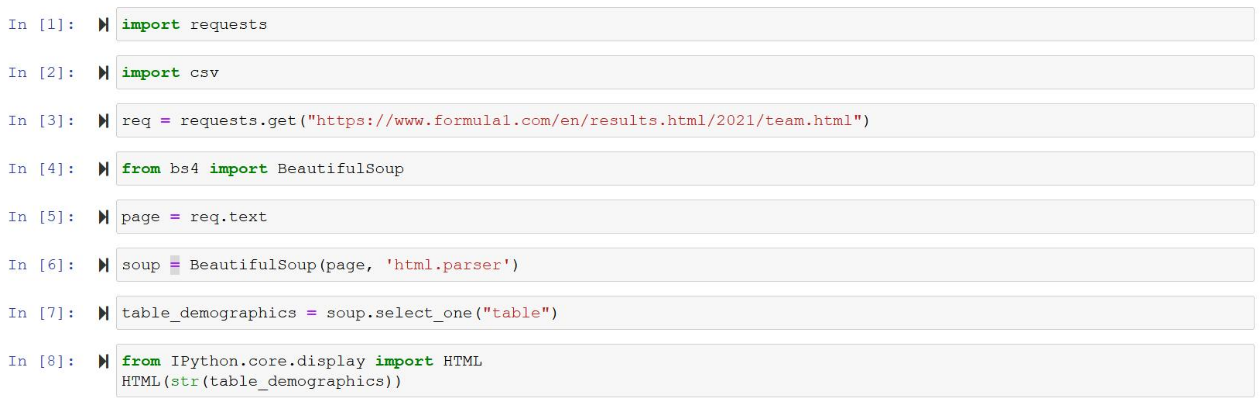
Code Explaination

1. Teams.ipynb



```
In [1]:  import requests

In [2]:  import csv

In [3]:  req = requests.get("https://www.formula1.com/en/results.html/2021/team.html")

In [4]:  from bs4 import BeautifulSoup

In [5]:  page = req.text

In [6]:  soup = BeautifulSoup(page, 'html.parser')

In [7]:  table_demographics = soup.select_one("table")

In [8]:  from IPython.core.display import HTML
         HTML(str(table_demographics))
```

We do import requests library to request the HTML file from a website and the csv library to write the scrapped cleaned data into a csv file

We save the HTML file that we request into a variable named req

We do import the BeautifulSoup library to enhance the handling of the website

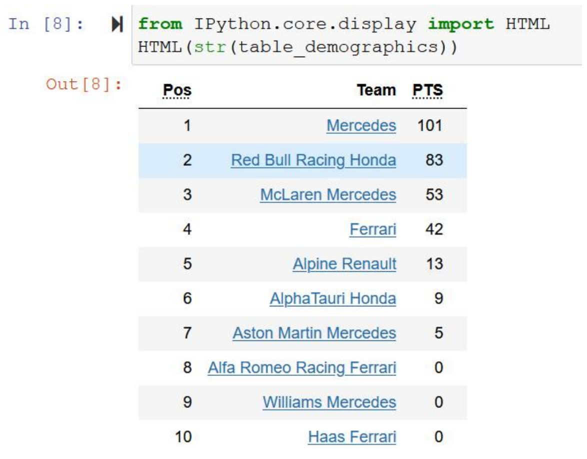
We do save the text version of the req variable into a new variable named page

We use the BeautifulSoup library to parser the variable page into a new variable named soup

We inspect the website to get the name of the table that handles the data within it which is “table”

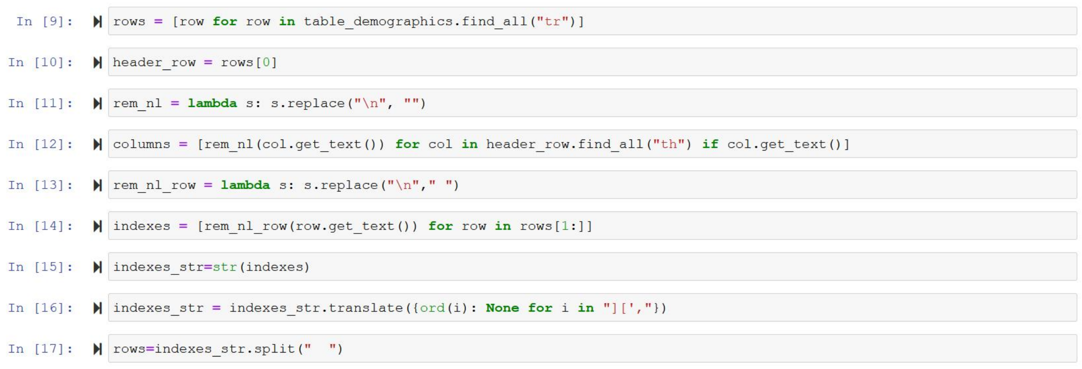
We use soup.select_one(“table”) to select only one object which carries the data within it of class “table” and saves it into a new variable named table_demographics

We import the HTML library to display the table that is saved into table_demographics



```
In [8]:   from IPython.core.display import HTML
          HTML(str(table_demographics))
Out[8]:
```

Pos	Team	PTS
1	Mercedes	101
2	Red Bull Racing Honda	83
3	McLaren Mercedes	53
4	Ferrari	42
5	Alpine Renault	13
6	AlphaTauri Honda	9
7	Aston Martin Mercedes	5
8	Alfa Romeo Racing Ferrari	0
9	Williams Mercedes	0
10	Haas Ferrari	0



```
In [9]:   rows = [row for row in table_demographics.find_all("tr")]

In [10]:  header_row = rows[0]

In [11]:  rem_nl = lambda s: s.replace("\n", "")

In [12]:  columns = [rem_nl(col.get_text()) for col in header_row.find_all("th") if col.get_text()]

In [13]:  rem_nl_row = lambda s: s.replace("\n"," ")

In [14]:  indexes = [rem_nl_row(row.get_text()) for row in rows[1:]]

In [15]:  indexes_str=str(indexes)

In [16]:  indexes_str = indexes_str.translate({ord(i): None for i in "][',"})

In [17]:  rows=indexes_str.split("  ")
```

