Homework12_Surista

November 26, 2020

1 Homework 12

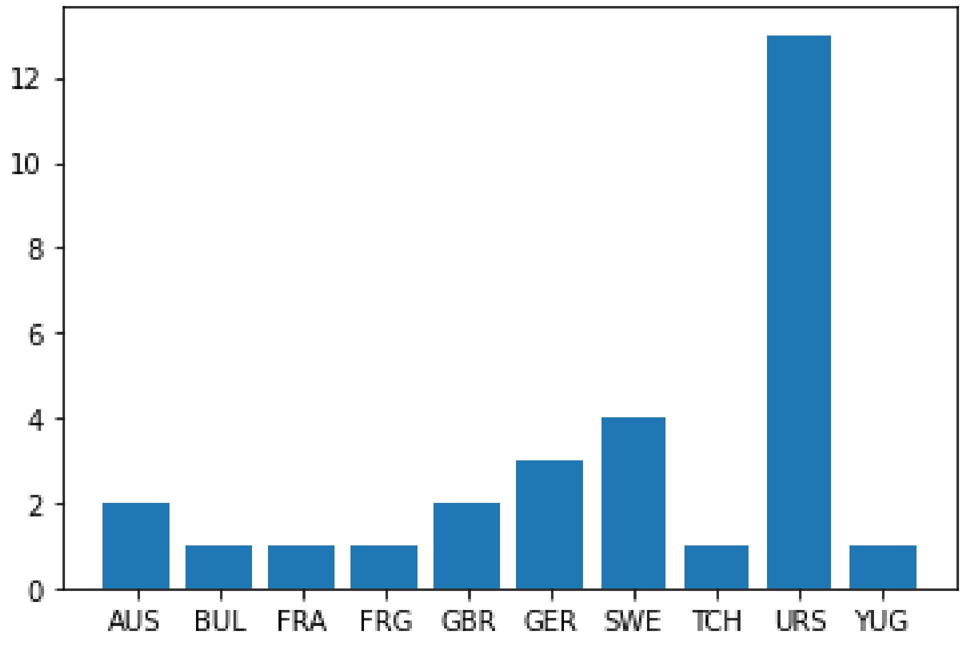
1.1 Due 4PM Nov 30, 2020

1.2 Problem 1: Women's 800 Meter

Which countries have done best at the Women's 800 Meter?

Gather the data from the World Records CSV, use a Dictionary to count the records, and create a bar chart showing the relative number of records per country. Sort the countries alphabetically, and make sure we can read the country names.

[98]:
```python
import pandas as pd
import matplotlib.pyplot as plt
filename = "WorldRecords.csv"

# read in data
df = pd.read_csv(filename)

# filter to women's 800m
df2 = df[(df['Event'] =='Womens 800m')]

# create dict to count records
nats = {}
for index, row in df2.iterrows():
    if row['Nationality'] not in nats:
        nats[row['Nationality']] = 1
    else:
        nats[row['Nationality']] += 1

# create & draw chart
nats = sorted(nats.items())
x = [k[0] for k in nats]
y = [k[1] for k in nats]

plt.bar(x,y)
plt.show()
```



1.3 Problem 2: Regular Expressions

We have used Beautiful Soup to scrape a website.

Let's see what we can do with just urlib and Regular Expressions

Take the DCE website, and find all the links. (Be sure to compare notes with Beautiful Soup)

[119]:
```
import urllib.request
import string
import re

def find_links(url):
    """Returns the first URL and link txt on page"""

    # read in url text
    with urllib.request.urlopen(url) as f:
        text = f.read().decode('utf-8')

    re_links = re.findall('<a\s+.*/a>', text)
    return re_links
```

