Homework 12

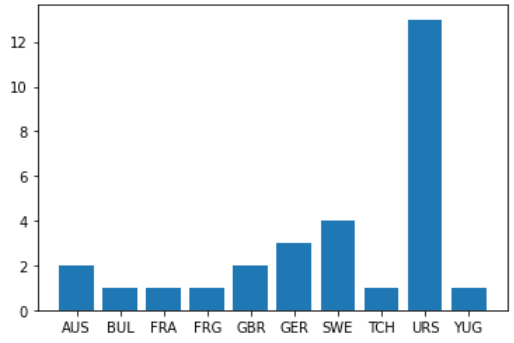
Due 4PM Nov 30, 2020

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.

In [46]:
```
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()
```



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)

In [34]:
```
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.*/a>', text)
    return re_links
```

Unit Test

In [35]:
```
website = 'https://www.extension.harvard.edu'
```

In [37]:
```
results = find_links(website)
print(len(results))
for link in results:
    print(link)
```

```
58
<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/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 S
chool</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 Cou
rses</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 Pro
fessional 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">Calli
ng 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-ca
n-support-online-students/582141/" target="_blank">How colleges with hybrid instruction this fall can su
pport online students</a>
<a class="h3" href="/about-us/press-announcements/michael-fabiano-joins-haa-board-directors" >Michael Fa
biano Joins HAA Board of Directors</a>
<a class="h3" href="https://blog.dce.harvard.edu/extension/announcing-new-graduate-and-undergraduate-cer
tificates-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="_bl
ank">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/buildin
g-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">Facebo
ok</a>
<a href="https://www.youtube.com/user/HarvardExtension" title="" class="menu__link i-social-youtube">You
Tube</a>
<a href="https://www.instagram.com/harvardextension/" title="" class="menu__link i-social-instagram">Ins
tagram</a>
<a class="menu__link" href="/privacy-policy" title="">Privacy</a>
<a class="menu__link" href="/resources-policies/accessibility-services-office-aso" title="">Accessibilit
y</a>
<a class="menu__link" href="/resources-policies/resources/rights-regulations" title="">Rights &amp; Regu
lations</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>
```

In [38]:
```
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)
```

```
link: <a class="skip" href="#main-menu">Jump to navigation</a>
results: <a href="#main-menu" class="skip">Jump to navigation</a>
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/extens
ion.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;amp; Admissions">Registration &amp;amp; Admissions</
a>
<a href="/resources-policies">Resources &amp;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;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 Cou
rses</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 Pro
fessional 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;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">Calli
ng 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-ca
n-support-online-students/582141/" target="_blank">How colleges with hybrid instruction this fall can su
pport online students</a>
<a class="h3" href="/about-us/press-announcements/michael-fabiano-joins-haa-board-directors">Michael Fab
iano Joins HAA Board of Directors</a>
<a class="h3" href="https://blog.dce.harvard.edu/extension/announcing-new-graduate-and-undergraduate-cer
tificates-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;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>
```

In [39]:
```
for x in range(10):
    print(results1[x],"\t", links[x])
```

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

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?

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.

In [40]:
```
import os

def find_files_gen(path, filename, filesize=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:
                matches.append((root+"/"+f, size))
    return matches

gen = find_files_gen('..', 'py',35000)
```

Unit Test

In [41]:
```
# Show recursive search.  Make sure we can see at least two directories of files
gen = find_files_gen('..', 'py',35000)

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

