Trabajo Práctico Final

Nombre: Ivanna López Barbona

Comisión: 22616

Docente: Matías Ignacio Escobar

Consigna

Mejorar/agregar contexto y estilo a las data-viz. - Formular 3 preguntas y resolverlas.

Análisis Exploratorio

El análisis exploratorio de datos es la forma de entender los sets de datos obteniendo un resumen de sus características principales. Esta exploración puede ser tanto analítica como visual.

```
# 1. Importar las librerias
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# Las 5 primeras filas
df.head()
```

	id object	gender object	race/ethnicity obj...	parental level of ...	lunch object	empl
0	10-5894942	male	group A	high school	standard	yes
1	41-1676468	female	group D	some high school	free/reduced	no
2	64-6396924	male	group E	some college	free/reduced	no
3	35-2426788	male	group B	high school	standard	yes
4	60-9387304	male	group E	associate's degree	standard	yes

```
# Las últimas 5 filas
df.tail()
```

	id object	gender object	race/ethnicity obj...	parental level of ...	lunch object	empl
1013	82-7312119	male	group E	associate's degree	standard	yes
1014	45-3445439	male	group E	some college	free/reduced	no
1015	02-3651562	male	group A	some college	standard	no
1016	05-5203587	female	group B	some college	standard	yes
1017	13-3347050	male	group D	some college	standard	no

```
# Resúmen estadístico
```

```
df.describe()
```

	math score float64	physics score flo...	chemistry score f...	algebra_score flo...	
count	1011.0	1011.0	1011.0	1011.0	
mean	66.4807121661721 1	69.063303659742 83	67.7893175074184	67.7784371909000 9	
std	15.3268797043793 37	14.6941070078516 35	15.5598532861405 2	14.4506798610410 94	
min	13.0	27.0	23.0	22.0	
25%	56.0	60.0	58.0	59.0	
50%	67.0	70.0	68.0	68.0	
75%	77.0	79.0	79.0	78.0	
max	100.0	100.0	100.0	100.0	

```
# 3. Revisar los tipos de datos
df.dtypes
```

```
id                             object
gender                         object
race/ethnicity                 object
parental level of education    object
lunch                          object
employed                       object
test preparation course        object
math score                    float64
physics score                 float64
chemistry score               float64
algebra_score                 float64
dtype: object
```

```
# 4. Eliminar los duplicados
print(f'Original: {df.id.count()} files')
duplicate_rows_df = df[df.duplicated() ]
print(f'Cantidad de filas duplicadas: {duplicate_rows_df.id.count()}')

# Eliminar los duplicados
df = df.drop_duplicates()
```

```
Original: 1018 files
Cantidad de filas duplicadas: 18
```

```
# Filas después de eliminar los duplicados
print(f'Final: {df.id.count()} files')
```

```
Final: 1000 files
```

```
# 5. Eliminar las columnas irrelevantes
print(df.columns)
df = df.drop(['id'], axis=1)
```

```
Index(['id', 'gender', 'race/ethnicity', 'parental level of education',
       'lunch', 'employed', 'test preparation course', 'math score',
       'physics score', 'chemistry score', 'algebra_score'],
      dtype='object')
```

```
print(df.columns)
```

```
Index(['gender', 'race/ethnicity', 'parental level of education', 'lunch',
       'employed', 'test preparation course', 'math score', 'physics score',
       'chemistry score', 'algebra_score'],
      dtype='object')
```

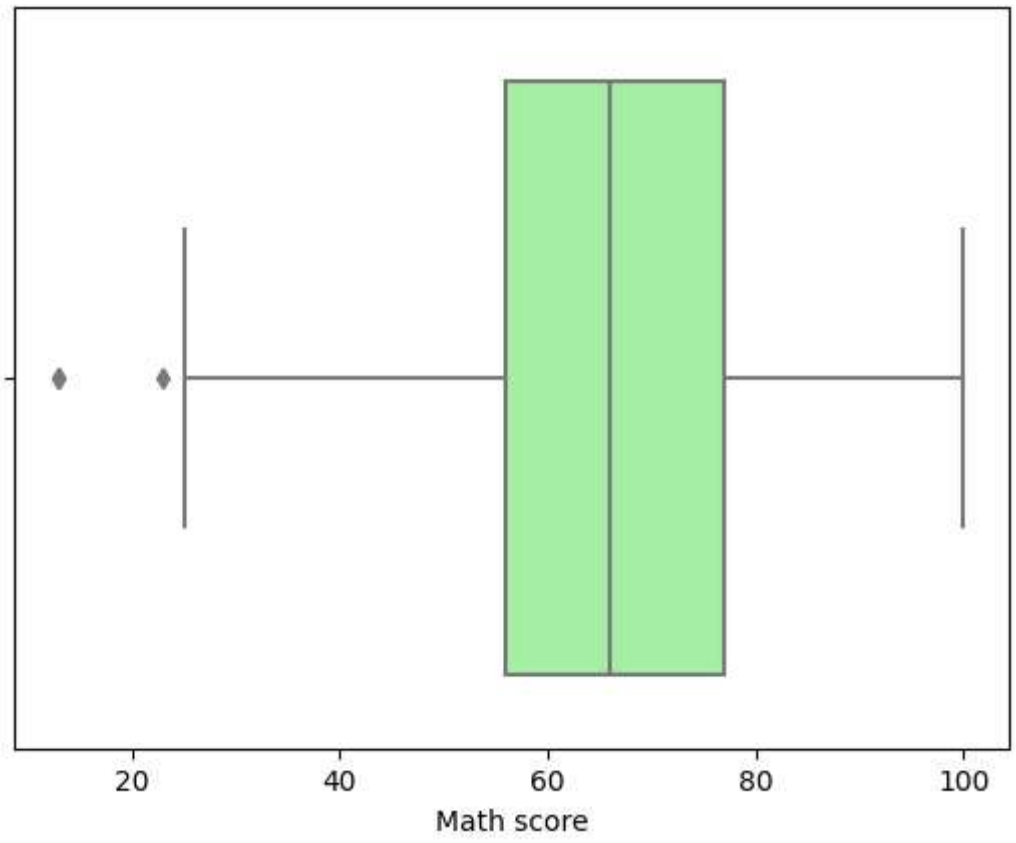
```
# 6. Renombrar las columnas
df = df.rename(columns = {
'gender' : 'Gender',
'race/ethnicity' : 'Ethnicity',
'parental level of education' : 'Parental level of education',
'lunch' :  'Lunch',
'employed' : 'Employed',
'test preparation course' : 'Test preparation course',
'math score' : 'Math score',
'physics score' : 'Physics score',
'chemistry score' : 'Chemistry score',
'algebra_score' : 'Algebra_score',})
df.columns
```

