

# Trabajo Práctico Integrador



## Análisis exploratorio

En este trabajo se analiza una estructura de datos que contiene calificaciones de alumnos y una serie de variables categóricas relativas a su vida personal y familiar (tales como género, estudios alcanzados por los padres, situación laboral, alimentación, grupo étnico al que pertenecen, etcétera).

A partir del análisis exploratorio se responderán una serie de interrogantes, tales como: ¿hay alguna relación entre el promedio de notas obtenidas y el hecho de haber realizado el curso preparatorio?; ¿existe correspondencia entre el grupo étnico y el promedio de notas alcanzado?; ¿existe relación entre el promedio de notas obtenidas y el hecho de que los alumnos trabajen o no?; ¿tienen mayor promedio los alumnos cuyos padres han alcanzado un nivel de estudio posuniversitario (en este caso, un "master degree")?; ¿hay diferencias significativas en las calificaciones, teniendo en consideración el género de los estudiantes?

	math score float64	physics score flo...	chemistry score f...	algebra_score flo...	
count	1011.0	1011.0	1011.0	1011.0	
mean	66.48071216617211	69.06330365974283	67.7893175074184	67.77843719090009	
std	15.326879704379337	14.694107007851635	15.55985328614052	14.450679861041094	
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	

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

```
object
object
object
object
object
object
object
float64
float64
float64
float64
```

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

```
# 7) ELIMINAR LOS VALORES PERDIDOS O FALSOS
```

```
# 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
```

```
# 8) DETECTAR LOS OUTLIERS
```

```
sns.set(style='dark')
```

```
sns.boxplot(x=df['Math score'], color='mediumslateblue')
```

```
plt.show()
```

```
sns.boxplot(x=df['Physics score'], color='mediumslateblue')
```

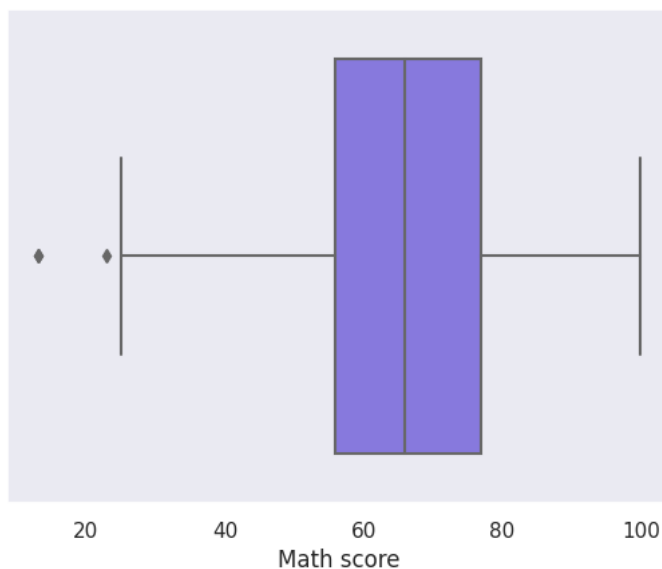
```
plt.show()
```

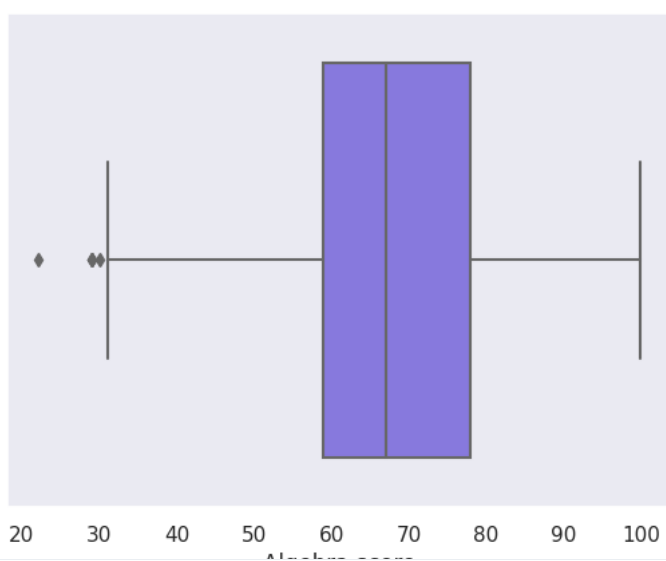
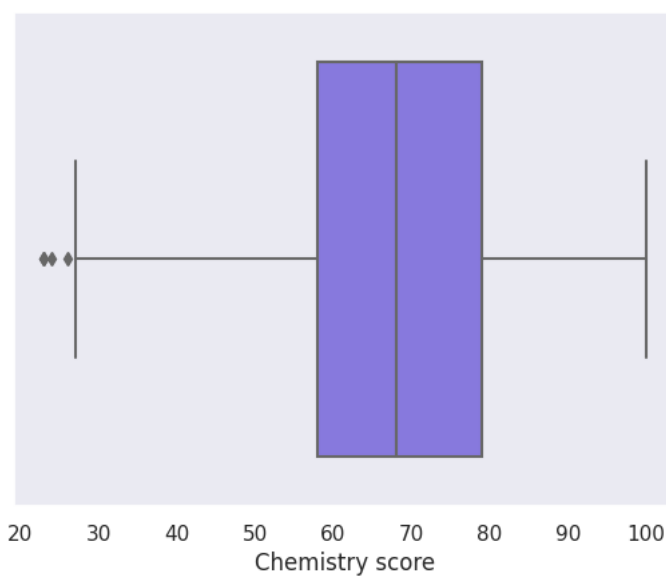
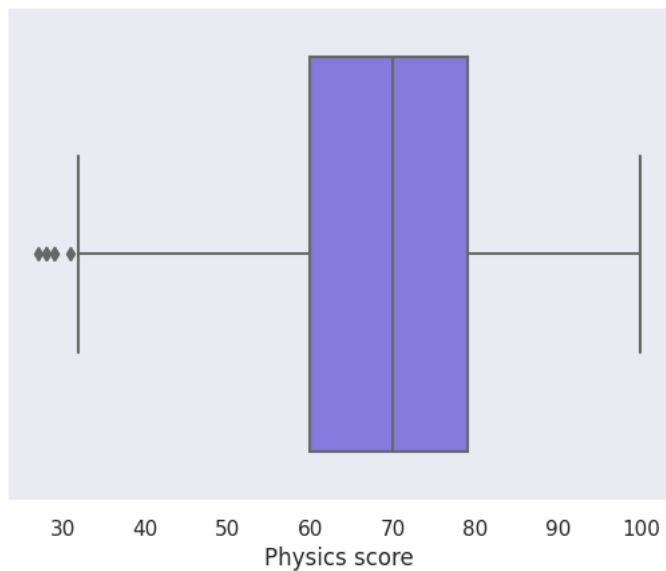
```
sns.boxplot(x=df['Chemistry score'], color='mediumslateblue')
```

```
plt.show()
```

```
sns.boxplot(x=df['Algebra score'], color='mediumslateblue')
```

```
plt.show()
```





```
print(f'Antes: {df.Lunch.count()} filas\n')
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1 # Rango intercuartil
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      21.0
Physics score   19.0
Chemistry score 21.0
Algebra score    19.0
dtype: float64
```