We do loop for all the rows that are defined by the tag “tr” by using the find_all() in the “table_demographics” and save them into a list named rows

We do get the first row which is the header row and save them into a string named header_rows

We do use the lambda function to replace all the newlines into space

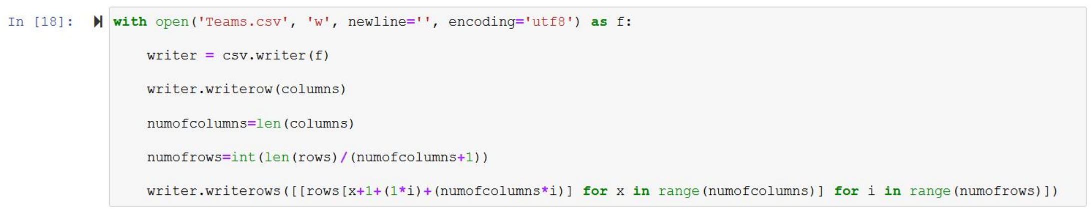
We do loop for all the headers that are defined by the tag “th” by using the find_all() in the “header_row” and save them into a list named columns

We do extract the data from rows after the first one which was the header data and put them into a list named indexes

We do transform the indexes list into a string and names it indexes_str

We do clean indexes_str from all the extra letters that are in the data and we don’t need to use the translate to None function

We do create a new list named rows by using the split(“ ”) function on the indexes_strNote in the split(“   ”) function we did use double space



```
In [18]:  with open('Teams.csv', 'w', newline='', encoding='utf8') as f:
              writer = csv.writer(f)
              writer.writerow(columns)
              numofcolumns=len(columns)
              numofrows=int(len(rows)/(numofcolumns+1))
              writer.writerows([[rows[x+1+(1*i)+(numofcolumns*i)] for x in range(numofcolumns)] for i in range(numofrows)])
```

We do open a new file named ‘Teams.csv’ with writer ability and the newline equals ‘’ so each new row will be just under the previous row and encoding equals to utf8 and we name the access to this file as f

We create a writer function to write in the csv file

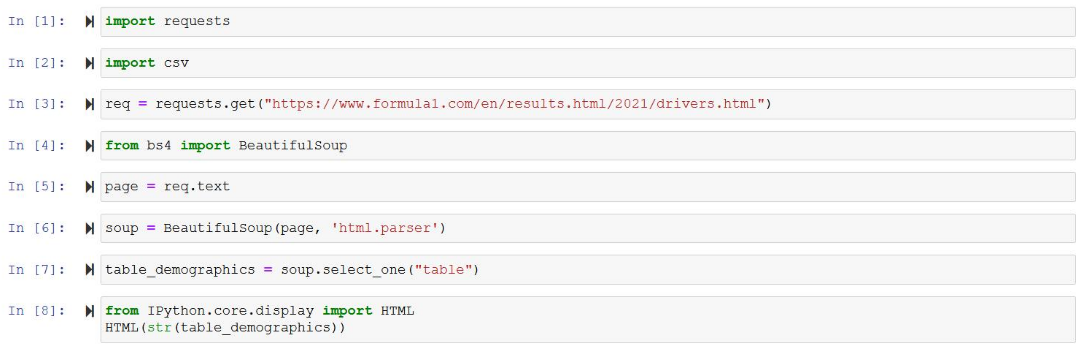
We get the number of columns

We get the number of rows by getting the total length of the rows dividing by the number of columns +1 this ‘1’ represents the header value then we do int parsing the value of this equation

We write in the rows[“number of columns done” added by one the header row multiplied by the number of rows done plus the total number of columns multiplied by the number of rows done]

Note: You can view the output by opening the Teams.csv file

2. Drivers.ipynb



```
In [1]:  import requests

In [2]:  import csv

In [3]:  req = requests.get("https://www.formula1.com/en/results.html/2021/drivers.html")

In [4]:  from bs4 import BeautifulSoup

In [5]:  page = req.text

In [6]:  soup = BeautifulSoup(page, 'html.parser')

In [7]:  table_demographics = soup.select_one("table")

In [8]:  from IPython.core.display import HTML
         HTML(str(table_demographics))
```

We do import requests library to request the HTML file from a website and the csv library to write the scrapped cleaned data into a csv file