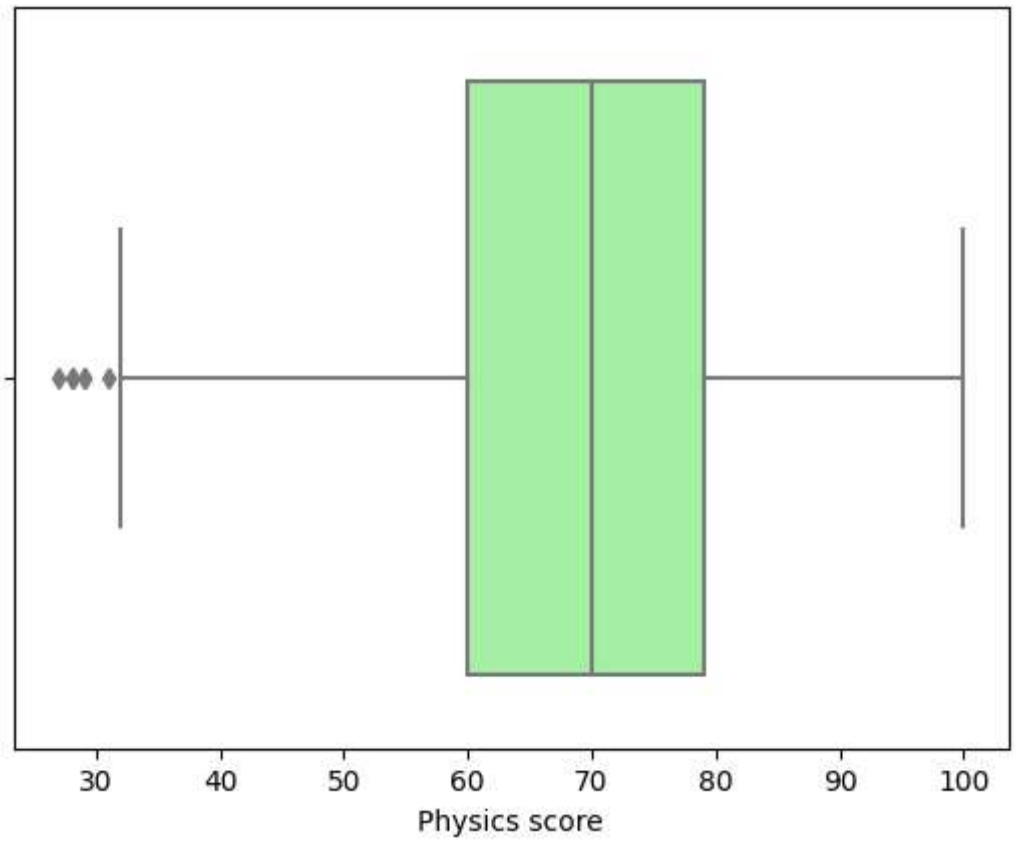
```
Index(['Gender', 'Ethnicity', 'Parental level of education', 'Lunch',
       'Employed', 'Test preparation course', 'Math score', 'Physics score',
       'Chemistry score', 'Algebra_score'],
      dtype='object')
```

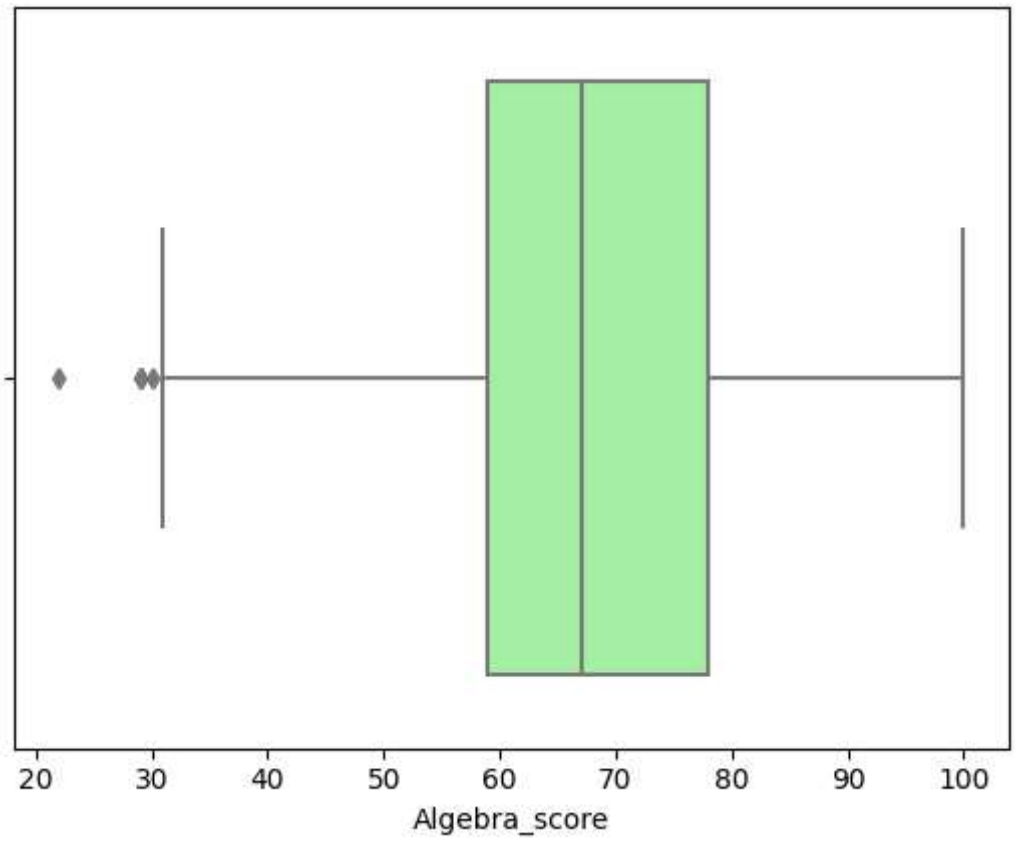
```python
# 7. Eliminar los valores perdidos o nulos
# Encontrar los valores nulos
print(df.isnull().sum())

#Eliminar los valores perdidos
df = df.dropna()
print()

# Después de eliminar los nulos
print(df.isnull().sum())
```