1.4 Unit Test

[120]:
```
website = 'https://www.extension.harvard.edu'
```

[121]:
```python
results = find_links(website)
print("Number of links:",len(results),"\n")
for link in results:
   print(link)
```

```
Number of links: 59

<a href="#main-menu" class="skip">Jump to navigation</a>
<a href="#main-content" class="skip">Skip to Main Content</a>
<a class="topbar__link" href="https://www.harvard.edu">Harvard.edu</a>
<a href="https://www.extension.harvard.edu">Harvard Extension School</a>
<a href="https://www.summer.harvard.edu">Harvard Summer School</a>
<a
              href="https://www.extension.harvard.edu/professional-
development">Professional Development</a>
<a href="https://www.extension.harvard.edu/hilr">Learning In Retirement</a>
<a href="https://alumni.extension.harvard.edu/">Extension Alumni Association</a>
<a class="header__mobile-menu ir i-hamburger" data-grunticon-embed
href="">Menu</a>
<a href="/academics">Academics</a>
<a href="/registration-admissions" title="Registration &amp;
Admissions">Registration &amp; Admissions</a>
<a href="/resources-policies">Resources &amp; Policies</a>
<a href="https://blog.dce.harvard.edu/extension" title="">Blog</a>
<a href="/request-information" title="">Get Info</a>
<a href="/about-us" title="">About</a>
<a href="/academic-calendar" title="">Calendar</a>
<a href="/completing-your-degree" title="">For Degree Candidates</a>
<a href="/academics/online-campus-courses" title="Link to courses">Courses</a>
<a href="/faculty-directory" title="">Faculty Directory</a>
<a href="https://www.extension.harvard.edu/login" title="">LOGIN</a>
<a id="main-content" tabindex="-1"></a>
<a href=https://www.extension.harvard.edu/covid-19-updates>latest COVID-19 news
from Harvard Extension School</a>
<a class="i-right-arrow" href="/academics/graduate-certificates">Graduate
Certificates</a>
<a class="i-right-arrow" href="/academics/graduate-degrees">Master&rsquo;s
Degrees</a>
<a class="i-right-arrow" href="/academics/academic-gap-year">Academic Gap
Year</a>
<a class="i-right-arrow" href="/academics/bachelor-liberal-arts-
degree">Bachelor&rsquo;s Degree</a>
<a class="i-right-arrow" href="/academics/undergraduate-
certificates">Undergraduate Certificates</a>
<a class="i-right-arrow" href="/joint-undergraduate-graduate-program">Joint
Undergraduate &amp; Graduate Programs</a>
```

```
<a class="i-right-arrow" href="/course-catalog">Course Catalog</a>
<a class="i-right-arrow" href="/course-
catalog/courses?subjects=Medical%20Sciences">Medical Sciences Courses</a>
<a class="i-right-arrow" href="/academics/premedical-program">Premedical
Program</a>
<a class="i-right-arrow" href="https://www.extension.harvard.edu/professional-
development">Noncredit Professional Development Programs</a>
<a class="i-right-arrow" href="https://www.extension.harvard.edu/hilr">Learn
about the program</a>
<a class="button-link" href="/about-us/why-hes">Find out why</a>
<a class="student-name h3" href="/about-us/peter-thielen">Peter Thielen</a>
<a class="student-name h3" href="/about-us/renee-m-greene">Renee M. Greene</a>
<a class="student-name h3" href="/about-us/diane-smith">Diane Smith</a>
<a class="button btn-outline-primary" href="/about-us/student-stories">Meet
Other Alumni &amp; Students <i class="far fa-long-arrow-alt-right"></i></a>
<a class="h3" href="https://harvardmagazine.com/2020/10/calling-
the-2020-election" target="_blank">Calling the 2020 Election</a>
<a class="h3" href="https://www.thecrimson.com/article/2020/9/18/extension-
school-new-programs/" target="_blank">Harvard Extension School Unveils New
Academic Gap Year, Undergraduate Certificate Programs</a>
<a class="h3" href="https://www.educationdive.com/news/how-colleges-with-hybrid-
instruction-this-fall-can-support-online-students/582141/" target="_blank">How
colleges with hybrid instruction this fall can support online students</a>
<a class="h3" href="/about-us/press-announcements/michael-fabiano-joins-haa-
board-directors" >Michael Fabiano Joins HAA Board of Directors</a>
<a class="h3" href="https://blog.dce.harvard.edu/extension/announcing-new-
graduate-and-undergraduate-certificates-online-courses-for-2020-21"
target="_blank">What&#039;s New for 2020-21</a>
<a class="h3" href="https://www.fas.harvard.edu/news/new-dean-division-
continuing-education" target="_blank">New Dean of the Division of Continuing
Education</a>
<a class="button-link"
href="https://blog.dce.harvard.edu/extension/6-strategies-for-staying-
productive-during-the-covid-19-crisis">Read the blog post</a>
<a class="button-link" href="https://www.extension.harvard.edu/course-
catalog">Course Catalog</a>
<a class="button-link" href="https://www.extension.harvard.edu/professional-
development/programs/building-organizational-cultures-framework-leaders-
online">Learn More</a>
<a href="/contact-us" title="" class="menu__link">Contact Us</a>
<a href="/forms" title="" class="menu__link">Forms</a>
<a href="/website-archives" title="" class="menu__link">Archives</a>
<a href="https://twitter.com/HarvardEXT" title="" class="menu__link i-social-
twitter">Twitter</a>
<a href="https://www.facebook.com/HarvardExtension" title="" class="menu__link
i-social-facebook">Facebook</a>
<a href="https://www.youtube.com/user/HarvardExtension" title=""
class="menu__link i-social-youtube">YouTube</a>
```

```
<a href="https://www.instagram.com/harvardextension/" title="" class="menu__link
i-social-instagram">Instagram</a>
<a class="menu__link" href="/privacy-policy" title="">Privacy</a>
<a class="menu__link" href="/resources-policies/accessibility-services-office-
aso" title="">Accessibility</a>
<a class="menu__link" href="/resources-policies/resources/rights-regulations"
title="">Rights &amp; Regulations</a>
<a class="menu__link" href="https://accessibility.huit.harvard.edu/digital-
accessibility-policy" title="">Digital Accessibility Policy</a>
<a class="menu__link ot-sdk-show-settings" href="#" title="">Cookie Settings</a>
```