We save the HTML file that we request into a variable named req

We do import the BeautifulSoup library to enhance the handling of the website

We do save the text version of the req variable into a new variable named page

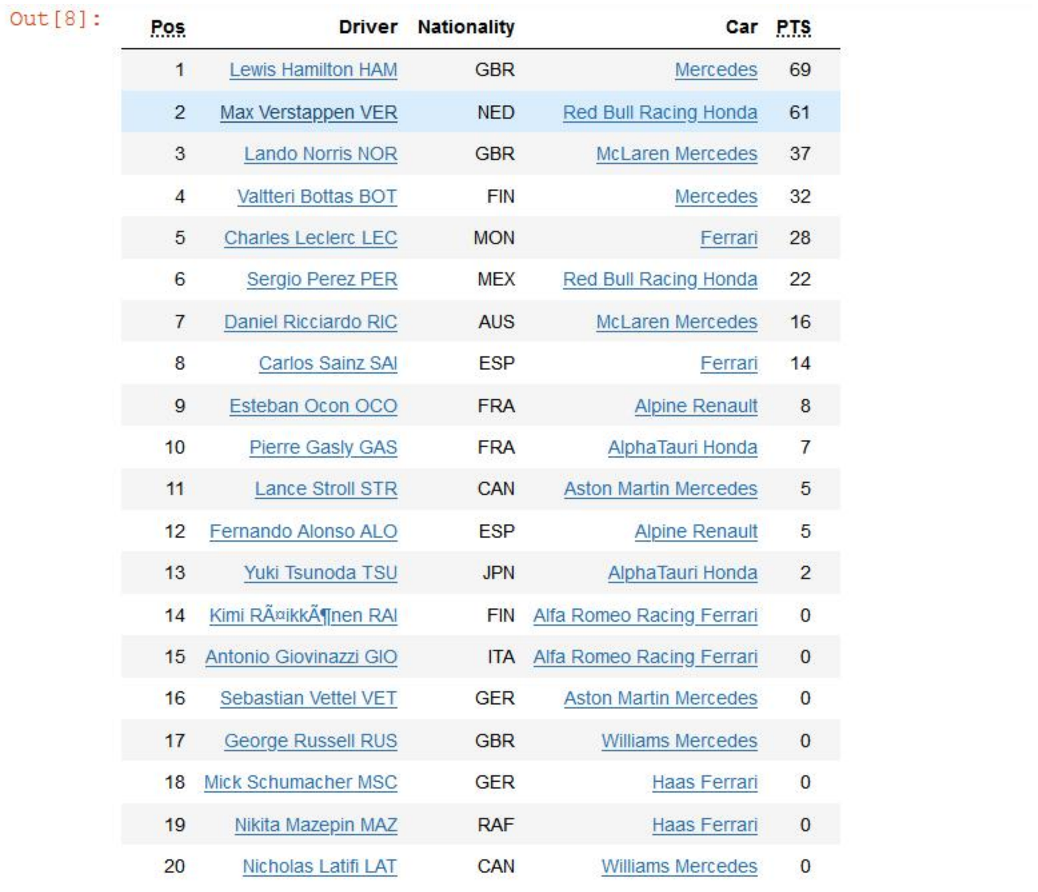
We use the BeautifulSoup library to parser the variable page into a new variable named soup

We inspect the website to get the name of the table that handles the data within it which is “table”

We use soup.select_one(“table”) to select only one object which carries the data within it of class “table” and saves it into a new variable named table_demographics

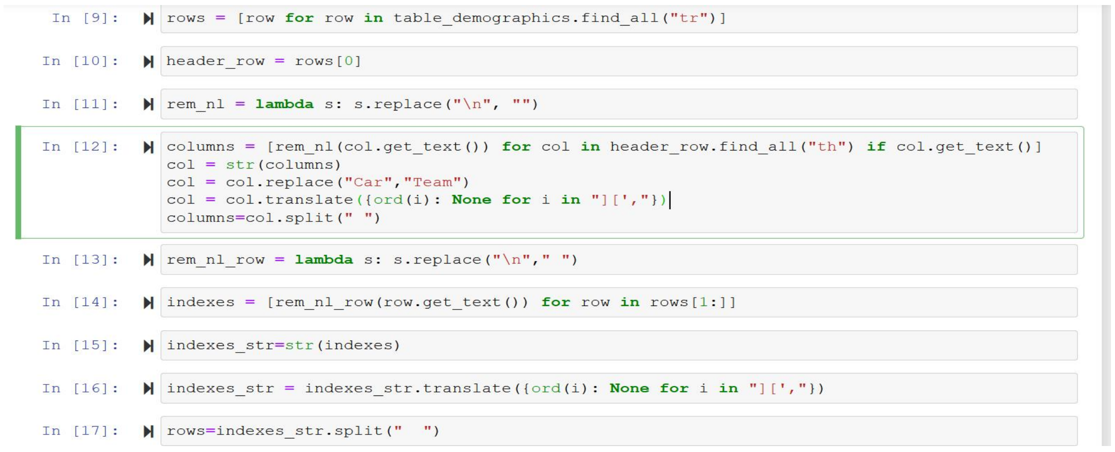
We import the HTML library to display the table that is saved into

table_demographics



```
Out[8]:
```

Pos	Driver	Nationality	Car	PTS
1	Lewis Hamilton HAM	GBR	Mercedes	69
2	Max Verstappen VER	NED	Red Bull Racing Honda	61
3	Lando Norris NOR	GBR	McLaren Mercedes	37
4	Valtteri Bottas BOT	FIN	Mercedes	32
5	Charles Leclerc LEC	MON	Ferrari	28
6	Sergio Perez PER	MEX	Red Bull Racing Honda	22
7	Daniel Ricciardo RIC	AUS	McLaren Mercedes	16
8	Carlos Sainz SAI	ESP	Ferrari	14
9	Esteban Ocon OCO	FRA	Alpine Renault	8
10	Pierre Gasly GAS	FRA	AlphaTauri Honda	7
11	Lance Stroll STR	CAN	Aston Martin Mercedes	5
12	Fernando Alonso ALO	ESP	Alpine Renault	5
13	Yuki Tsunoda TSU	JPN	AlphaTauri Honda	2
14	Kimi RÃ¤ikkÃ¶nen RAI	FIN	Alfa Romeo Racing Ferrari	0
15	Antonio Giovinazzi GIO	ITA	Alfa Romeo Racing Ferrari	0
16	Sebastian Vettel VET	GER	Aston Martin Mercedes	0
17	George Russell RUS	GBR	Williams Mercedes	0
18	Mick Schumacher MSC	GER	Haas Ferrari	0
19	Nikita Mazepin MAZ	RAF	Haas Ferrari	0
20	Nicholas Latifi LAT	CAN	Williams Mercedes	0



```
In [9]:   rows = [row for row in table_demographics.find_all("tr")]

In [10]:  header_row = rows[0]

In [11]:  rem_nl = lambda s: s.replace("\n", "")

In [12]:  columns = [rem_nl(col.get_text()) for col in header_row.find_all("th") if col.get_text()]
          col = str(columns)
          col = col.replace("Car","Team")
          col = col.translate({ord(i): None for i in "]['","})
          columns=col.split(" ")

In [13]:  rem_nl_row = lambda s: s.replace("\n"," ")

In [14]:  indexes = [rem_nl_row(row.get_text()) for row in rows[1:]]

In [15]:  indexes_str=str(indexes)

In [16]:  indexes_str = indexes_str.translate({ord(i): None for i in "]['","})

In [17]:  rows=indexes_str.split("  ")
```

