

plt.style.use('seaborn-darkgrid')

geoeditor['continent'] =
geoeditor['country'].apply(country_to_continent)

asia_df=geoeditor[geoeditor['continent'] == 'Asia']
col = geoeditor.loc[: , "lower_bound":"upper_bound"]
geoeditor['bound_mean'] = col.mean(axis=1)
country= geoeditor.groupby(asia_df["country"])['bound_mean'].count()
mylabels =
["Afghanistan","Armenia","Bangladesh","Bhutan","Brunei","Cambodia","Cyp
rus","Georgia","Hong Kong","India",

"Indonesia","Israel","Japan","Jordan","Kuwait","Kyrgyzstan","Lebanon","
Macao","Malaysia","Maldives","Mongolia",
           "Nepal","Oman","Palestine","Philippines","Qatar","South
Korea","Sri Lanka","Taiwan","Timor-Leste"]
percent = 100.*country/country.sum()
patches, texts = plt.pie(country,  startangle=90, radius=1.2)
labels = ['{0} - {1:1.2f} %'.format(i,j) for i,j in zip(mylabels,
percent)]
sort_legend = True
if sort_legend:
    patches, labels, dummy =  zip(*sorted(zip(patches, labels,
country),
                                          key=lambda mylabels:
mylabels[2],
                                          reverse=True))
plt.legend(patches, labels, loc='upper right', bbox_to_anchor=(-0.1,
1.),
           fontsize=9)
plt.title('The average of lower and upper bound in Asia')
plt.show()
```

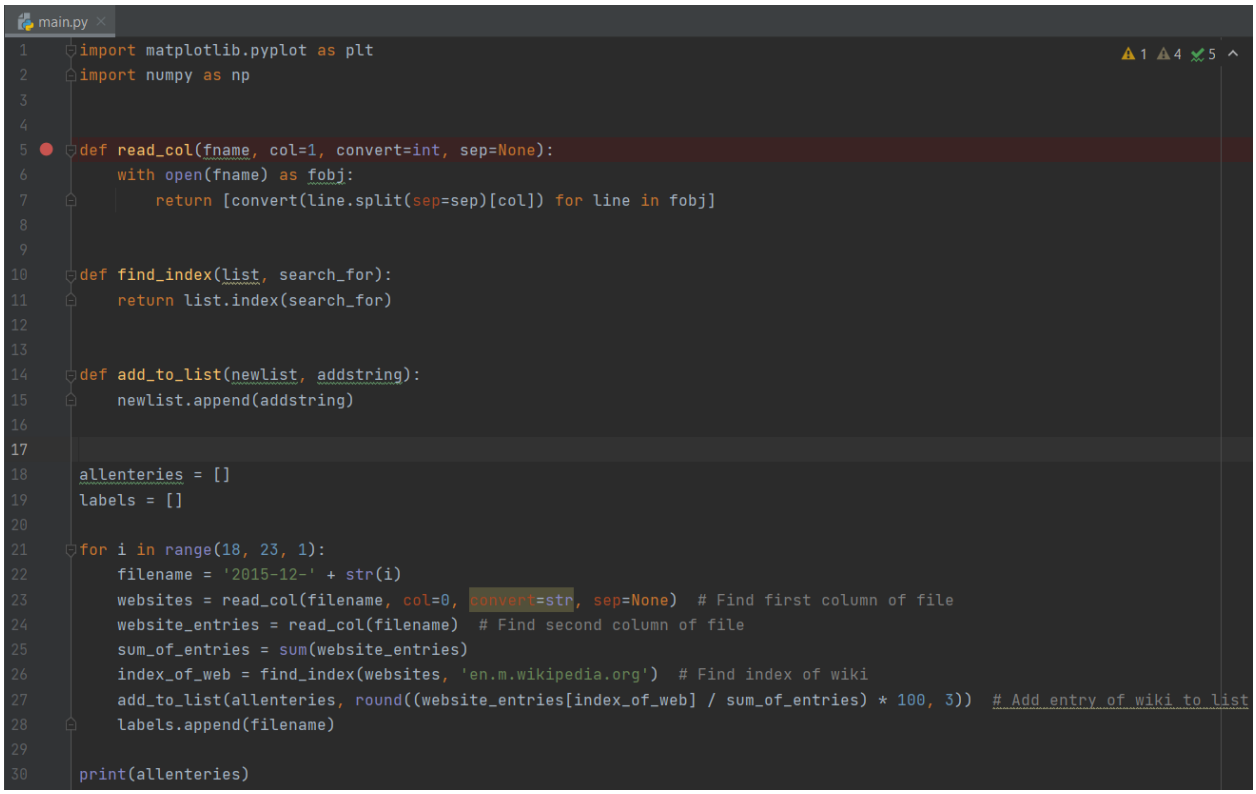
Yussif Abdalla 2180142

My role in this project was analyzing the traces of data related to the unique devices dataset (3rd question). I made 2 plots using 5 files (dates from 2015-12-18 to 2015-12-22).

Here's a flowchart of my program.



The code used for reading the files, creating lists, applying arithmetic operations, and searching.

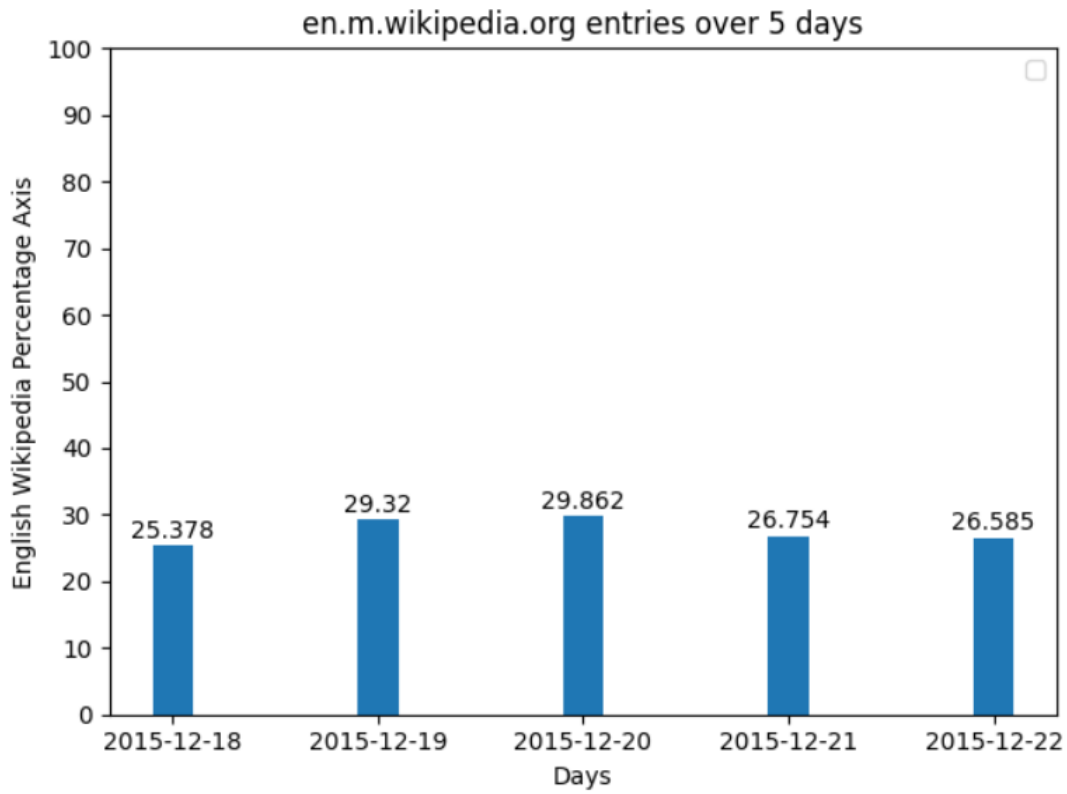