Después: 984 filas

```
/tmp/ipykernel_1373/3704891740.py:6: FutureWarning: Automatic reindexing on DataFrame vs Series comparisons is deprecated and will raise ValueError in a futur
df = df[~((df<(Q1-1.5 * IQR)) | (df>(Q3+1.5 * IQR))).any(axis=1)]
/tmp/ipykernel_1373/3704891740.py:6: FutureWarning: Automatic reindexing on DataFrame vs Series comparisons is deprecated and will raise ValueError in a futur
df = df[~((df<(Q1-1.5 * IQR)) | (df>(Q3+1.5 * IQR))).any(axis=1)]
```

# 9) ENCONTRAR CORRELACIONES Y FRECUENCIAS

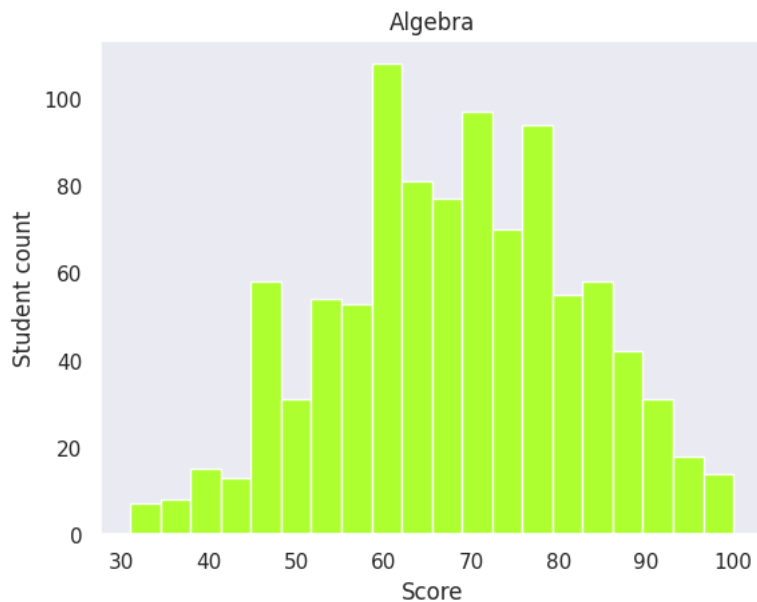
# Para frecuencias (histogramas)

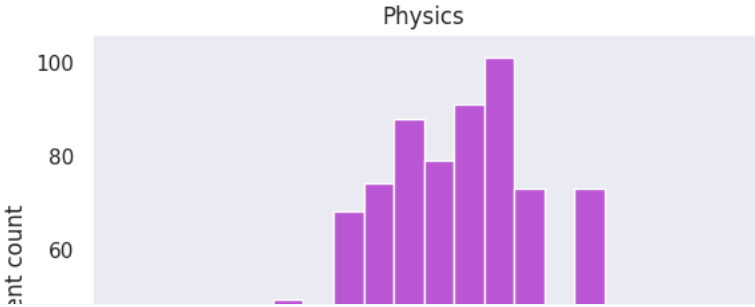
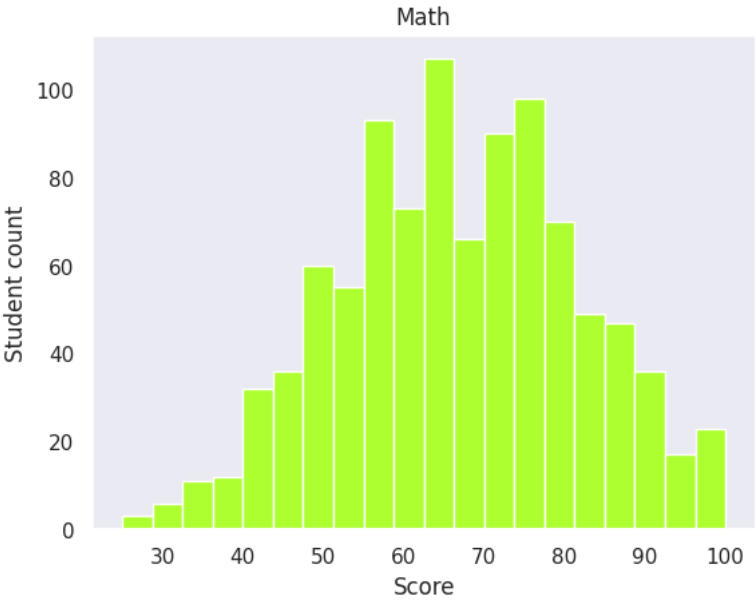
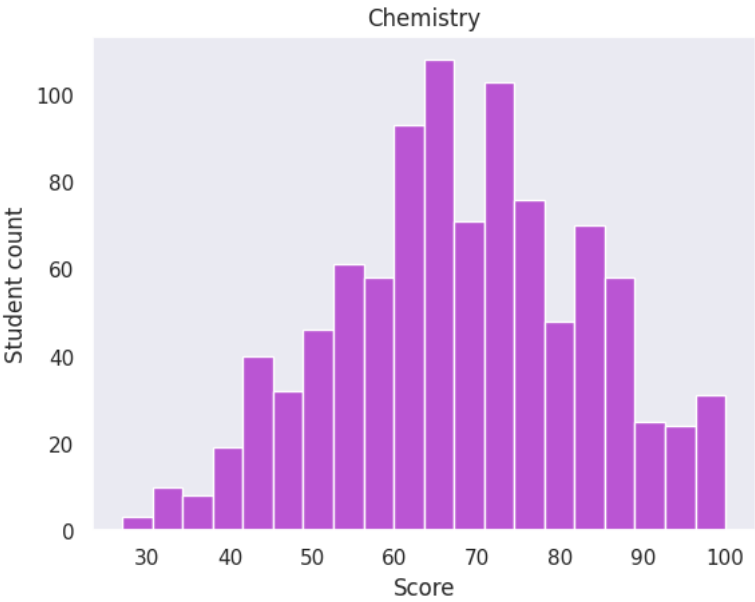
```
plt.hist(df['Algebra score'], bins=20, color='greenyellow')
plt.title('Algebra')
plt.ylabel('Student count')
plt.xlabel('Score')
plt.show()
```

