In [38]:

```
import pandas as pd
import numpy as np
import math as mt

def reemplazoVacios(vector, promedio):
    for i in range(len(vector)):
        if mt.isnan(vector[i]):
            print(i)
            vector[i] = promedio

notas = pd.read_csv("datos_notas_PC1_PC2_EP.csv")
notas_aux = notas
notasEF = pd.read_csv("datos_notas_EF.csv")
print(notasEF.shape)
notas.head()

#-------#

print(notas)
print("\n")
print(notas["PC01"])
print("\n")
print(notas["PC01"].array)
print("\n")

#--Deshacerse de los vacios: drop--#
print("-----Nulos por columnas-----")
print(len(notas["PC01"][notas["PC01"].isnull()]))
print(len(notas["PC02"][notas["PC02"].isnull()]))
print(len(notas["EP"][notas["EP"].isnull()]))

notas1 = notas.dropna()
print(notas1)

#-Promedio de notas PC1-#
print("\nPromedio de PC1")
prom_pc1 = np.mean(notas1["PC01"])
prom_pc1 = round(prom_pc1, 2)
print(prom_pc1)

#--Bucles--#
s = 0
notas_aux = notas1["PC01"].array
for i in range(len(notas_aux)):
    s = s + notas_aux[i]

prom1 = s/len(notas_aux)

print(round(prom1, 2))

#-Promedio de notas PC2-#
print("\nPromedio de PC2")
prom_pc2 = np.mean(notas1["PC02"])
prom_pc2 = round(prom_pc2, 2)
print(prom_pc2)

#-Promedio de notas EP-#
print("\nPromedio de EP")
prom_ep = np.mean(notas1["EP"])
prom_ep = round(prom_ep, 2)
print(prom_ep)
```

```
#-Reemplazo por el promedio-#
print("\nReemplazo de notas PC01")
notas_aux = notas["PC01"].array
reemplazoVacios(notas_aux, prom_pc1)
notas["PC01"] = notas_aux
print(len(notas["PC01"][notas["PC01"].isnull()]))

print("\nReemplazo de notas PC02")
notas_aux = notas["PC02"].array
reemplazoVacios(notas_aux, prom_pc2)
notas["PC02"] = notas_aux
print(len(notas["PC02"][notas["PC02"].isnull()]))

print("\nReemplazo de notas EP")
notas_aux = notas["EP"].array
reemplazoVacios(notas_aux, prom_ep)
notas["EP"] = notas_aux
print(len(notas["EP"][notas["EP"].isnull()]))

print(notas)
```

```
(135, 4)
            Apellidos   Nombres Seccion  PC01  PC02    EP
0    Vergara Bautista     Elena       B  15.0   9.0   5.0
1        Pont Ramirez     David       A  16.0  12.0  13.0
2    Escolano Palacio   Araceli       B  16.0  17.0   7.0
3        Catalán Nole    Bernie       A  17.0  18.0  11.0
4      Vaquero Querol      Paco       C  17.0  17.0  18.0
..                ...       ...     ...   ...   ...   ...
145  Vallenas Atienza      Cloe       C  19.0   8.0  13.0
146         Gual Soto   Cecilia       D  18.0  13.0   5.0
147     Batalla Gallo     Delia       D  20.0  10.0   5.0
148   Cabezas Beltrán      José       B  16.0  12.0   9.0
149     Morera Teruel  Clemente       D  12.0   7.0  17.0

[150 rows x 6 columns]


0      15.0
1      16.0
2      16.0
3      17.0
4      17.0
       ...
145    19.0
146    18.0
147    20.0
148    16.0
149    12.0
Name: PC01, Length: 150, dtype: float64


<PandasArray>
[15.0, 16.0, 16.0, 17.0, 17.0, 10.0, 13.0, 13.0,  nan, 10.0,
 ...
 17.0, 17.0, 18.0, 15.0, 18.0, 19.0, 18.0, 20.0, 16.0, 12.0]
Length: 150, dtype: float64


-----Nulos por columnas-----
4
3
4
            Apellidos   Nombres Seccion  PC01  PC02    EP
0    Vergara Bautista     Elena       B  15.0   9.0   5.0
1        Pont Ramirez     David       A  16.0  12.0  13.0
2    Escolano Palacio   Araceli       B  16.0  17.0   7.0
3        Catalán Nole    Bernie       A  17.0  18.0  11.0
4      Vaquero Querol      Paco       C  17.0  17.0  18.0
..                ...       ...     ...   ...   ...   ...
145  Vallenas Atienza      Cloe       C  19.0   8.0  13.0
146         Gual Soto   Cecilia       D  18.0  13.0   5.0
147     Batalla Gallo     Delia       D  20.0  10.0   5.0
148   Cabezas Beltrán      José       B  16.0  12.0   9.0
149     Morera Teruel  Clemente       D  12.0   7.0  17.0

[139 rows x 6 columns]

Promedio de PC1
14.45
14.45

Promedio de PC2
13.54
```

```
Promedio de EP
11.23

Reemplazo de notas PC01
8
30
47
66
0

Reemplazo de notas PC02
36
57
88
0

Reemplazo de notas EP
5
15
116
142
0
```