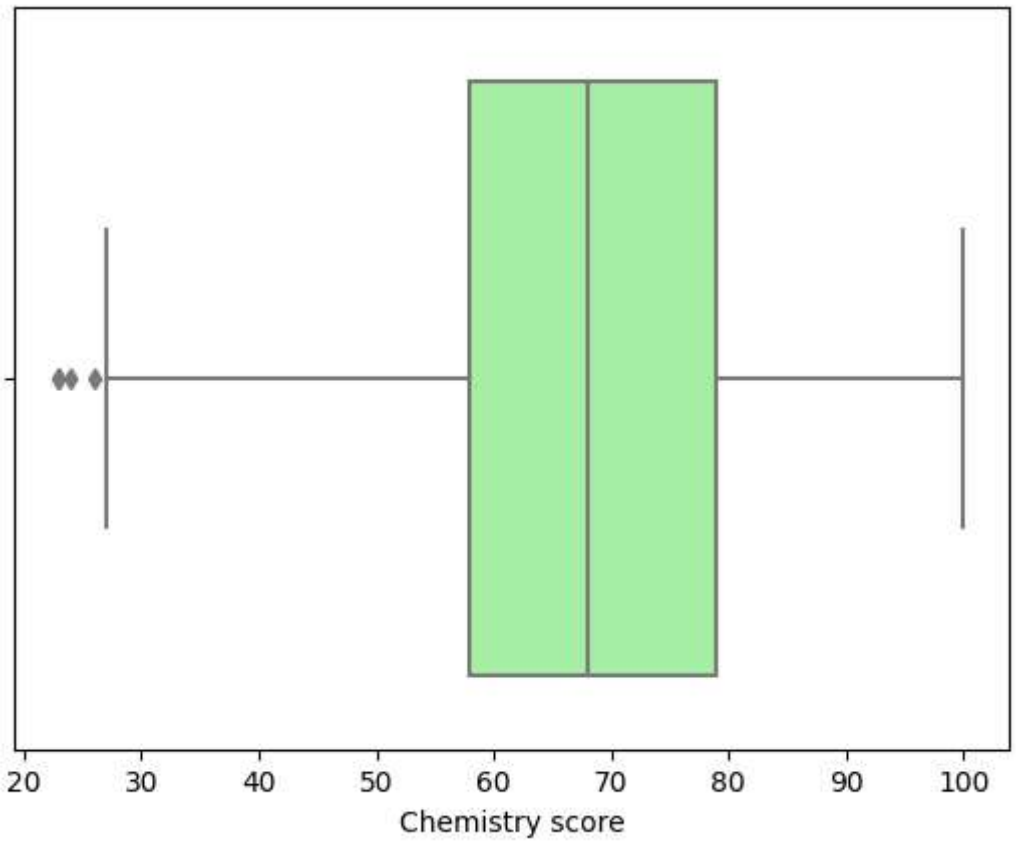
```
Gender                        0
Ethnicity                     0
Parental level of education   0
Lunch                         0
Employed                      0
Test preparation course       0
Math score                    7
Physics score                 7
Chemistry score               7
Algebra_score                 7
dtype: int64

Gender                        0
Ethnicity                     0
Parental level of education   0
Lunch                         0
Employed                      0
Test preparation course       0
Math score                    0
Physics score                 0
Chemistry score               0
Algebra_score                 0
dtype: int64
```

```python
# 8. Detectar los outliers
sns.boxplot(x=df['Math score'] , color= 'palegreen')
plt.show()
sns.boxplot(x=df['Physics score'] , color= 'palegreen')
plt.show()
sns.boxplot(x=df['Algebra_score'], color= 'palegreen')
plt.show()
sns.boxplot(x=df['Chemistry score'], color= 'palegreen')
plt.show()
```