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

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

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



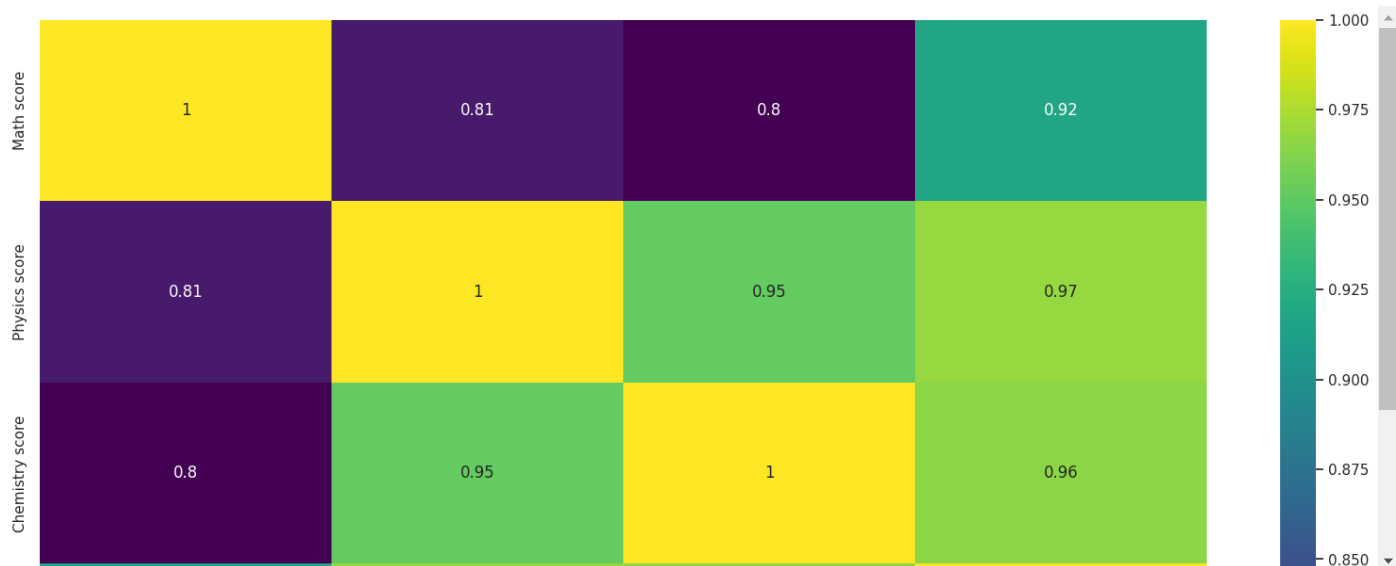


```
# Correlaciones entre los datos (mapas de calor)
c = df.corr()
print(c)
```

	Math score	Physics score	Chemistry score	Algebra score
Math score	1.000000	0.812055	0.798312	0.916674
Physics score	0.812055	1.000000	0.951536	0.968358
Chemistry score	0.798312	0.951536	1.000000	0.964652
Algebra score	0.916674	0.968358	0.964652	1.000000

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

plt.show()



```
# Gráficas para variables categóricas
# pandas.value_counts() -> devuelve una Serie con valores únicos en un orden descendente de frecuencia
labels = df['Gender'].value_counts().index
sizes = df['Gender'].value_counts()
c = ['paleturquoise', 'lightpink']
plt.pie(sizes, labels=labels, colors=c, autopct='%1.2f%%', pctdistance = 0.75, shadow=True)
plt.title('Gender')
cc = plt.Circle((0,0), 0.5, fc='white')
fig=plt.gcf()
fig.gca().add_artist(cc)
plt.savefig('plot.png', dpi=300)
plt.legend(loc="upper right")
plt.show()

# Repetimos para el resto de las variables categóricas
labels = df['Ethnicity'].value_counts().index
sizes = df['Ethnicity'].value_counts()
c = ['gold', 'yellowgreen', 'lightcoral', 'plum', 'paleturquoise']
e = (0,0,0,0,0.2)
plt.pie(sizes, explode = e, labels=labels, colors=c, autopct='%1.2f%%', shadow=True)
plt.title('Ethnicity')
plt.show()

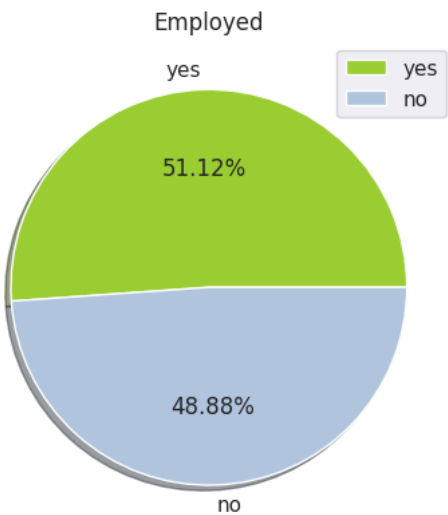
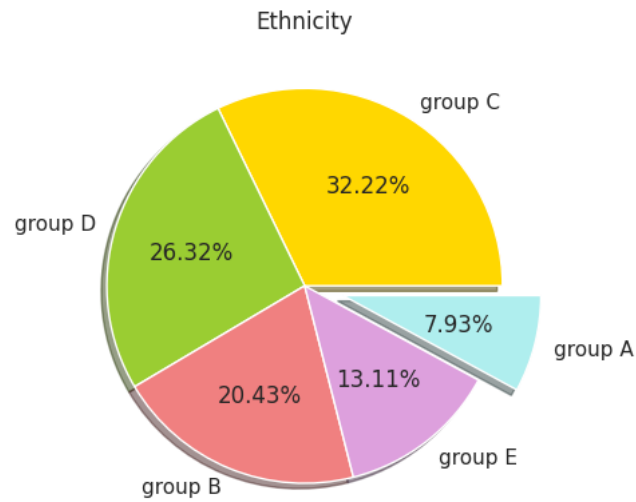
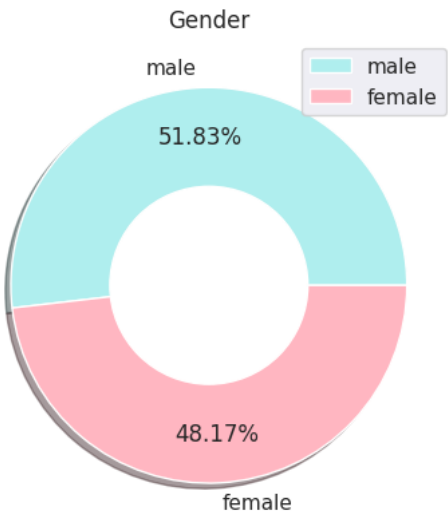
labels = df['Employed'].value_counts().index
sizes = df['Employed'].value_counts()
c = ['yellowgreen', 'lightsteelblue']
plt.pie(sizes, labels=labels, colors=c, autopct='%1.2f%%', shadow=True)
plt.title('Employed')
plt.legend(loc="upper right")
plt.show()

labels = df['Test preparation course'].value_counts().index
sizes = df['Test preparation course'].value_counts()
c = ['lightsteelblue', 'yellowgreen']
plt.pie(sizes, labels=labels, colors=c, autopct='%1.2f%%', shadow=True)
plt.title('Test preparation course')
plt.legend(loc="upper right")
plt.show()

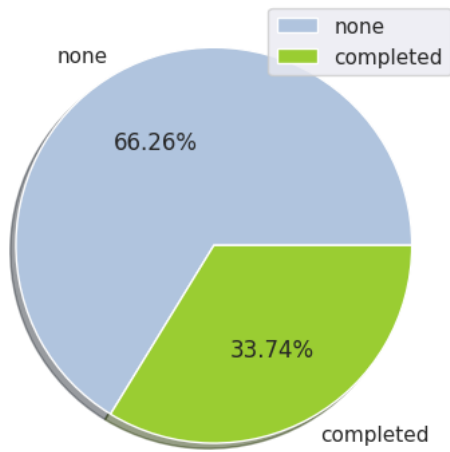
labels = df['Parental level of education'].value_counts().index
sizes = df['Parental level of education'].value_counts()
c = ['gold', 'yellowgreen', 'lightcoral', 'plum', 'paleturquoise', 'mediumslateblue']
e = (0,0,0,0,0,0.2)
plt.pie(sizes, explode=e, labels=labels, colors=c, autopct='%1.2f%%', pctdistance = 0.75, shadow=True)
plt.title('Parental level of education')
cc = plt.Circle((0,0), 0.5, fc='white')
fig=plt.gcf()
fig.gca().add_artist(cc)
plt.savefig('plot.png', dpi=300)
plt.show()

labels = df['Lunch'].value_counts().index
sizes = df['Lunch'].value_counts()
c = ['gold', 'yellowgreen']
plt.pie(sizes, labels=labels, colors=c, autopct='%1.2f%%', shadow=True)
plt.title('Lunch')
```