[122]:
```
import requests
from bs4 import BeautifulSoup

"prettify print the html of a given url"

url = "https://www.extension.harvard.edu"
html_content = requests.get(url).text
soup = BeautifulSoup(html_content, 'html.parser')
pretty_soup = soup.prettify()

links = soup.find_all("a")


# print("link:", links[0])
# print("results:", results[0])

print("Number of links:",len(links),"\n")
for x in links:
    print(x)
```

```
Number of links: 62

<a class="skip" href="#main-menu">Jump to navigation</a>
<a class="skip" href="#main-content">Skip to Main Content</a>
<a class="topbar__logo i-harvard-logo ir" href="https://dce.harvard.edu"
target="_blank">
        Harvard Division of Continuing Education
      </a>
<a class="topbar__link" href="https://www.harvard.edu">Harvard.edu</a>
<a href="https://www.extension.harvard.edu">Harvard Extension School</a>
<a href="https://www.summer.harvard.edu">Harvard Summer School</a>
<a href="https://www.extension.harvard.edu/professional-
development">Professional Development</a>
<a href="https://www.extension.harvard.edu/hilr">Learning In Retirement</a>
<a href="https://alumni.extension.harvard.edu/">Extension Alumni Association</a>
<a class="header__mobile-menu ir i-hamburger" data-grunticon-embed=""
href="">Menu</a>
```

```
<a class="header__logo i-hes-logo" href="/" id="logo" rel="home" title="Home">
<noscript><img alt="Home" class="header__logo-image" src="https://www.extension.
harvard.edu/sites/extension.harvard.edu/themes/extension/logo.png"/></noscript>
</a>
<a class="header__site-link" href="/" rel="home" title="Home"><span>Harvard
Extension School</span></a>
<a href="/academics">Academics</a>
<a href="/registration-admissions" title="Registration &amp;
Admissions">Registration &amp; Admissions</a>
<a href="/resources-policies">Resources &amp; Policies</a>
<a href="https://blog.dce.harvard.edu/extension" title="">Blog</a>
<a href="/request-information" title="">Get Info</a>
<a href="/about-us" title="">About</a>
<a href="/academic-calendar" title="">Calendar</a>
<a href="/completing-your-degree" title="">For Degree Candidates</a>
<a href="/academics/online-campus-courses" title="Link to courses">Courses</a>
<a href="/faculty-directory" title="">Faculty Directory</a>
<a href="https://www.extension.harvard.edu/login" title="">LOGIN</a>
<a id="main-content" tabindex="-1"></a>
<a href="https://www.extension.harvard.edu/covid-19-updates">latest COVID-19
news from Harvard Extension School</a>
<a class="i-right-arrow" href="/academics/graduate-certificates">Graduate
Certificates</a>
<a class="i-right-arrow" href="/academics/graduate-degrees">Master’s Degrees</a>
<a class="i-right-arrow" href="/academics/academic-gap-year">Academic Gap
Year</a>
<a class="i-right-arrow" href="/academics/bachelor-liberal-arts-
degree">Bachelor’s Degree</a>
<a class="i-right-arrow" href="/academics/undergraduate-
certificates">Undergraduate Certificates</a>
<a class="i-right-arrow" href="/joint-undergraduate-graduate-program">Joint
Undergraduate &amp; Graduate Programs</a>
<a class="i-right-arrow" href="/course-catalog">Course Catalog</a>
<a class="i-right-arrow" href="/course-
catalog/courses?subjects=Medical%20Sciences">Medical Sciences Courses</a>
<a class="i-right-arrow" href="/academics/premedical-program">Premedical
Program</a>
<a class="i-right-arrow" href="https://www.extension.harvard.edu/professional-
development">Noncredit Professional Development Programs</a>
<a class="i-right-arrow" href="https://www.extension.harvard.edu/hilr">Learn
about the program</a>
<a class="button-link" href="/about-us/why-hes">Find out why</a>
<a class="student-name h3" href="/about-us/peter-thielen">Peter Thielen</a>
<a class="student-name h3" href="/about-us/renee-m-greene">Renee M. Greene</a>
<a class="student-name h3" href="/about-us/diane-smith">Diane Smith</a>
<a class="button btn-outline-primary" href="/about-us/student-stories">Meet
Other Alumni &amp; Students <i class="far fa-long-arrow-alt-right"></i></a>
<a class="h3" href="https://harvardmagazine.com/2020/10/calling-
```

```
the-2020-election" target="_blank">Calling the 2020 Election</a>
<a class="h3" href="https://www.thecrimson.com/article/2020/9/18/extension-
school-new-programs/" target="_blank">Harvard Extension School Unveils New
Academic Gap Year, Undergraduate Certificate Programs</a>
<a class="h3" href="https://www.educationdive.com/news/how-colleges-with-hybrid-
instruction-this-fall-can-support-online-students/582141/" target="_blank">How
colleges with hybrid instruction this fall can support online students</a>
<a class="h3" href="/about-us/press-announcements/michael-fabiano-joins-haa-
board-directors">Michael Fabiano Joins HAA Board of Directors</a>
<a class="h3" href="https://blog.dce.harvard.edu/extension/announcing-new-
graduate-and-undergraduate-certificates-online-courses-for-2020-21"
target="_blank">What's New for 2020-21</a>
<a class="h3" href="https://www.fas.harvard.edu/news/new-dean-division-
continuing-education" target="_blank">New Dean of the Division of Continuing
Education</a>
<a class="button-link"
href="https://blog.dce.harvard.edu/extension/6-strategies-for-staying-
productive-during-the-covid-19-crisis">Read the blog post</a>
<a class="button-link" href="https://www.extension.harvard.edu/course-
catalog">Course Catalog</a>
<a class="button-link" href="https://www.extension.harvard.edu/professional-
development/programs/building-organizational-cultures-framework-leaders-
online">Learn More</a>
<a class="menu__link" href="/contact-us" title="">Contact Us</a>
<a class="menu__link" href="/forms" title="">Forms</a>
<a class="menu__link" href="/website-archives" title="">Archives</a>
<a class="menu__link i-social-twitter" href="https://twitter.com/HarvardEXT"
title="">Twitter</a>
<a class="menu__link i-social-facebook"
href="https://www.facebook.com/HarvardExtension" title="">Facebook</a>
<a class="menu__link i-social-youtube"
href="https://www.youtube.com/user/HarvardExtension" title="">YouTube</a>
<a class="menu__link i-social-instagram"
href="https://www.instagram.com/harvardextension/" title="">Instagram</a>
<a class="menu__link" href="/privacy-policy" title="">Privacy</a>
<a class="menu__link" href="/resources-policies/accessibility-services-office-
aso" title="">Accessibility</a>
<a class="menu__link" href="/resources-policies/resources/rights-regulations"
title="">Rights &amp; Regulations</a>
<a class="menu__link" href="https://accessibility.huit.harvard.edu/digital-
accessibility-policy" title="">Digital Accessibility Policy</a>
<a class="menu__link ot-sdk-show-settings" href="#" title="">Cookie Settings</a>
```

[123]:
```
# I do not get the same number of links. I miss three using RE: the Harvard␣
↪Division of Continuing Education
# link, the second Harvard Extension School link, and the HES logo that links␣
↪back to the home page.
```

```
# I spent hours reading up on this, and but cannot figure out a reliable way to␣
↪use RE
# to get the links I first miss, without missing other links I didn't miss the␣
↪first time.

# It's a bit baffling. All three of the links I miss using RE span multiple␣
↪lines.
# but so does the Professional Development link, that I'm able to catch.

# I suspect the moral of the story is - don't use RE for url scraping, use␣
↪tools specifically built
# for that, like Beautiful Soup...
```

1.5 Compare your program with the results from Beautiful Soup

Do you get the same number of links? If not:

1) How many do you miss?
2) Can you explain why you miss them?
3) Can you fix it?

1.6 Problem 3: File Name Generator

Write a Generator that takes a directory, a file extension, and, optionally, a file size, and then yields a stream of tuples, (path, filename) so that path/filename is a legal path to a file that meets the conditions.

Use os.walk(dir) to create a generator that gives all files and directories below dir. Call this generator, and yield files (not directories) with the right extension and a file size greater than the given size.

We have three unit tests: demonstrate that you can walk recursivly through two or more directories, and that you can filter by file extension and filter by extension and by size.