```python
import matplotlib.pyplot as plt
import numpy as np


def read_col(fname, col=1, convert=int, sep=None):
    with open(fname) as fobj:
        return [convert(line.split(sep=sep)[col]) for line in fobj]


def find_index(list, search_for):
    return list.index(search_for)


def add_to_list(newlist, addstring):
    newlist.append(addstring)


allenteries = []
labels = []

for i in range(18, 23, 1):
    filename = '2015-12-' + str(i)
    websites = read_col(filename, col=0, convert=str, sep=None)  # Find first column of file
    website_entries = read_col(filename)  # Find second column of file
    sum_of_entries = sum(website_entries)
    index_of_web = find_index(websites, 'en.m.wikipedia.org')  # Find index of wiki
    add_to_list(allenteries, round((website_entries[index_of_web] / sum_of_entries) * 100, 3))  # Add entry of wiki to list
    labels.append(filename)

print(allenteries)
```

The codes used for graphing the plots, and the final result.

```
print(allenteries)
x = np.arange(len(labels))
width = 0.20

fig, ax = plt.subplots()
rects1 = ax.bar(x, allenteries, width)
ax.set_ylabel('English Wikipedia Percentage Axis')
ax.set_xlabel('Days')
ax.set_title('en.m.wikipedia.org entries over 5 days')
ax.set_xticks(x)
ax.set_xticklabels(labels)
ax.set_yticks(np.arange(0, 101, 10))
ax.legend()

ax.bar_label(rects1, padding=1)


fig.tight_layout()

plt.show()

```

Plot shown on Pycharm scientific mode



Percentage of “en.m.wikipedia.org” entries, and other entries for the five days

```
labels = 'en.m.wikipedia.org', 'Other'
wiki_ratio = (sum_of_entries/sum_all_files) * 100
other_ratio = 100 - wiki_ratio
sizes = [wiki_ratio, other_ratio]
explode = (0.1, 0)  # only "explode" the 2nd slice (i.e. 'Hogs')

fig1, ax1 = plt.subplots()
ax1.pie(sizes, explode=explode, labels=labels, autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

plt.show()
```