20
40
60
80
100
Math score


30
40
50
60
70
80
90
100
Physics score

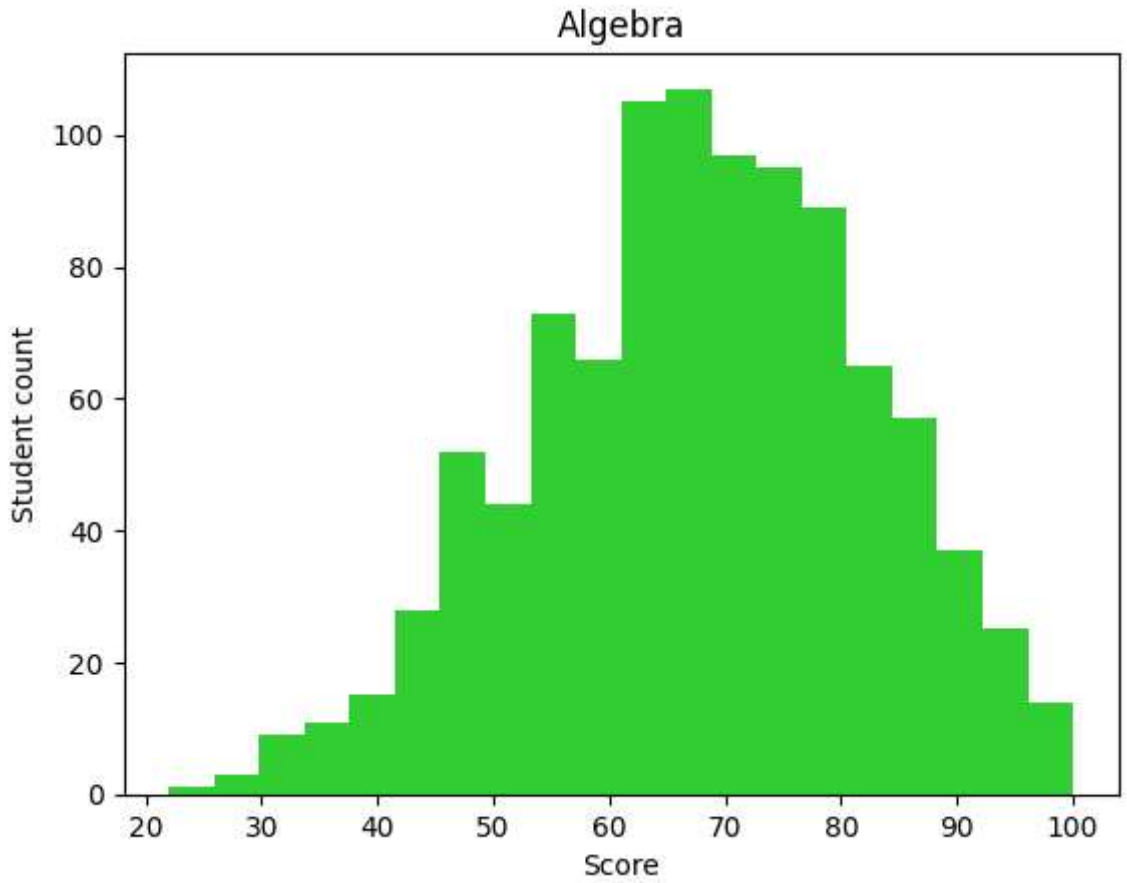

20
30
40
50
60
70
80
90
100
Algebra_score

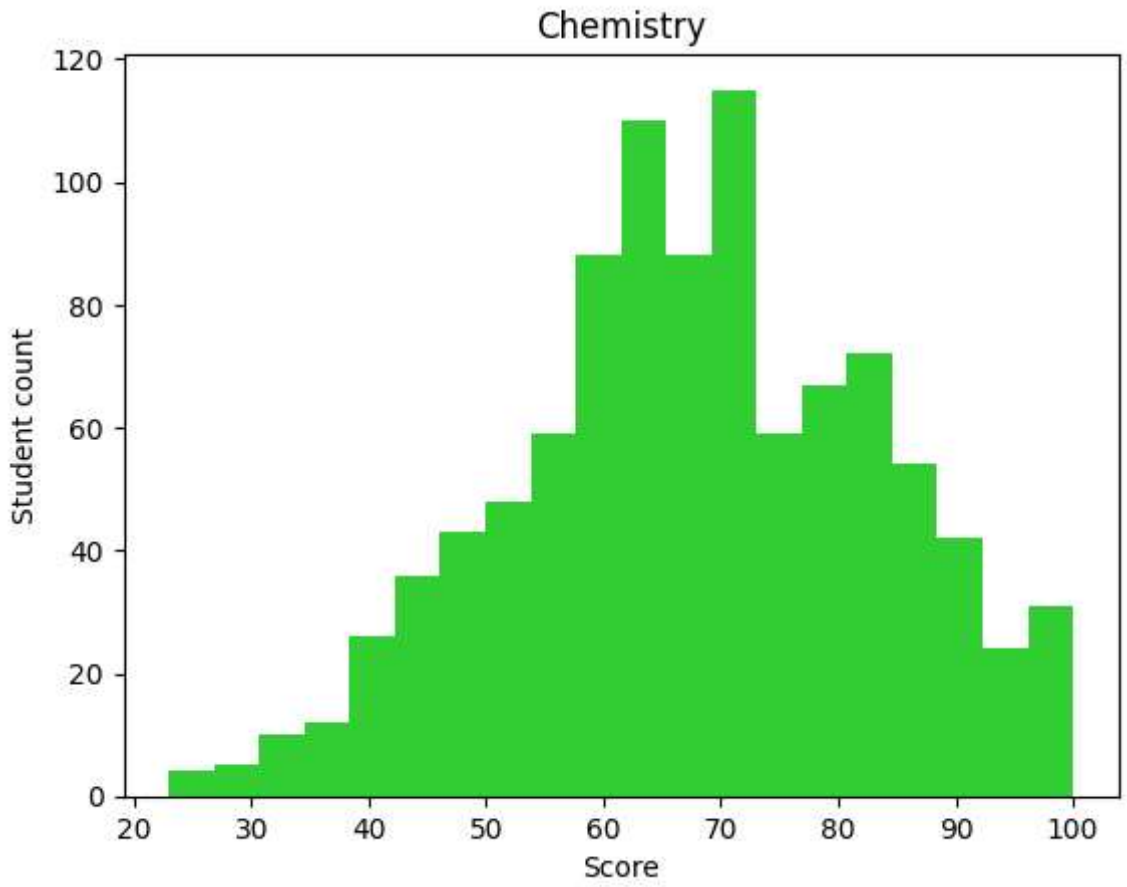

20
30
40
50
60
70
80
90
100
Chemistry score

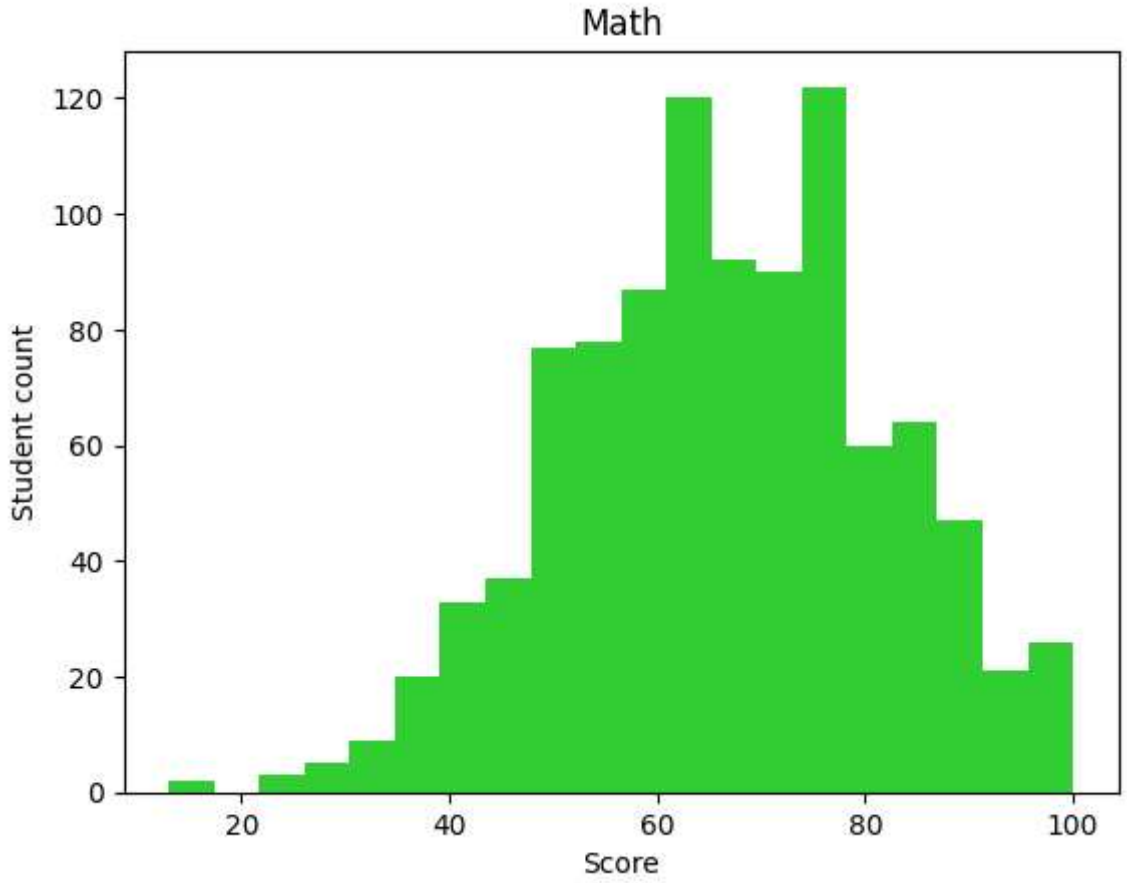
```python
print(f'Antes: {df.Lunch.count()} filas\n')
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 = Q1
print(IQR)
df = df[~((df < (Q1-1.5 * IQR)) |(df > (Q3 + 1.5 * IQR))).any(axis=1)]
print(f'\nDespués: {df.Lunch.count()} filas')
```