	Apellidos	Nombres	Seccion	PC01	PC02	EP
0	Vergara Bautista	Elena	B	15.0	9.0	5.0
1	Pont Ramirez	David	A	16.0	12.0	13.0
2	Escolano Palacio	Araceli	B	16.0	17.0	7.0
3	Catalán Nole	Bernie	A	17.0	18.0	11.0
4	Vaquero Querol	Paco	C	17.0	17.0	18.0
..	...	...	...	...	...	...
145	Vallenas Atienza	Cloe	C	19.0	8.0	13.0
146	Gual Soto	Cecilia	D	18.0	13.0	5.0
147	Batalla Gallo	Delia	D	20.0	10.0	5.0
148	Cabezas Beltrán	José	B	16.0	12.0	9.0
149	Morera Teruel	Clemente	D	12.0	7.0	17.0

[150 rows x 6 columns]

In [33]:
```
#-Ordenamiento de datos-#
notas = notas.sort_values(["Seccion", "Apellidos"])
print(notas)
```

	Apellidos	Nombres	Seccion	PC01	PC02	EP
5	Alegria Carmona	Ignacio	A	10.00	8.0	11.23
38	Bonet Alvarez	Felicidad	A	12.00	16.0	18.00
3	Catalán Nole	Bernie	A	17.00	18.0	11.00
41	Chaves Gimeno	Elisa	A	12.00	14.0	9.00
51	Donoso Izquierdo	Alfonso	A	18.00	13.0	10.00
..	...	...	...	...	...	...
47	Ramos Puig	Juan	E	14.45	7.0	17.00
86	Rueda Casanovas	Leonel	E	11.00	13.0	8.00
6	Salazar Lopez	Rodrigo	E	13.00	12.0	7.00
48	Salinas López	Óscar	E	12.00	15.0	10.00
72	Sosa Barón	Alfonso	E	13.00	19.0	13.00

[150 rows x 6 columns]

In [59]:
```
#-Union de tablas notas, notasEF-#
print(notas)
print(notasEF)
notas_x = pd.merge(notas, notasEF, on = ["Apellidos", "Nombres", "Seccion"], how =
print(notas_x)

pos = []
print(len(notas_x["EF"][notas_x["EF"].isnull()]))
```

```
vx = notas_x["EF"].array

for i in range(len(vx)):
    if mt.isnan(vx[i]):
        print(i)

promT = (notas_x["PC01"] + notas_x["PC02"] + notas_x["EP"])/3
promT = round(promT, 3)

print(promT)
print(pos)

promT = promT.array

j = 0
for i in range(len(promT)):
    if i == pos[j]:
        vx[i] = promT[pos[i]]

        if pos[j] == 146:
            break
        j += 1

notas_x["EF1"] = vx
notas_x = notas_x.drop["EF", axis == 1]

print(vx)
```

```
           Apellidos   Nombres Seccion  PC01  PC02    EP
0    Vergara Bautista     Elena       B  15.0   9.0   5.0
1        Pont Ramirez     David       A  16.0  12.0  13.0
2    Escolano Palacio   Araceli       B  16.0  17.0   7.0
3        Catalán Nole    Bernie       A  17.0  18.0  11.0
4      Vaquero Querol      Paco       C  17.0  17.0  18.0
..                ...       ...     ...   ...   ...   ...
145  Vallenas Atienza      Cloe       C  19.0   8.0  13.0
146         Gual Soto   Cecilia       D  18.0  13.0   5.0
147     Batalla Gallo     Delia       D  20.0  10.0   5.0
148   Cabezas Beltrán      José       B  16.0  12.0   9.0
149     Morera Teruel  Clemente       D  12.0   7.0  17.0

[150 rows x 6 columns]
           Apellidos   Nombres Seccion  EF
0    Vergara Bautista     Elena       B  15
1        Pont Ramirez     David       A  11
2        Catalán Nole    Bernie       A  14
3      Vaquero Querol      Paco       C   8
4     Alegria Carmona   Ignacio       A   9
..                ...       ...     ...  ..
130  Vallenas Atienza      Cloe       C  12
131         Gual Soto   Cecilia       D  10
132     Batalla Gallo     Delia       D  15
133   Cabezas Beltrán      José       B  18
134     Morera Teruel  Clemente       D  12

[135 rows x 4 columns]
           Apellidos   Nombres Seccion  PC01  PC02    EP    EF
0    Vergara Bautista     Elena       B  15.0   9.0   5.0  15.0
1        Pont Ramirez     David       A  16.0  12.0  13.0  11.0
2    Escolano Palacio   Araceli       B  16.0  17.0   7.0   NaN
3        Catalán Nole    Bernie       A  17.0  18.0  11.0  14.0
4      Vaquero Querol      Paco       C  17.0  17.0  18.0   8.0
..                ...       ...     ...   ...   ...   ...   ...
145  Vallenas Atienza      Cloe       C  19.0   8.0  13.0  12.0
146         Gual Soto   Cecilia       D  18.0  13.0   5.0  10.0
147     Batalla Gallo     Delia       D  20.0  10.0   5.0  15.0
148   Cabezas Beltrán      José       B  16.0  12.0   9.0  18.0
149     Morera Teruel  Clemente       D  12.0   7.0  17.0  12.0

[150 rows x 7 columns]
15
2
11
16
18
21
38
53
60
74
86
95
106
110
137
140
0       9.667
1      13.667
2      13.333
3      15.333
4      17.333
        ...
```

```
145    13.333
146    12.000
147    11.667
148    12.333
149    12.000
Length: 150, dtype: float64
[]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[59], line 26
     24 j = 0
     25 for i in range(len(promT)):
---> 26     if i == pos[j]:
     27         vx[i] = promT[pos[i]]
     29         if pos[j] == 146:

IndexError: list index out of range
```

