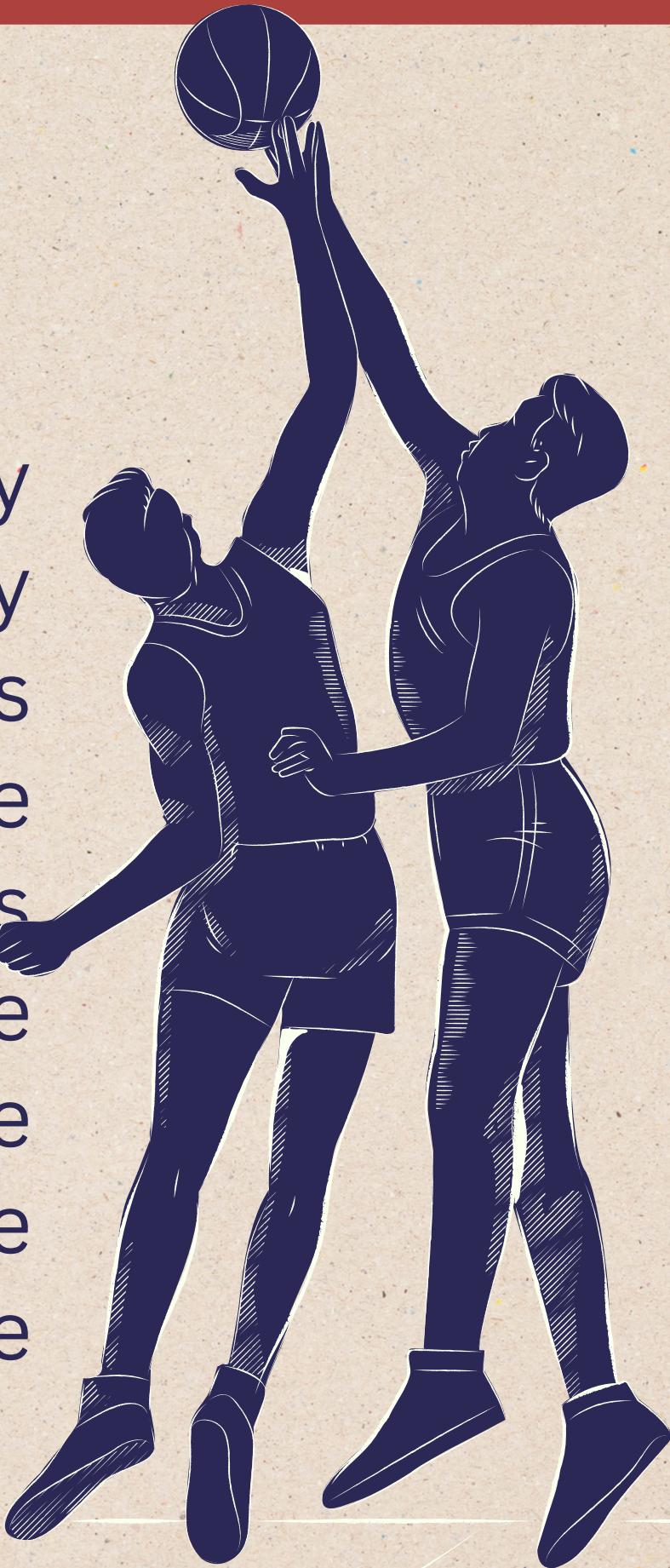


# Olimpiadas

*Diplomado de Ciencias de Datos  
Proyecto Modulo 2 Procesamiento de datos con Python  
Cinthya Simoneen*

# Introducción

El objetivo principal del proyecto es recompilar y analizar datos históricos de los Juegos Olímpicos y compararlos con el Producto Interno Bruto de los países con el objetivo de analizar la relación entre medallas ganadas y dicho indicador. Utilizando los conocimientos obtenidos en el Módulo 2 de este Diplomado (Procesamiento de Datos con Python), se ha consultado mediante API y formateado los datos de los logros olímpicos a lo largo de las Olimpiadas que han transcurrido desde el año 2000.