We do a loop for all the rows that are defined by the tag “tr” by using the find_all() in the “table_demographics” and save them into a list named rows

We do get the first row which is the header row and save them into a string named header_rows

We do use the lambda function to replace all the newlines into space

We do loop for all the headers that are defined by the tag “th” by using the find_all() in the “header_row” and save them into a list named columns

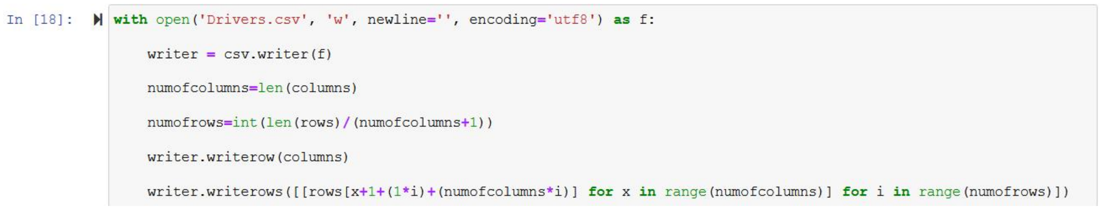
We did transform the columns list to string change the name of Columns Car to Team then cleaned the data and transfers it back to a list

We do extract the data from rows after the first one which was the header data and put them into a list named indexes

We do transform the indexes list into a string and names it indexes_str

We do clean indexes_str from all the extra letters that are in the data and we don’t need to use the translate to None function

We do create a new list named rows by using the split(“ ”) function on the indexes_str Note in the split(“   ”) function we did use double space



```
In [18]:  with open('Drivers.csv', 'w', newline='', encoding='utf8') as f:
              writer = csv.writer(f)
              numofcolumns=len(columns)
              numofrows=int(len(rows)/(numofcolumns+1))
              writer.writerow(columns)
              writer.writerows([[rows[x+1+(1*i)+(numofcolumns*i)] for x in range(numofcolumns)] for i in range(numofrows)])
```

We do open a new file named ‘Teams.csv’ with writer ability and the newline equals ‘’ so each new row will be just under the previous row and encoding equals to utf8 and we name the access to this file as f

We create a writer function to write in the csv file

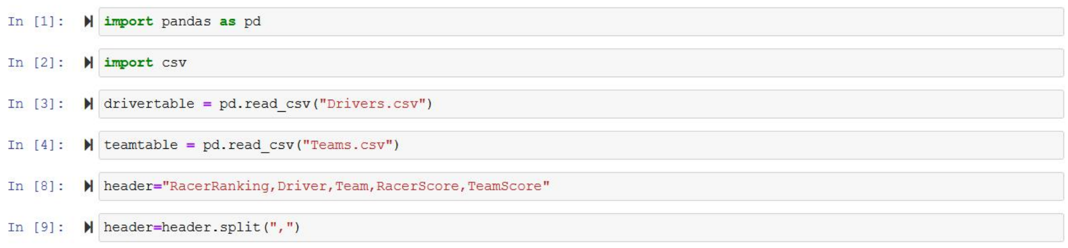
We get the number of columns

We get the number of rows by getting the total length of the rows dividing by the number of columns +1 this ‘1’ represents the header value then we do int parsing the value of this equation

We write in the rows[“number of columns done” added by one the header row multiplied by the number of rows done plus the total number of columns multiplied by the number of rows done]

Note: You can view the output by opening the Driver.csv file

3. Merge.ipynb



```
In [1]:  import pandas as pd

In [2]:  import csv

In [3]:  drivertable = pd.read_csv("Drivers.csv")

In [4]:  teamtable = pd.read_csv("Teams.csv")

In [8]:  header="RacerRanking,Driver,Team,RacerScore,TeamScore"

In [9]:  header=header.split(",")
```

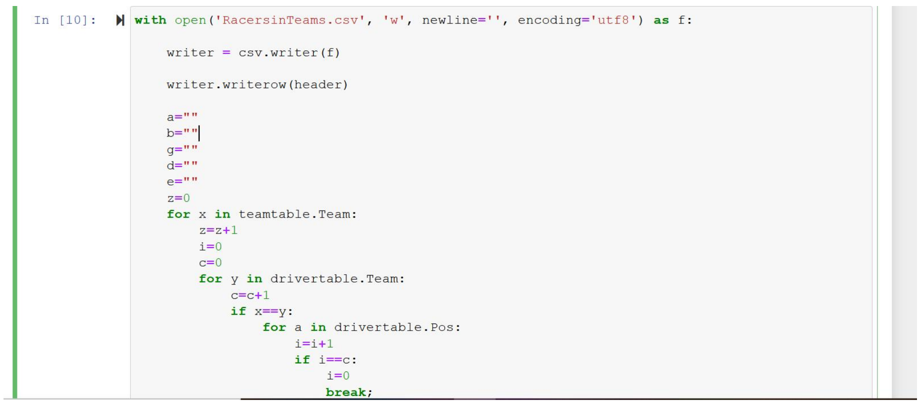
We do import pandas library to read the data from the csv file and import csv file to be able to write a new csv file