[124]:
```
import os

def find_files_gen(path, filename, filesize=0):
    "yields specific files larger than given size (default size is 0)"

    matches = []
    for root,dir, files in os.walk(path):
        for f in files:
            path = os.path.join(root, f)
            size = os.stat(path).st_size
            if filename in f and size > filesize:
                yield(root, f)
```

```python
gen = find_files_gen('..', 'py')
print(next(gen))
print(next(gen))
print(next(gen))
print(next(gen))
print(next(gen))
print(next(gen))
```

```
('..\\.ipynb_checkpoints', 'Homework12_Surista-checkpoint.ipynb')
('..\\Day01', 'Day1.ipynb')
('..\\Day01', 'day1.py')
('..\\Day01', 'Day_01.py')
('..\\Day01', 'Day_01a.py')
('..\\Day01', 'E7 Day1.ipynb')
```

1.7 Unit Test

[104]:
```python
# Show recursive search.  Make sure we can see at least two directories of files
gen = find_files_gen('..', '.py')

for path, filename in gen:
    print(path, filename)
```

```
..\Day01 day1.py
..\Day01 Day_01.py
..\Day01 Day_01a.py
..\Day02 bamba.py
..\Day02 contract.py
..\Day02 day2.py
..\Day02 double.py
..\Day02 example.py
..\Day02 greet.py
..\Day02 greet2.py
..\Day02 kc.py
..\Day02 plat.py
..\Day02 script.py
..\Day02 sum.py
..\Day02 system.py
..\Day03 binary_search.py
..\Day03 binary_search_test.py
..\Day03 freestyle.py
..\Day03 grid.py
..\Day03 honeycomb.py
..\Day03 paint.py
..\Day03 pentagram.py
..\Day03 turtle2.py
..\Day03 turtlenew.py
```

```
..\Day03 TurtleOne.py
..\Day03 TurtleTwo.py
..\Day03 TurtleZero.py
..\Day03 turtle_random_shapes.py
..\Day03 yinyang.py
..\Day04 cli.py
..\Day04 cross.py
..\Day04 file .py
..\Day04 file3.py
..\Day04 hanoi.py
..\Day04 Koch.py
..\Day04 mycopy.py
..\Day05 od.py
..\Day05 search.py
..\Day05 stripComments.py
..\Day05 walk.py
..\Day05 which.py
..\Day06 FailedBanks.py
..\Day06 game.py
..\Day06 homework_copyright.py
..\Day06 probems_day_6.py
..\Day06 reversalDict.py
..\Day06 reversals.py
..\Day06 reversalsShortList.py
..\Day06 save_url.py
..\Day06 walktest.py
..\Day07 day_7_lecture.py
..\Day07 day_8_lecture.py
..\Day07 doublesPick.py
..\Day07 FailedBanks.py
..\Day07 wines_new.py
..\Day08 bubble_sort.py
..\Day08 dates.py
..\Day08 dna_complement.py
..\Day08 find_first_link.py
..\Day08 find_y.py
..\Day08 Koch.py
..\Day08 koch_snowflake.py
..\Day08 odds_n_evens.py
..\Day08 Point1.py
..\Day08 Point2.py
..\Day08 Time1.py
..\Day08 Time2.py
..\Day08 time_after_time.py
..\Day09 collatz.py
..\Day09 next_month.py
..\Day09 Phone.py
..\Day09 point.py
```

```
..\Day10 anagrams.py
..\Day10 beautiful_Jam.py
..\Day10 plot_histogram.py
..\Day10 wordLengths.py
..\Day11 anagram_d11.py
..\Day11 global_mile.py
..\Day11 people.py
..\Day11 sudoku.py
..\Day12 binary_fasta.py
..\Day12 defaultDict.py
..\Day12 file_name.py
..\Day12 parkers_double_table.py
..\Day12 People.py
..\Day12 records_w800.py
..\Day12 re_urls.py
```

[105]:
```
# Show all notebooks in this directory
gen = find_files_gen('..', '.ipynb')

for path, filename in gen:
    print(path, filename)
```

```
..\.ipynb_checkpoints Homework12_Surista-checkpoint.ipynb
..\Day01 Day1.ipynb
..\Day01 E7 Day1.ipynb
..\Day01 Homework1.ipynb
..\Day01 Homework_01.ipynb
..\Day01 HW1Share.ipynb
..\Day01 HW1Share_solutions.ipynb
..\Day02 Day2.ipynb
..\Day02 Homework2.ipynb
..\Day02 HW2Share.ipynb
..\Day03 Day3.ipynb
..\Day03 Debug.ipynb
..\Day03 HW2Share_sol.ipynb
..\Day03 HW3Share.ipynb
..\Day03\.ipynb_checkpoints HW2Share_sol-checkpoint.ipynb
..\Day04 Day4 (1).ipynb
..\Day04 Homework4.ipynb
..\Day04 HW4Share.ipynb
..\Day04 PyCharmVEnv.ipynb
..\Day04\.ipynb_checkpoints Day4 (1)-checkpoint.ipynb
..\Day04\.ipynb_checkpoints Day4-checkpoint.ipynb
..\Day05 Day5.ipynb
..\Day05 homework5.ipynb
..\Day05 HW4Share (1).ipynb
..\Day05 HW5Share.ipynb
..\Day05 Solutions4 (1).ipynb
```

```
..\Day05\.ipynb_checkpoints Day5-checkpoint.ipynb
..\Day06 Day6.ipynb
..\Day06 Homework6.ipynb
..\Day06 HW5Share (2).ipynb
..\Day06\.ipynb_checkpoints Homework6_SUrista-checkpoint.ipynb
..\Day06\.ipynb_checkpoints Homework6_SUrista_new-checkpoint.ipynb
..\Day07 Day7.ipynb
..\Day07 Homework7_SUrista.ipynb
..\Day07 Solution6.ipynb
..\Day07\.ipynb_checkpoints Homework7_SUrista-checkpoint.ipynb
..\Day07\.ipynb_checkpoints Solution6-checkpoint.ipynb
..\Day08 Day8.ipynb
..\Day08 Homework8 (1).ipynb
..\Day08 Homework8.ipynb
..\Day08 Homework8FirstDraft.ipynb
..\Day08 While.ipynb
..\Day08\.ipynb_checkpoints Day8-checkpoint.ipynb
..\Day08\.ipynb_checkpoints Homework8 (1)-checkpoint.ipynb
..\Day08\.ipynb_checkpoints Homework8-checkpoint.ipynb
..\Day08\.ipynb_checkpoints Homework8FirstDraft-checkpoint.ipynb
..\Day09 Assignment_9.ipynb
..\Day09 Assignment_9_original.ipynb
..\Day09 Homework8Solutions.ipynb
..\Day09 HW9Share.ipynb
..\Day09 Iterator (1).ipynb
..\Day09\.ipynb_checkpoints Assignment_9 (1)-checkpoint.ipynb
..\Day09\.ipynb_checkpoints Assignment_9 (2)-checkpoint.ipynb
..\Day09\.ipynb_checkpoints Assignment_9-checkpoint.ipynb
..\Day09\.ipynb_checkpoints Day10-checkpoint.ipynb
..\Day09\.ipynb_checkpoints Homework8Solutions-checkpoint.ipynb
..\Day09\.ipynb_checkpoints HW9Share-checkpoint.ipynb
..\Day09\.ipynb_checkpoints Iterator-checkpoint.ipynb
..\Day09\.ipynb_checkpoints SourceControl-checkpoint.ipynb
..\Day10 BeautifulSoup-Lena.ipynb
..\Day10 Homework10.ipynb
..\Day10 Homework10_original.ipynb
..\Day10 Homework10_S_Urista.ipynb
..\Day10 Scipy Tutorial.ipynb
..\Day10\.ipynb_checkpoints 2016_US_County_Level_Presidential_Results-
checkpoint.ipynb
..\Day10\.ipynb_checkpoints Homework10 (3)-checkpoint.ipynb
..\Day10\.ipynb_checkpoints Homework10 (4)-checkpoint.ipynb
..\Day10\.ipynb_checkpoints Homework10-checkpoint.ipynb
..\Day10\.ipynb_checkpoints Homework10_original-checkpoint.ipynb
..\Day11 Homework11 _original.ipynb
..\Day11 Homework11_SUrista.ipynb
..\Day11\.ipynb_checkpoints Homework11-checkpoint.ipynb
..\Day11\.ipynb_checkpoints Homework11SUrista-checkpoint.ipynb
```

```
..\Day11\.ipynb_checkpoints Homework11_original-checkpoint.ipynb
..\Day11\.ipynb_checkpoints Homework11_SUrista-checkpoint.ipynb
..\Day12 Anagrams.ipynb
..\Day12 Homework12_original.ipynb
..\Day12 Homework12_Surista.ipynb
..\Day12 Pickle.ipynb
..\Day12 Repeats.ipynb
..\Day12\.ipynb_checkpoints Homework12 (1)-checkpoint.ipynb
..\Day12\.ipynb_checkpoints Homework12 (5)-checkpoint.ipynb
..\Day12\.ipynb_checkpoints Homework12 (6)-checkpoint.ipynb
..\Day12\.ipynb_checkpoints Homework12_original-checkpoint.ipynb
..\Day12\.ipynb_checkpoints Homework12_Surista-checkpoint.ipynb
..\Day12\.ipynb_checkpoints Repeats-checkpoint.ipynb
```