Se abre un archivo CSV para cargar los datos de PIB per Capita (Banco Mundial) por cada país y se genera un dataframe con estos datos.

```
## abrir el csv de PIBS para preparar la información de Productos Internos Brutos por país
df_PIBS=pd.read_csv('C:\BEDU_CIENCIA_DATOS\Modulo2\Proyecto_M2\Data\PIBSXPAIS.csv',delimiter=';', encoding="Latin-1")
#df_PIBS
```

✓ 0.2s

```
#Se toman solo las series que nos sirven del archivo CSV
df_PIB_Olimpiadas=pd.DataFrame()
df_PIB_Olimpiadas=df.concat([df_PIBS["PAIS"],
| | | | | df_PIBS["2000"],
| | | | | df_PIBS["2004"],
| | | | | df_PIBS["2008"],
| | | | | df_PIBS["2012"],
| | | | | df_PIBS["2016"],
| | | | | df_PIBS["2020"],
| | | | | df_PIBS["2024"],axis=1)

#type(df_PIB_Olimpiadas["2000"])
#Valores vacios a cero
df_PIB_Olimpiadas.fillna(0)
#Pasamos los valores no data que vienen en el csv a cero
df_PIB_Olimpiadas['2000'] = df_PIB_Olimpiadas['2000'].str.replace('no data', '0')
df_PIB_Olimpiadas['2004'] = df_PIB_Olimpiadas['2004'].str.replace('no data', '0')
df_PIB_Olimpiadas['2008'] = df_PIB_Olimpiadas['2008'].str.replace('no data', '0')
df_PIB_Olimpiadas['2012'] = df_PIB_Olimpiadas['2012'].str.replace('no data', '0')
df_PIB_Olimpiadas['2016'] = df_PIB_Olimpiadas['2016'].str.replace('no data', '0')
df_PIB_Olimpiadas['2020'] = df_PIB_Olimpiadas['2020'].str.replace('no data', '0')
df_PIB_Olimpiadas['2024'] = df_PIB_Olimpiadas['2024'].str.replace('no data', '0')
#Quitamos las comas
df_PIB_Olimpiadas['2000'] = df_PIB_Olimpiadas['2000'].str.replace(',', '')
df_PIB_Olimpiadas['2004'] = df_PIB_Olimpiadas['2004'].str.replace(',', '')
df_PIB_Olimpiadas['2008'] = df_PIB_Olimpiadas['2008'].str.replace(',', '')
df_PIB_Olimpiadas['2012'] = df_PIB_Olimpiadas['2012'].str.replace(',', '')
df_PIB_Olimpiadas['2016'] = df_PIB_Olimpiadas['2016'].str.replace(',', '')
df_PIB_Olimpiadas['2020'] = df_PIB_Olimpiadas['2020'].str.replace(',', '')
df_PIB_Olimpiadas['2024'] = df_PIB_Olimpiadas['2024'].str.replace(',', '')

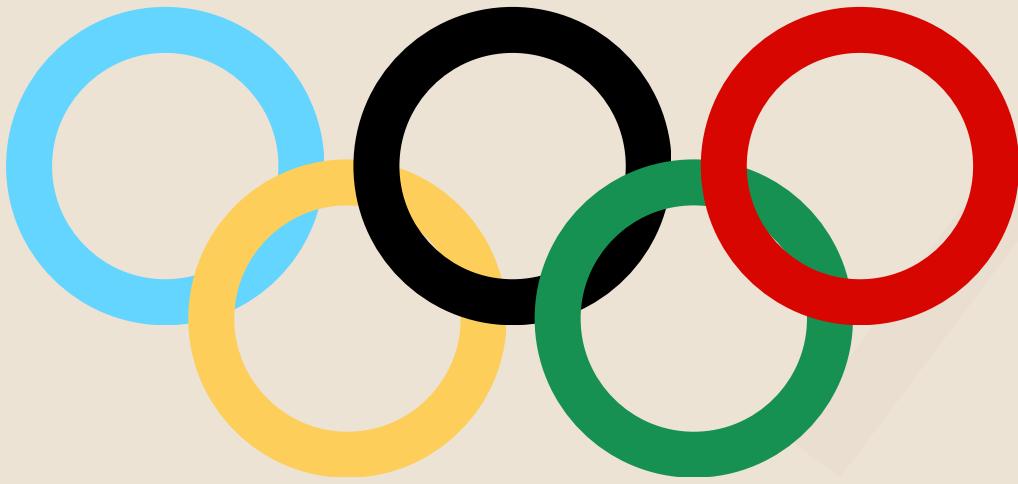
#Pasamos los valores de PIB a entero
df_PIB_Olimpiadas['2000'] = df_PIB_Olimpiadas["2000"].astype(int)
df_PIB_Olimpiadas['2004'] = df_PIB_Olimpiadas["2004"].astype(int)
df_PIB_Olimpiadas['2008'] = df_PIB_Olimpiadas["2008"].astype(int)
df_PIB_Olimpiadas['2012'] = df_PIB_Olimpiadas["2012"].astype(int)
df_PIB_Olimpiadas['2016'] = df_PIB_Olimpiadas["2016"].astype(int)
df_PIB_Olimpiadas['2020'] = df_PIB_Olimpiadas["2020"].astype(int)
df_PIB_Olimpiadas['2024'] = df_PIB_Olimpiadas["2024"].astype(int)
```