We do read the Drivers.csv and save it into a drivertable object

We do read the Teams.csv and save it into a teamstable object

We do create a header string for the new csv file

We do transform the header into a list using the split(",") function to split the string using the commas that we used in writing the header

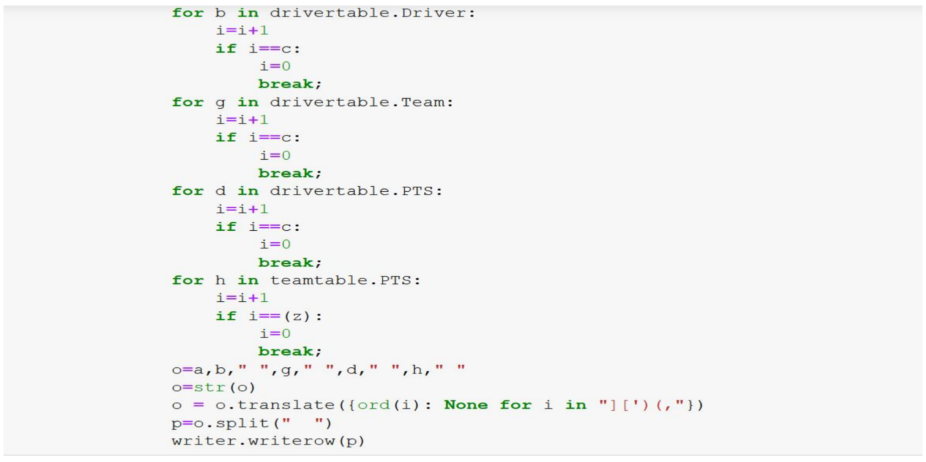


```
In [10]:  with open('RacersinTeams.csv', 'w', newline='', encoding='utf8') as f:

              writer = csv.writer(f)

              writer.writerow(header)

              a=""
              b=""
              g=""
              d=""
              e=""
              z=0
              for x in teamtable.Team:
                  z=z+1
                  i=0
                  c=0
                  for y in drivertable.Team:
                      c=c+1
                      if x==y:
                          for a in drivertable.Pos:
                              i=i+1
                              if i==c:
                                  i=0
                                  break;
```



```
                          for b in drivertable.Driver:
                              i=i+1
                              if i==c:
                                  i=0
                                  break;
                          for g in drivertable.Team:
                              i=i+1
                              if i==c:
                                  i=0
                                  break;
                          for d in drivertable.PTS:
                              i=i+1
                              if i==c:
                                  i=0
                                  break;
                          for h in teamtable.PTS:
                              i=i+1
                              if i==(z):
                                  i=0
                                  break;
                          o=a,b,"  ",g,"  ",d,"  ",h,"  "
                          o=str(o)
                          o = o.translate({ord(i): None for i in "][')(,"})
                          p=o.split("  ")
                          writer.writerow(p)
```

We do open a new file named ‘RacersinTeams.csv’ with writer ability and the newline equals ‘’ so each new row will be just under the previous row and encoding equals to utf8 and we name the access to this file as f

We create a writer function to write in the csv file

We get the number of columns

We do create an empty string variable that we will use for later and set the z to zero to count the outer loop

The outer loop will loop for teams table column Teams then increment the z value by one and set i to 0 and c to 0 then loops for drivers table column Teams and see if the output of the outer loop exist or not and increment the c value by 1 for every loop for later use.

If the Teams of the drivertable equals the Teams in the teamtable the loop will enter the if function inside the if function there will be a loop for each column to get the value of the column that’s in the same row as the Teams of the drivertable that equals to the Teams in the teamtable

We write a list that includes the strings that we did got from the if function Note we did write space between some of the data to enhance and make the splitting easier

We transfer the list into a string to manipulate the data in it

We do clean from all the extra letters that are in the data and we don’t need to use the translate to None function

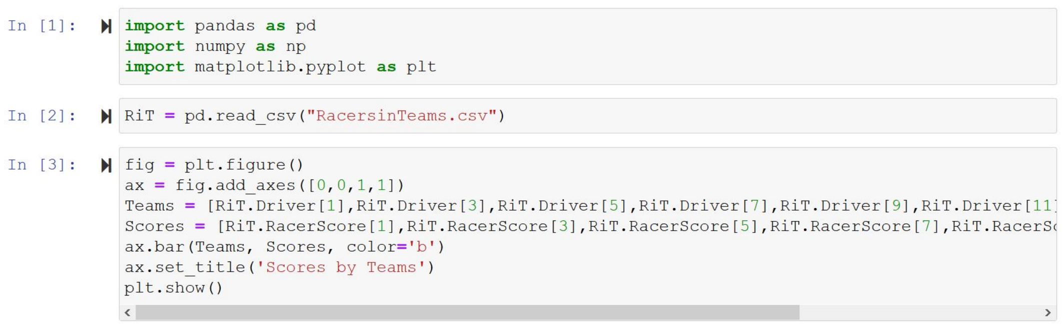
We split the string into a list by using a double-spacing list remember that we did write space between some of the data to enhance the splitting

