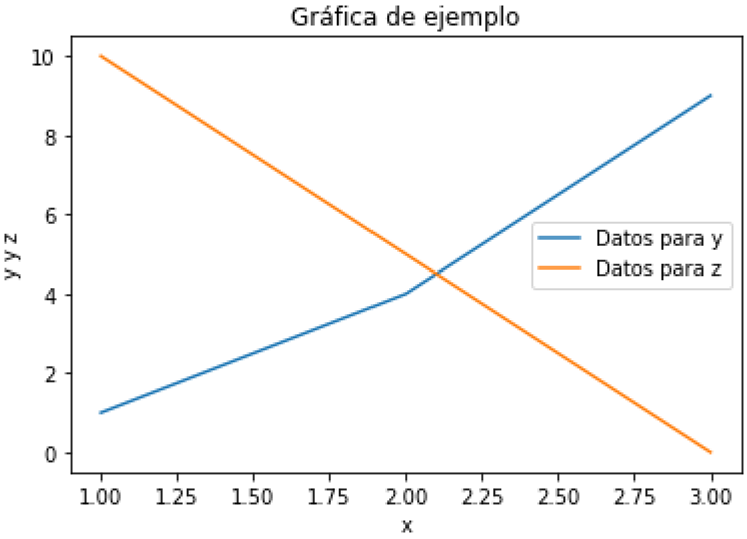
Ejercicio PANDAS

Andrea González Arredondo A01351820 Xavier Alfonso Barrera Ruiz A01702869
Olivia Araceli Morales Quezada A01707371 José Pablo Cabos Austria A01274631 RETO 6

1.-En este primer pequeño programa podemos observar como al principio importamosciertas librerías para que pueda funcionar, despúés le asignamos ciertos vectores a las variables, y con ello graficamos, lo demás son letreros que se encuentran en la gráfica.

In [108…

```
import pandas as pd
from matplotlib import pyplot as plt
x =[1, 2, 3]
y = [1, 4, 9]
z = [10, 5, 0]
plt.plot (x,y)
plt.plot (x,z)
plt.title("Gráfica de ejemplo")
plt.xlabel("x")
plt.ylabel("y y z")
plt.legend(["Datos para y", "Datos para z"])
plt.show()
```



2.- Para esta parte usamos una función de la librería panda para que leyera nuestro archivo.

In [109…

```
data=pd.read_csv('Videojuegos.csv')
data
```

Out[109…

	Platform	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Rating	C
0	Wii	Sports	Nintendo	41.36	28.96	3.77	8.45	82.54	E	
1	Wii	Racing	Nintendo	15.68	12.80	3.79	3.29	35.57	E	
2	Wii	Sports	Nintendo	15.61	10.95	3.28	2.95	32.78	E	
3	DS	Platform	Nintendo	11.28	9.15	6.50	2.88	29.81	E	
4	Wii	Misc	Nintendo	13.96	9.18	2.93	2.84	28.92	E	
...	...	...	...	...	...	...	...	...	...	

	Platform	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Rating	C
7107	PC	Action	Ubisoft	0.00	0.00	0.00	0.00	0.01	T	
7108	PC	Shooter	Midway Games	0.00	0.00	0.00	0.00	0.01	T	
7109	PC	Sports	Sega	0.00	0.00	0.00	0.00	0.01	E	
7110	PC	Strategy	Take-Two Interactive	0.00	0.00	0.00	0.00	0.01	E10+	
7111	PS4	Platform	Team Meat	0.00	0.00	0.00	0.00	0.01	T	

7112 rows × 10 columns

3.- En la siguiente parte del código, usamos "len" para saber de que tamaño es nuestra tabla, y cuántas filas existen

```
In [110…  filas = len(data)
          filas
```

```
Out[110…  7112
```

4.- Con las siguientes dos partes del código que usan "type", lo usamos para saber de que tipo de variable se está usando

```
In [73]:  type (data)
```

```
Out[73]:  pandas.core.frame.DataFrame
```

```
In [111…  type (data.Platform)
          type (data.Genre)
          type (data.Publisher)
          type (data.NA_Sales)
          type (data.EU_Sales)
          type (data.JP_Sales)
          type (data.Other_Sales)
          type (data.Global_Sales)
          type (data.Rating)
          type (data.Critic_Score_Class)
```

```
Out[111…  pandas.core.series.Series
```

5.- La siguientes variables que contienen este formato "data.nombre_de_columna", es para analizar las variables y buscar el rancgo en el que se encuentra.

```
In [75]:  data.Platform
```

```
Out[75]:  0       Wii
          1       Wii
          2       Wii
          3        DS
          4       Wii
                 ...
          7107     PC
```

```
7108     PC
7109     PC
7110     PC
7111    PS4
Name: Platform, Length: 7112, dtype: object
```