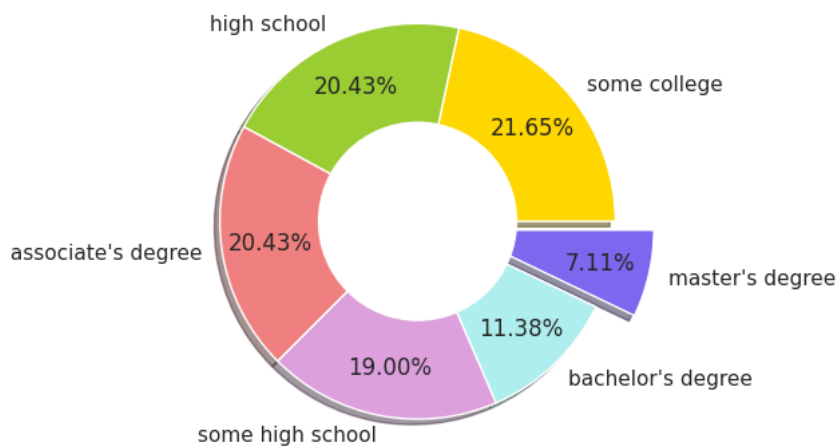
```
plt.legend(loc="upper right")
plt.show()
```



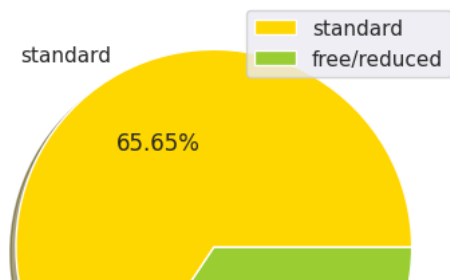
Test preparation course



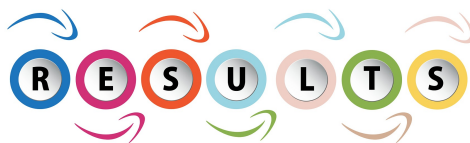
Parental level of education



Lunch





## Respondiendo preguntas



**Pregunta 1 (ejemplo del video): ¿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) # creo esta nueva columna "Average Score"
# axis = 1 -> hace que aplique la función sobre los valores numéricos de la fila, en lugar de las columnas (es decir, busca el promedio
df
```

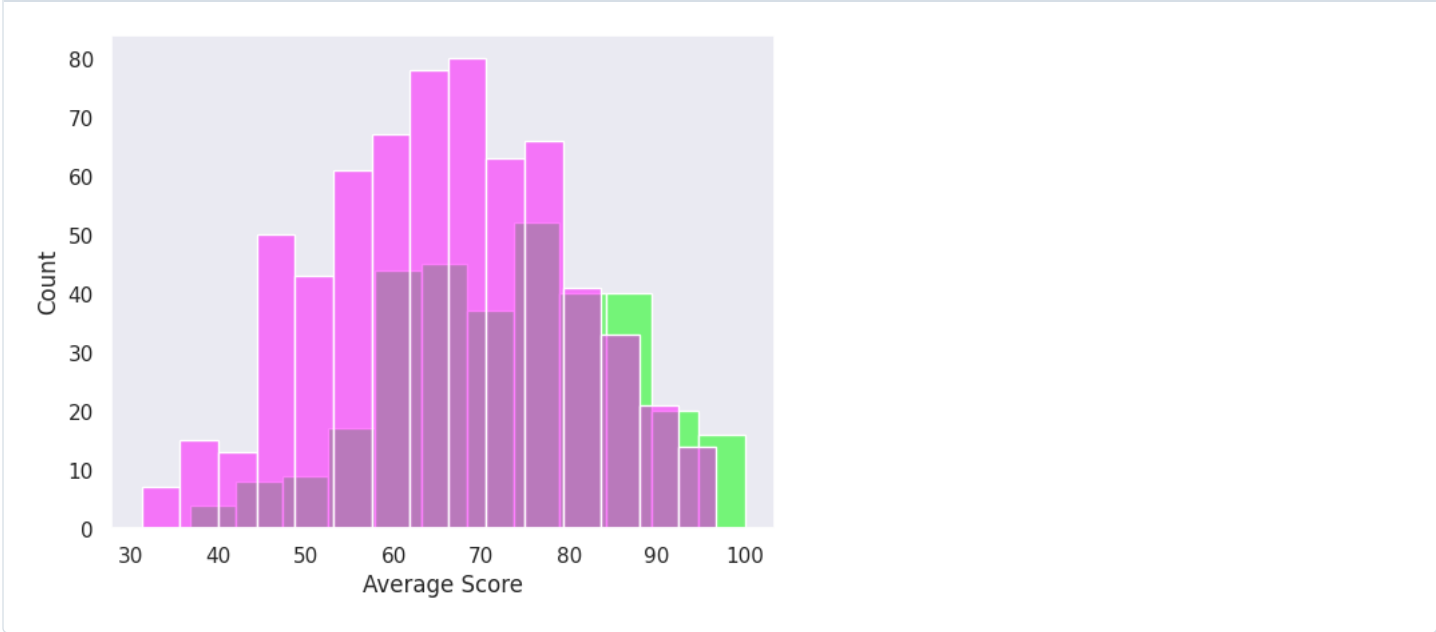
	Gender object	Ethnicity object	Parental level of ...	Lunch object	Employed object	Test preparation...	Math score float64	Physics score flo...
	male ..... 51.8% female ..... 48.2%	group C ..... 32.2% group D ..... 26.3% 3 others ..... 41.5%	some college _ 21.6% high school _ _ 20.4% 4 others ..... 57.9%	standard ..... 65.7% free/reduced . 34.3%	yes ..... 51.1% no ..... 48.9%	none ..... 66.3% completed ..... 33.7%		
0	male	group A	high school	standard	yes	completed	67.0	67.
1	female	group D	some high school	free/reduced	no	none	40.0	59.
2	male	group E	some college	free/reduced	no	none	59.0	60.
3	male	group B	high school	standard	yes	none	77.0	78.
4	male	group E	associate's degree	standard	yes	completed	78.0	73.
5	female	group D	high school	standard	yes	none	63.0	77.
6	female	group A	bachelor's degree	standard	yes	none	62.0	59.
7	male	group E	some college	standard	yes	completed	93.0	88.
8	male	group D	high school	standard	no	none	63.0	56.
9	male	group C	some college	free/reduced	no	none	47.0	42.

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

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

sns.set(style='dark')
sns.histplot(si['Average Score'], color = 'lime', alpha = .5, fill = True)
sns.histplot(no['Average Score'], color = 'fuchsia', alpha = .5, fill = True)
plt.show()
```