We write the list into a row

Repeat

Note: You can view the output by opening the RacersinTeams.csv file

4. Graphs.ipynb



```
In [1]:  import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt

In [2]:  RiT = pd.read_csv("RacersinTeams.csv")

In [3]:  fig = plt.figure()
         ax = fig.add_axes([0,0,1,1])
         Teams = [RiT.Driver[1],RiT.Driver[3],RiT.Driver[5],RiT.Driver[7],RiT.Driver[9],RiT.Driver[11
         Scores = [RiT.RacerScore[1],RiT.RacerScore[3],RiT.RacerScore[5],RiT.RacerScore[7],RiT.RacerSc
         ax.bar(Teams, Scores, color='b')
         ax.set_title('Scores by Teams')
         plt.show()
```

We do import pandas library to read the data from the csv file and import numpy to do mathematical ops and matplotlib library to plot the data into graphs

We do read the RacersinTeams.csv and save it into a RiT named object

We did create a figure then created its axes by giving it 0 0 1 1 which means that this graph will only the display the +ve part of x-axis and y-axis

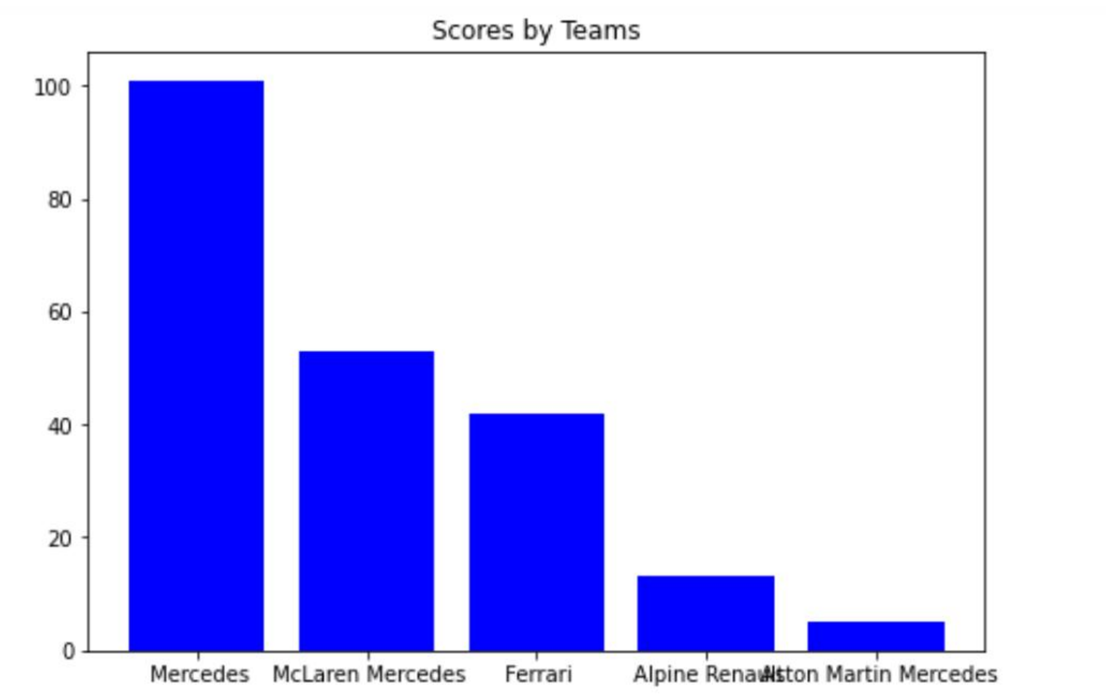
We did create a list named Teams added the Teams name into it Notice and please forgive us that there’s a bug in the RacersinTeams.csv file that names the column as the previous column header name while it does display the data correctly in the csv file

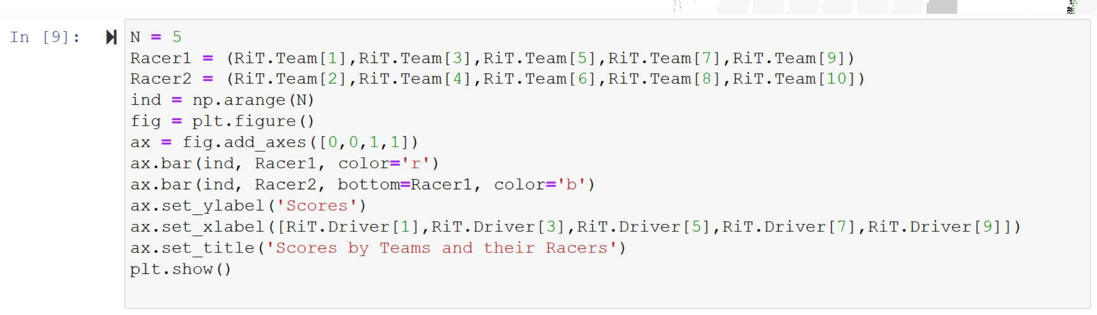
We did create a list named Scores added the Teams total Scores into it Notice and please forgive us that there’s a bug in the RacersinTeams.csv file that names the column as the previous column header name while it does display the data correctly in the csv file

We used the bar(Teams, Scores, color) function to build the bar graph using Teams list on the x-axis, Scores in the y-axis and give it the color blue

We did sit the title to Scores by Teams

Then showed the bar graph





```
In [9]:  N = 5
         Racer1 = (RiT.Team[1],RiT.Team[3],RiT.Team[5],RiT.Team[7],RiT.Team[9])
         Racer2 = (RiT.Team[2],RiT.Team[4],RiT.Team[6],RiT.Team[8],RiT.Team[10])
         ind = np.arange(N)
         fig = plt.figure()
         ax = fig.add_axes([0,0,1,1])
         ax.bar(ind, Racer1, color='r')
         ax.bar(ind, Racer2, bottom=Racer1, color='b')
         ax.set_ylabel('Scores')
         ax.set_xlabel([RiT.Driver[1],RiT.Driver[3],RiT.Driver[5],RiT.Driver[7],RiT.Driver[9]])
         ax.set_title('Scores by Teams and their Racers')
         plt.show()
```