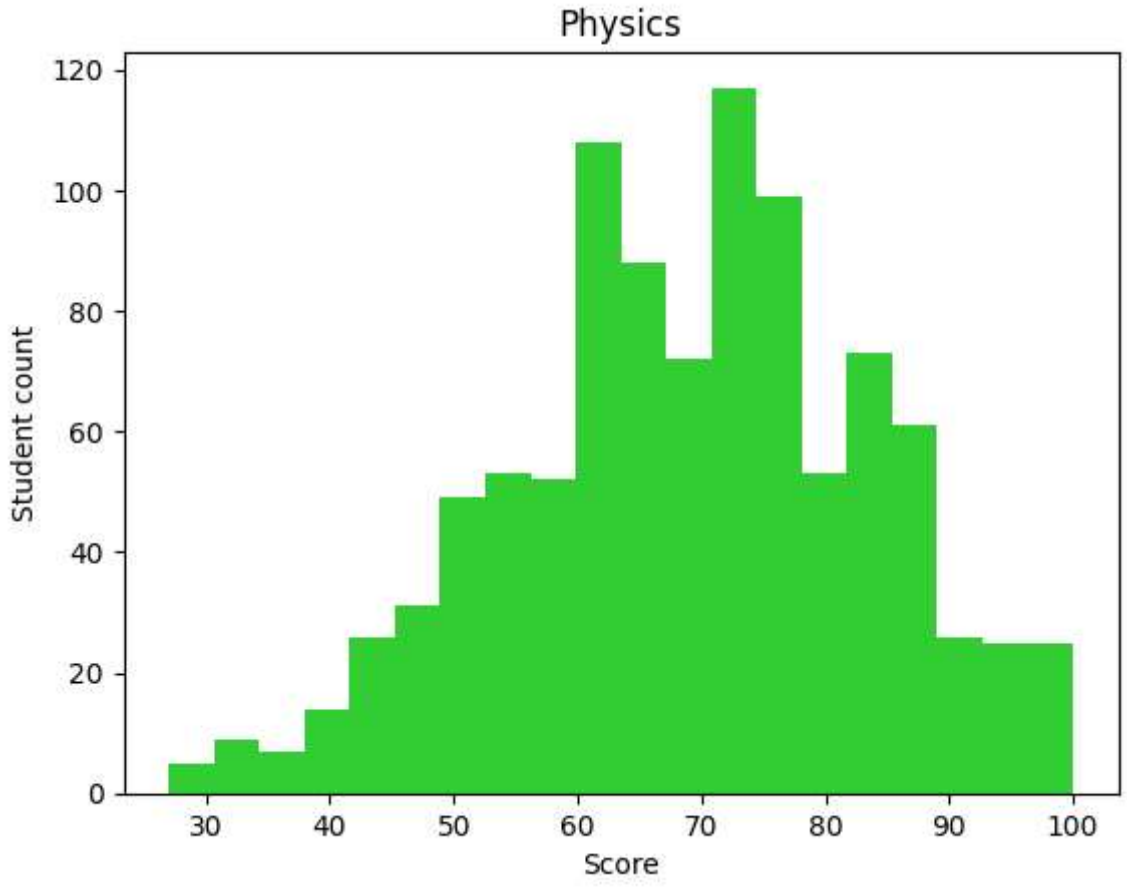
```
Antes: 993 filas

Math score         56.0
Physics score      60.0
Chemistry score    58.0
Algebra_score      59.0
Name: 0.25, dtype: float64

Después: 993 filas
/tmp/ipykernel_140/2524283831.py:6: FutureWarning: Automatic reindexing on DataFrame vs Series comparisons i
  df = df[~((df < (Q1-1.5 * IQR)) |(df > (Q3 + 1.5 * IQR))).any(axis=1)]
/tmp/ipykernel_140/2524283831.py:6: FutureWarning: Automatic reindexing on DataFrame vs Series comparisons i
  df = df[~((df < (Q1-1.5 * IQR)) |(df > (Q3 + 1.5 * IQR))).any(axis=1)]
```

```python
# 9. Encontrar correlaciones y tendencias
plt.hist(df['Algebra_score'], bins=20, color= 'limegreen')
plt.title("Algebra")
plt.ylabel("Student count")
plt.xlabel("Score")
plt.show()

plt.hist(df['Chemistry score'], bins=20, color= 'limegreen')
plt.title("Chemistry")
plt.ylabel("Student count")
plt.xlabel("Score")
plt.show()

plt.hist(df['Math score'], bins=20, color= 'limegreen')
plt.title("Math")
plt.ylabel("Student count")
plt.xlabel("Score")
plt.show()

plt.hist(df['Physics score'], bins=20, color= 'limegreen')
plt.title("Physics")
plt.ylabel("Student count")
plt.xlabel("Score")
plt.show()
```