Realizaron el curso: 332 alumnos.  
No realizaron el curso: 652 alumnos.



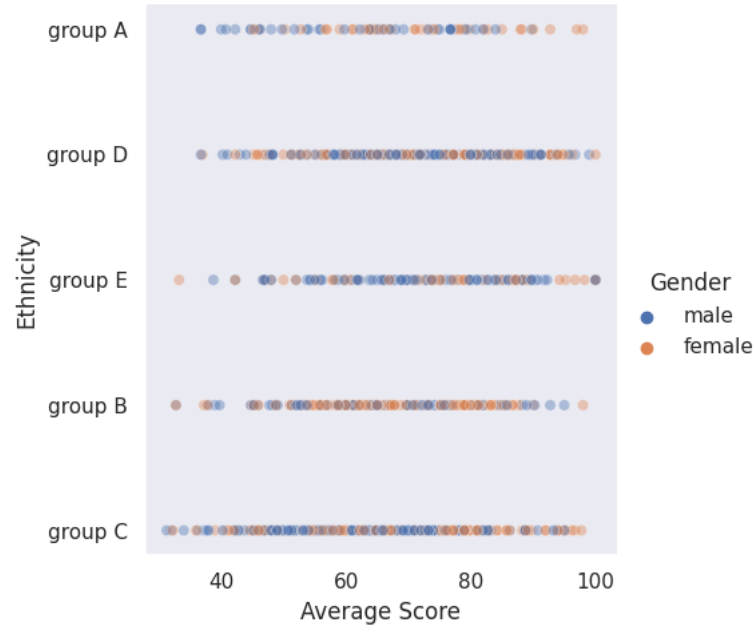
**Conclusión:** si bien, la cantidad de alumnos que no realizó el curso preparatorio duplica, casi, a la de quienes lo han completado, esta diferencia no se ve reflejada, significativamente, en el promedio de notas.

**Recomendación:** podrían auditarse los contenidos del curso, a fines de lograr una mejora en el rendimiento académico y, así, incrementar el interés del alumnado.

Pregunta 2: ¿Existe correspondencia entre el grupo étnico y el promedio de notas alcanzado?

```
sns.relplot(x = 'Average Score', hue='Gender', y='Ethnicity', alpha=.4, data=df)
```

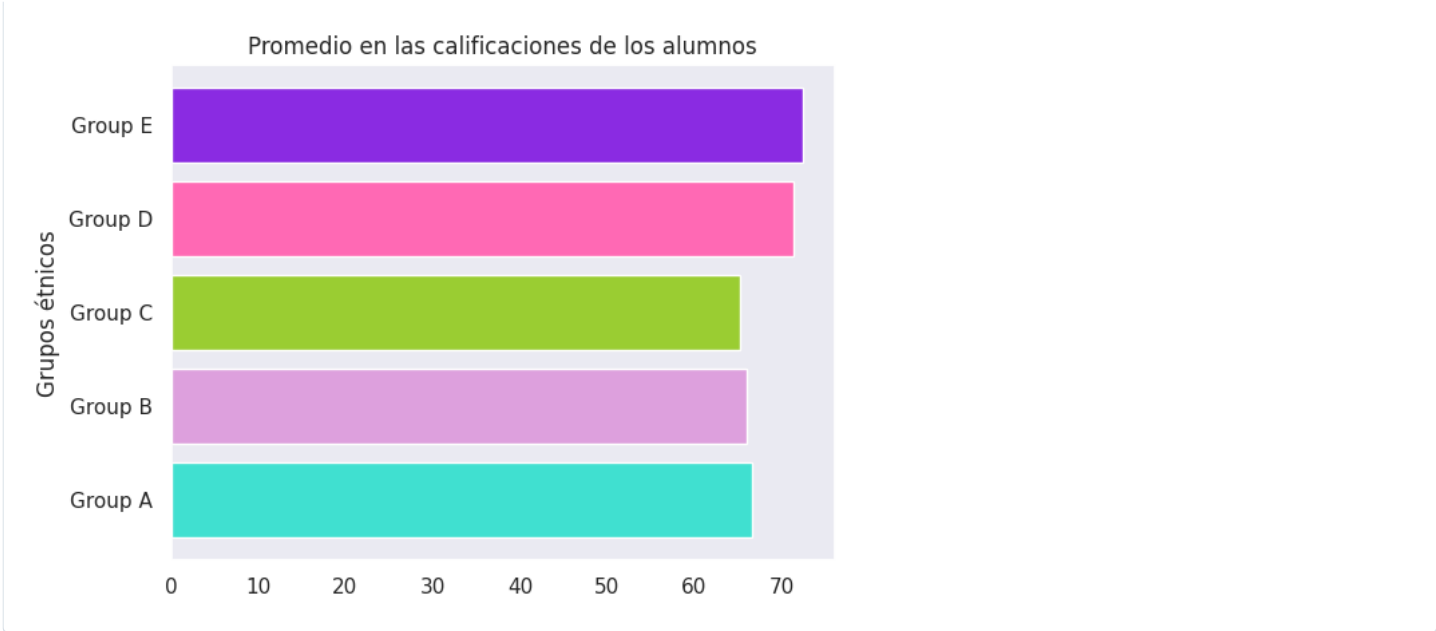
<seaborn.axisgrid.FacetGrid at 0x7f8339127d30>



```
# Cuento la cantidad de alumnos dentro de cada grupo étnico y, dentro de cada uno de estos grupos, saco el promedio de notas (teniendo en cuenta el género)
grupoA = df[df['Ethnicity'] == 'group A']
grupoB = df[df['Ethnicity'] == 'group B']
grupoC = df[df['Ethnicity'] == 'group C']
grupoD = df[df['Ethnicity'] == 'group D']
grupoE = df[df['Ethnicity'] == 'group E']
print('Cantidad de alumnos Group A:', grupoA['Ethnicity'].count())
print('Cantidad de alumnos Group B:', grupoB['Ethnicity'].count())
print('Cantidad de alumnos Group C:', grupoC['Ethnicity'].count())
print('Cantidad de alumnos Group D:', grupoD['Ethnicity'].count())
print('Cantidad de alumnos Group E:', grupoE['Ethnicity'].count())
grupoA_total = np.sum(grupoA['Average Score']).mean()
grupoB_total = np.sum(grupoB['Average Score']).mean()
grupoC_total = np.sum(grupoC['Average Score']).mean()
grupoD_total = np.sum(grupoD['Average Score']).mean()
grupoE_total = np.sum(grupoE['Average Score']).mean()
print(f'Promedio de calificaciones de alumnos Group A: {grupoA_total:.2f}')
print(f'Promedio de calificaciones de alumnos Group B: {grupoB_total:.2f}')
print(f'Promedio de calificaciones de alumnos Group C: {grupoC_total:.2f}')
print(f'Promedio de calificaciones de alumnos Group D: {grupoD_total:.2f}')
print(f'Promedio de calificaciones de alumnos Group E: {grupoE_total:.2f}')
```