We did set the N to 5 which are the number of bars that will be shown in the graph

Added Racer 1 and Racer 2 list

Racer 1 contained the scores of the first racer in each team and Racer 2 contained the scores of the second of each team Notice and please forgive us that there's a bug in the RacersinTeams.csv file that names the column as the previous column header name while it does display the data correctly in the csv file

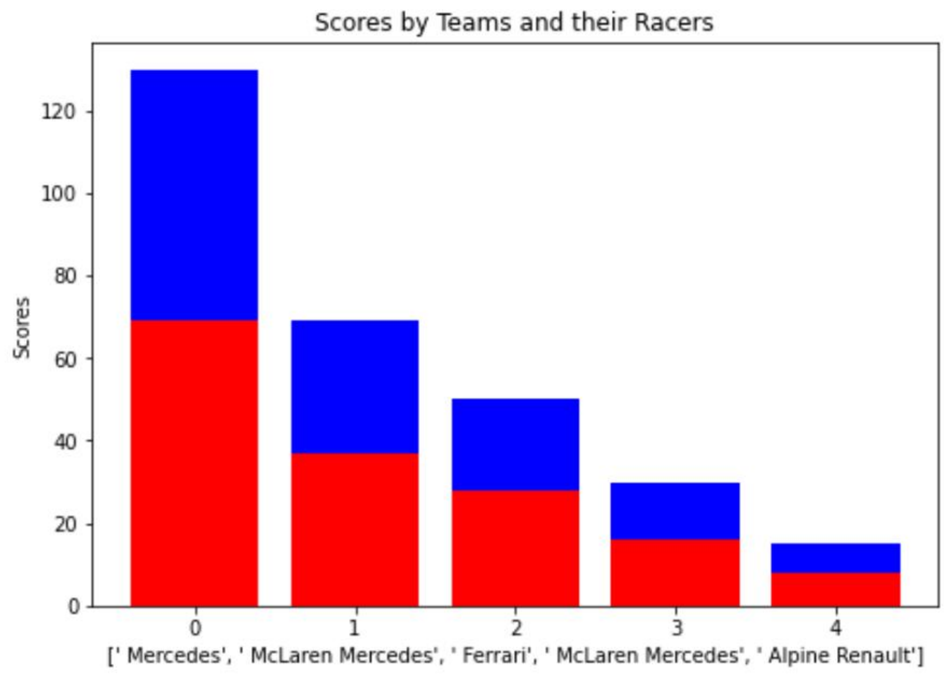
Then we did set the ind to n by using the arrange function to arrange the bar graph

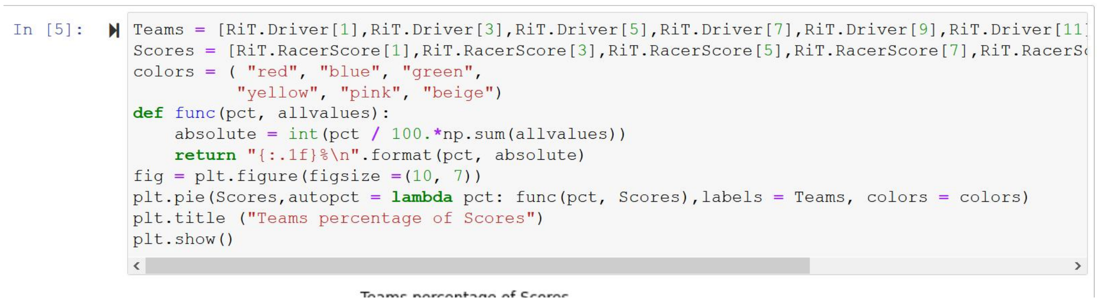
We did create a figure then created its axes by giving it 0 0 1 1 which means that this graph will only the display the +ve part of the x-axis and y-axis

Used the bar function to give Racer 1 the color red and the place that has been arranged for him by using the arrange graph

Used the bar function to give Racer 2 the color blue and the place that has been arranged for him by using the arrange graph and an extra function which bottom= Racer 1 to set Racer 2 above Racer 1

We did set the ylabel to Scores and assign the Teams name to the xlabel we did set the title then we showed the graph Notice and please forgive us that there's a bug in the RacersinTeams.csv file that names the column as the previous column header name while it does display the data correctly in the csv file





```
In [5]:  Teams = [RiT.Driver[1],RiT.Driver[3],RiT.Driver[5],RiT.Driver[7],RiT.Driver[9],RiT.Driver[11
          Scores = [RiT.RacerScore[1],RiT.RacerScore[3],RiT.RacerScore[5],RiT.RacerScore[7],RiT.RacerSc
          colors = ( "red", "blue", "green",
                    "yellow", "pink", "beige")
          def func(pct, allvalues):
              absolute = int(pct / 100.*np.sum(allvalues))
              return "{:.1f}%\n".format(pct, absolute)
          fig = plt.figure(figsize =(10, 7))
          plt.pie(Scores,autopct = lambda pct: func(pct, Scores),labels = Teams, colors = colors)
          plt.title ("Teams percentage of Scores")
          plt.show()
```

Teams percentage of Scores

We enter a list of the Team names as Teams Notice and please forgive us that there's a bug in the RacersinTeams.csv file that names the column as the previous column header name while it does display the data correctly in the csv file

