Chicagomarathon Website Scrapping

Code Scenerio: We have fetched the links of first 50 male runners in Chicago Marathon 2019 site and store it in a list,Then we use the libraries and for loop to pass each instance of the list with the delay of 2 seconds to our GetData Function.After each iteration of for loop, we get in return of the function list of the required data then we add the data one by one in the DataFrame to show it as the requirement.In additional the dataframe is exported to the csv file to use this data for further use.

As we need to fetch data of first 50 male marathon runners, we have written a script in python for given requirement. Required Libraires: 1)BeautifulSoup

2)request

3)pandas

4)Time

Above libraires are used in development of our program. Following is the little explanation of code along with

In [ ]:
```
#Importing All Required Libraries
from bs4 import BeautifulSoup
from urllib import request
import pandas as pd
import time
```

Requesting html page using website Link

By using given website, we are getting response using request. Then reading it, after that parsing it into html using BeautifulSoup. It parse the given response into html like page.

In [4]:
```
#With Request Library we are opening the Website Link and converting ists meshy output to beautifull Html code
url="https://results.chicagomarathon.com/2019/?page=1&event=MAR&lang=EN_CAP&num_results=50&pid=list&search%5Bsex%5D=
responce=request.urlopen(url)
responce=responce.read()
soup=BeautifulSoup(responce,'html.parser')
```

Collecting links from html page.

After getting soup we used find_all() method which helps us to find all tags with specified attribues and its value. Here we first get all 'ul' tags having class 'list-group list-group-multicolumn' then finding of inside it we get 'li' and inside we have 'href' attribute containing the required link. We loop this process to get all the links of runners data page.(that are total 50)

In [5]:
```
#Getting the required class from the soup and further getting the table and further anker tag's
uls = soup.find_all('ul', {'class': 'list-group list-group-multicolumn'})
Links=[]
for ul in uls:
    for li in ul.find_all('li'):
        for link in li.find_all('a'):
            url = link.get('href')
            Links.append(url)
            break
```

GetData Function

GetData is funtion which takes input of link and process it to get desired data. It goes to link, read the page, parse it into HTML and get values with find/findall method. Then it picks the table containing our desired data then gets values and add append them into a list. We have two tables here which keeps the data so we fetch data one by one and append them in same list. At the end it return the list of data, fetched from the passed link.

In [6]:
```
#Function to process a single link after exectly 2 seconds of interval
def GetData(link):
#Another soup of HTML is created on the behalf of link getted from the previous page
    responce2=request.urlopen("https://results.chicagomarathon.com/2019/"+link)
    responce2=responce2.read()
    soup2=BeautifulSoup(responce2,'html.parser')
#Moving toward the required table to get the split and Time Information
    table = soup2.find("table", {"class":"table-condensed"})
    rows = []
    for tr in table.select('tr'):
        rows.append([td.get_text(strip=True) for td in tr.select('td')])
    flat_list = [item for sublist in rows for item in sublist]
    flat_list.pop()

    Time = soup2.find_all("td", {"class":"time"})
    for i in Time:
        flat_list.append(i.get_text(strip=True))
    return flat_list
```

Getting Heading of Dataframe.

we have so far go the process of getting data from website. Now its time to add the data into DataFrame.So we go to website with random person selected. We fetch our headings of data frame. We append them into a list. We have done this for both tables, because our dataframe contains the heading of both

tables.

In [8]:
```
#Getting the Main Headings or Columns of the Pandas DataFrame
responce2=request.urlopen("https://results.chicagomarathon.com/2019/"+Links[0])
responce2=responce2.read()
soup2=BeautifulSoup(responce2,'html.parser')
Heading=[]
table = soup2.find("table", {"class":"table-condensed"})
rows = []
for tr in table.select('tr'):
    rows.append([td.get_text(strip=True) for td in tr.select('th')])
Heading = [item for sublist in rows for item in sublist]
Heading.pop()
table2 = soup2.find("table", {"class":"table table-condensed table-striped"})
split = table2.find_all("th", {"class":"desc"})
for i in split:
    Heading.append(i.get_text(strip=True))
Heading.pop(5)
print("")
```

Creating DataFrame

In this last section of code we make DataFrame using two previous lists we make(one of heading other of data). As data is different for all so we fetch thier data in a loop and add it into DaraFrame one by one. Here we make our program sleep for two seconds(As per our given instructions).

In [10]:
```
#Creating a Dataframe and after Getting the Data of every single Runner then append it
Data=pd.DataFrame(columns=Heading)
print("Working...")
for i in Links:
    d=GetData(i)
    new={}
    for key in Heading:
        for value in d:
            new[key]=value
            d.remove(value)
            break
    Data=Data.append(new,ignore_index=True)
    time.sleep(2)
print("Done")
```

```
Working...
Done
```

Exporting DataFrame into CSV

For futher usage of data we have exported the DataFrame of values into a .csv file.

In [11]:
```
Data.to_csv('frame.csv', index = False)
```

In [12]:
```
Data
```

Out[12]:

	Name (CTZ)	Age Group	Bib Number	Age	City, State	05K	10K	15K	20K	HALF	25K	30K	35K	40K	Finish
0	Cherono, Lawrence (KEN)	30-34	4	31	Chepkoilel	00:14:45	00:29:28	00:44:11	00:59:01	01:02:15	01:13:57	01:28:59	01:43:54	01:59:10	02:05:45
1	Debela, Dejene (ETH)	20-24	38	24	West Chester	00:14:45	00:29:29	00:44:12	00:59:02	01:02:16	01:13:56	01:28:59	01:43:53	01:59:08	02:05:46
2	Mengstu, Asefa (ETH)	30-34	5	31	Addis Ababa	00:14:46	00:29:29	00:44:12	00:59:02	01:02:15	01:13:55	01:28:59	01:43:53	01:59:08	02:05:48
3	Karoki, Bedan (KEN)	25-29	9	29	Mbuyu	00:14:45	00:29:27	00:44:10	00:59:02	01:02:15	01:13:54	01:28:59	01:43:53	01:59:09	02:05:53
4	Abdi, Bashir (BEL)	30-34	10	30	Nijmegen	00:14:47	00:29:30	00:44:23	00:59:32	01:02:54	01:14:53	01:30:22	01:45:15	01:59:53	02:06:14
5	Tura, Seifu (ETH)	20-24	39	22	Addis Abeba	00:14:46	00:29:29	00:44:12	00:59:02	01:02:15	01:13:56	01:28:59	01:43:53	02:00:13	02:08:35
6	Chumba, Dickson (KEN)	30-34	6	32	Kipngoror	00:14:45	00:29:27	00:44:11	00:59:01	01:02:14	01:13:55	01:28:58	01:44:27	02:01:05	02:09:11

In [ ]:
```
```