	PAIS	2000	2004	2008	2012	2016	2020	2024
0	Afghanistan	0	254259	447746	784611	617126	611268	0
1	Albania	112764	237358	4370562	4248909	4124405	5278986	8924317
2	Algeria	1947755	283982	5214762	6057974	442658	3757585	5721678
3	Andorra	0	0	0	4571397	39595317	36973845	44899596
4	Angola	681129	1254696	4081718	5083827	3468518	1709283	243158
...	...	...	...	...	...	...	...	...
189	West Bank and Gaza	141275	1350906	1913316	2888598	3325846	3044744	0
190	Yemen	526881	677899	1170964	1369687	1083887	633311	486382
191	Zambia	364026	55605	139378	172958	1250324	958265	1413421
192	Zimbabwe	969824	806898	553376	1310442	1444716	1769564	2087614
193	United States	36312782	41641617	48470553	51736738	58179697	64367435	85372686

```
#Inicio variables
opcion_pais=[]
anio_olimpiada=""
opcion_olimpiada=""

#Usuario elige la Olimpiada que quiere consultar
opcion_pais=["Sidney","Atenas","Beijing","Londres","Tokyo","Paris"]
opcion_olimpiada = input(f"""Ingrese la Olimpiada que desea:
{opcion_pais[0]}
{opcion_pais[1]}
{opcion_pais[2]}
{opcion_pais[3]}
{opcion_pais[4]}
{opcion_pais[5]}
""")

match opcion_olimpiada:
    case "Sidney":
        anio_olimpiada="2000"
    case "Atenas":
        anio_olimpiada="2004"
    case "Beijing":
        anio_olimpiada="2008"
    case "Londres":
        anio_olimpiada="2012"
    case "Tokyo":
        anio_olimpiada="2020"
    case "Paris":
        anio_olimpiada="2024"
    case "":
        anio_olimpiada="2012" #El default de la API
```



Mediante un input y match se permite al usuario elegir la Olimpiada de la que se obtendrán los datos