```
../E-7-Fall2020/Homework12_Surista.ipynb 35092
../E-7-Fall2020/Day08/Day8.ipynb 43290
../E-7-Fall2020/Day08/CopyBox.jpg 52884
../E-7-Fall2020/Day08/.ipynb_checkpoints/Day8-checkpoint.ipynb 43290
../E-7-Fall2020/Day07/Day7.ipynb 61025
../E-7-Fall2020/Day11/Homework11_SUrista.ipynb 41789
../E-7-Fall2020/Day11/.ipynb_checkpoints/Homework11-checkpoint.ipynb 41729
../E-7-Fall2020/Day11/.ipynb_checkpoints/Homework11_original-checkpoint.ipynb 41848
../E-7-Fall2020/Day11/.ipynb_checkpoints/Homework11_SUrista-checkpoint.ipynb 41789
../E-7-Fall2020/Day11/.ipynb_checkpoints/Homework11SUrista-checkpoint.ipynb 43057
../E-7-Fall2020/Day02/Day2.ipynb 40949
../E-7-Fall2020/Day06/pywiki.txt 46990
../E-7-Fall2020/Day06/pytext.txt 49208
../E-7-Fall2020/Day06/Day6.ipynb 73647
../E-7-Fall2020/Day12/.ipynb_checkpoints/Homework12_Surista-checkpoint.ipynb 44408
../E-7-Fall2020/Day10/Homework10_S_Urista.ipynb 43354
../E-7-Fall2020/Day10/BeautifulSoup-Lena.ipynb 991090
../E-7-Fall2020/Day10/Homework10.ipynb 42601
../E-7-Fall2020/Day10/.ipynb_checkpoints/Homework10-checkpoint.ipynb 35055
../E-7-Fall2020/Day10/.ipynb_checkpoints/Homework10 (4)-checkpoint.ipynb 43354
../E-7-Fall2020/Day10/.ipynb_checkpoints/2016_US_County_Level_Presidential_Results-checkpoint.ipynb 9803
3
../E-7-Fall2020/Day04/Day4 (1).ipynb 60195
../E-7-Fall2020/Day04/.ipynb_checkpoints/Day4-checkpoint.ipynb 60057
../E-7-Fall2020/Day04/.ipynb_checkpoints/Day4 (1)-checkpoint.ipynb 60195
../E-7-Fall2020/Day09/Iterator (1).ipynb 52895
../E-7-Fall2020/Day09/.ipynb_checkpoints/Day10-checkpoint.ipynb 46483
../E-7-Fall2020/Day09/.ipynb_checkpoints/Iterator-checkpoint.ipynb 115049
../E-7-Fall2020/Day05/Day5.ipynb 61784
```

In [28]:
```
# Show all notebooks in this directory
gen = find_files_gen('.', '.ipynb', 70000)

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

```
./Day06/Day6.ipynb 73647
./Day10/BeautifulSoup-Lena.ipynb 991090
./Day10/.ipynb_checkpoints/2016_US_County_Level_Presidential_Results-checkpoint.ipynb 98033
./Day09/.ipynb_checkpoints/Iterator-checkpoint.ipynb 115049
```

In [29]:
```
# Show all notebooks in this directory with at least 1K bytes
gen = find_files_gen('.', '.ipynb', 70000)

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

```
./Day06/Day6.ipynb 73647
./Day10/BeautifulSoup-Lena.ipynb 991090
./Day10/.ipynb_checkpoints/2016_US_County_Level_Presidential_Results-checkpoint.ipynb 98033
./Day09/.ipynb_checkpoints/Iterator-checkpoint.ipynb 115049
```

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.

Add to the cell below

In [15]:
```
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)
```

Unit Test

In [16]:
```
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!
```

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
```

In [17]:
```
# 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
```

In [49]:
```
# 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
```

Unit tests

In [53]:
```
%%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
CPU times: user 55.9 ms, sys: 9 µs, total: 55.9 ms
Wall time: 56.2 ms
```

In [54]:
```
%%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)
AGCTCCTTCCGGTGGGCGCGGGGCATGACTATCGTCGCCGCACTTATGACTGTCTTCTTTATCATGCAACTCGTAGGACAGGTGCCGGCAGCGC
AGCTCCTTCCGGTGGGCGCGGGGCATGACTATCGTCGCCGCACTTATGACTGTCTTCTTTATCATGCAACTCGTAGGACAGGTGCCGGCAGCGC
CPU times: user 157 ms, sys: 17 µs, total: 157 ms
Wall time: 156 ms
```