Cantidad de alumnos Group A: 78  
Cantidad de alumnos Group B: 201  
Cantidad de alumnos Group C: 317  
Cantidad de alumnos Group D: 259  
Cantidad de alumnos Group E: 129  
Promedio de calificaciones de alumnos Group A: 66.57  
Promedio de calificaciones de alumnos Group B: 66.03  
Promedio de calificaciones de alumnos Group C: 65.22  
Promedio de calificaciones de alumnos Group D: 71.29  
Promedio de calificaciones de alumnos Group E: 72.41

```
grupos = ['Group A', 'Group B', 'Group C', 'Group D', 'Group E']
promedio = [grupoA_total, grupoB_total, grupoC_total, grupoD_total, grupoE_total]
plt.barh(grupos, promedio, color=['turquoise', 'plum', 'yellowgreen', 'hotpink', 'blueviolet'])
plt.ylabel("Grupos étnicos")
plt.title("Promedio en las calificaciones de los alumnos")
plt.show()
```



**Conclusión:** los grupos E y D poseen un promedio general mayor en las calificaciones. Mientras que, los 3 grupos étnicos restantes (A, B, C) alcanzan un promedio menor y bastante parecido entre ellos.

**Recomendación:** generar acciones (de acompañamiento académico) que fortalezcan las competencias y habilidades de los alumnos que pertenecen a los grupos étnicos con menores calificaciones.

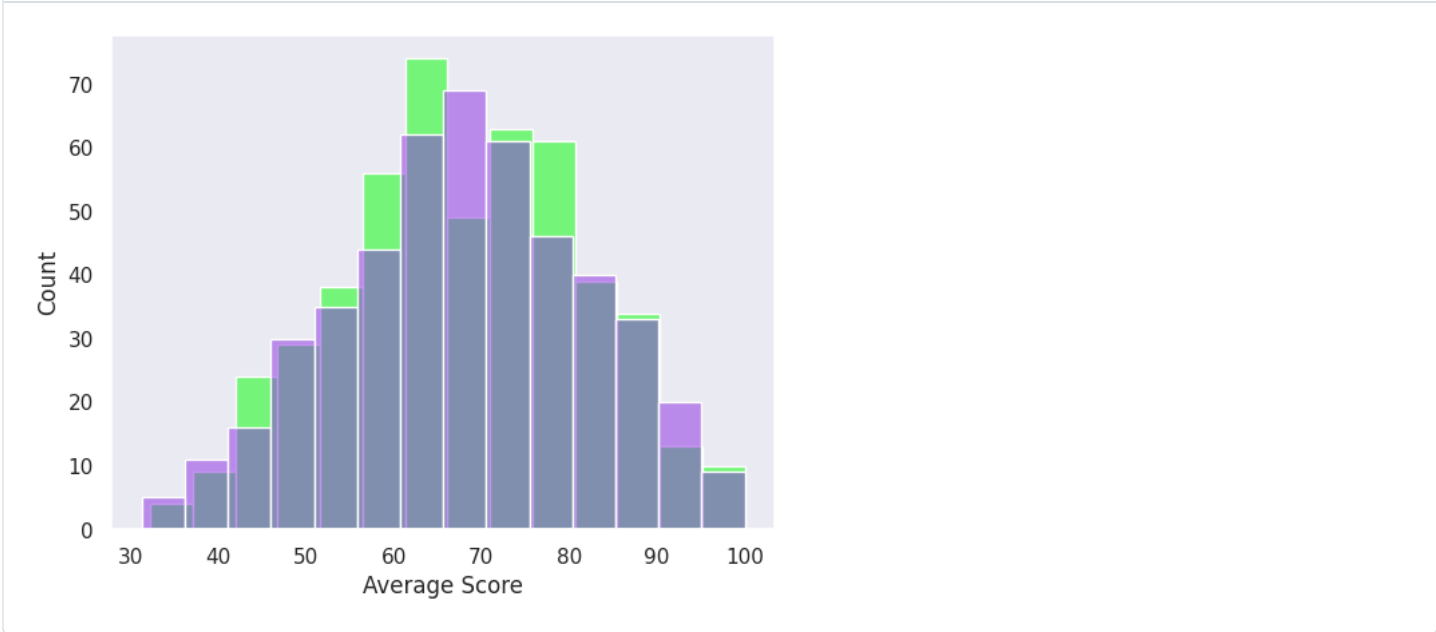
**Pregunta 3: ¿Existe relación entre el promedio de notas obtenidas y el hecho de que los alumnos trabajen o no?**

```
trabaja = df[df['Employed'] == 'yes']
noTrabaja = df[df['Employed'] == 'no']

print('Trabajan:', trabaja['Employed'].count(), 'alumnos.')
print('No trabajan:', noTrabaja['Employed'].count(), 'alumnos.')

sns.histplot(trabaja['Average Score'], color = 'lime', alpha = .5, fill = True)
sns.histplot(noTrabaja['Average Score'], color = 'blueviolet', alpha = .5, fill = True)
plt.show()
```

Trabajan: 503 alumnos.  
No trabajan: 481 alumnos.