In [57]:
```
#-Ejemplos-#
datos = {"ventas": [10, 20, 30], "descuentos": [9, 3, 4], "bonus": [0, 0, 0]}
datos = pd.DataFrame(datos)
print(datos)

vz = datos["ventas"]*datos["descuentos"]+datos["bonus"]
datos["total"] = vz

print(datos)
```

```
   ventas  descuentos  bonus
0      10           9      0
1      20           3      0
2      30           4      0
   ventas  descuentos  bonus  total
0      10           9      0     90
1      20           3      0     60
2      30           4      0    120
```

In [58]:
```
print(notas_x)
```

```
           Apellidos   Nombres Seccion  PC01  PC02    EP    EF
0    Vergara Bautista     Elena       B  15.0   9.0   5.0  15.0
1        Pont Ramirez     David       A  16.0  12.0  13.0  11.0
2    Escolano Palacio   Araceli       B  16.0  17.0   7.0   NaN
3        Catalán Nole    Bernie       A  17.0  18.0  11.0  14.0
4      Vaquero Querol      Paco       C  17.0  17.0  18.0   8.0
..                ...       ...     ...   ...   ...   ...   ...
145  Vallenas Atienza      Cloe       C  19.0   8.0  13.0  12.0
146         Gual Soto   Cecilia       D  18.0  13.0   5.0  10.0
147     Batalla Gallo     Delia       D  20.0  10.0   5.0  15.0
148   Cabezas Beltrán      José       B  16.0  12.0   9.0  18.0
149     Morera Teruel  Clemente       D  12.0   7.0  17.0  12.0

[150 rows x 7 columns]
```

In [60]:
```
NF = 0.3*(notas_x["PC01"] + notas_x["PC02"]/2 + 0.3*notas_x["EP"] + 0.4*notas_x["E

notas_x["NF"] = NF
print(notas_x)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~\anaconda3\lib\site-packages\pandas\core\indexes\base.py:3802, in Index.get_
loc(self, key, method, tolerance)
   3801 try:
-> 3802     return self._engine.get_loc(casted_key)
   3803 except KeyError as err:

File ~\anaconda3\lib\site-packages\pandas\_libs\index.pyx:138, in pandas._libs.ind
ex.IndexEngine.get_loc()

File ~\anaconda3\lib\site-packages\pandas\_libs\index.pyx:165, in pandas._libs.ind
ex.IndexEngine.get_loc()

File pandas\_libs\hashtable_class_helper.pxi:5745, in pandas._libs.hashtable.PyObj
ectHashTable.get_item()

File pandas\_libs\hashtable_class_helper.pxi:5753, in pandas._libs.hashtable.PyObj
ectHashTable.get_item()

KeyError: 'EF1'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[60], line 1
----> 1 NF = 0.3*(notas_x["PC01"] + notas_x["PC02"]/2 + 0.3*notas_x["EP"] + 0.4*no
tas_x["EF1"])
      3 notas_x["NF"] = NF
      4 print(notas_x)

File ~\anaconda3\lib\site-packages\pandas\core\frame.py:3807, in DataFrame.__getit
em__(self, key)
   3805 if self.columns.nlevels > 1:
   3806     return self._getitem_multilevel(key)
-> 3807 indexer = self.columns.get_loc(key)
   3808 if is_integer(indexer):
   3809     indexer = [indexer]

File ~\anaconda3\lib\site-packages\pandas\core\indexes\base.py:3804, in Index.get_
loc(self, key, method, tolerance)
   3802     return self._engine.get_loc(casted_key)
   3803 except KeyError as err:
-> 3804     raise KeyError(key) from err
   3805 except TypeError:
   3806     # If we have a listlike key, _check_indexing_error will raise
   3807     #  InvalidIndexError. Otherwise we fall through and re-raise
   3808     #  the TypeError.
   3809     self._check_indexing_error(key)

KeyError: 'EF1'
```