Extra credit: Find the longest repeat in EColi

In [52]:
```
%%time
filename = 'ecoli.fasta'

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

tup = longest_repeat(text, 2810)
print(tup)

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

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)
AAGAAACATCTTCGGGTTGTGAGGTTAAGCGACTAAGCGTACACGGTGGATGCCCTGGCAGTCAGAGGCGATGAAGGACGTGCTAATCTGCGATAAGCGTCGGT
AAGGTGATATGAACCGTTATAACCGGCGATTTCCGAATGGGGAAACCCAGTGTGTTTCGACACACTATCATTAACTGAATCCATAGGTTAATGAGGCGAACCGG
GGGAACTGAAACATCTAAGTACCCCGAGGAAAAGAAATCAACCGAGATTCCCCCAGTAGCGGCGAGCGAACGGGGAGCAGCCCAGAGCCTGAATCAGTGTGTGT
GTTAGTGGAAGCGTCTGGAAAGGCGCGCGATACAGGGTGACAGCCCCGTACACAAAAATGCACATGCTGTGAGCTCGATGAGTAGGGCGGGACACGTGGTATCC
TGTCTGAATATGGGGGGACCATCCTCCAAGGCTAAATACTCCTGACTGACCGATAGTGAACCAGTACCGTGAGGGAAAGGCGAAAAGAACCCCGGCGAGGGGAG
TGAAAAAGAACCTGAAACCGTGTACGTACAAGCAGTGGGAGCACGCTTAGGCGTGTGACTGCGTACCTTTTGTATAATGGGTCAGCGACTTATATTCTGTAGCA
AGGTTAACCGAATAGGGGAGCCGAAGGGAAACCGAGTCTTAACTGGGCGTTAAGTTGCAGGGTATAGACCCGAAACCCGGTGATCTAGCCATGGGCAGGTTGAA
GGTTGGGTAACACTAACTGGAGGACCGAACCGACTAATGTTGAAAAATTAGCGGATGACTTGTGGCTGGGGGTGAAAGGCCAATCAAACCGGGAGATAGCTGGT
TCTCCCCGAAAGCTATTTAGGTAGCGCCTCGTGAATTCATCTCCGGGGGTAGAGCACTGTTTCGGCAAGGGGGTCATCCCGACTTACCAACCCGATGCAAACTG
CGAATACCGGAGAATGTTATCACGGGAGACACACGGCGGGTGCTAACGTCCGTCGTGAAGAGGGAAACAACCCAGACCGCCAGCTAAGGTCCCAAAGTCATGGT
TAAGTGGGAAACGATGTGGGAAGGCCCAGACAGCCAGGATGTTGGCTTAGAAGCAGCCATCATTTAAAGAAAGCGTAATAGCTCACTGGTCGAGTCGGCCTGCG
CGGAAGATGTAACGGGGCTAAACCATGCACCGAAGCTGCGGCAGCGACGCTTATGCGTTGTTGGGTAGGGGAGCGTTCTGTAAGCCTGCGAAGGTGTGCTGTGA
GGCATGCTGGAGGTATCAGAAGTGCGAATGCTGACATAAGTAACGATAAAGCGGGTGAAAAGCCCGCTCGCCGGAAGACCAAGGGTTCCTGTCCAACGTTAATC
GGGGCAGGGTGAGTCGACCCCTAAGGCGAGGCCGAAAGGCGTAGTCGATGGGAAACAGGTTAATATTCCTGTACTTGGTGTTACTGCGAAGGGGGGACGGAGAA
GGCTATGTTGGCCGGGCGACGGTTGTCCCGGTTTAAGCGTGTAGGCTGGTTTTCCAGGCAAATCCGGAAAATCAAGGCTGAGGCGTGATGACGAGGCACTACGG
TGCTGAAGCAACAAATGCCCTGCTTCCAGGAAAAGCCTCTAAGCATCAGGTAACATCAAATCGTACCCCAAACCGACACAGGTGGTCAGGTAGAGAATACCAAG
GCGCTTGAGAGAACTCGGGTGAAGGAACTAGGCAAAATGGTGCCGTAACTTCGGGAGAAGGCACGCTGATATGTAGGTGAGGTCCCTCGCGGATGGAGCTGAAA