[106]:

```python
# Show all notebooks in this directory with at least 50K bytes
gen = find_files_gen('..', '.ipynb', 50000)

for path, filename in gen:
    print(path, filename)
```

```
..\Day04 Day4 (1).ipynb
..\Day04\.ipynb_checkpoints Day4 (1)-checkpoint.ipynb
..\Day04\.ipynb_checkpoints Day4-checkpoint.ipynb
..\Day05 Day5.ipynb
..\Day06 Day6.ipynb
..\Day07 Day7.ipynb
..\Day09 Iterator (1).ipynb
..\Day09\.ipynb_checkpoints Iterator-checkpoint.ipynb
..\Day10 BeautifulSoup-Lena.ipynb
..\Day10\.ipynb_checkpoints 2016_US_County_Level_Presidential_Results-
checkpoint.ipynb
..\Day12 Homework12_Surista.ipynb
..\Day12\.ipynb_checkpoints Homework12_Surista-checkpoint.ipynb
```

1.8 Problem 4: Sorting Employees

We wish to take an unordered list of Employees, and get a list sorted by Company and Id.

Everyone who works at 'Springfield Department of Motor Vehicles' should be in one group. Everyone who works at 'Springfield Nuclear Power' would be in another group, later in the list, and everyone who works from the Mafia would be in a group earlier in the list. Within each group, we want to see the low ID numbers before this high ones.

For this problem, we do not want you to write a sorting program. You will use Python's sort. You just need to define the magic method dunder lt(), less than, for the class Employee.

Once you have defined dunder lt(), calling Python's sorted() on a list of Employees will return a sorted list.

1.8.1 Add to the cell below

[107]:
```python
class Person:

    def __init__(self, first, last):
        self.firstname = first.capitalize()
        self.lastname = last.capitalize()

    def __str__(self):
        return self.firstname + " " + self.lastname


class Employee(Person):

    def __init__(self, first, last, company, id):
        # Call Superclass to set common information
        super().__init__(first, last)
        self.id = id
        self.company = company

    def __str__(self):
        # Call Superclass to dispaly common information
        return super().__str__() + ", " + str(self.id) + ' at ' + self.company

    def __lt__(self, other):
        "Is self less than other?"

        if not isinstance(other, Employee):
            return False
        return (self.company, self.id) < (other.company, other.id)
```

1.9 Unit Test

[108]:
```python
lst = [
    Employee('Homer', 'Simpson', 'Springfield Nuclear Power', 1005),
    Employee('Barney', 'Gumble', 'Plow King', 1),
    Employee('Clancy', 'Wiggum', 'Police Department', 1),
    Employee('Edna', 'Krabapple', 'Springfield Elementary School', 39),
    Employee('Seymour', 'Skinner', 'Springfield Elementary School', 1),
    Employee('Charles', 'Burns', 'Springfield Nuclear Power', 1),
    Employee('Waylon', 'Smithers', 'Springfield Nuclear Power', 2),
    Employee('Patty', 'Bouvier', 'Springfield Department of Motor Vehicles',␣
 ↪39),
    Employee('Selma', 'Bouvier', 'Springfield Department of Motor Vehicles',␣
 ↪38),
    Employee('Robert', 'Terwilliger', 'Channel 6', 31),
    Employee('Herschel', 'Krustofsky', 'Channel 6', 2),
    Employee('Lois', 'Pennycandy', 'Channel 6', 46),
    Employee('Johnny', 'Cevasco', 'Mafia', 2),
    Employee('Fat', 'Tony', 'Mafia', 1),
    Employee('Max', 'Legman', 'Mafia', 3 ),
    Employee('Louie', 'Walters', 'Mafia', 4)
    ]

for emp in lst:
    print(emp)

print('==========================')

# Sort the people
lst = sorted(lst)

# Now check that the list is sorted
for first, second in zip(lst[:-1], lst[1:]):
    assert (first.company, first.id) < (second.company, second.id)

for emp in lst:
    print(emp)

print("\n\tSuccess!")
```

```
Homer Simpson, 1005 at Springfield Nuclear Power
Barney Gumble, 1 at Plow King
Clancy Wiggum, 1 at Police Department
Edna Krabapple, 39 at Springfield Elementary School
Seymour Skinner, 1 at Springfield Elementary School
Charles Burns, 1 at Springfield Nuclear Power
Waylon Smithers, 2 at Springfield Nuclear Power
Patty Bouvier, 39 at Springfield Department of Motor Vehicles
Selma Bouvier, 38 at Springfield Department of Motor Vehicles
Robert Terwilliger, 31 at Channel 6
Herschel Krustofsky, 2 at Channel 6
Lois Pennycandy, 46 at Channel 6
Johnny Cevasco, 2 at Mafia
Fat Tony, 1 at Mafia
Max Legman, 3 at Mafia
Louie Walters, 4 at Mafia
==========================
Herschel Krustofsky, 2 at Channel 6
Robert Terwilliger, 31 at Channel 6
Lois Pennycandy, 46 at Channel 6
Fat Tony, 1 at Mafia
Johnny Cevasco, 2 at Mafia
Max Legman, 3 at Mafia
```

```
Louie Walters, 4 at Mafia
Barney Gumble, 1 at Plow King
Clancy Wiggum, 1 at Police Department
Selma Bouvier, 38 at Springfield Department of Motor Vehicles
Patty Bouvier, 39 at Springfield Department of Motor Vehicles
Seymour Skinner, 1 at Springfield Elementary School
Edna Krabapple, 39 at Springfield Elementary School
Charles Burns, 1 at Springfield Nuclear Power
Waylon Smithers, 2 at Springfield Nuclear Power
Homer Simpson, 1005 at Springfield Nuclear Power

        Success!
```