We enter a list of the Team total scores as Scores Notice and please forgive us that there's a bug in the RacersinTeams.csv file that names the column as the previous column header name while it does display the data correctly in the csv file

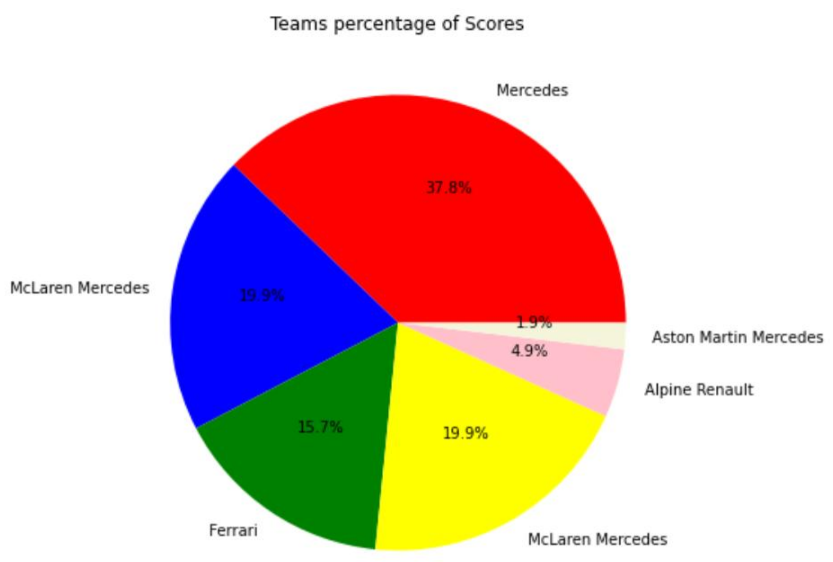
Added 6 color to the only 6 teams that have scores

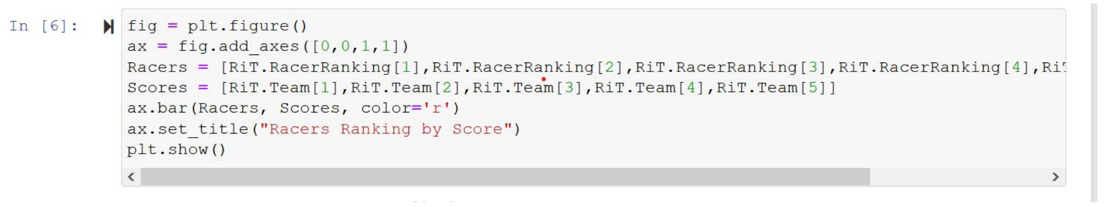
Created a function that will the display the percentage of the team scores from the total scores of all the teams

Create the pi fig and set its size

Entered the data to the pi

Showed the pi



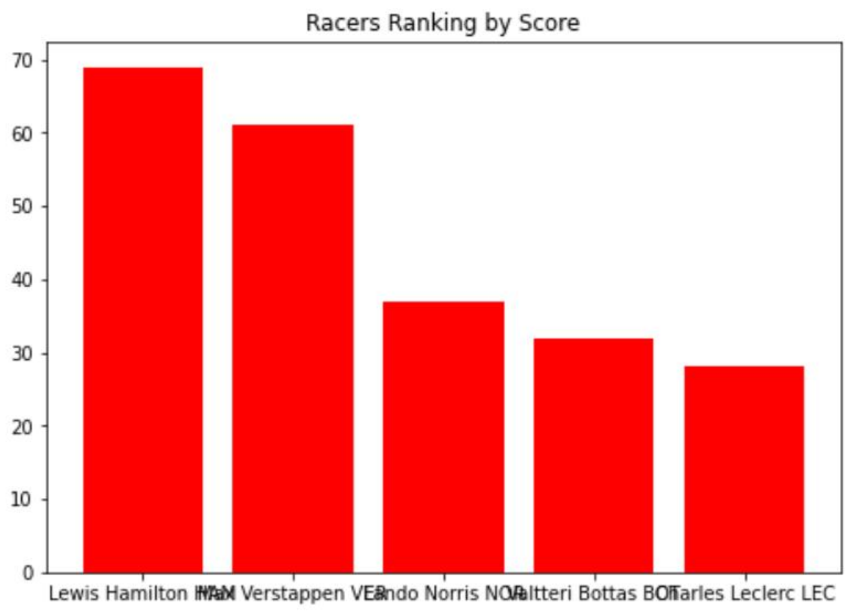

```
In [6]:  fig = plt.figure()
         ax = fig.add_axes([0,0,1,1])
         Racers = [RiT.RacerRanking[1],RiT.RacerRanking[2],RiT.RacerRanking[3],RiT.RacerRanking[4],Ri
         Scores = [RiT.Team[1],RiT.Team[2],RiT.Team[3],RiT.Team[4],RiT.Team[5]]
         ax.bar(Racers, Scores, color='r')
         ax.set_title("Racers Ranking by Score")
         plt.show()
```

We did create a figure then created its axes by giving it 0 0 1 1 which means that this graph will only display the +ve part of the x-axis and y-axis

We enter a list of the Racer names as Racers Notice and please forgive us that there’s a bug in the RacersinTeams.csv file that names the column as the previous column header name while it does display the data correctly in the csv file

We enter a list of the Racer total scores as Scores Notice and please forgive us that there’s a bug in the RacersinTeams.csv file that names the column as the previous column header name while it does display the data correctly in the csv file

Added the data into the bar() then colored it red then we give it a title then showed it


Racers Ranking by Score
Lewis Hamilton HAM
Max Verstappen VER
Lando Norris NOR
Valtteri Bottas BOT
Charles Leclerc LEC