TCAGTCGAAGATACCAGCTGGCTGCAACTGTTTATTAAAAACACAGCACTGTGCAAACACGAAAGTGGACGTATACGGTGTGACGCCTGCCCGGTGCCGGAAGG
TTAATTGATGGGGTTAGCGCAAGCGAAGCTCTTGATCGAAGCCCCGGTAAACGGCGGCCGTAACTATAACGGTCCTAAGGTAGCGAAATTCCTTGTCGGGTAAG
TTCCGACCTGCACGAATGGCGTAATGATGGCCAGGCTGTCTCCACCCGAGACTCAGTGAAATTGAACTCGCTGTGAAGATGCAGTGTACCCGCGGCAAGACGGA
AAGACCCCGTGAACCTTTACTATAGCTTGACACTGAACATTGAGCCTTGATGTGTAGGATAGGTGGGAGGCTTTGAAGTGTGGACGCCAGTCTGCATGGAGCCG
ACCTTGAAATACCACCCTTTAATGTTTGATGTTCTAACGTTGACCCGTAATCCGGGTTGCGGACAGTGTCTGGTGGGTAGTTTGACTGGGGCGGTCTCCTCCTA
AAGAGTAACGGAGGAGCACGAAGGTTGGCTAATCCTGGTCGGACATCAGGAGGTTAGTGCAATGGCATAAGCCAGCTTGACTGCGAGCGTGACGGCGCGAGCAG
GTGCGAAAGCAGGTCATAGTGATCCGGTGGTTCTGAATGGAAGGGCCATCGCTCAACGGATAAAAGGTACTCCGGGGATAACAGGCTGATACCGCCCAAGAGTT
CATATCGACGGCGGTGTTTGGCACCTCGATGTCGGCTCATCACATCCTGGGGCTGAAGTAGGTCCCAAGGGTATGGCTGTTCGCCATTTAAAGTGGTACGCGAG
CTGGGTTTAGAACGTCGTGAGACAGTTCGGTCCCTATCTGCCGTGGGCGCTGGAGAACTGAGGGGGGCTGCTCCTAGTACGAGAGGACCGGAGTGGACGCATCA
CTGGTGTTCGGGTTGTCATGCCAATGGCACTGCCCGGTAGCTAAATGCGGAAGAGATAAGTGCTGAAAGCATCTAAGCACGAAACTTGCCCCGAGATGAGTTCT
CCCTGAC
AAGAAACATCTTCGGGTTGTGAGGTTAAGCGACTAAGCGTACACGGTGGATGCCCTGGCAGTCAGAGGCGATGAAGGACGTGCTAATCTGCGATAAGCGTCGGT
AAGGTGATATGAACCGTTATAACCGGCGATTTCCGAATGGGGAAACCCAGTGTGTTTCGACACACTATCATTAACTGAATCCATAGGTTAATGAGGCGAACCGG
GGGAACTGAAACATCTAAGTACCCCGAGGAAAAGAAATCAACCGAGATTCCCCCAGTAGCGGCGAGCGAACGGGGAGCAGCCCAGAGCCTGAATCAGTGTGTGT
GTTAGTGGAAGCGTCTGGAAAGGCGCGCGATACAGGGTGACAGCCCCGTACACAAAAATGCACATGCTGTGAGCTCGATGAGTAGGGCGGGACACGTGGTATCC
TGTCTGAATATGGGGGGACCATCCTCCAAGGCTAAATACTCCTGACTGACCGATAGTGAACCAGTACCGTGAGGGAAAGGCGAAAAGAACCCCGGCGAGGGGAG
TGAAAAAGAACCTGAAACCGTGTACGTACAAGCAGTGGGAGCACGCTTAGGCGTGTGACTGCGTACCTTTTGTATAATGGGTCAGCGACTTATATTCTGTAGCA
AGGTTAACCGAATAGGGGAGCCGAAGGGAAACCGAGTCTTAACTGGGCGTTAAGTTGCAGGGTATAGACCCGAAACCCGGTGATCTAGCCATGGGCAGGTTGAA
GGTTGGGTAACACTAACTGGAGGACCGAACCGACTAATGTTGAAAAATTAGCGGATGACTTGTGGCTGGGGGTGAAAGGCCAATCAAACCGGGAGATAGCTGGT
TCTCCCCGAAAGCTATTTAGGTAGCGCCTCGTGAATTCATCTCCGGGGGTAGAGCACTGTTTCGGCAAGGGGGTCATCCCGACTTACCAACCCGATGCAAACTG
CGAATACCGGAGAATGTTATCACGGGAGACACACGGCGGGTGCTAACGTCCGTCGTGAAGAGGGAAACAACCCAGACCGCCAGCTAAGGTCCCAAAGTCATGGT
TAAGTGGGAAACGATGTGGGAAGGCCCAGACAGCCAGGATGTTGGCTTAGAAGCAGCCATCATTTAAAGAAAGCGTAATAGCTCACTGGTCGAGTCGGCCTGCG