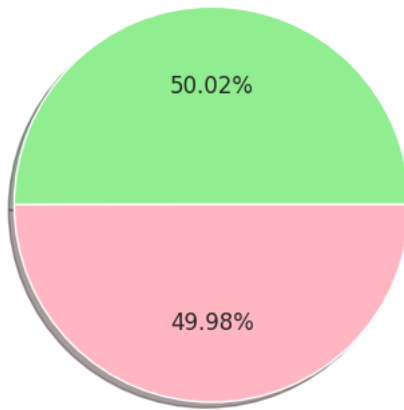
```
# Cuento la cantidad de alumnos que trabajan y los que no trabajan y, dentro de cada uno de estos grupos, saco el promedio de notas (ter
trabaja_total = np.sum(trabaja['Average Score']) / trabaja['Employed'].count()
noTrabaja_total = np.sum(noTrabaja['Average Score']) / noTrabaja['Employed'].count()
print(f'El promedio de notas de las personas que trabajan es: {trabaja_total:.2f}')
print(f'El promedio de notas de las personas que no trabajan es: {noTrabaja_total:.2f}')
```

El promedio de notas de las personas que trabajan es: 68.06  
 El promedio de notas de las personas que no trabajan es: 68.01

```
labels = [f'Trabaja (promedio de nota: {trabaja_total:.2f})', f'No trabaja (promedio de nota: {noTrabaja_total:.2f})']
sizes = [trabaja_total, noTrabaja_total]
c = ['lightgreen', 'lightpink']
plt.pie(sizes, labels=labels, colors=c, autopct='%1.2f%%', shadow=True)
plt.title('¿Qué grupo tiene un mayor promedio en el total de las notas?\n (Porcentaje que exhibe la diferencia)', loc = 'center')
plt.show()
```

¿Qué grupo tiene un mayor promedio en el total de las notas?  
 (Porcentaje que exhibe la diferencia)

Trabaja (promedio de nota: 68.06)



No trabaja (promedio de nota: 68.01)

**Conclusión:** si tenemos en cuenta el promedio total de las notas dentro de cada grupo (trabajadores y desempleados), no es posible advertir una diferencia significativa entre las mismas, ya que son prácticamente iguales.

En suma, el hecho de trabajar no implica un menor desenvolvimiento académico.

**Recomendación:** sabiendo que el hecho de trabajar no genera una merma en las calificaciones, podrían generarse prácticas especiales para aquellos alumnos que no trabajan, para que -a la par de la cursada- puedan adquirir experiencia laboral en el ámbito de sus estudios.

## Pregunta 4: ¿Tienen mayor promedio los alumnos cuyos padres han alcanzado un nivel de estudio posuniversitario (en este caso, un "master degree")?

```
padres_master = df[df['Parental level of education'] == "master's degree" ]
padres_no_master = df[df['Parental level of education'] != "master's degree"]

print('Padres que han alcanzado un máster:', padres_master['Parental level of education'].count())
print('Padres que no han alcanzado un máster:', padres_no_master['Parental level of education'].count())

sns.histplot(padres_master['Average Score'], color = 'deeppink', alpha = .5, fill = True)
sns.histplot(padres_no_master['Average Score'], color = 'blueviolet', alpha = .5, fill = True)
plt.show()
```

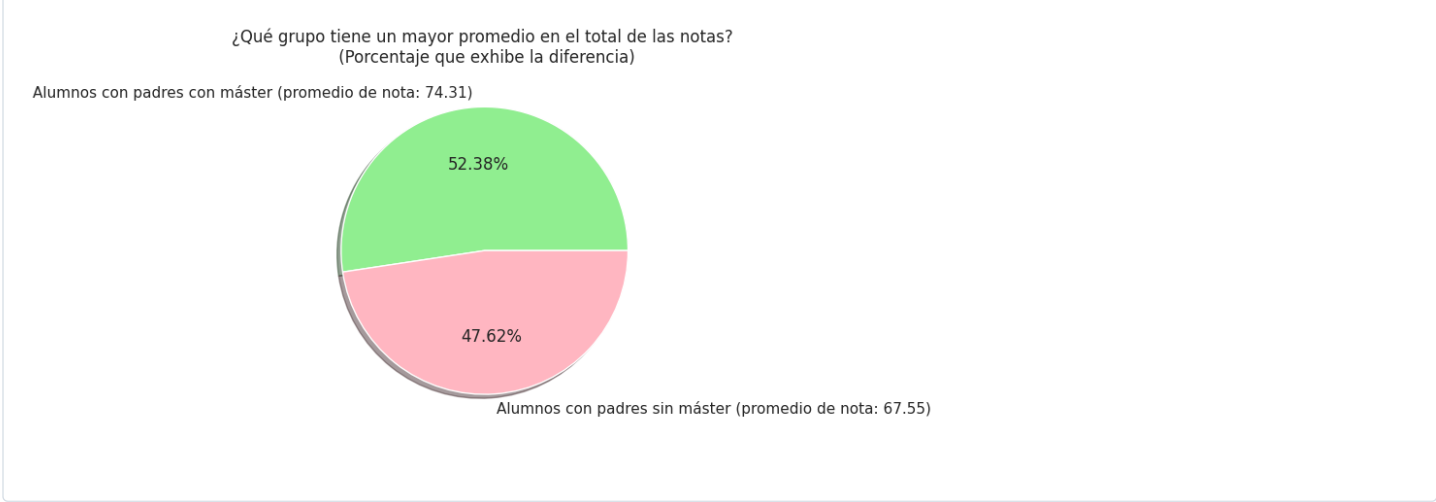
Padres que han alcanzado un máster: 70  
 Padres que no han alcanzado un máster: 914



```
# Cuento la cantidad de alumnos cuyos padres alcanzaron un máster y, dentro de cada uno de estos grupos, saco el promedio de notas (teniendo en cuenta los alumnos con y sin padres con máster)
padres_master_total = np.sum(padres_master['Average Score'].mean())
padres_no_master_total = np.sum(padres_no_master['Average Score'].mean())
print(f'El promedio de notas de los alumnos con padres con máster es: {padres_master_total:.2f}')
print(f'El promedio de notas de los alumnos cuyos padres no alcanzaron un máster es: {padres_no_master_total:.2f}')
```

El promedio de notas de los alumnos con padres con máster es: 74.31  
El promedio de notas de los alumnos cuyos padres no alcanzaron un máster es: 67.55

```
labels = [f'Alumnos con padres con máster (promedio de nota: {padres_master_total:.2f})', f'Alumnos con padres sin máster (promedio de nota: {padres_no_master_total:.2f})']
sizes = [padres_master_total, padres_no_master_total]
c = ['lightgreen', 'lightpink']
plt.pie(sizes, labels=labels, colors=c, autopct='%1.2f%%', shadow=True)
plt.title('¿Qué grupo tiene un mayor promedio en el total de las notas? \n (Porcentaje que exhibe la diferencia)', loc = 'center')
plt.show()
```



**Conclusión:** puede advertirse una superioridad en el promedio total de las calificaciones de los alumnos, de quienes algunos de sus padres han alcanzado un grado académico de máster (74.31), frente al de los alumnos cuyos padres tienen un grado académico menor (67.55).

**Recomendación:** adoptar programas de incentivos para los alumnos (por ejemplo, mentorías) que puedan robustecer su confianza y sus destrezas en el estudio, para que esta diferencia pueda reducirse. Y, de esta forma, que este tipo de estímulos externos -más allá de la familia- les permitan mejorar sus calificaciones.