Algebra
Student count
Score
0
20
40
60
80
100
20
30
40
50
60
70
80
90
100


Chemistry
Student count
Score
0
20
40
60
80
100
120
20
30
40
50
60
70
80
90
100

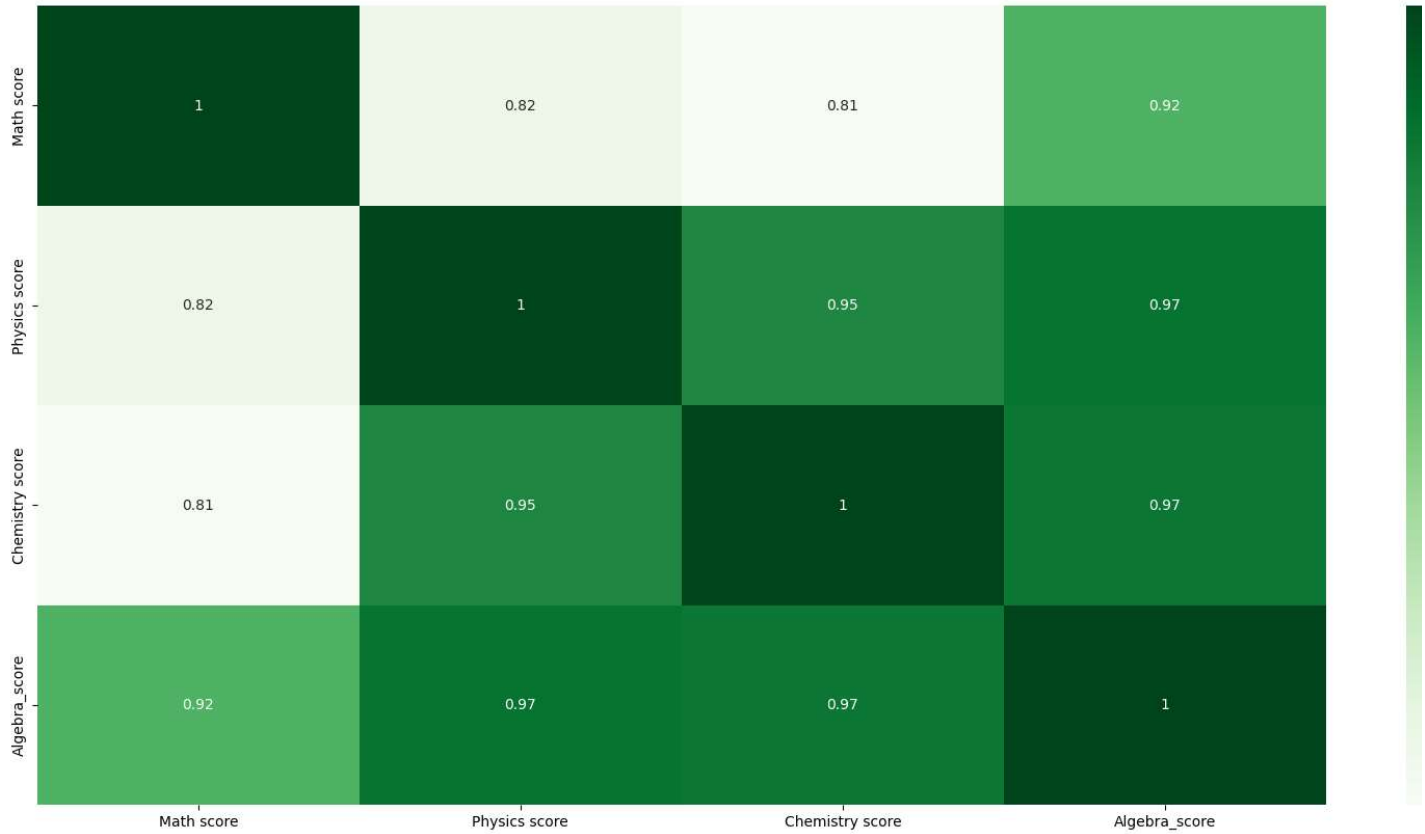

Math
Student count
Score
0
20
40
60
80
100
120
20
40
60
80
100


Physics
Student count
Score
0
20
40
60
80
100
120
30
40
50
60
70
80
90
100

```
# Correlacion entre los datos = Mapas de calor
c = df.corr()
print(c)
```

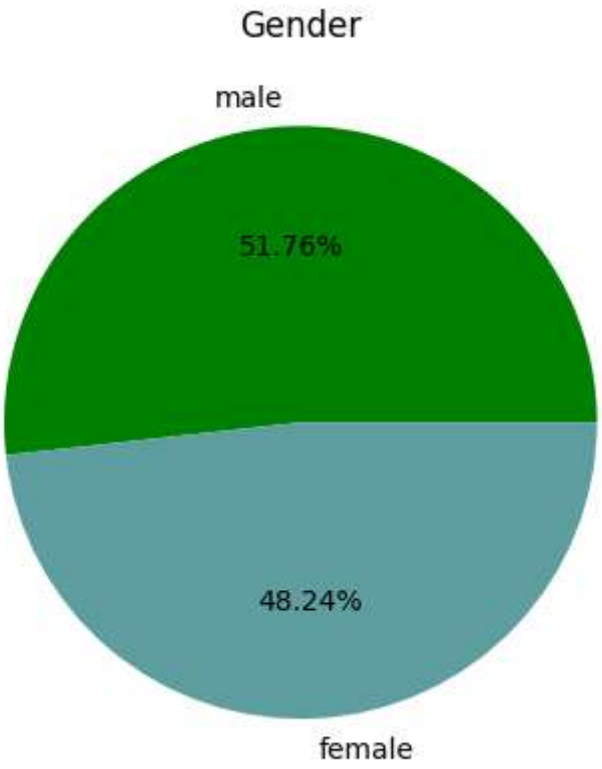
	Math score	Physics score	Chemistry score	Algebra_score
Math score	1.000000	0.818925	0.805260	0.919585
Physics score	0.818925	1.000000	0.954054	0.969793
Chemistry score	0.805260	0.954054	1.000000	0.966060
Algebra_score	0.919585	0.969793	0.966060	1.000000

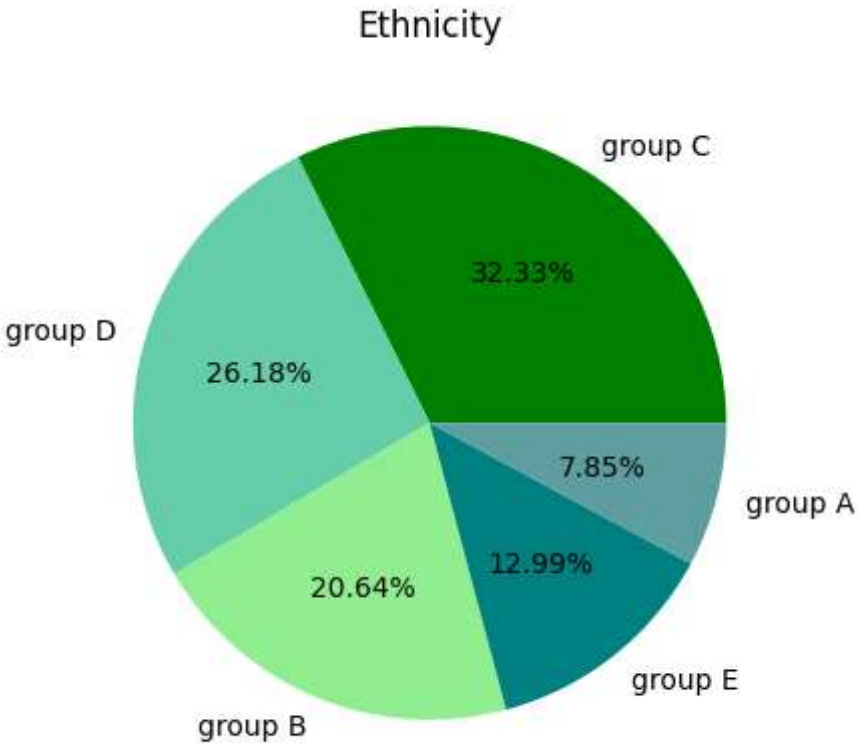
```
plt.figure(figsize=(20,10))
sns.heatmap(c,cmap="Greens", annot=True,)
plt.show()
```