CGGAAGATGTAACGGGGCTAAACCATGCACCGAAGCTGCGGCAGCGACGCTTATGCGTTGTTGGGTAGGGGAGCGTTCTGTAAGCCTGCGAAGGTGTGCTGTGA
GGCATGCTGGAGGTATCAGAAGTGCGAATGCTGACATAAGTAACGATAAAGCGGGTGAAAAGCCCGCTCGCCGGAAGACCAAGGGTTCCTGTCCAACGTTAATC
GGGGCAGGGTGAGTCGACCCCTAAGGCGAGGCCGAAAGGCGTAGTCGATGGGAAACAGGTTAATATTCCTGTACTTGGTGTTACTGCGAAGGGGGGACGGAGAA
GGCTATGTTGGCCGGGCGACGGTTGTCCCGGTTTAAGCGTGTAGGCTGGTTTTCCAGGCAAATCCGGAAAATCAAGGCTGAGGCGTGATGACGAGGCACTACGG
TGCTGAAGCAACAAATGCCCTGCTTCCAGGAAAAGCCTCTAAGCATCAGGTAACATCAAATCGTACCCCAAACCGACACAGGTGGTCAGGTAGAGAATACCAAG
GCGCTTGAGAGAACTCGGGTGAAGGAACTAGGCAAAATGGTGCCGTAACTTCGGGAGAAGGCACGCTGATATGTAGGTGAGGTCCCTCGCGGATGGAGCTGAAA
TCAGTCGAAGATACCAGCTGGCTGCAACTGTTTATTAAAAACACAGCACTGTGCAAACACGAAAGTGGACGTATACGGTGTGACGCCTGCCCGGTGCCGGAAGG
TTAATTGATGGGGTTAGCGCAAGCGAAGCTCTTGATCGAAGCCCCGGTAAACGGCGGCCGTAACTATAACGGTCCTAAGGTAGCGAAATTCCTTGTCGGGTAAG
TTCCGACCTGCACGAATGGCGTAATGATGGCCAGGCTGTCTCCACCCGAGACTCAGTGAAATTGAACTCGCTGTGAAGATGCAGTGTACCCGCGGCAAGACGGA
AAGACCCCGTGAACCTTTACTATAGCTTGACACTGAACATTGAGCCTTGATGTGTAGGATAGGTGGGAGGCTTTGAAGTGTGGACGCCAGTCTGCATGGAGCCG
ACCTTGAAATACCACCCTTTAATGTTTGATGTTCTAACGTTGACCCGTAATCCGGGTTGCGGACAGTGTCTGGTGGGTAGTTTGACTGGGGCGGTCTCCTCCTA
AAGAGTAACGGAGGAGCACGAAGGTTGGCTAATCCTGGTCGGACATCAGGAGGTTAGTGCAATGGCATAAGCCAGCTTGACTGCGAGCGTGACGGCGCGAGCAG
GTGCGAAAGCAGGTCATAGTGATCCGGTGGTTCTGAATGGAAGGGCCATCGCTCAACGGATAAAAGGTACTCCGGGGATAACAGGCTGATACCGCCCAAGAGTT
CATATCGACGGCGGTGTTTGGCACCTCGATGTCGGCTCATCACATCCTGGGGCTGAAGTAGGTCCCAAGGGTATGGCTGTTCGCCATTTAAAGTGGTACGCGAG
CTGGGTTTAGAACGTCGTGAGACAGTTCGGTCCCTATCTGCCGTGGGCGCTGGAGAACTGAGGGGGGCTGCTCCTAGTACGAGAGGACCGGAGTGGACGCATCA
CTGGTGTTCGGGTTGTCATGCCAATGGCACTGCCCGGTAGCTAAATGCGGAAGAGATAAGTGCTGAAAGCATCTAAGCACGAAACTTGCCCCGAGATGAGTTCT
CCCTGAC
CPU times: user 1min 14s, sys: 1.29 s, total: 1min 15s
Wall time: 1min 15s
```

In [15]:
```
# Well - technically I did indeed find the longest repeat in ecoli, although a) it took an
# ungodly amount of time and b) that was only after tweaking the unit test to start at a higher
# assumed repeat string length. Without that cheat / hack, the program took over six hours
# yes, I ran it overnight....
```