1.10 Problem 5: Finding Repeats

DNA has a great deal of structure. DNA often contains repeats: this is a fascinating area that we are not going to explore. Investigate 'transposons'.

Write a program that finds the longest repeat in a sequence of DNA stored in a FASTA file.

There will be a single string of DNA in the file. The first line has a description of the contents, while the remainder is a string of A, C, G, and T with line breaks. Be sure to remove the line breaks.

Here is a sample run on pKLMF-FX.fasta

```
10089
(5535, 5541, 15)
CACGGGCACGGGCAC
CACGGGCACGGGCAC
CPU times: user 191 ms, sys: 2.49 ms, total: 193 ms
Wall time: 193 ms
```

[109]:
```python
# Read contents of fasta file with a single sequence
# Skip the first line, and return a string holding the contents

import re

def read_fasta_file(filename: str) -> str:
    with open(filename, 'r') as f:
        temp = [line.strip() for line in f]
        seq = ''.join(temp[1:])

    return seq
```

[110]:
```python
# Take a string and look for the longest repeat
# Return a tuple: (pos1, pos2, length) or None if there are no repeats
#    pos1 != pos2 and text[pos1:pos1+length)] == text[pos2:pos2+length]

from collections import defaultdict
```

```
def longest_repeat(text, cntr = 2):

    sol = (0,0,0)
    while True:

        d = defaultdict(list)
        for i in range(len(text)):
            d[text[i:i+cntr]].append(i)
        del_list = [(item, d[item]) for item in d if len(item) > 1 and␣
 ↪len(d[item]) > 1]
        if len(del_list) == 0:
            return sol
        else:
            temp = [(item[1][0], item[1][1], len(item[0])) for item in del_list]
            sol = (temp[0][0], temp[0][1], temp[0][2])
        cntr += 1

    return sol
```

1.11 Unit tests

[114]:
```
%%time
filename = 'pKLMF-FX.fasta'

text = read_fasta_file(filename)
print(len(text))
assert len(text) == 9988

tup = longest_repeat(text)

print(tup)
assert len(tup) == 3
assert isinstance(tup, tuple)

print(text[tup[0]:tup[0]+tup[2]])
print(text[tup[1]:tup[1]+tup[2]])

assert text[tup[0]:tup[0]+tup[2]] == text[tup[1]:tup[1]+tup[2]]
```

```
9988
(5434, 5440, 15)
CACGGGCACGGGCAC
CACGGGCACGGGCAC
Wall time: 87.8 ms
```

[112]:
```
%%time
filename = 'pACYC184.fasta'         # An EColi plasmid cloning vector

# See https://www.snapgene.com/resources/plasmid-files/?
 ↪set=basic_cloning_vectors&plasmid=pACYC184


text = read_fasta_file(filename)
print(len(text))                    # DNA is 4289 Bytes long: remove first line␣
 ↪and \n
assert len(text) == 4245

tup = longest_repeat(text)
print(tup)
assert len(tup) == 3
assert isinstance(tup, tuple)

print(text[tup[0]:tup[0]+tup[2]])
print(text[tup[1]:tup[1]+tup[2]])

assert tup[2] == 94
assert text[tup[0]:tup[0]+tup[2]] == text[tup[1]:tup[1]+tup[2]]
```

```
4245
(2180, 3274, 94)
AGCTCCTTCCGGTGGGCGCGGGGCATGACTATCGTCGCCGCACTTATGACTGTCTTCTTTATCATGCAACTCGTAGGACA
GGTGCCGGCAGCGC
AGCTCCTTCCGGTGGGCGCGGGGCATGACTATCGTCGCCGCACTTATGACTGTCTTCTTTATCATGCAACTCGTAGGACA
GGTGCCGGCAGCGC
Wall time: 247 ms
```

1.12 Extra credit: Find the longest repeat in EColi

[48]:
```
%%time
filename = 'ecoli.fasta'

text = read_fasta_file(filename)
print(len(text))
assert len(text) == 4641652

tup = longest_repeat(text, 2812)
print(tup)

assert len(tup) == 3
assert isinstance(tup, tuple)
assert len(text) == 4641652