In [41]:

```
data.Genre
```

Out[41]:

```
0         Sports
1         Racing
2         Sports
3       Platform
4           Misc
          ...   
7107      Action
7108     Shooter
7109      Sports
7110    Strategy
7111    Platform
Name: Genre, Length: 7112, dtype: object
```

In [43]:

```
data.Publisher
```

Out[43]:

```
0                   Nintendo
1                   Nintendo
2                   Nintendo
3                   Nintendo
4                   Nintendo
                ...         
7107                 Ubisoft
7108            Midway Games
7109                    Sega
7110    Take-Two Interactive
7111               Team Meat
Name: Publisher, Length: 7112, dtype: object
```

In [44]:

```
data.NA_Sales
```

Out[44]:

```
0       41,36
1       15,68
2       15,61
3       11,28
4       13,96
        ...  
7107        0
7108        0
7109        0
7110        0
7111        0
Name: NA_Sales, Length: 7112, dtype: object
```

In [45]:

```
data.EU_Sales
```

Out[45]:

```
0       28,96
1        12,8
2       10,95
3        9,15
4        9,18
        ...  
7107        0
7108        0
7109        0
7110        0
```

```
7111        0
Name: EU_Sales, Length: 7112, dtype: object
```

In [46]:
```
data.JP_Sales
```

Out[46]:
```
0       3,77
1       3,79
2       3,28
3        6,5
4       2,93
        ...
7107       0
7108       0
7109       0
7110       0
7111       0
Name: JP_Sales, Length: 7112, dtype: object
```

In [47]:
```
data.Other_Sales
```

Out[47]:
```
0       8,45
1       3,29
2       2,95
3       2,88
4       2,84
        ...
7107       0
7108       0
7109       0
7110       0
7111       0
Name: Other_Sales, Length: 7112, dtype: object
```

In [49]:
```
data.Global_Sales
```

Out[49]:
```
0       82,54
1       35,57
2       32,78
3       29,81
4       28,92
        ...
7107     0,01
7108     0,01
7109     0,01
7110     0,01
7111     0,01
Name: Global_Sales, Length: 7112, dtype: object
```

In [50]:
```
data.Rating
```

Out[50]:
```
0          E
1          E
2          E
3          E
4          E
        ...
7107       T
7108       T
7109       E
7110    E10+
7111       T
Name: Rating, Length: 7112, dtype: object
```

In [112…
```
data.Critic_Score_Class
```

Out[112…

```
0           Bueno
1       Excelente
2       Excelente
3       Excelente
4            Malo
          ...    
7107    Excelente
7108    Aceptable
7109    Aceptable
7110    Aceptable
7111    Excelente
Name: Critic_Score_Class, Length: 7112, dtype: object
```

6.- Esta parte que sigue es para conocer cuáles columnas están en nuestra base de datos.

In [113…

```
data.columns
```

Out[113…

```
Index(['Platform', 'Genre', 'Publisher', 'NA_Sales', 'EU_Sales', 'JP_Sales',
       'Other_Sales', 'Global_Sales', 'Rating', 'Critic_Score_Class'],
      dtype='object')
```

7.- El formato siguiente de data.(nombre_de_columna).median, es para calcular la mediana de cada columna que está en la base de datos.

In [97]:

```
data['Platform'].median
```

Out[97]:

```
<bound method Series.median of 0       Wii
1       Wii
2       Wii
3        DS
4       Wii
       ... 
7107     PC
7108     PC
7109     PC
7110     PC
7111    PS4
Name: Platform, Length: 7112, dtype: object>
```

In [98]:

```
data['Genre'].median
```

Out[98]:

```
<bound method Series.median of 0         Sports
1         Racing
2         Sports
3       Platform
4           Misc
          ...   
7107      Action
7108     Shooter
7109      Sports
7110    Strategy
7111    Platform
Name: Genre, Length: 7112, dtype: object>
```

In [99]:

```
data['Publisher'].median
```

Out[99]:

```
<bound method Series.median of 0                   Nintendo
1                   Nintendo
2                   Nintendo
3                   Nintendo
4                   Nintendo
                ...
```

```
7107                  Ubisoft
7108             Midway Games
7109                     Sega
7110    Take-Two Interactive
7111                Team Meat
Name: Publisher, Length: 7112, dtype: object>
```

In [100…

```
data['NA_Sales'].median
```

Out[100…

```
<bound method Series.median of 0       41.36
1       15.68
2       15.61
3       11.28
4       13.96
        ...  
7107     0.00
7108     0.00
7109     0.00
7110     0.00
7111     0.00
Name: NA_Sales, Length: 7112, dtype: float64>
```