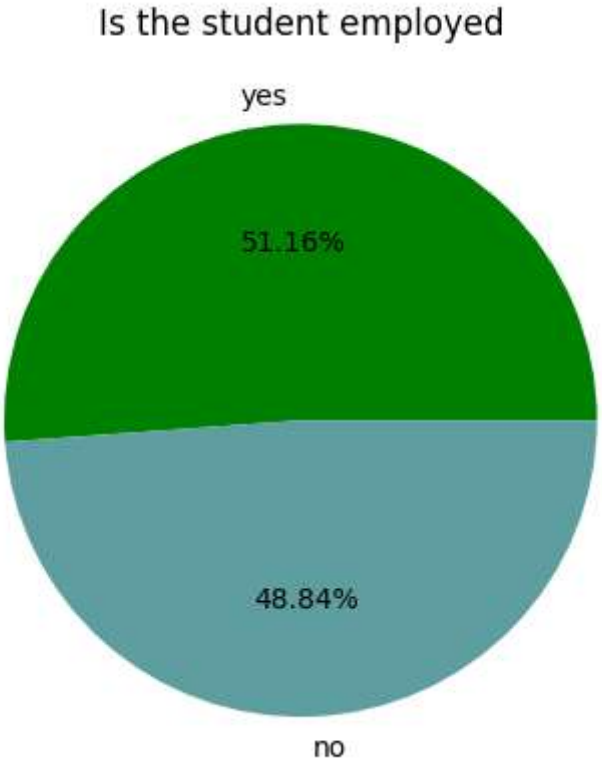


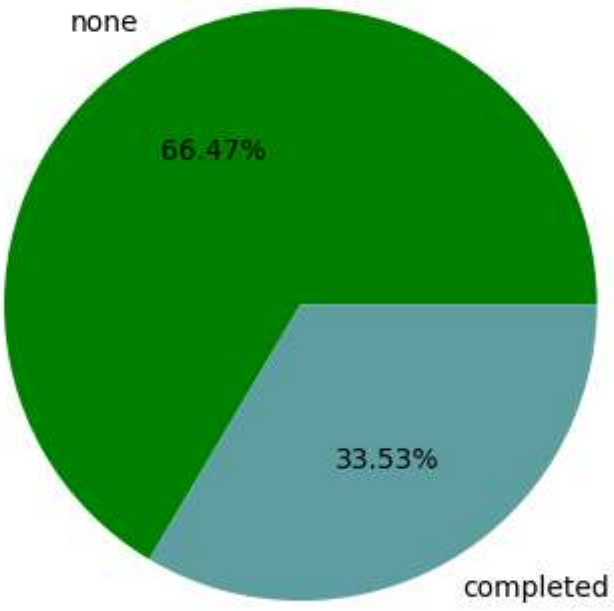

```
# pandas.value_counts() -> devuelve una Serie con valores únicos en orden descendente de frec
labels = df["Gender"].value_counts().index
sizes = df["Gender"].value_counts()
colors=['green','cadetblue']
plt.pie(sizes,labels=labels, autopct='%1.2f%%', colors=colors)
plt.title('Gender')
plt.show()

# repetimos para ETNIA, EMPLEO y el resto de variables categóricas
```

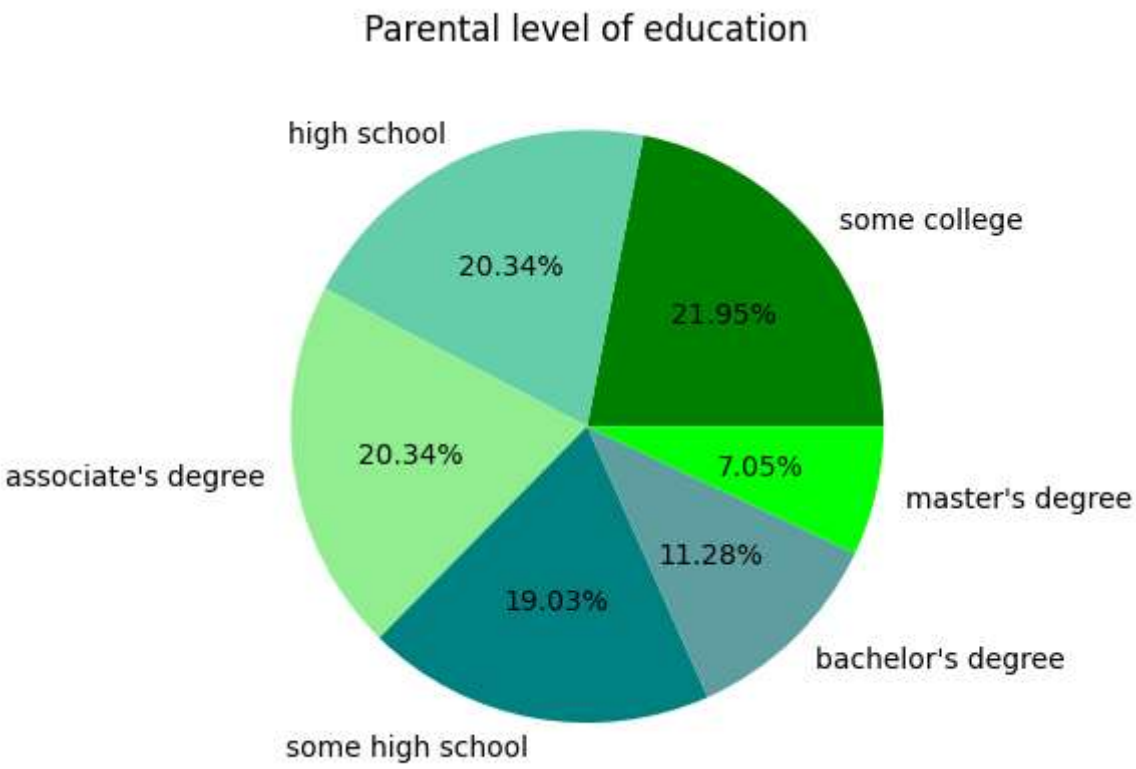
```
labels = df["Ethnicity"].value_counts().index
sizes = df["Ethnicity"].value_counts()
colors=['green','mediumaquamarine', 'lightgreen','teal','cadetblue']
plt.pie(sizes, labels=labels, autopct='%1.2f%%', colors=colors)
plt.title('Ethnicity')
plt.show()


labels = df["Employed"].value_counts().index
sizes = df["Employed"].value_counts()
colors=['green','cadetblue']
plt.pie(sizes, labels=labels, autopct='%1.2f%%', colors=colors)
plt.title('Is the student employed')
plt.show()

labels = df["Test preparation course"].value_counts().index
sizes = df["Test preparation course"].value_counts()
colors=['green','cadetblue']
plt.pie(sizes, labels=labels, autopct='%1.2f%%', colors=colors)
plt.title('Test preparation course')
plt.show()

labels = df["Parental level of education"].value_counts().index
sizes = df["Parental level of education"].value_counts()
colors=['green','mediumaquamarine', 'lightgreen','teal','cadetblue', 'lime']
plt.pie(sizes, labels=labels, autopct='%1.2f%%', colors=colors)
plt.title('Parental level of education')
plt.show()

labels = df["Lunch"].value_counts().index
sizes = df["Lunch"].value_counts()
colors=['green','cadetblue']
plt.pie(sizes, labels=labels, autopct='%1.2f%%', colors=colors)
plt.title('Lunch')
plt.show()
```