```

```python
print(text[tup[0]:tup[0]+tup[2]])
print(text[tup[1]:tup[1]+tup[2]])

assert text[tup[0]:tup[0]+tup[2]] == text[tup[1]:tup[1]+tup[2]]
```

```
4641652
(4168618, 4210020, 2815)
AAGAAACATCTTCGGGTTGTGAGGTTAAGCGACTAAGCGTACACGGTGGATGCCCTGGCAGTCAGAGGCGATGAAGGACG
TGCTAATCTGCGATAAGCGTCGGTAAGGTGATATGAACCGTTATAACCGGCGATTTCCGAATGGGGAAACCCAGTGTGTT
TCGACACACTATCATTAACTGAATCCATAGGTTAATGAGGCGAACCGGGGGAACTGAAACATCTAAGTACCCCGAGGAAA
AGAAATCAACCGAGATTCCCCCAGTAGCGGCGAGCGAACGGGGAGCAGCCCAGAGCCTGAATCAGTGTGTGTGTTAGTGG
AAGCGTCTGGAAAGGCGCGCGATACAGGGTGACAGCCCCGTACACAAAAATGCACATGCTGTGAGCTCGATGAGTAGGGC
GGGACACGTGGTATCCTGTCTGAATATGGGGGGACCATCCTCCAAGGCTAAATACTCCTGACTGACCGATAGTGAACCAG
TACCGTGAGGGAAAGGCGAAAAGAACCCCGGCGAGGGGAGTGAAAAAGAACCTGAAACCGTGTACGTACAAGCAGTGGGA
GCACGCTTAGGCGTGTGACTGCGTACCTTTTGTATAATGGGTCAGCGACTTATATTCTGTAGCAAGGTTAACCGAATAGG
GGAGCCGAAGGGAAACCGAGTCTTAACTGGGCGTTAAGTTGCAGGGTATAGACCCGAAACCCGGTGATCTAGCCATGGGC
AGGTTGAAGGTTGGGTAACACTAACTGGAGGACCGAACCGACTAATGTTGAAAAATTAGCGGATGACTTGTGGCTGGGGG
TGAAAGGCCAATCAAACCGGGAGATAGCTGGTTCTCCCCGAAAGCTATTTAGGTAGCGCCTCGTGAATTCATCTCCGGGG
GTAGAGCACTGTTTCGGCAAGGGGGTCATCCCGACTTACCAACCCGATGCAAACTGCGAATACCGGAGAATGTTATCACG
GGAGACACACGGCGGGTGCTAACGTCCGTCGTGAAGAGGGAAACAACCCAGACCGCCAGCTAAGGTCCCAAAGTCATGGT
TAAGTGGGAAACGATGTGGGAAGGCCCAGACAGCCAGGATGTTGGCTTAGAAGCAGCCATCATTTAAAGAAAGCGTAATA
GCTCACTGGTCGAGTCGGCCTGCGCGGAAGATGTAACGGGGCTAAACCATGCACCGAAGCTGCGGCAGCGACGCTTATGC
GTTGTTGGGTAGGGGAGCGTTCTGTAAGCCTGCGAAGGTGTGCTGTGAGGCATGCTGGAGGTATCAGAAGTGCGAATGCT
GACATAAGTAACGATAAAGCGGGTGAAAAGCCCGCTCGCCGGAAGACCAAGGGTTCCTGTCCAACGTTAATCGGGGCAGG
GTGAGTCGACCCCTAAGGCGAGGCCGAAAGGCGTAGTCGATGGGAAACAGGTTAATATTCCTGTACTTGGTGTTACTGCG
AAGGGGGGACGGAGAAGGCTATGTTGGCCGGGCGACGGTTGTCCCGGTTTAAGCGTGTAGGCTGGTTTTCCAGGCAAATC
CGGAAAATCAAGGCTGAGGCGTGATGACGAGGCACTACGGTGCTGAAGCAACAAATGCCCTGCTTCCAGGAAAAGCCTCT
AAGCATCAGGTAACATCAAATCGTACCCCAAACCGACACAGGTGGTCAGGTAGAGAATACCAAGGCGCTTGAGAGAACTC
GGGTGAAGGAACTAGGCAAAATGGTGCCGTAACTTCGGGAGAAGGCACGCTGATATGTAGGTGAGGTCCCTCGCGGATGG
AGCTGAAATCAGTCGAAGATACCAGCTGGCTGCAACTGTTTATTAAAAACACAGCACTGTGCAAACACGAAAGTGGACGT
ATACGGTGTGACGCCTGCCCGGTGCCGGAAGGTTAATTGATGGGGTTAGCGCAAGCGAAGCTCTTGATCGAAGCCCCGGT
AAACGGCGGCCGTAACTATAACGGTCCTAAGGTAGCGAAATTCCTTGTCGGGTAAGTTCCGACCTGCACGAATGGCGTAA
TGATGGCCAGGCTGTCTCCACCCGAGACTCAGTGAAATTGAACTCGCTGTGAAGATGCAGTGTACCCGCGGCAAGACGGA
AAGACCCCGTGAACCTTTACTATAGCTTGACACTGAACATTGAGCCTTGATGTGTAGGATAGGTGGGAGGCTTTGAAGTG
TGGACGCCAGTCTGCATGGAGCCGACCTTGAAATACCACCCTTTAATGTTTGATGTTCTAACGTTGACCCGTAATCCGGG
TTGCGGACAGTGTCTGGTGGGTAGTTTGACTGGGGCGGTCTCCTCCTAAAGAGTAACGGAGGAGCACGAAGGTTGGCTAA
TCCTGGTCGGACATCAGGAGGTTAGTGCAATGGCATAAGCCAGCTTGACTGCGAGCGTGACGGCGCGAGCAGGTGCGAAA
GCAGGTCATAGTGATCCGGTGGTTCTGAATGGAAGGGCCATCGCTCAACGGATAAAAGGTACTCCGGGGATAACAGGCTG
ATACCGCCCAAGAGTTCATATCGACGGCGGTGTTTGGCACCTCGATGTCGGCTCATCACATCCTGGGGCTGAAGTAGGTC
CCAAGGGTATGGCTGTTCGCCATTTAAAGTGGTACGCGAGCTGGGTTTAGAACGTCGTGAGACAGTTCGGTCCCTATCTG
CCGTGGGCGCTGGAGAACTGAGGGGGGCTGCTCCTAGTACGAGAGGACCGGAGTGGACGCATCACTGGTGTTCGGGTTGT
CATGCCAATGGCACTGCCCGGTAGCTAAATGCGGAAGAGATAAGTGCTGAAAGCATCTAAGCACGAAACTTGCCCCGAGA
TGAGTTCTCCCTGAC
AAGAAACATCTTCGGGTTGTGAGGTTAAGCGACTAAGCGTACACGGTGGATGCCCTGGCAGTCAGAGGCGATGAAGGACG