# Usando Olympic Sports API se obtiene un JSON que trae datos de:

Categories > Sports >  Olympic Sports API > GET Medal By Countries

```
## Importar librerias a usar
import requests
import pandas as pd
import json
```

```
## Funcion para importar usando una API los datos de los ganadores de las medallas olimpicas (Oro,plata,Bronce)
def carga_datos_API(anio_consulta):
    url = "https://olympic-sports-api.p.rapidapi.com/medals/countries"

    ## querystring = {"year":"2012"} marca el año de la olimpiada, si es inválido marca 2012 de default
    querystring= {"year":anio_consulta}

    headers = {
        "x-rapidapi-key": "30a604f21amshb618d82fe73fc18p10597ejsnd68459afa5b1",
        "x-rapidapi-host": "olympic-sports-api.p.rapidapi.com"
    }

    response = requests.get(url, headers=headers, params=querystring)
    data_response=json.loads(response.text)
    return data_response
```

01

## Olimpiada(año)

02

## Lideres

03

## Medallas Oro

04

## Medallas Plata

05

## Medallas Bronce

06

## Medallas Totales

```
{ "year": "2012",
  "leaders": { },
  "medals": [ ],
  "table": { }
    "cols": { }
      "country": { },
      "gold": { },
      "silver": { },
      "bronze": { },
      "total": { }
    },
    "sortOrder": [ ]
      "country",
      "gold",
      "silver",
      "bronze",
      "total"
    ]
}
```

Función que llama a API cargando la información de medallas ganadas por país en la Olimpiada elegida, para esto se pasa el parámetro año.

# SERIES

```
## Manipular la información obtenida por la API  
data_olimpiada=data["year"]  
data_lideres=data["leaders"]  
data_medallas=data["medals"]  
#print(data_medallas)
```

✓ 0.1s

```
## Obtener series y dataframes  
datos={}  
  
#variables tipo lista para armar series del dataframe  
  
lista_paises=[]  
lista_medallas_oro=[]  
lista_medallas_plata=[]  
lista_medallas_bronce=[]  
  
for registro in data_medallas:  
    lista_paises.append(registro["name"])  
    lista_medallas_oro.append(registro["medalStandings"]["goldMedalCount"])  
    lista_medallas_plata.append(registro["medalStandings"]["silverMedalCount"])  
    lista_medallas_bronce.append(registro["medalStandings"]["bronzeMedalCount"])
```

Se recorre el JSON generado desde la API y se arman listas para obtener la información requerida registro por registro

# Dataframe

```
# Con las listas armadas al recorrer la información de la API, obtenemos el DataFrame  
datos={"País":lista_paises,"Medallas_Oro":lista_medallas_oro,"Medallas_Plata":lista_medallas_plata,"Medallas_Bronce":lista_medallas_bronce}  
df=pd.DataFrame(datos)  
df
```

	País	Medallas_Oro	Medallas_Plata	Medallas_Bronce
0	United States	46	29	29
1	China	38	27	23
2	Russia	24	26	32
3	Great Britain	29	17	19
4	Germany	11	19	14
--	--	--	--	--
80	Saudi Arabia	0	0	1
81	Afghanistan	0	0	1
82	Tajikistan	0	0	1
83	Kuwait	0	0	1
84	Hong Kong	0	0	1

Con las listas creadas con anterioridad, se genera el Dataframe



	Pais	Medallas_Oro	Medallas_Plata	Medallas_Bronce
0	United States	46	29	29
1	China	38	27	23
2	Russia	24	26	32
3	Great Britain	29	17	19
4	Germany	11	19	14
5	Japan	7	14	17
6	Australia	7	16	12
7	France	11	11	12
8	South Korea	13	8	7
9	Italy	8	9	11

USA=51736738

	Pais	Medallas_Oro	Medallas_Plata	Medallas_Bronce	PIB
64	Qatar	0	0	2	101933117
42	Norway	2	1	1	101779186
41	Switzerland	2	2	0	86284413
6	Australia	7	16	12	68485597
27	Denmark	2	4	3	58623414
28	Sweden	1	4	3	57816023
63	Singapore	0	0	2	55547546
13	Canada	1	5	12	52745057
10	Netherlands	6	6	8	50175559
5	Japan	7	14	17	49175035