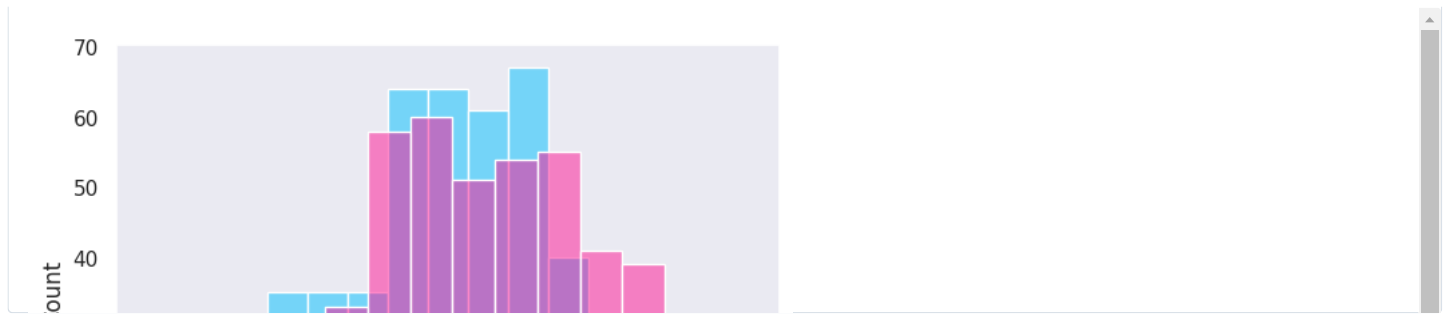
## Pregunta 5: De acuerdo al género, ¿existen diferencias en las calificaciones?

```
varones = df[df['Gender'] == "male" ]
mujeres = df[df['Gender'] == "female" ]

print('Varones:', varones['Gender'].count())
print('Mujeres:', mujeres['Gender'].count())

sns.histplot(varones['Average Score'], color = 'deepskyblue', alpha = .5, fill = True)
sns.histplot(mujeres['Average Score'], color = 'deeppink', alpha = .5, fill = True)
plt.show()
```

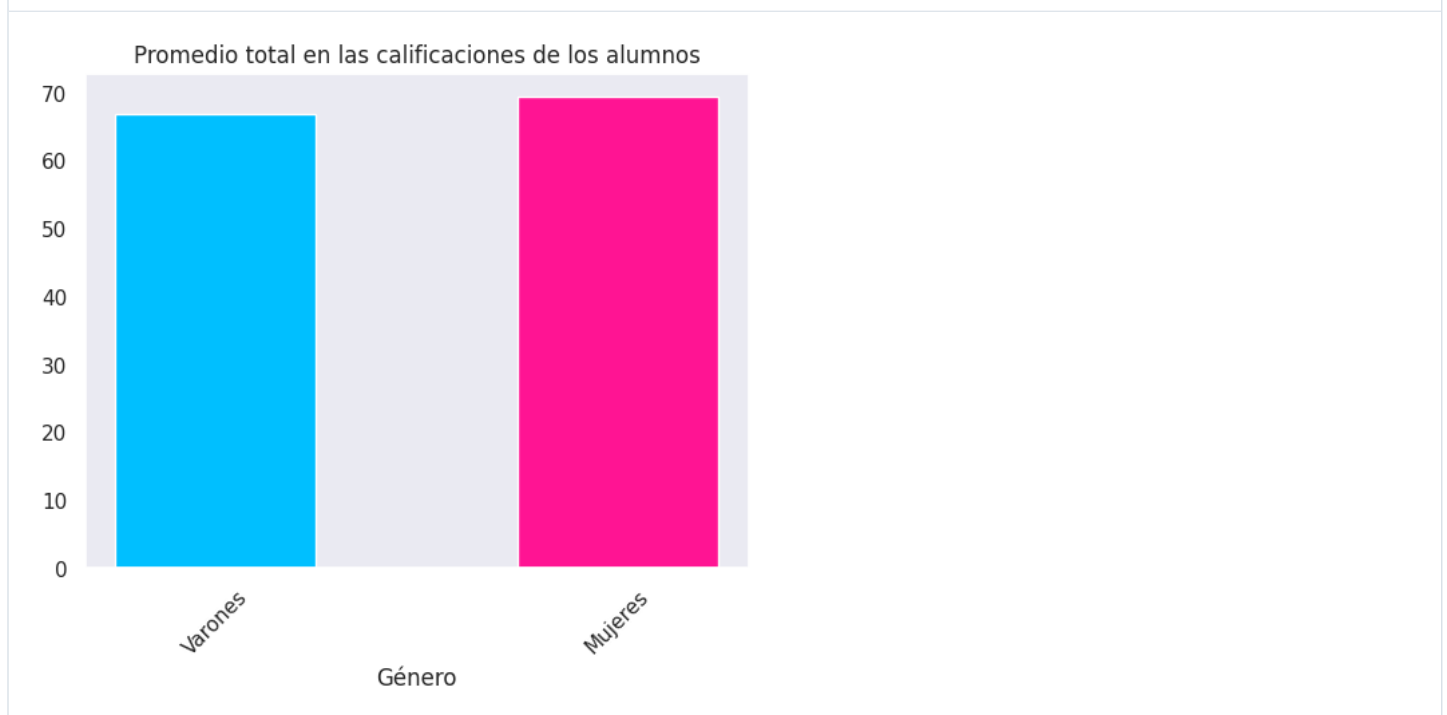
Varones: 510  
Mujeres: 474



```
# Cuento la cantidad de alumnos en cada género y, dentro de cada uno de estos grupos, saco el promedio de notas (teniendo en cuenta el A
varones_total = np.sum(varones['Average Score']).mean()
mujeres_total = np.sum(mujeres['Average Score']).mean()
print(f'El promedio de notas de los alumnos varones es: {varones_total:.2f}')
print(f'El promedio de notas de las alumnas mujeres es: {mujeres_total:.2f}')
```

El promedio de notas de los alumnos varones es: 66.80  
El promedio de notas de las alumnas mujeres es: 69.37

```
grupos = ['Varones', 'Mujeres']
promedio = [varones_total, mujeres_total]
plt.bar(grupos, promedio, width = 0.5, color=['deepskyblue', 'deeppink'])
plt.xticks(np.arange(2), ('Varones', 'Mujeres'), rotation = 45)
plt.xlabel("Género")
plt.title("Promedio total en las calificaciones de los alumnos")
plt.show()
```



**Conclusión:** puede observarse que el promedio general de calificaciones de las mujeres (69.37) es mayor al propio de los varones (66.80).

**Información adicional:** en consonancia con las recomendaciones expuestas en las conclusiones de este apartado, en el sitio oficial del [BID](#) se describen acciones desarrolladas en la región, que permiten fortalecer las habilidades de los estudiantes como, también, su inserción en el ámbito laboral.

**María Inés Abarrategui Fernández** [Linkedin](#)

(Trabajo presentado en el curso "Big Data". Codo a Codo 4.0, 2° cuatrimestre, 2022)