TGCTAATCTGCGATAAGCGTCGGTAAGGTGATATGAACCGTTATAACCGGCGATTTCCGAATGGGGAAACCCAGTGTGTT
TCGACACACTATCATTAACTGAATCCATAGGTTAATGAGGCGAACCGGGGGAACTGAAACATCTAAGTACCCCGAGGAAA
AGAAATCAACCGAGATTCCCCCAGTAGCGGCGAGCGAACGGGGAGCAGCCCAGAGCCTGAATCAGTGTGTGTGTTAGTGG
```

```
AAGCGTCTGGAAAGGCGCGCGATACAGGGTGACAGCCCCGTACACAAAAATGCACATGCTGTGAGCTCGATGAGTAGGGC
GGGACACGTGGTATCCTGTCTGAATATGGGGGGACCATCCTCCAAGGCTAAATACTCCTGACTGACCGATAGTGAACCAG
TACCGTGAGGGAAAGGCGAAAAGAACCCCGGCGAGGGGAGTGAAAAAGAACCTGAAACCGTGTACGTACAAGCAGTGGGA
GCACGCTTAGGCGTGTGACTGCGTACCTTTTGTATAATGGGTCAGCGACTTATATTCTGTAGCAAGGTTAACCGAATAGG
GGAGCCGAAGGGAAACCGAGTCTTAACTGGGCGTTAAGTTGCAGGGTATAGACCCGAAACCCGGTGATCTAGCCATGGGC
AGGTTGAAGGTTGGGTAACACTAACTGGAGGACCGAACCGACTAATGTTGAAAAATTAGCGGATGACTTGTGGCTGGGGG
TGAAAGGCCAATCAAACCGGGAGATAGCTGGTTCTCCCCGAAAGCTATTTAGGTAGCGCCTCGTGAATTCATCTCCGGGG
GTAGAGCACTGTTTCGGCAAGGGGGTCATCCCGACTTACCAACCCGATGCAAACTGCGAATACCGGAGAATGTTATCACG
GGAGACACACGGCGGGTGCTAACGTCCGTCGTGAAGAGGGAAACAACCCAGACCGCCAGCTAAGGTCCCAAAGTCATGGT
TAAGTGGGAAACGATGTGGGAAGGCCCAGACAGCCAGGATGTTGGCTTAGAAGCAGCCATCATTTAAAGAAAGCGTAATA
GCTCACTGGTCGAGTCGGCCTGCGCGGAAGATGTAACGGGGCTAAACCATGCACCGAAGCTGCGGCAGCGACGCTTATGC
GTTGTTGGGTAGGGGAGCGTTCTGTAAGCCTGCGAAGGTGTGCTGTGAGGCATGCTGGAGGTATCAGAAGTGCGAATGCT
GACATAAGTAACGATAAAGCGGGTGAAAAGCCCGCTCGCCGGAAGACCAAGGGTTCCTGTCCAACGTTAATCGGGGCAGG
GTGAGTCGACCCCTAAGGCGAGGCCGAAAGGCGTAGTCGATGGGAAACAGGTTAATATTCCTGTACTTGGTGTTACTGCG
AAGGGGGGACGGAGAAGGCTATGTTGGCCGGGCGACGGTTGTCCCGGTTTAAGCGTGTAGGCTGGTTTTCCAGGCAAATC
CGGAAAATCAAGGCTGAGGCGTGATGACGAGGCACTACGGTGCTGAAGCAACAAATGCCCTGCTTCCAGGAAAAGCCTCT
AAGCATCAGGTAACATCAAATCGTACCCCAAACCGACACAGGTGGTCAGGTAGAGAATACCAAGGCGCTTGAGAGAACTC
GGGTGAAGGAACTAGGCAAAATGGTGCCGTAACTTCGGGAGAAGGCACGCTGATATGTAGGTGAGGTCCCTCGCGGATGG
AGCTGAAATCAGTCGAAGATACCAGCTGGCTGCAACTGTTTATTAAAAACACAGCACTGTGCAAACACGAAAGTGGACGT
ATACGGTGTGACGCCTGCCCGGTGCCGGAAGGTTAATTGATGGGGTTAGCGCAAGCGAAGCTCTTGATCGAAGCCCCGGT
AAACGGCGGCCGTAACTATAACGGTCCTAAGGTAGCGAAATTCCTTGTCGGGTAAGTTCCGACCTGCACGAATGGCGTAA
TGATGGCCAGGCTGTCTCCACCCGAGACTCAGTGAAATTGAACTCGCTGTGAAGATGCAGTGTACCCGCGGCAAGACGGA
AAGACCCCGTGAACCTTTACTATAGCTTGACACTGAACATTGAGCCTTGATGTGTAGGATAGGTGGGAGGCTTTGAAGTG
TGGACGCCAGTCTGCATGGAGCCGACCTTGAAATACCACCCTTTAATGTTTGATGTTCTAACGTTGACCCGTAATCCGGG
TTGCGGACAGTGTCTGGTGGGTAGTTTGACTGGGGCGGTCTCCTCCTAAAGAGTAACGGAGGAGCACGAAGGTTGGCTAA
TCCTGGTCGGACATCAGGAGGTTAGTGCAATGGCATAAGCCAGCTTGACTGCGAGCGTGACGGCGCGAGCAGGTGCGAAA
GCAGGTCATAGTGATCCGGTGGTTCTGAATGGAAGGGCCATCGCTCAACGGATAAAAGGTACTCCGGGGATAACAGGCTG
ATACCGCCCAAGAGTTCATATCGACGGCGGTGTTTGGCACCTCGATGTCGGCTCATCACATCCTGGGGCTGAAGTAGGTC
CCAAGGGTATGGCTGTTCGCCATTTAAAGTGGTACGCGAGCTGGGTTTAGAACGTCGTGAGACAGTTCGGTCCCTATCTG
CCGTGGGCGCTGGAGAACTGAGGGGGGCTGCTCCTAGTACGAGAGGACCGGAGTGGACGCATCACTGGTGTTCGGGTTGT
CATGCCAATGGCACTGCCCGGTAGCTAAATGCGGAAGAGATAAGTGCTGAAAGCATCTAAGCACGAAACTTGCCCCGAGA
TGAGTTCTCCCTGAC
Wall time: 2min 26s
```

[ ]:
```
# Well - technically I did indeed find the longest repeat in ecoli, although a)␣
 ↪if I do it from 'scratch',
# so to speak, it took an ungodly amount of time and b) I get a faster answer␣
 ↪only after tweaking the
# unit test to start at a higher assumed repeat string length.
```