In [101…

```
data['EU_Sales'].median
```

Out[101…

```
<bound method Series.median of 0       28.96
1       12.80
2       10.95
3        9.15
4        9.18
        ...  
7107     0.00
7108     0.00
7109     0.00
7110     0.00
7111     0.00
Name: EU_Sales, Length: 7112, dtype: float64>
```

In [102…

```
data['JP_Sales'].median
```

Out[102…

```
<bound method Series.median of 0       3.77
1       3.79
2       3.28
3       6.50
4       2.93
        ... 
7107    0.00
7108    0.00
7109    0.00
7110    0.00
7111    0.00
Name: JP_Sales, Length: 7112, dtype: float64>
```

In [103…

```
data['Other_Sales'].median
```

Out[103…

```
<bound method Series.median of 0       8.45
1       3.29
2       2.95
3       2.88
4       2.84
        ... 
7107    0.00
7108    0.00
7109    0.00
7110    0.00
```

```
7111    0.00
Name: Other_Sales, Length: 7112, dtype: float64>
```

In [104…

```
data['Global_Sales'].median
```

Out[104…

```
<bound method Series.median of 0       82.54
1       35.57
2       32.78
3       29.81
4       28.92
        ...
7107     0.01
7108     0.01
7109     0.01
7110     0.01
7111     0.01
Name: Global_Sales, Length: 7112, dtype: float64>
```

In [106…

```
data['Rating'].median
```

Out[106…

```
<bound method Series.median of 0          E
1          E
2          E
3          E
4          E
        ...
7107       T
7108       T
7109       E
7110    E10+
7111       T
Name: Rating, Length: 7112, dtype: object>
```

In [114…

```
data['Critic_Score_Class'].median
```

Out[114…

```
<bound method Series.median of 0           Bueno
1       Excelente
2       Excelente
3       Excelente
4            Malo
          ...
7107    Excelente
7108    Aceptable
7109    Aceptable
7110    Aceptable
7111    Excelente
Name: Critic_Score_Class, Length: 7112, dtype: object>
```

8.- El siguiente formato de función data(nombre_de_columna).describe es para encontrar ciertas características de cada columna, como la media, mediana, la frecuencia, máximos, mínimos y desviación estandar.

In [92]:

```
data['Platform'].describe()
```

Out[92]:

```
count     7112
unique      17
top        PS2
freq      1169
Name: Platform, dtype: object
```

In [77]:

```
data['Genre'].describe()
```

```
count       7112
```

```
Out[77]: unique        12
         top       Action
         freq        1698
         Name: Genre, dtype: object
```

```
In [57]:  data['Publisher'].describe()
```

```
Out[57]: count                 7112
         unique                 278
         top       Electronic Arts
         freq                   971
         Name: Publisher, dtype: object
```

```
In [83]:  data['NA_Sales'].describe()
```

```
Out[83]: count    7112.000000
         mean        0.388567
         std         0.953982
         min         0.000000
         25%         0.060000
         50%         0.150000
         75%         0.390000
         max        41.360000
         Name: NA_Sales, dtype: float64
```

```
In [84]:  data['EU_Sales'].describe()
```

```
Out[84]: count    7112.000000
         mean        0.232537
         std         0.680028
         min         0.000000
         25%         0.020000
         50%         0.060000
         75%         0.202500
         max        28.960000
         Name: EU_Sales, dtype: float64
```

```
In [86]:  data['JP_Sales'].describe()
```

```
Out[86]: count    7112.000000
         mean        0.062652
         std         0.283475
         min         0.000000
         25%         0.000000
         50%         0.000000
         75%         0.010000
         max         6.500000
         Name: JP_Sales, dtype: float64
```

```
In [85]:  data['Other_Sales'].describe()
```

```
Out[85]: count    7112.000000
         mean        0.081347
         std         0.265864
         min         0.000000
         25%         0.010000
         50%         0.020000
         75%         0.070000
         max        10.570000
         Name: Other_Sales, dtype: float64
```

```
In [87]:  data['Global_Sales'].describe()
```

```
Out[87]: count    7112.000000
         mean        0.765307
         std         1.936692
         min         0.010000
         25%         0.110000
         50%         0.290000
         75%         0.742500
         max        82.540000
         Name: Global_Sales, dtype: float64
```

```
In [88]: data['Rating'].describe()
```

```
Out[88]: count     7112
         unique       7
         top          T
         freq      2489
         Name: Rating, dtype: object
```

```
In [69]: data['Critic_Score_Class'].describe()
```

```
Out[69]: count           7112
         unique             4
         top        Excelente
         freq            1997
         Name: Critic_Score_Class, dtype: object
```