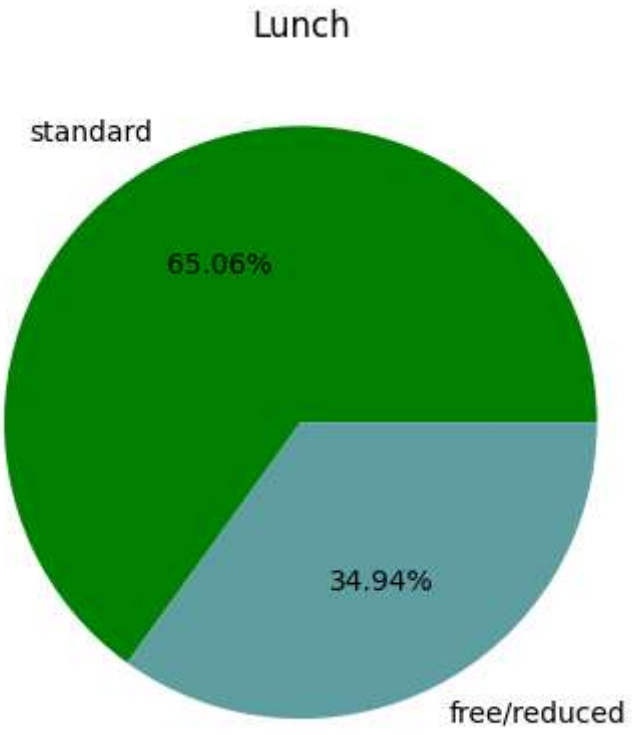

Gender
male
51.76%
48.24%
female


Ethnicity
group C
32.33%
group D
26.18%
7.85%
group A
12.99%
20.64%
group E
group B


Is the student employed
yes
51.16%
48.84%
no


Test preparation course
none
66.47%
33.53%
completed





Respondiendo preguntas

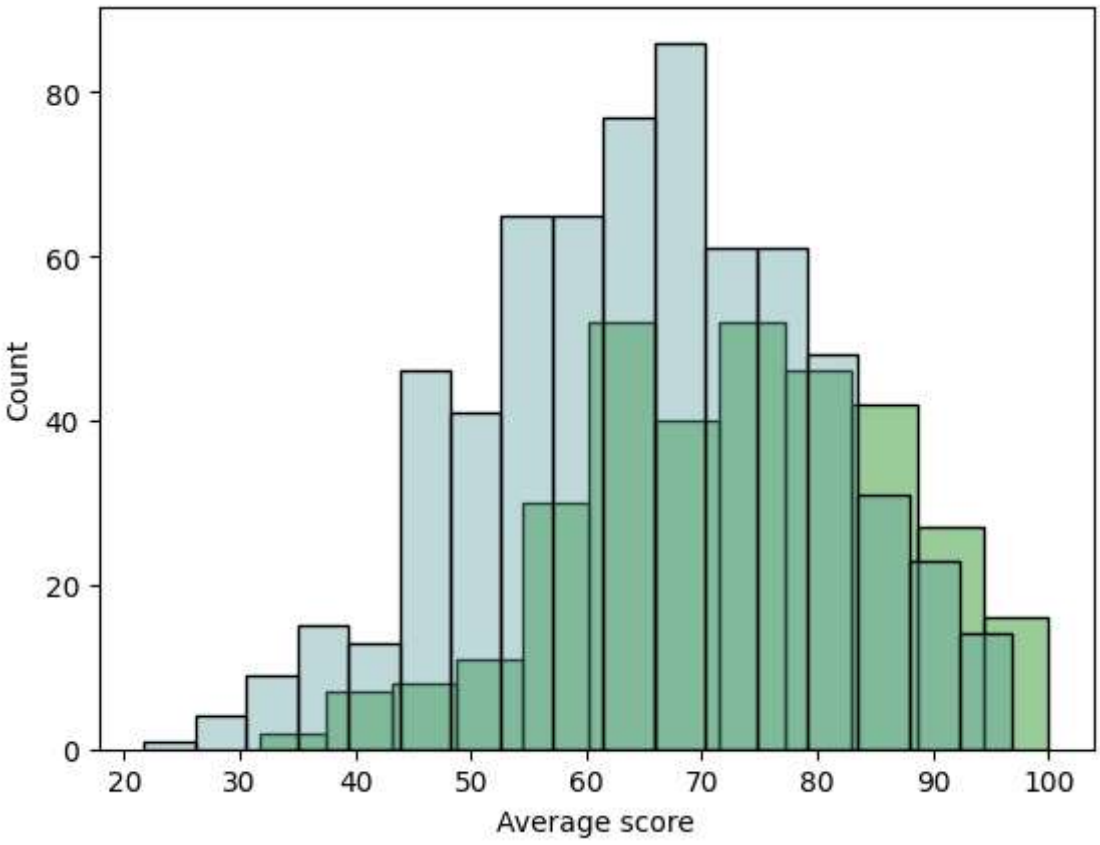
Ejemplo ¿Hay alguna relación entre el promedio de notas obtenidas y el hecho de haber realizado el curso preparatorio?

```
df['Average score'] = df.mean(axis = 1)
# axis = 1 hace que aplique la función sobre los valores numéricos de la fila en lugar de las
df
```

	Gender object male 51.8% female 48.2%	Ethnicity object group C 32.3% group D 26.2% 3 others 41.5%	Parental level of ... some college 22% high school 20.3% 4 others 57.7%	Lunch object standard 65.1% free/reduced 34.9%	Employed object yes 51.2% no 48.8%	Test none comp
0	male	group A	high school	standard	yes	comp
1	female	group D	some high school	free/reduced	no	none
2	male	group E	some college	free/reduced	no	none
3	male	group B	high school	standard	yes	none
4	male	group E	associate's degree	standard	yes	comp
5	female	group D	high school	standard	yes	none
6	female	group A	bachelor's degree	standard	yes	none
7	male	group E	some college	standard	yes	comp
8	male	group D	high school	standard	no	none
9	male	group C	some college	free/reduced	no	none

```
si = df[df['Test preparation course'] =='completed']
no = df[df['Test preparation course'] =='none']
```

```
sns.histplot(si['Average score'], color= 'green', alpha=.4, fill = True)
sns.histplot(no['Average score'], color= 'cadetblue', alpha=.4, fill = True)
plt.show()
```




```
print('Realizaron el curso:', si['Test preparation course'].count())
print('No realizaron el curso:', no['Test preparation course'].count())
```

```
Realizaron el curso: 333
No realizaron el curso: 660
```

Conclusión: Si bien la cantidad de alumnos que no realizó el curso preparatorio casi duplica a la de quienes lo han completado, ésta diferencia no se ve relejada significativamente en el promedio de notas. Se recomienda auditar los contenidos del curso, a fines de lograr una mejora en el rendimiento académico y aumentar el interés del alumnado.