In [64]:

```
print(notas_x)

promedios = notas_x.groupby(["Seccion"]).mean()
print(promedios)

cantidad = notas_x.groupby(["Seccion"]).count()
print(cantidad)
cantidadAprobados = notas_x.groupby(["Seccion", "condicion"]).count()
print(cantidadAprobados)
```

```
           Apellidos   Nombres Seccion  PC01  PC02    EP    EF
0    Vergara Bautista     Elena       B  15.0   9.0   5.0  15.0
1        Pont Ramirez     David       A  16.0  12.0  13.0  11.0
2    Escolano Palacio   Araceli       B  16.0  17.0   7.0   NaN
3        Catalán Nole    Bernie       A  17.0  18.0  11.0  14.0
4      Vaquero Querol      Paco       C  17.0  17.0  18.0   8.0
..                ...       ...     ...   ...   ...   ...   ...
145  Vallenas Atienza      Cloe       C  19.0   8.0  13.0  12.0
146         Gual Soto   Cecilia       D  18.0  13.0   5.0  10.0
147     Batalla Gallo     Delia       D  20.0  10.0   5.0  15.0
148   Cabezas Beltrán      José       B  16.0  12.0   9.0  18.0
149     Morera Teruel  Clemente       D  12.0   7.0  17.0  12.0

[150 rows x 7 columns]
              PC01       PC02         EP         EF
Seccion
A        14.766667  13.966667  11.974333  12.428571
B        14.315000  13.166667  10.956333  12.080000
C        15.133333  13.851333  12.633333  11.535714
D        14.357812  12.940000  10.312500  11.965517
E        13.210714  13.428571  10.642857  12.560000
         Apellidos  Nombres  PC01  PC02  EP  EF
Seccion
A               30       30    30    30  30  28
B               30       30    30    30  30  25
C               30       30    30    30  30  28
D               32       32    32    32  32  29
E               28       28    28    28  28  25
C:\Users\PROFESOR\AppData\Local\Temp\ipykernel_11528\851056341.py:3: FutureWarnin
g: The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a
future version, numeric_only will default to False. Either specify numeric_only or
select only columns which should be valid for the function.
  promedios = notas_x.groupby(["Seccion"]).mean()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[64], line 8
      6 cantidad = notas_x.groupby(["Seccion"]).count()
      7 print(cantidad)
----> 8 cantidadAprobados = notas_x.groupby(["Seccion", "condicion"]).count()
      9 print(cantidadAprobados)

File ~\anaconda3\lib\site-packages\pandas\core\frame.py:8402, in DataFrame.groupby
(self, by, axis, level, as_index, sort, group_keys, squeeze, observed, dropna)
   8399     raise TypeError("You have to supply one of 'by' and 'level'")
   8400 axis = self._get_axis_number(axis)
-> 8402 return DataFrameGroupBy(
   8403     obj=self,
   8404     keys=by,
   8405     axis=axis,
   8406     level=level,
   8407     as_index=as_index,
   8408     sort=sort,
   8409     group_keys=group_keys,
   8410     squeeze=squeeze,
   8411     observed=observed,
   8412     dropna=dropna,
   8413 )

File ~\anaconda3\lib\site-packages\pandas\core\groupby\groupby.py:965, in GroupBy.
__init__(self, obj, keys, axis, level, grouper, exclusions, selection, as_index, s
ort, group_keys, squeeze, observed, mutated, dropna)
    962 if grouper is None:
    963     from pandas.core.groupby.grouper import get_grouper
--> 965     grouper, exclusions, obj = get_grouper(
    966         obj,
    967         keys,
    968         axis=axis,
    969         level=level,
    970         sort=sort,
    971         observed=observed,
    972         mutated=self.mutated,
    973         dropna=self.dropna,
    974     )
    976 self.obj = obj
    977 self.axis = obj._get_axis_number(axis)

File ~\anaconda3\lib\site-packages\pandas\core\groupby\grouper.py:888, in get_grou
per(obj, key, axis, level, sort, observed, mutated, validate, dropna)
    886         in_axis, level, gpr = False, gpr, None
    887     else:
--> 888         raise KeyError(gpr)
    889 elif isinstance(gpr, Grouper) and gpr.key is not None:
    890     # Add key to exclusions
    891     exclusions.add(gpr.key)

KeyError: 'condicion'
```

In [ ]: