"Understanding high school students grades from socio-economic factors such as family size, internet access, parents education levels, alcohol consumption patterns"

Importing Dependencies

In [32]:

```
import pandas as pd
import numpy as np
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('seaborn-whitegrid')
```

Reading dataset

In [33]:

```
student_M = pd.read_csv('student-mat.csv')
student_P= pd.read_csv('student-por.csv')
```

Finding size of the dataset and counting null values

In [34]:

```
#Finding size of data set 1
shape= student_M.shape
#print(alcohol)
rowsnum = shape[0]
columnsnum = shape[1]
print('Rowsnum : ', rowsnum , 'Columnsnum : ', columnsnum)
print('Null values in dataset 1 =',student_M.isnull().sum().sum()) #printing how many nu

#Finding size of data set 2
shape= student_P.shape
#print(alcohol)
rowsnum = shape[0]
columnsnum = shape[1]
print('Rowsnum : ', rowsnum , 'Columnsnum : ', columnsnum)
print('Null values in dataset 2 =',student_P.isnull().sum().sum()) #printing how many nu
```

```
Rowsnum :  395 Columnsnum :  33
Null values in dataset 1 = 0
Rowsnum :  649 Columnsnum :  33
Null values in dataset 2 = 0
```

Description of data

View of data

In [35]:

```
#Maths Dataset
student_M.head()
```

Out[35]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	freetime	goou
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	3	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	3	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	3	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	2	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	3	

5 rows × 33 columns

In [36]:

```
#Portuguese Dataset
student_P.head()
```

Out[36]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	freetime	goou
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	3	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	3	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	3	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	2	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	3	

5 rows × 33 columns

Columns in dataset

In [37]:

```
#Dataset 1
print(student_M.columns)
#Dataset 2
print(student_P.columns)
```

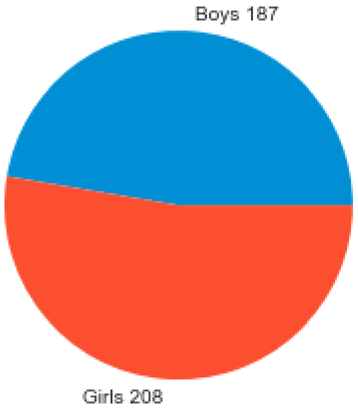
```
Index(['school', 'sex', 'age', 'address', 'famsize', 'Pstatus', 'Medu', 'Fedu',
       'Mjob', 'Fjob', 'reason', 'guardian', 'traveltime', 'studytime',
       'failures', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery',
       'higher', 'internet', 'romantic', 'famrel', 'freetime', 'goout', 'Dalc',
       'Walc', 'health', 'absences', 'G1', 'G2', 'G3'],
      dtype='object')
Index(['school', 'sex', 'age', 'address', 'famsize', 'Pstatus', 'Medu', 'Fedu',
       'Mjob', 'Fjob', 'reason', 'guardian', 'traveltime', 'studytime',
       'failures', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery',
       'higher', 'internet', 'romantic', 'famrel', 'freetime', 'goout', 'Dalc',
       'Walc', 'health', 'absences', 'G1', 'G2', 'G3'],
      dtype='object')
```

Number of students

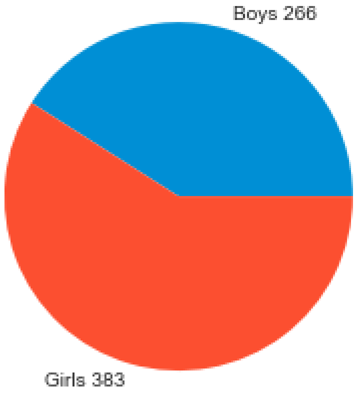
In [38]:

```
#For Maths Dataset
print("For dataset 1")
def countStudent( stra):
    return student_M['G3'].loc[(student_M['sex']==stra)].count()


y = np.array([ countStudent('M'),countStudent('F') ])
mylabels = ["Boys "+ str(countStudent('M')), "Girls " +  str(countStudent('F'))]

plt.pie(y, labels = mylabels)
plt.show()

#For portuguese Dataset
print("For dataset 2")
def countStudent( stra):
    return student_P['G3'].loc[(student_P['sex']==stra)].count()


y = np.array([ countStudent('M'),countStudent('F') ])
mylabels = ["Boys "+ str(countStudent('M')), "Girls " +  str(countStudent('F'))]

plt.pie(y, labels = mylabels)
plt.show()
```

```
For dataset 1
```



```
For dataset 2
```

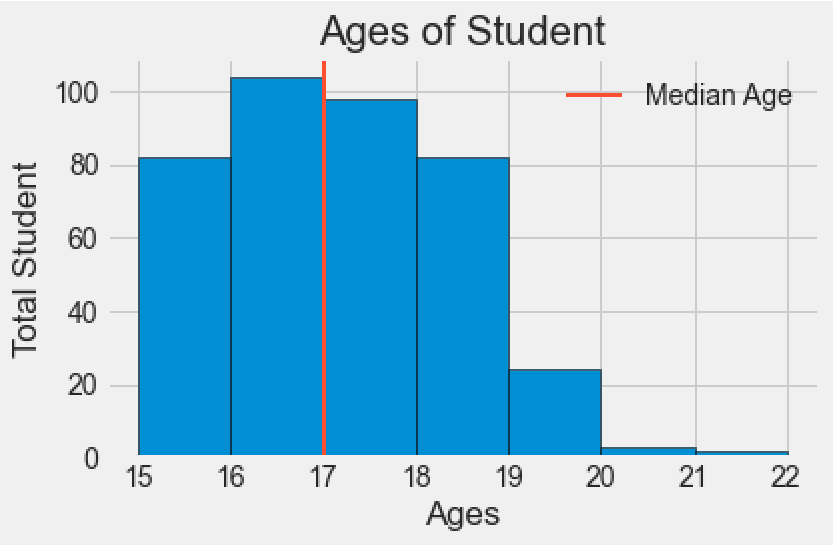


Age of the students

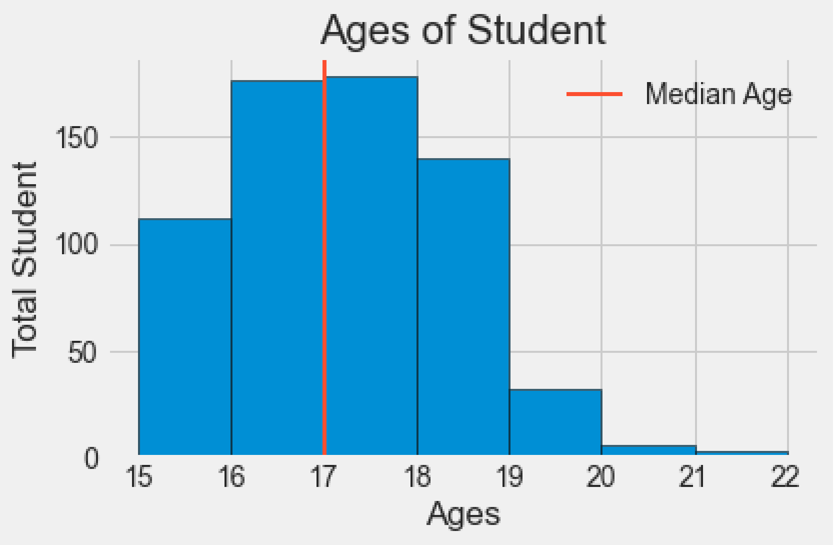
In [39]:

```
# For dataset 1
print("For dataset 1")
student_M['Responder_id'] = range(1, len(student_M) + 1)

plt.style.use('fivethirtyeight')

ages = student_M['age']

bins = [15,16,17,18,19,20,21,22]

plt.hist(ages, bins=bins, edgecolor='black')

median_age = student_M['age'].median()
print('median age -',median_age)
color = '#fc4f30'

plt.axvline(median_age, color=color, label='Median Age', linewidth=2)

plt.legend()

plt.title('Ages of Student')
plt.xlabel('Ages')
plt.ylabel('Total Student')

plt.tight_layout()

plt.show()

# For dataset 2
print("For dataset 2")

student_P['Responder_id'] = range(1, len(student_P) + 1)

plt.style.use('fivethirtyeight')

ages = student_P['age']

bins = [15,16,17,18,19,20,21,22]

plt.hist(ages, bins=bins, edgecolor='black')

median_age = student_P['age'].median()
print('median age -',median_age)
color = '#fc4f30'

plt.axvline(median_age, color=color, label='Median Age', linewidth=2)

plt.legend()

plt.title('Ages of Student')
plt.xlabel('Ages')
plt.ylabel('Total Student')

plt.tight_layout()

plt.show()
```

```
For dataset 1
median age - 17.0
```



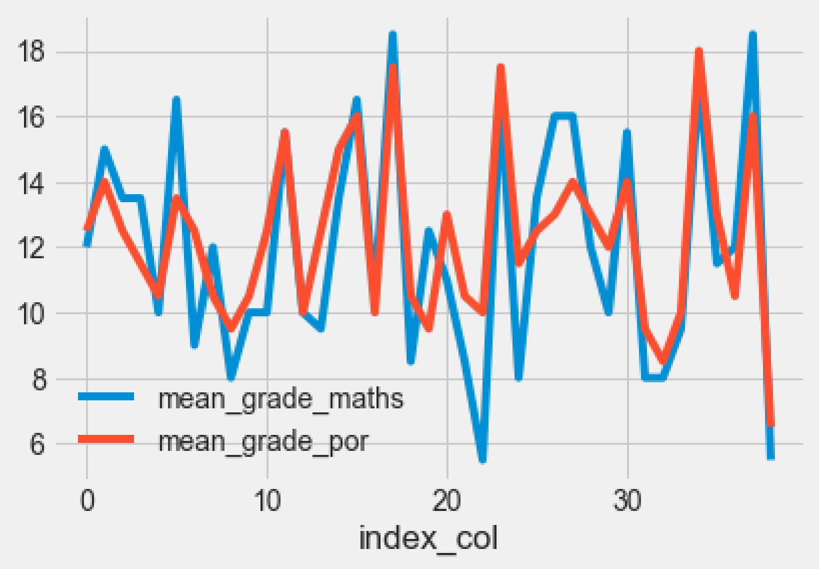
```
For dataset 2
median age - 17.0
```



Merging both dataset to make some unique inferances

```
In [40]: merge_data = pd.merge(student_M, student_P , how ='inner', left_on = ['school', 'sex',
                'Mjob', 'Fjob', 'reason', 'guardian', 'traveltime', 'studytime',
                'failures', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery',
                'higher', 'internet', 'romantic', 'famrel', 'freetime', 'goout', 'Dalc',
                'Walc', 'health', 'absences'] ,
                              right_on = ['school', 'sex', 'age', 'address', 'famsize', 'Pstatus
                'Mjob', 'Fjob', 'reason', 'guardian', 'traveltime', 'studytime',
                'failures', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery',
                'higher', 'internet', 'romantic', 'famrel', 'freetime', 'goout', 'Dalc',
                'Walc', 'health', 'absences'])

         print(merge_data.columns)


         merge_data['mean_grade_maths'] = merge_data[['G1_x','G2_x']].mean(axis=1)
         merge_data['mean_grade_por'] = merge_data[['G1_y', 'G2_y']].mean(axis=1)
         merge_data['mean_grade'] = merge_data[['G1_x','G2_x','G1_y', 'G2_y']].mean(axis=1)
         student_M['mean_grade'] = student_M[['G1','G2']].mean(axis=1)
         student_P['mean_grade'] = student_P[['G1','G2']].mean(axis=1)

         #Performance in both the subject when student is unique
         merge_data['index_col'] = merge_data.index
         merge_data.plot(x="index_col", y=["mean_grade_maths", "mean_grade_por"])
         plt.show()

         # The graph shows performance of the student dosent change radically with cahnge in subj
```

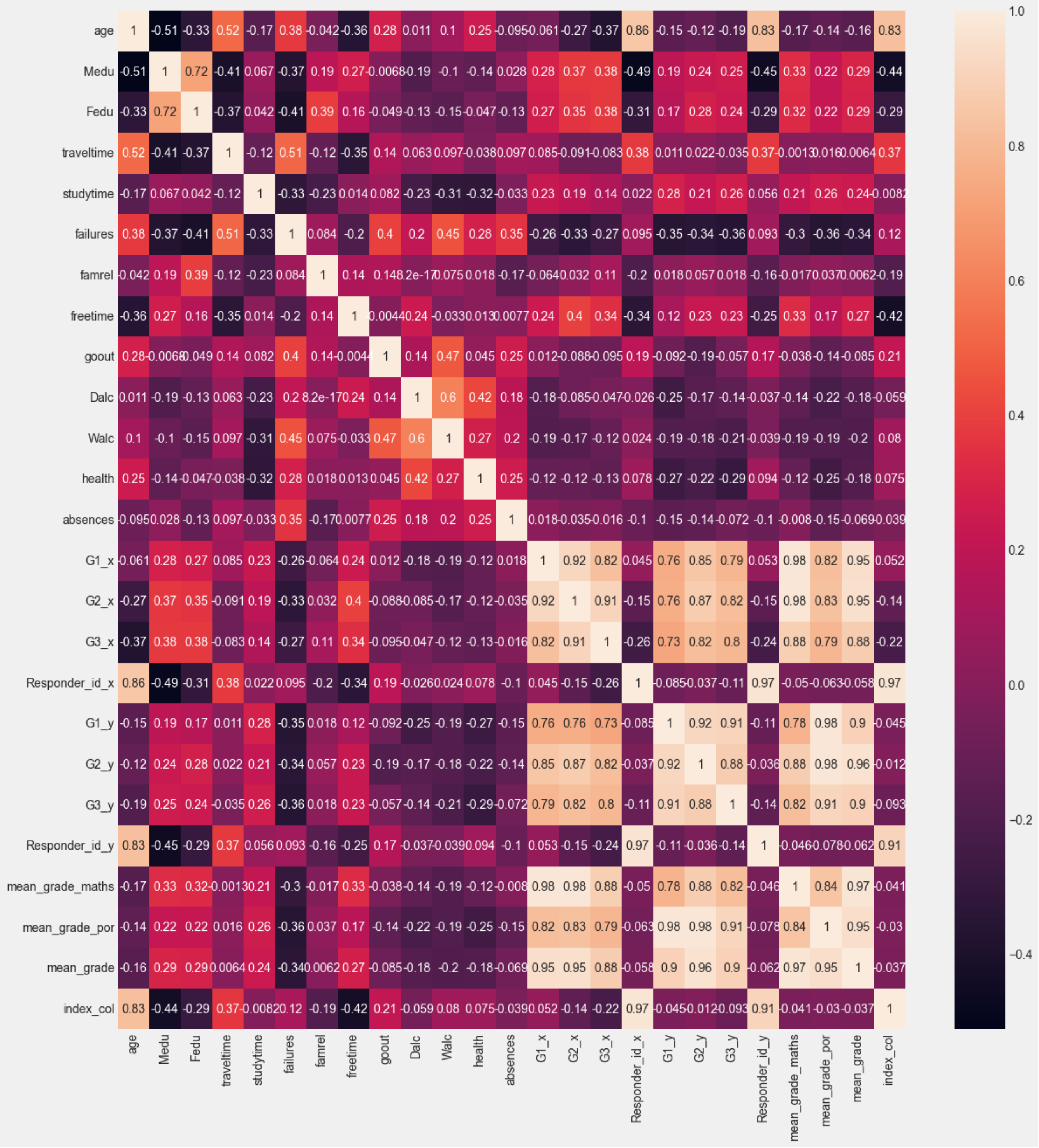
```
Index(['school', 'sex', 'age', 'address', 'famsize', 'Pstatus', 'Medu', 'Fedu',
       'Mjob', 'Fjob', 'reason', 'guardian', 'traveltime', 'studytime',
       'failures', 'schoolsup', 'famsup', 'paid', 'activities', 'nursery',
       'higher', 'internet', 'romantic', 'famrel', 'freetime', 'goout', 'Dalc',
       'Walc', 'health', 'absences', 'G1_x', 'G2_x', 'G3_x', 'Responder_id_x',
       'G1_y', 'G2_y', 'G3_y', 'Responder_id_y'],
      dtype='object')
```



Correlation Mattrix to find out correlated attributes

```
In [11]: plt.figure(figsize=(18, 20))
         import seaborn as sns
         sns.heatmap(merge_data.corr(),annot=True)
```

```
Out[11]: <AxesSubplot:>
```



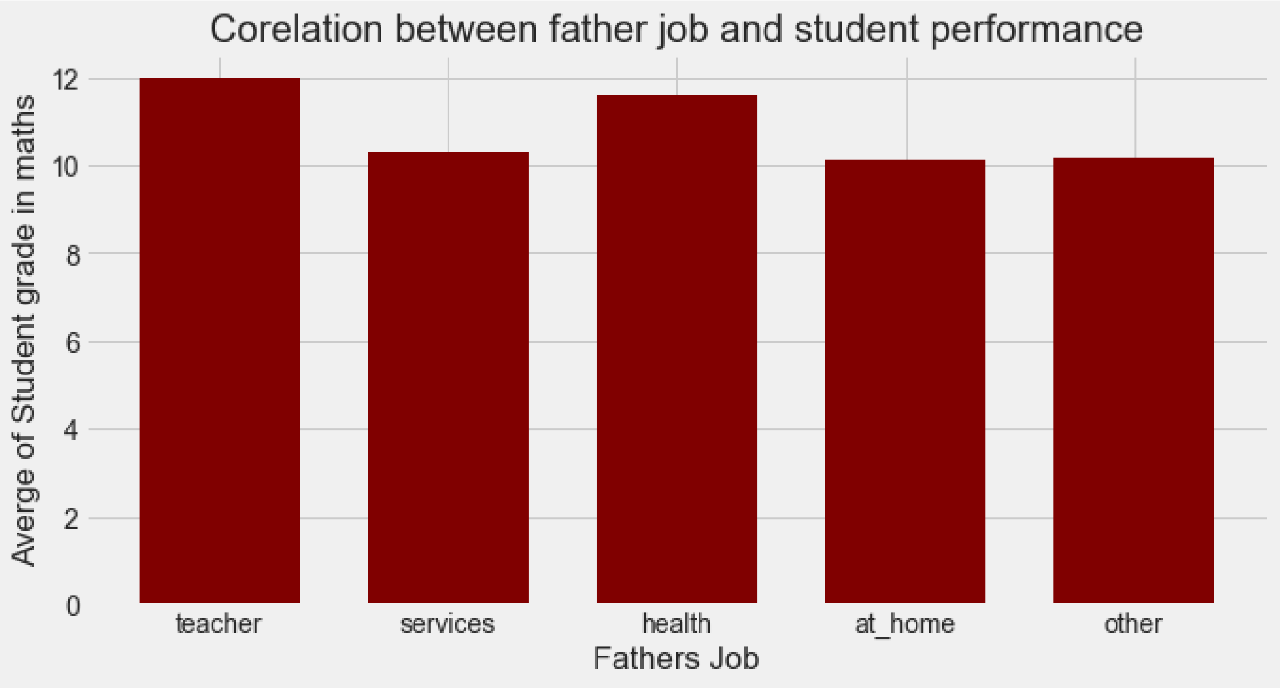
Some of the inferances made using correlation matrix

1. Does Father profession affect student performance

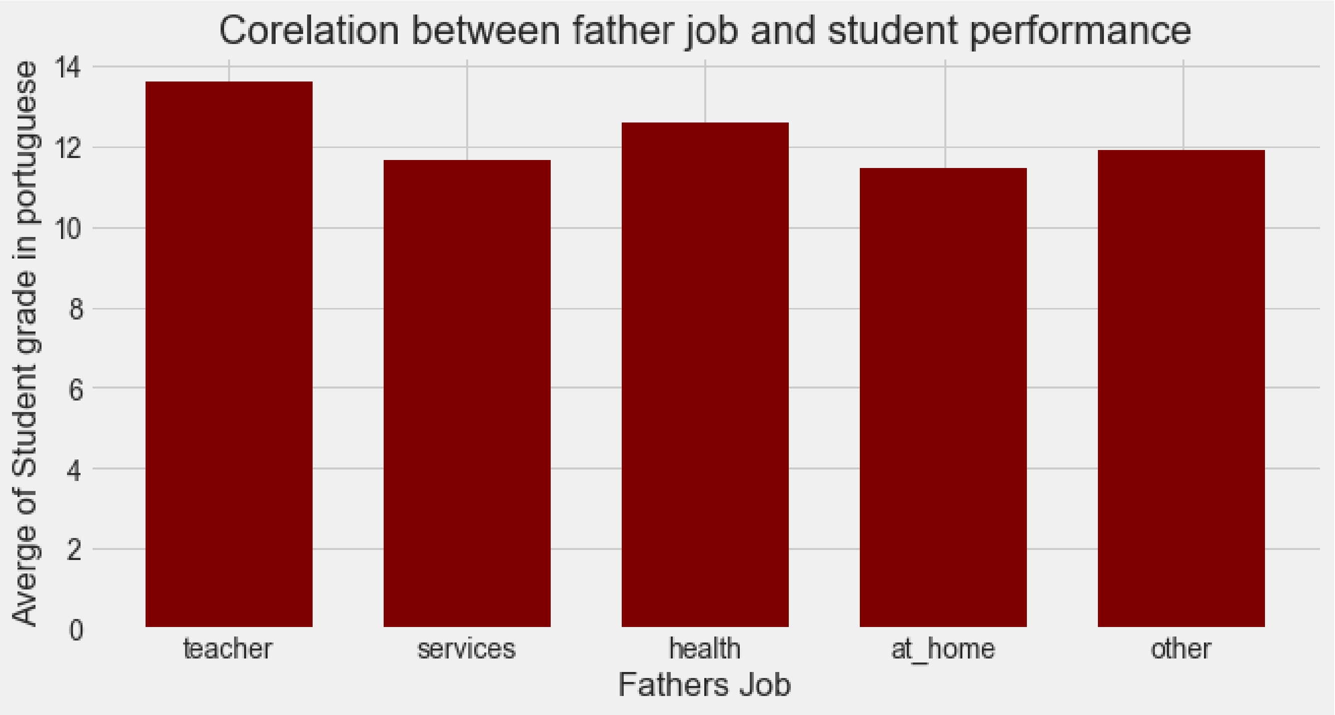
In [41]:

```
# For maths dataset
print("For maths dataset")
def fjob(stra):
    return student_M['G3'].loc[(student_M['Fjob']==stra)].mean()


Fjob = {'teacher':fjob('teacher'), 'services':fjob('services'), 'health':fjob('health')
        'at_home':fjob('at_home') , 'other':fjob('other')}
Father_job = list(Fjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (10, 5))

# creating the bar plot
plt.bar(Father_job, Student_grade, color ='maroon',
        width = 0.7)

plt.xlabel("Fathers Job")
plt.ylabel("Averge of Student grade in maths")
plt.title("Corelation between father job and student performance")
plt.show()

# For portuguese dataset
print("For portuguese dataset")
def fjob(stra):
    return student_P['G3'].loc[(student_P['Fjob']==stra)].mean()


Fjob = {'teacher':fjob('teacher'), 'services':fjob('services'), 'health':fjob('health')
        'at_home':fjob('at_home') , 'other':fjob('other')}
Father_job = list(Fjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (10, 5))

# creating the bar plot
plt.bar(Father_job, Student_grade, color ='maroon',
        width = 0.7)

plt.xlabel("Fathers Job")
plt.ylabel("Averge of Student grade in portuguese")
plt.title("Corelation between father job and student performance")
plt.show()

#Having father profession as teacher seems to affect student performance
```

```
For maths dataset
```



```
For portuguese dataset
```

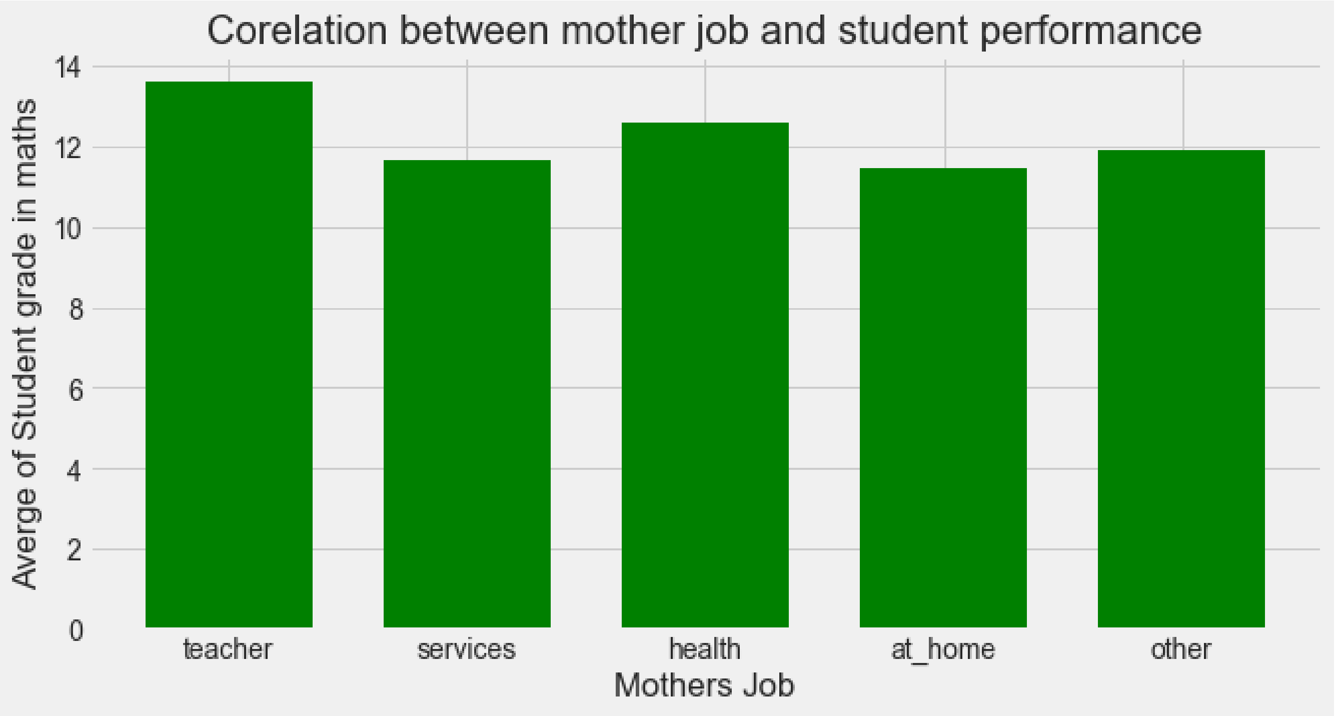


2. Effect of Mother job on student Grade

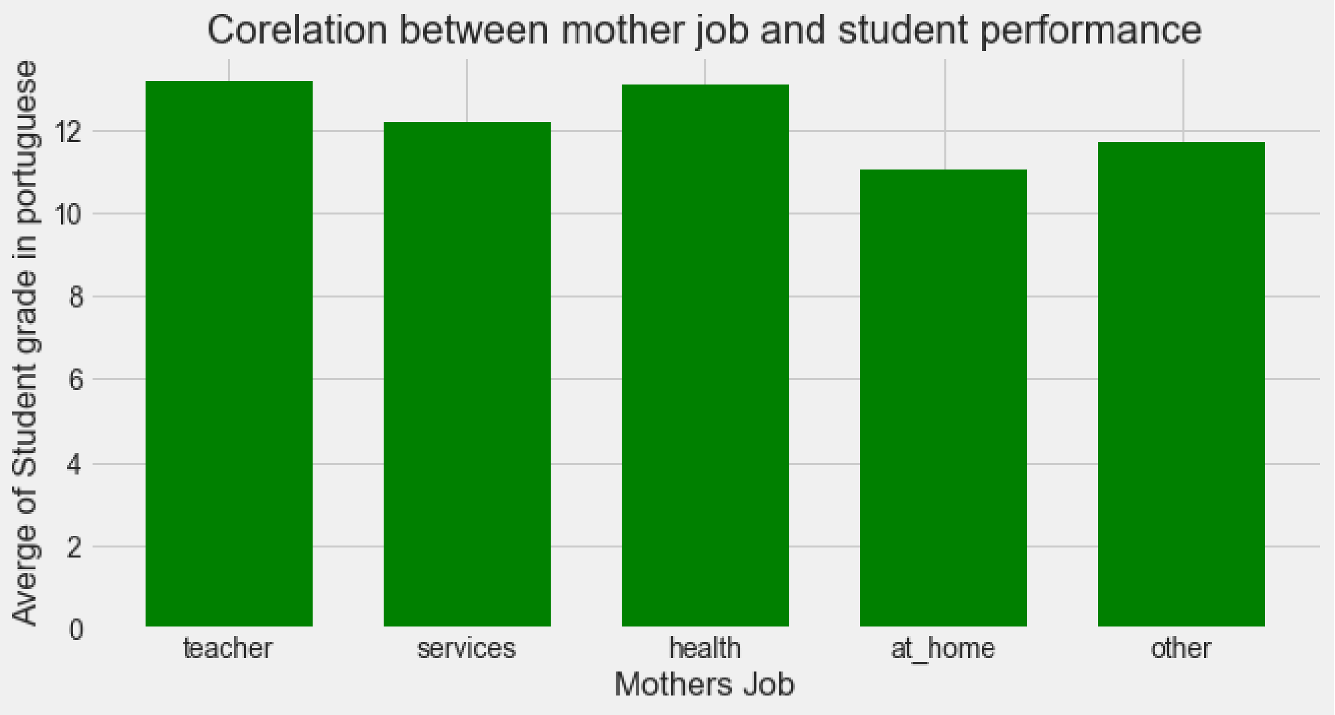
In [42]:

```
# For maths dataset
print("For maths dataset")
def Mjob(stra):
    return student_M['G3'].loc[(student_M['Mjob']==stra)].mean()


Mjob = {'teacher':Mjob('teacher'), 'services':Mjob('services'), 'health':Mjob('health')
        'at_home':Mjob('at_home') , 'other':Mjob('other')}
Mother_job = list(Mjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (10, 5))

# creating the bar plot
plt.bar(Mother_job, Student_grade, color ='green',
        width = 0.7)

plt.xlabel("Mothers Job")
plt.ylabel("Averge of Student grade in maths")
plt.title("Corelation between mother job and student performance")
plt.show()

# For portuguese dataset
print("For portuguese dataset")
def fjob(stra):
    return student_P['G3'].loc[(student_P['Mjob']==stra)].mean()


Fjob = {'teacher':fjob('teacher'), 'services':fjob('services'), 'health':fjob('health')
        'at_home':fjob('at_home') , 'other':fjob('other')}
Father_job = list(Fjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (10, 5))

# creating the bar plot
plt.bar(Father_job, Student_grade, color ='green',
        width = 0.7)

plt.xlabel("Mothers Job")
plt.ylabel("Averge of Student grade in portuguese")
plt.title("Corelation between mother job and student performance")
plt.show()

#So both mother and father profession as teacher seems to affect student performance
```

```
For maths dataset
```



```
For portuguese dataset
```

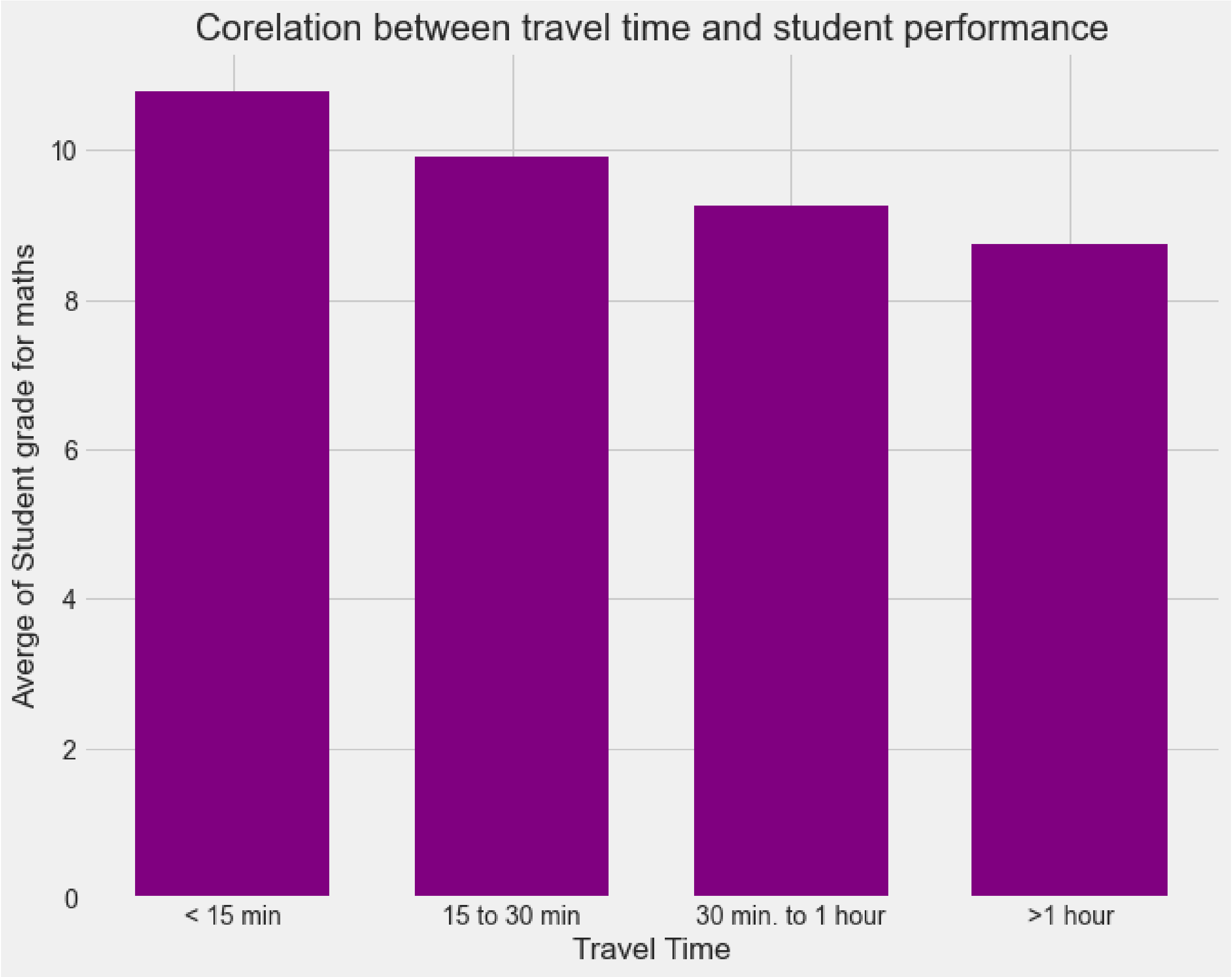


3.Does traveltime affects performance of students

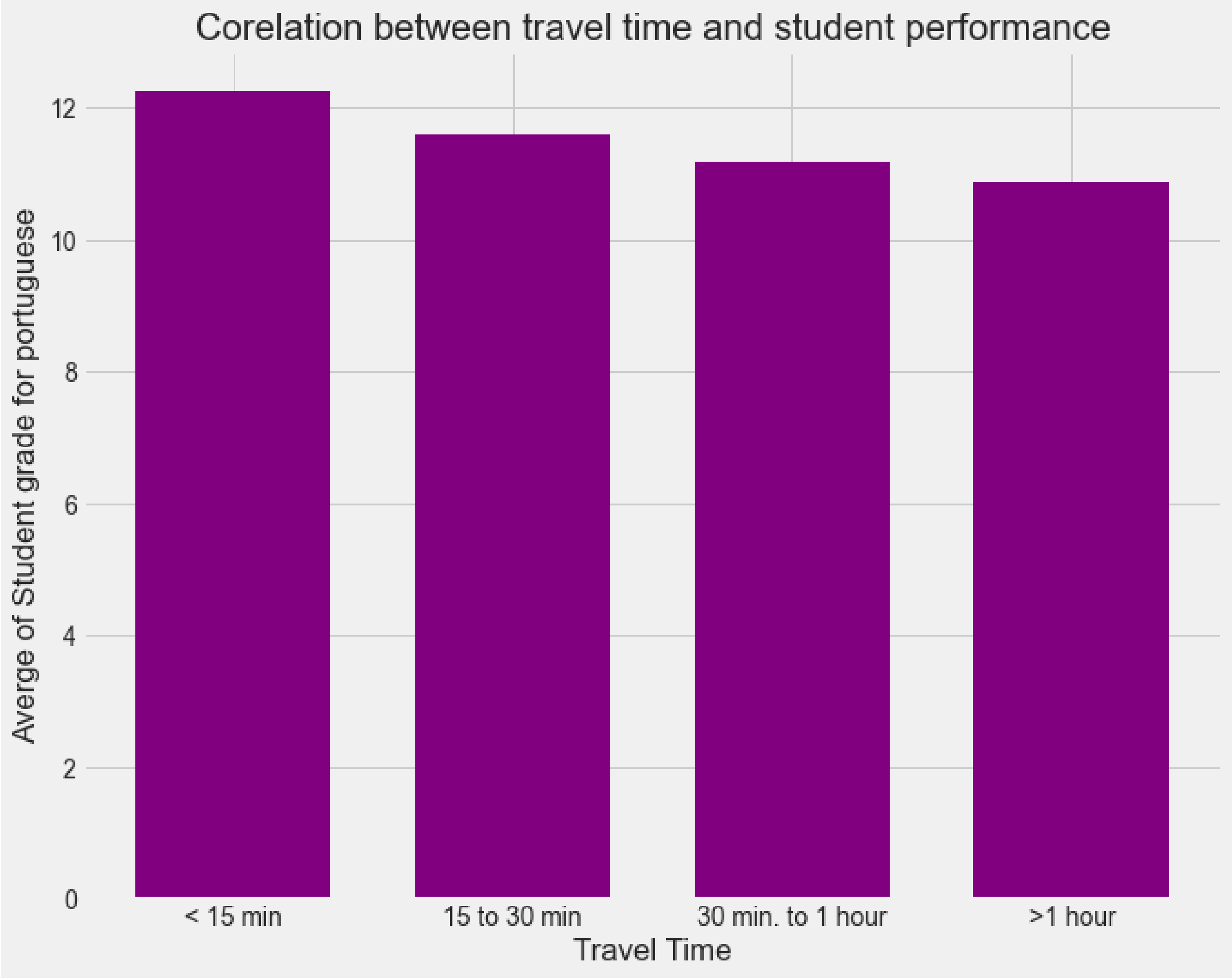
In [43]:

```
# for dataset 1
print("For maths dataset")


def perform(stra):
    return student_M['G3'].loc[(student_M['traveltime']==stra)].mean()

Fjob = {'< 15 min':perform(1), '15 to 30 min':perform(2), '30 min. to 1 hour':perform(3
        '>1 hour':perform(4) }

perform = list(Fjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (10, 8))

# creating the bar plot
plt.bar(perform, Student_grade, color ='purple',
        width = 0.7)

plt.xlabel("Travel Time")
plt.ylabel("Averge of Student grade for maths")
plt.title("Corelation between travel time and student performance")
plt.show()

# for dataset 2

print("For portuguese dataset")

def perform(stra):
    return student_P['G3'].loc[(student_P['traveltime']==stra)].mean()

Fjob = {'< 15 min':perform(1), '15 to 30 min':perform(2), '30 min. to 1 hour':perform(3
        '>1 hour':perform(4) }

perform = list(Fjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (10, 8))

# creating the bar plot
plt.bar(perform, Student_grade, color ='purple',
        width = 0.7)

plt.xlabel("Travel Time")
plt.ylabel("Averge of Student grade for portuguese")
plt.title("Corelation between travel time and student performance")
plt.show()

## Travel time seems to affect student performance in both subjects
```

```
For maths dataset
```



```
For portuguese dataset
```

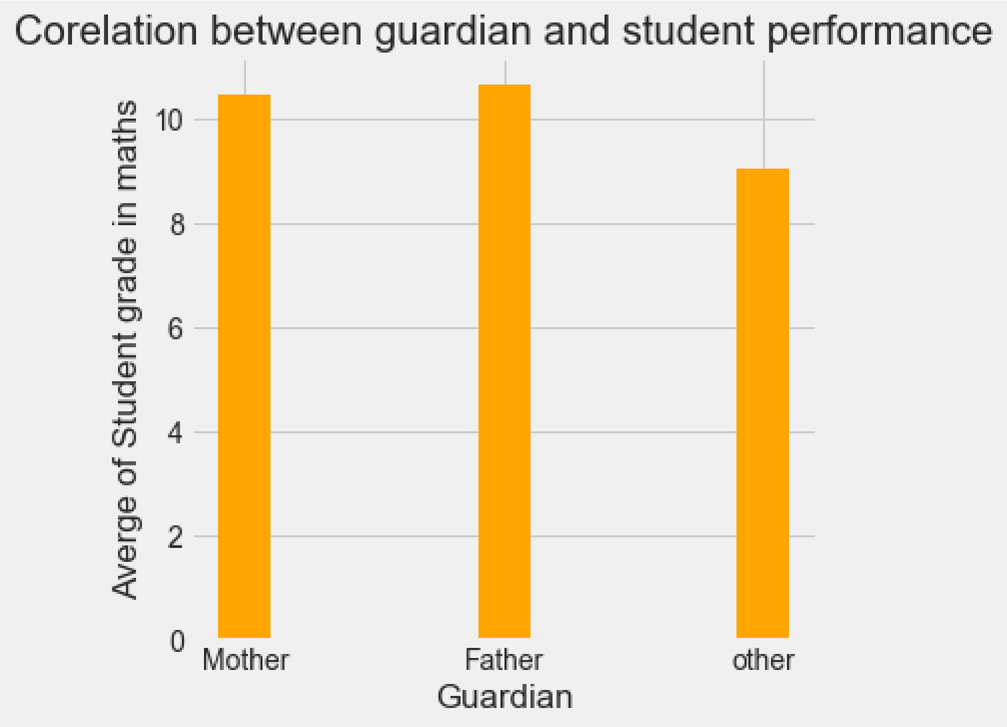


4.Does guardian other than father and mother affects student performance

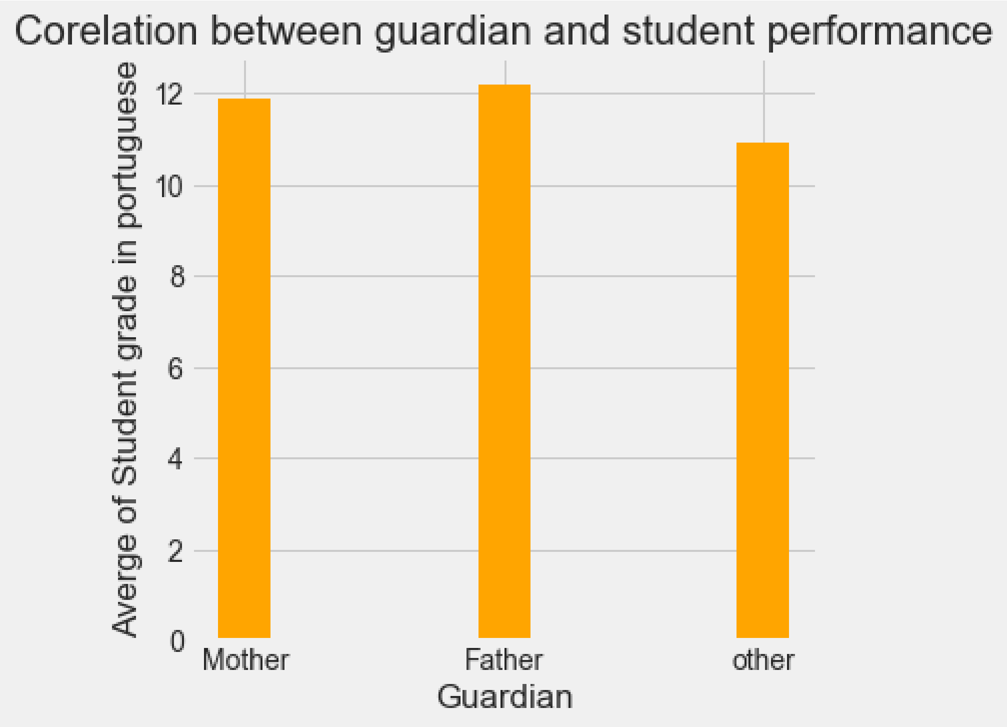
In [44]:

```
# For dataset 1
print("For maths dataset")

def perform(stra):
    return student_M['G3'].loc[(student_M['guardian']==stra)].mean()


Fjob = {'Mother':perform('mother'), 'Father':perform('father') ,  'other':perform('other
perform = list(Fjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (5, 5))

# creating the bar plot
plt.bar(perform, Student_grade, color ='orange',
        width = 0.2)

plt.xlabel("Guardian")
plt.ylabel("Averge of Student grade in maths")
plt.title("Corelation between guardian and student performance")
plt.show()

# For dataset 2

print("For portuguese dataset")
def perform(stra):
    return student_P['G3'].loc[(student_P['guardian']==stra)].mean()


Fjob = {'Mother':perform('mother'), 'Father':perform('father') ,  'other':perform('other
perform = list(Fjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (5, 5))

# creating the bar plot
plt.bar(perform, Student_grade, color ='orange',
        width = 0.2)

plt.xlabel("Guardian")
plt.ylabel("Averge of Student grade in portuguese")
plt.title("Corelation between guardian and student performance")
plt.show()

#Having guardian other than student immediate parents seems to affet student performance
```

```
For maths dataset
```



```
For portuguese dataset
```

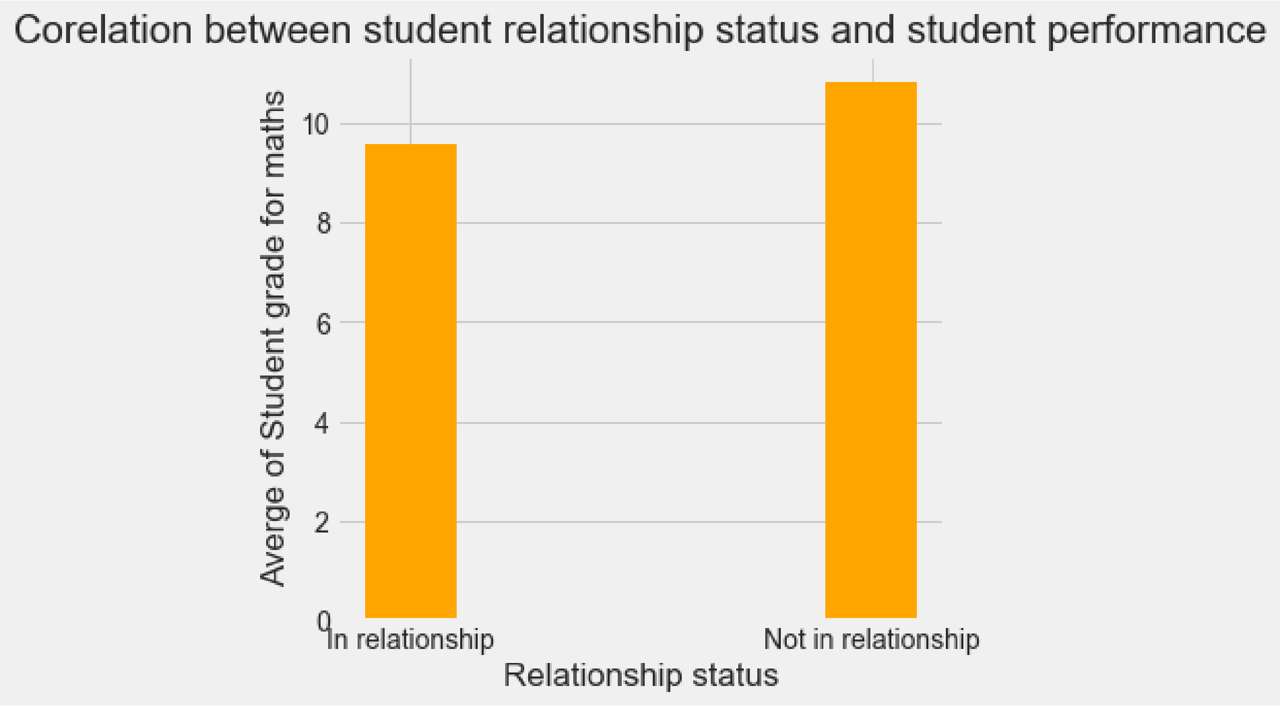


5.Does being in relationship have any effect on student performance

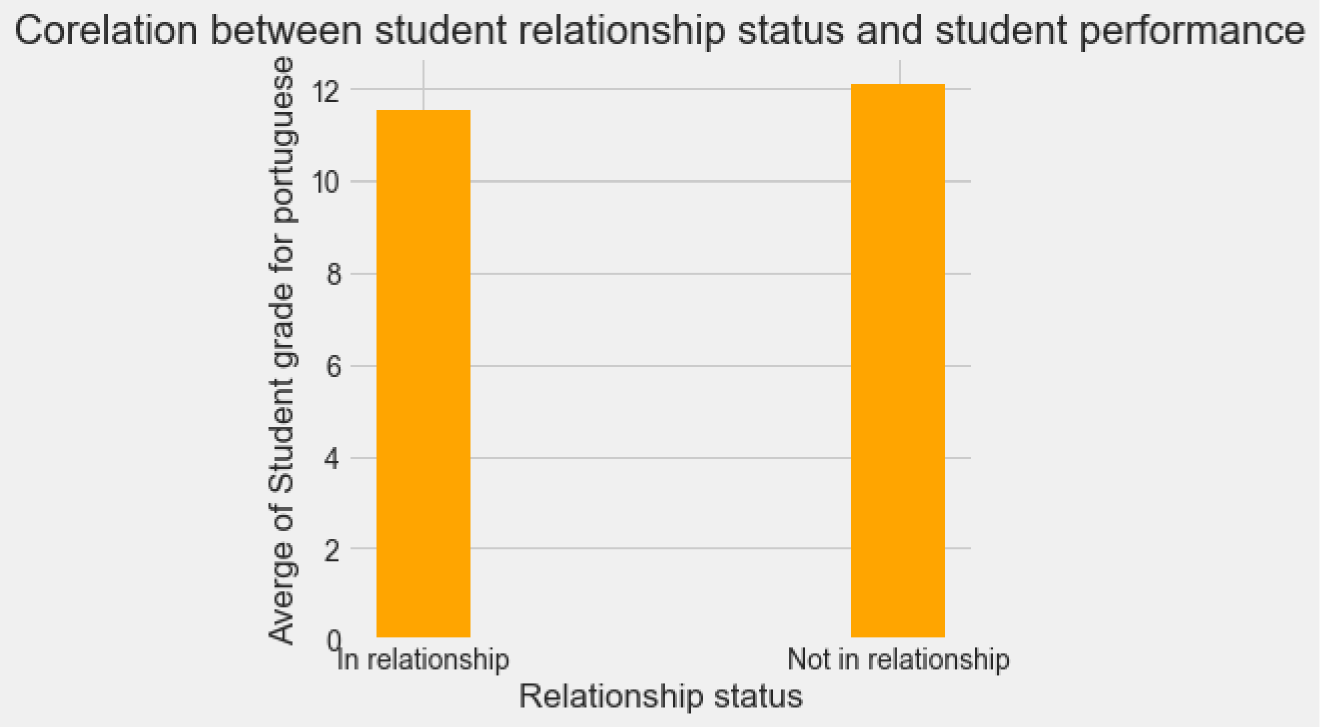
In [45]:

```
# For Dataset 1
print("For maths dataset")

def perform(stra):
    return student_M['G3'].loc[(student_M['romantic']==stra)].mean()


Fjob = {'In relationship':perform('yes'), 'Not in relationship':perform('no')}
performL = list(Fjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (5, 5))

# creating the bar plot
plt.bar(performL, Student_grade, color ='orange',
        width = 0.2)

plt.xlabel("Relationship status")
plt.ylabel("Averge of Student grade for maths")
plt.title("Corelation between student relationship status and student performance")
plt.show()

# For Dataset 2

print("For portuguese dataset")
def perform(stra):
    return student_P['G3'].loc[(student_P['romantic']==stra)].mean()


Fjob = {'In relationship':perform('yes'), 'Not in relationship':perform('no')}
performL = list(Fjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (5, 5))

# creating the bar plot
plt.bar(performL, Student_grade, color ='orange',
        width = 0.2)

plt.xlabel("Relationship status")
plt.ylabel("Averge of Student grade for portuguese")
plt.title("Corelation between student relationship status and student performance")
plt.show()
```

```
For maths dataset
```



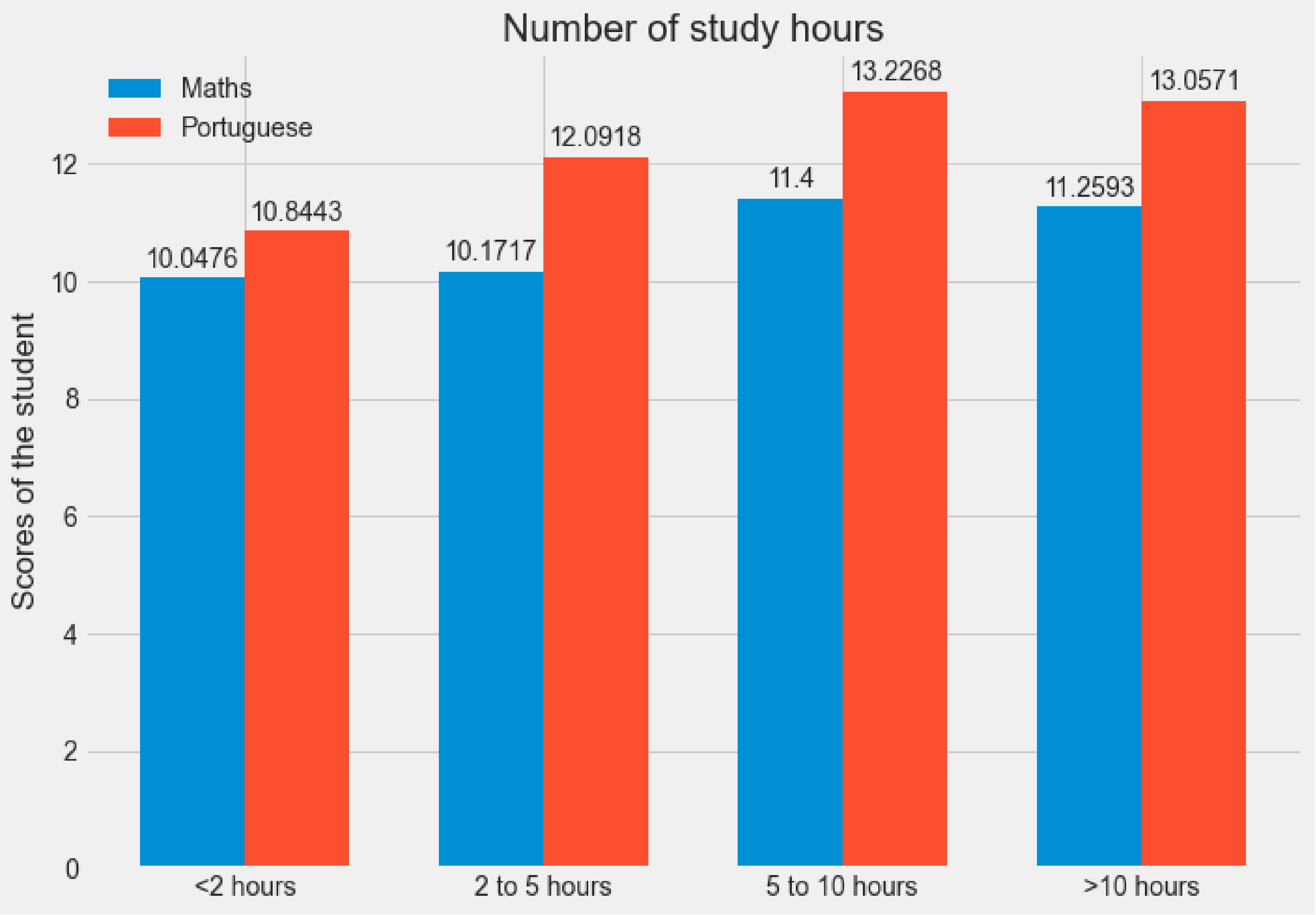
```
For portuguese dataset
```



6.Does number of study hours affects performance of students

In [46]:

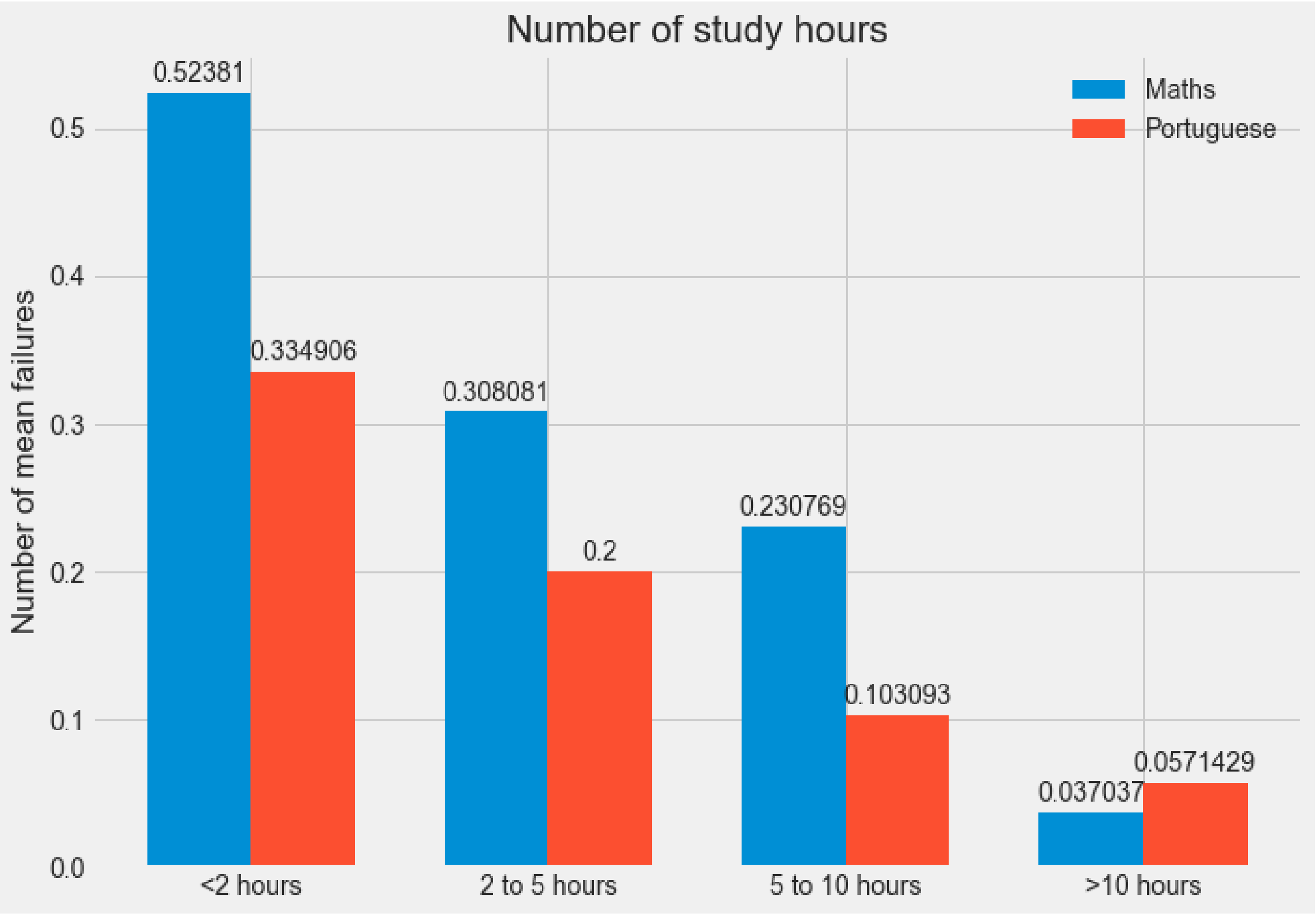
```
def g_math(stra):
    return student_M['G3'].loc[(student_M['studytime']==stra)].mean()

def g_por(stra):
    return student_P['G3'].loc[(student_P['studytime']==stra)].mean()

N = 3

math_means = (g_math(1), g_math(2), g_math(3),g_math(4))
por_means = (g_por(1), g_por(2), g_por(3),g_por(4))

labels = ['<2 hours', '2 to 5 hours', '5 to 10 hours','>10 hours']

x = np.arange(len(labels))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots()
fig.set_size_inches(10, 7)
rects1 = ax.bar(x - width/2, math_means, width, label='Maths')
rects2 = ax.bar(x + width/2, por_means, width, label='Portuguese')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Scores of the student')
ax.set_title('Number of study hours')
ax.set_xticks(x, labels)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)

fig.tight_layout()

plt.show()

# Increasing study hours will certainly help student to increase his grades
```



7.Does high study hours means lower failures

In [47]:

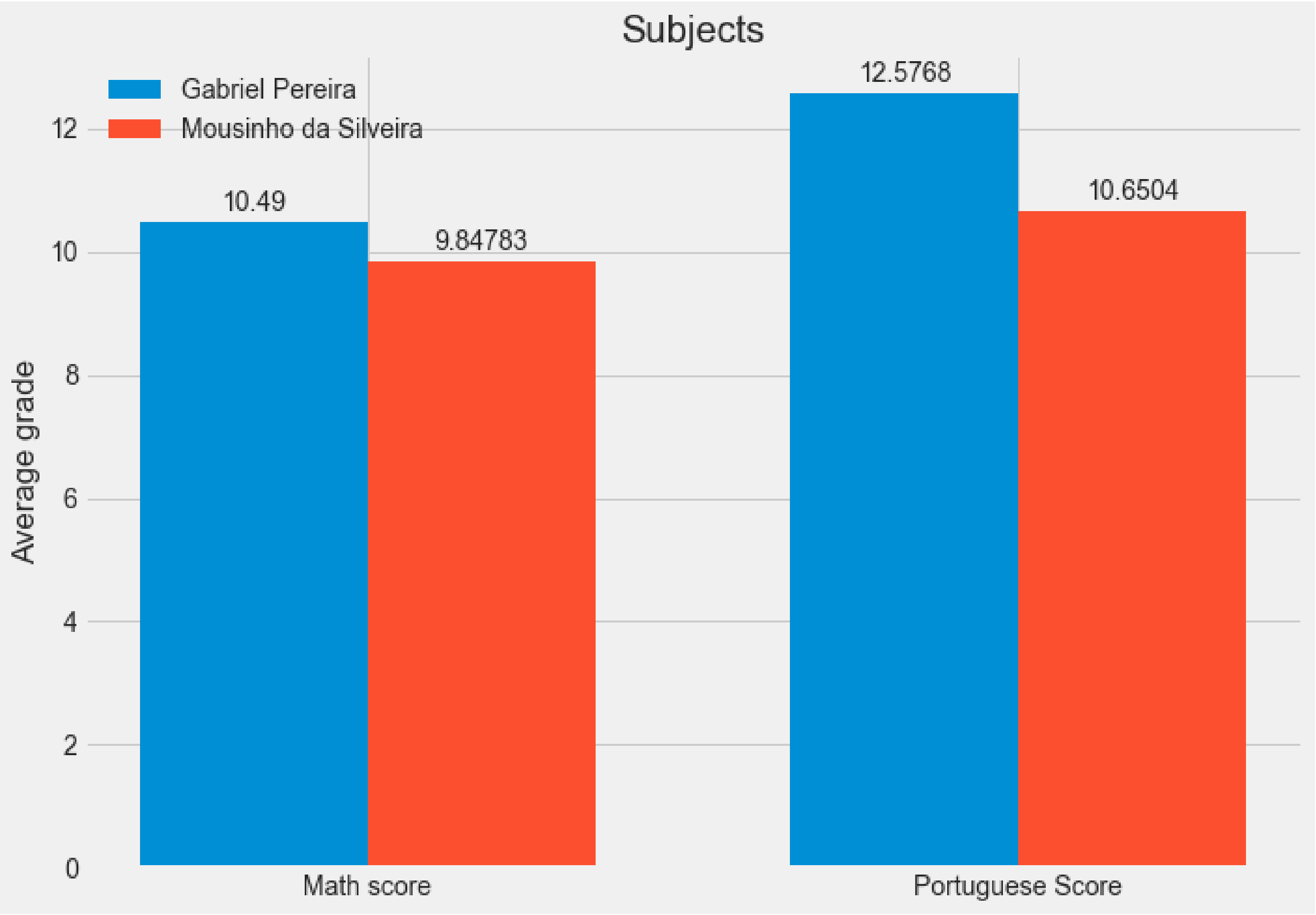
```
def g_math(stra):
    return student_M['failures'].loc[(student_M['studytime']==stra)].mean()

def g_por(stra):
    return student_P['failures'].loc[(student_P['studytime']==stra)].mean()

N = 3

math_means = (g_math(1), g_math(2), g_math(3),g_math(4))
por_means = (g_por(1), g_por(2), g_por(3),g_por(4))

labels = ['<2 hours', '2 to 5 hours', '5 to 10 hours','>10 hours']

x = np.arange(len(labels))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots()
fig.set_size_inches(10, 7)
rects1 = ax.bar(x - width/2, math_means, width, label='Maths')
rects2 = ax.bar(x + width/2, por_means, width, label='Portuguese')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Number of mean failures')
ax.set_title('Number of study hours')
ax.set_xticks(x, labels)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)

fig.tight_layout()

plt.show()

# Increasing study hours will certainly help student to reduce his failures
```



8.Does student performance also depends on school

In [48]:

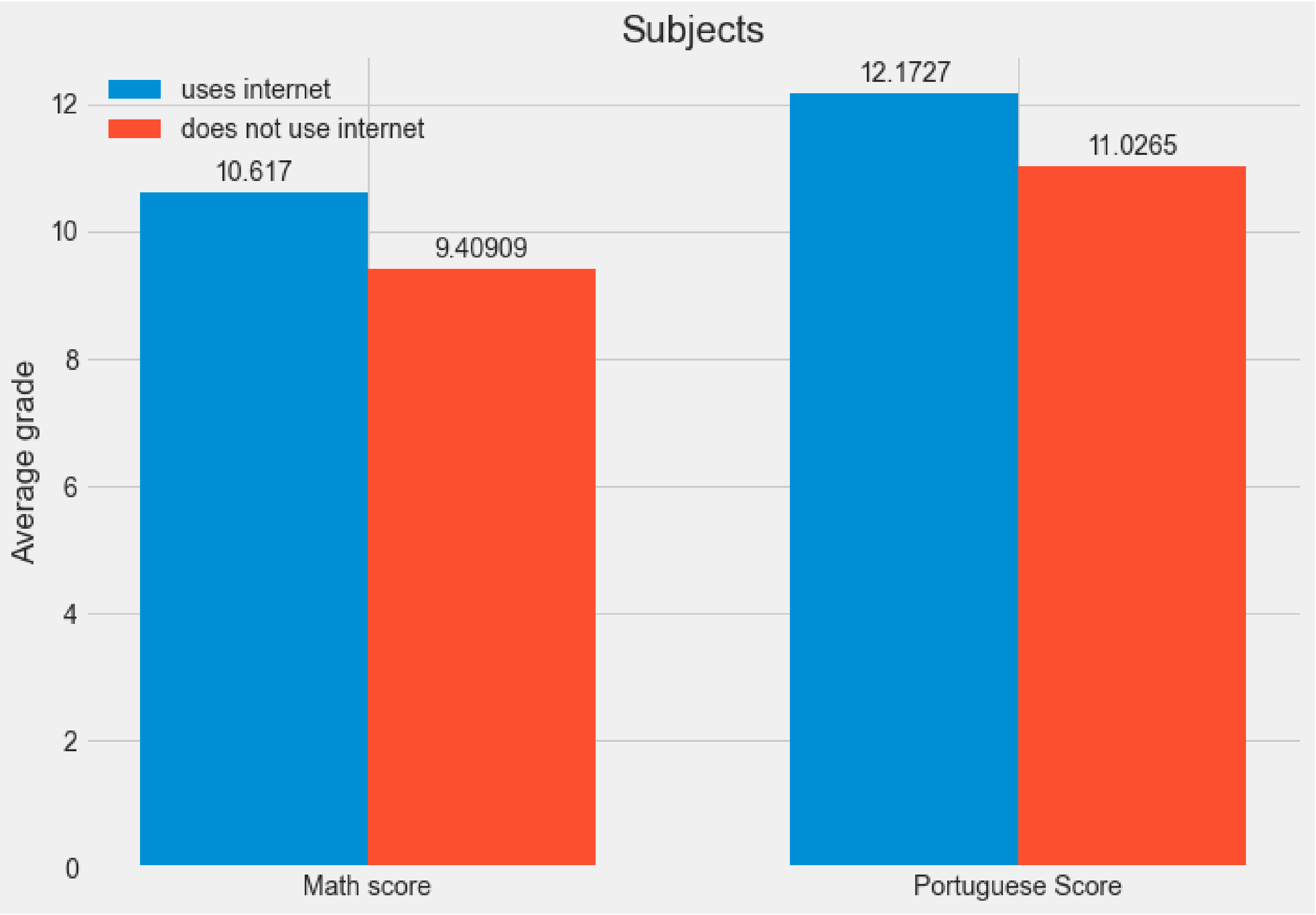
```
def g_math(stra):
    return student_M['G3'].loc[(student_M['school']==stra)].mean()

def g_por(stra):
    return student_P['G3'].loc[(student_P['school']==stra)].mean()

N = 3

GP = (g_math('GP'), g_por('GP'))
MS = (g_math('MS'), g_por('MS'))

labels = ['Math score', 'Portuguese Score']

x = np.arange(len(labels))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots()
fig.set_size_inches(10, 7)
rects1 = ax.bar(x - width/2, GP, width, label='Gabriel Pereira')
rects2 = ax.bar(x + width/2, MS, width, label='Mousinho da Silveira')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Average grade')
ax.set_title('Subjects')
ax.set_xticks(x, labels)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)

fig.tight_layout()

plt.show()

# School does affect student performance here GP school seems to have greater average sc
```



9. How internet affect the student performance

In [49]:

```
def g_math(stra):
    return student_M['G3'].loc[(student_M['internet']==stra)].mean()

def g_por(stra):
    return student_P['G3'].loc[(student_P['internet']==stra)].mean()

N = 3

uses_internet = (g_math('yes'), g_por('yes'))
Nuse_internet = (g_math('no'), g_por('no'))

labels = ['Math score', 'Portuguese Score']

x = np.arange(len(labels))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots()
fig.set_size_inches(10, 7)
rects1 = ax.bar(x - width/2, uses_internet, width, label='uses internet')
rects2 = ax.bar(x + width/2, Nuse_internet, width, label='does not use internet')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Average grade')
ax.set_title('Subjects')
ax.set_xticks(x, labels)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)

fig.tight_layout()

plt.show()

# Using internet services seems to improve performance of the student
```

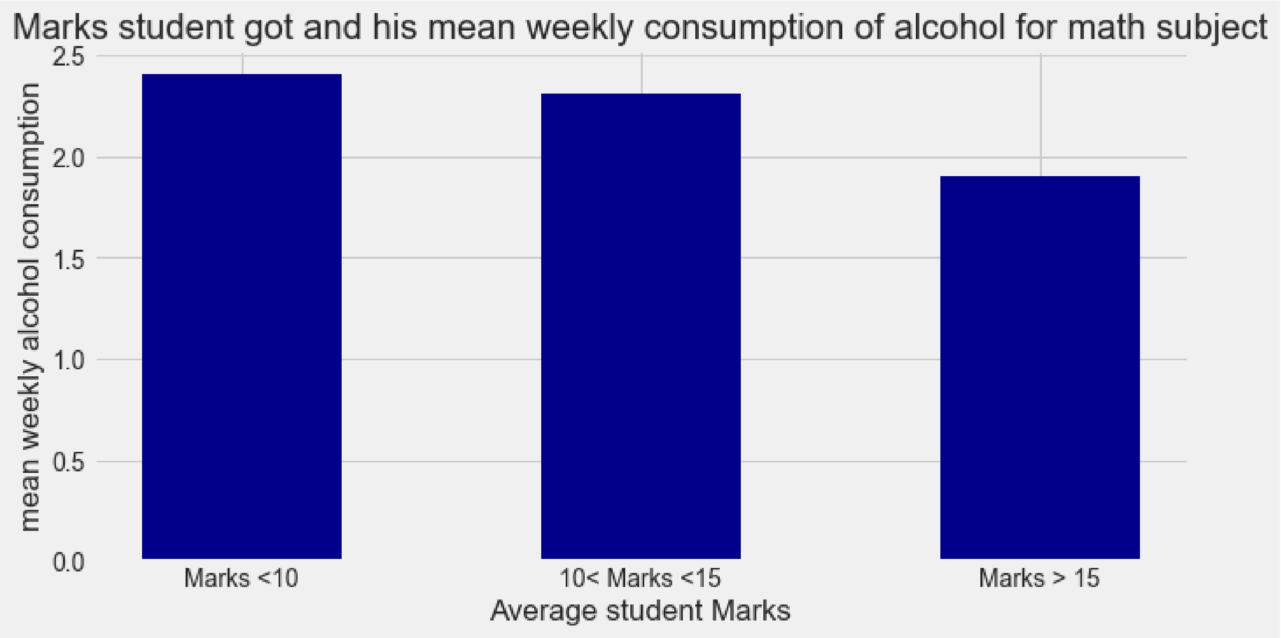


10. Marks student got and his weekly consumption of alcohol

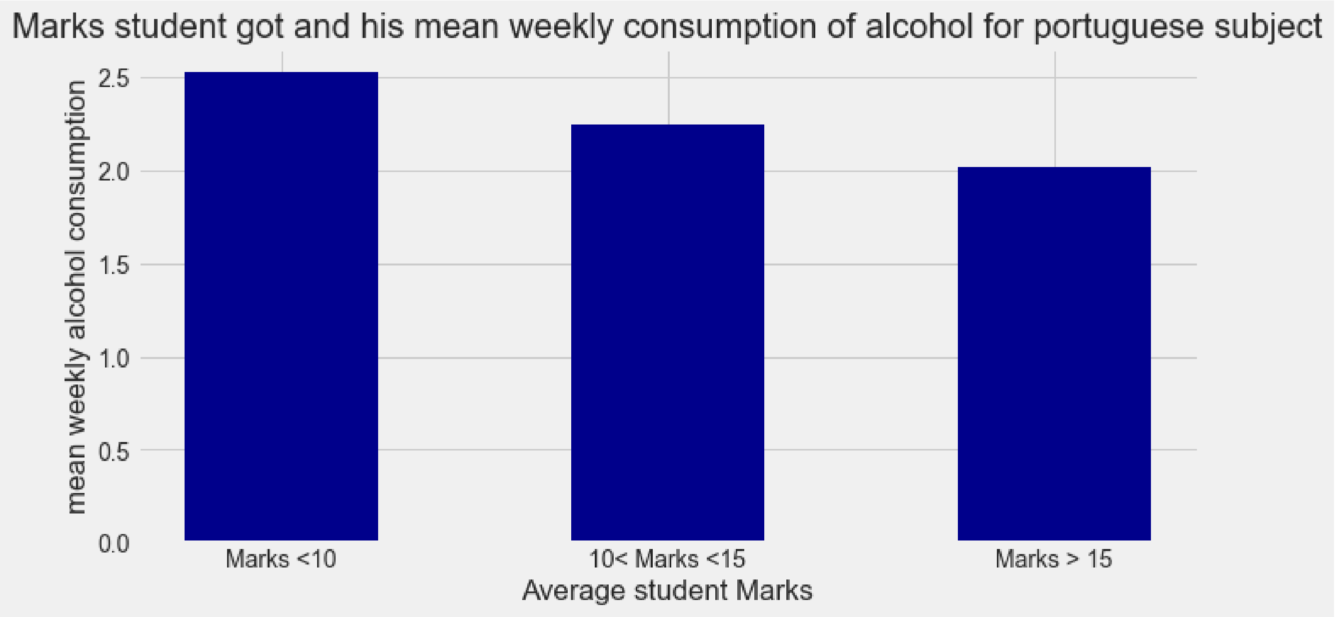
In [50]:

```
# For dataset 1
print("For maths dataset")

def perform(stra ,stra1):
    return student_M['Walc'].loc[(student_M['G3']>=stra) & (student_M['mean_grade']< str

Fjob = {'Marks <10':perform(0,11), '10< Marks <15':perform(10,16), 'Marks > 15':perform(

performL = list(Fjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (10, 5))

# creating the bar plot
plt.bar(performL, Student_grade, color ='DarkBlue',
        width = 0.5)

plt.xlabel("Average student Marks")
plt.ylabel("mean weekly alcohol consumption")
plt.title("Marks student got and his mean weekly consumption of alcohol for math subject
plt.show()

# For dataset 2

print("For portuguese dataset")
def perform(stra ,stra1):
    return student_P['Walc'].loc[(student_P['G3']>=stra) & (student_P['mean_grade']< str

Fjob = {'Marks <10':perform(0,11), '10< Marks <15':perform(10,16), 'Marks > 15':perform(

performL = list(Fjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (10, 5))

# creating the bar plot
plt.bar(performL, Student_grade, color ='DarkBlue',
        width = 0.5)

plt.xlabel("Average student Marks")
plt.ylabel("mean weekly alcohol consumption")
plt.title("Marks student got and his mean weekly consumption of alcohol for portuguese s

plt.show()

# Reducing consumption of the alcohol will certainly help student to increase performanc
```

```
For maths dataset
```



```
For portuguese dataset
```

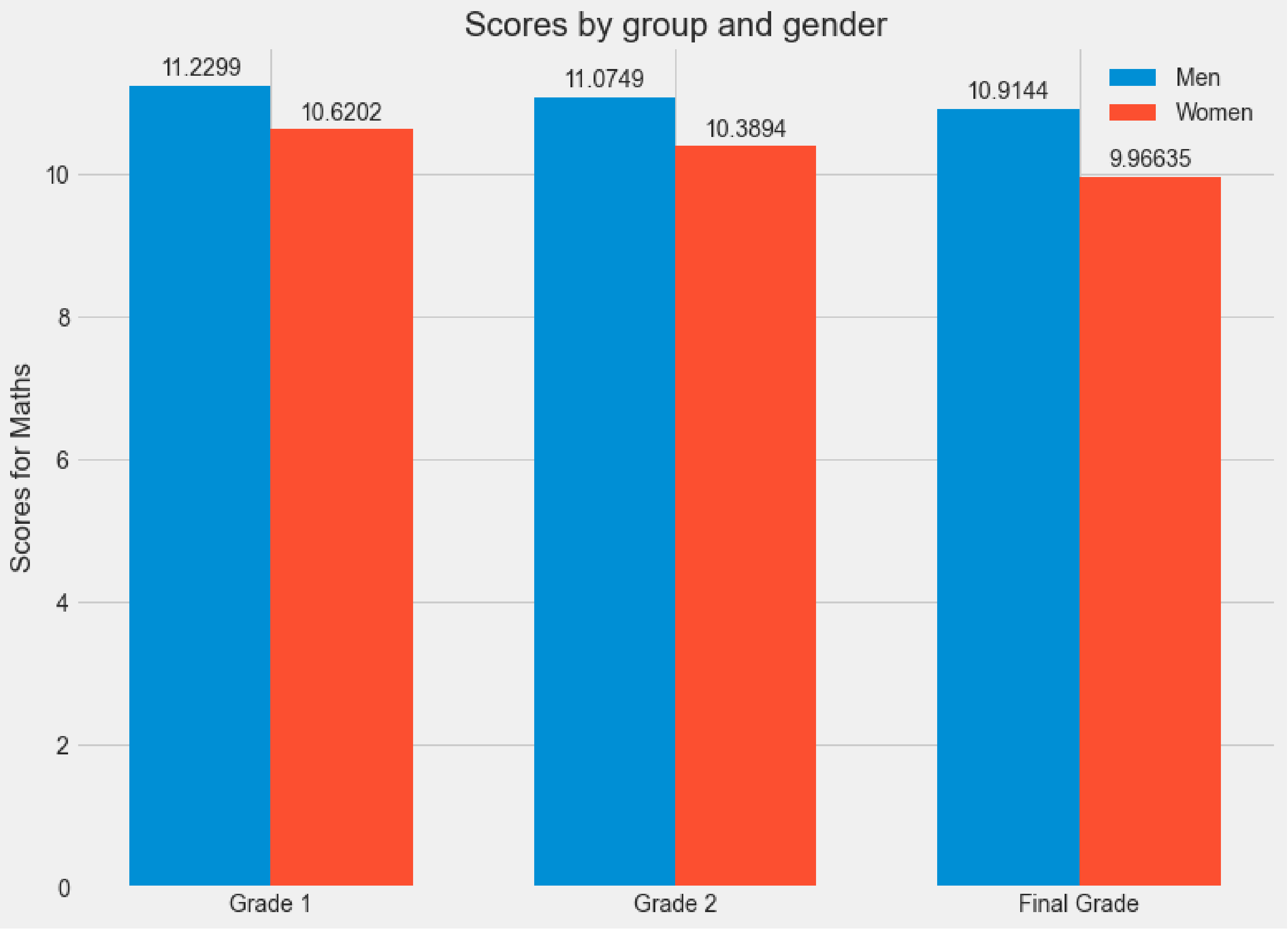


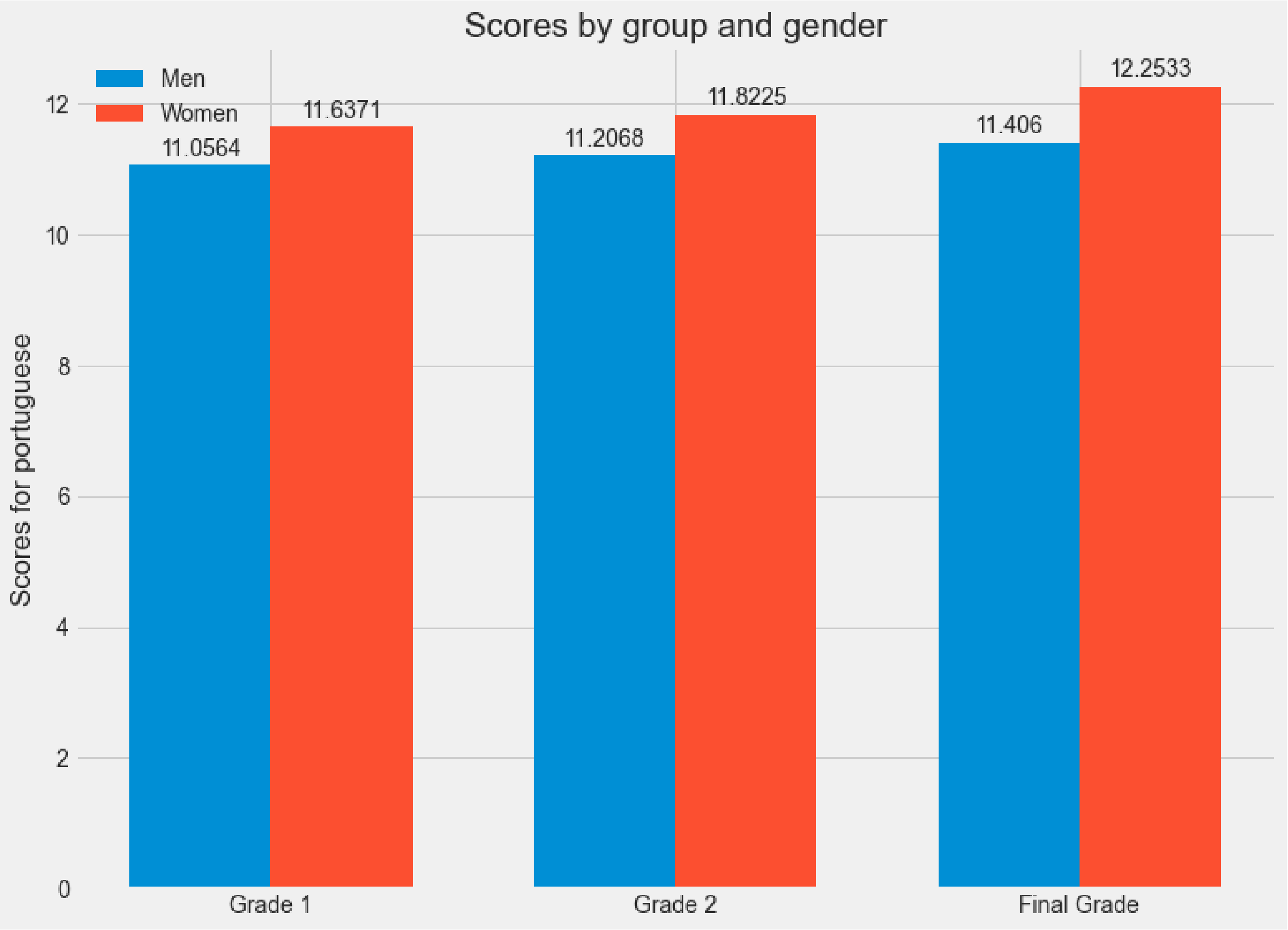
Insights

1. Are boys good in maths and girls in languages ?

In [52]:

```
# Finding for maths subject

def g1(stra):
    return student_M['G1'].loc[(student_M['sex']==stra)].mean()
def g2(stra):
    return student_M['G2'].loc[(student_M['sex']==stra)].mean()
def g3(stra):
    return student_M['G3'].loc[(student_M['sex']==stra)].mean()
N = 3

men_means = (g1('M'), g2('M'), g3('M'))
women_means = (g1('F'), g2('F'), g3('F'))

labels = ['Grade 1', 'Grade 2', 'Final Grade']

x = np.arange(len(labels))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots()
fig.set_size_inches(11, 8)
rects1 = ax.bar(x - width/2, men_means, width, label='Men')
rects2 = ax.bar(x + width/2, women_means, width, label='Women')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Scores for Maths')
ax.set_title('Scores by group and gender')
ax.set_xticks(x, labels)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)

fig.tight_layout()

plt.show()

# Finding for portuguese subject

def g1(stra):
    return student_P['G1'].loc[(student_P['sex']==stra)].mean()
def g2(stra):
    return student_P['G2'].loc[(student_P['sex']==stra)].mean()
def g3(stra):
    return student_P['G3'].loc[(student_P['sex']==stra)].mean()
N = 3

men_means = (g1('M'), g2('M'), g3('M'))
women_means = (g1('F'), g2('F'), g3('F'))

labels = ['Grade 1', 'Grade 2', 'Final Grade']

x = np.arange(len(labels))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots()
fig.set_size_inches(11, 8)
rects1 = ax.bar(x - width/2, men_means, width, label='Men')
rects2 = ax.bar(x + width/2, women_means, width, label='Women')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Scores for portuguese')
ax.set_title('Scores by group and gender')
ax.set_xticks(x, labels)
ax.legend()
ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)

fig.tight_layout()

plt.show()

# Boys seems to perform better in maths while girls in portuguese
```

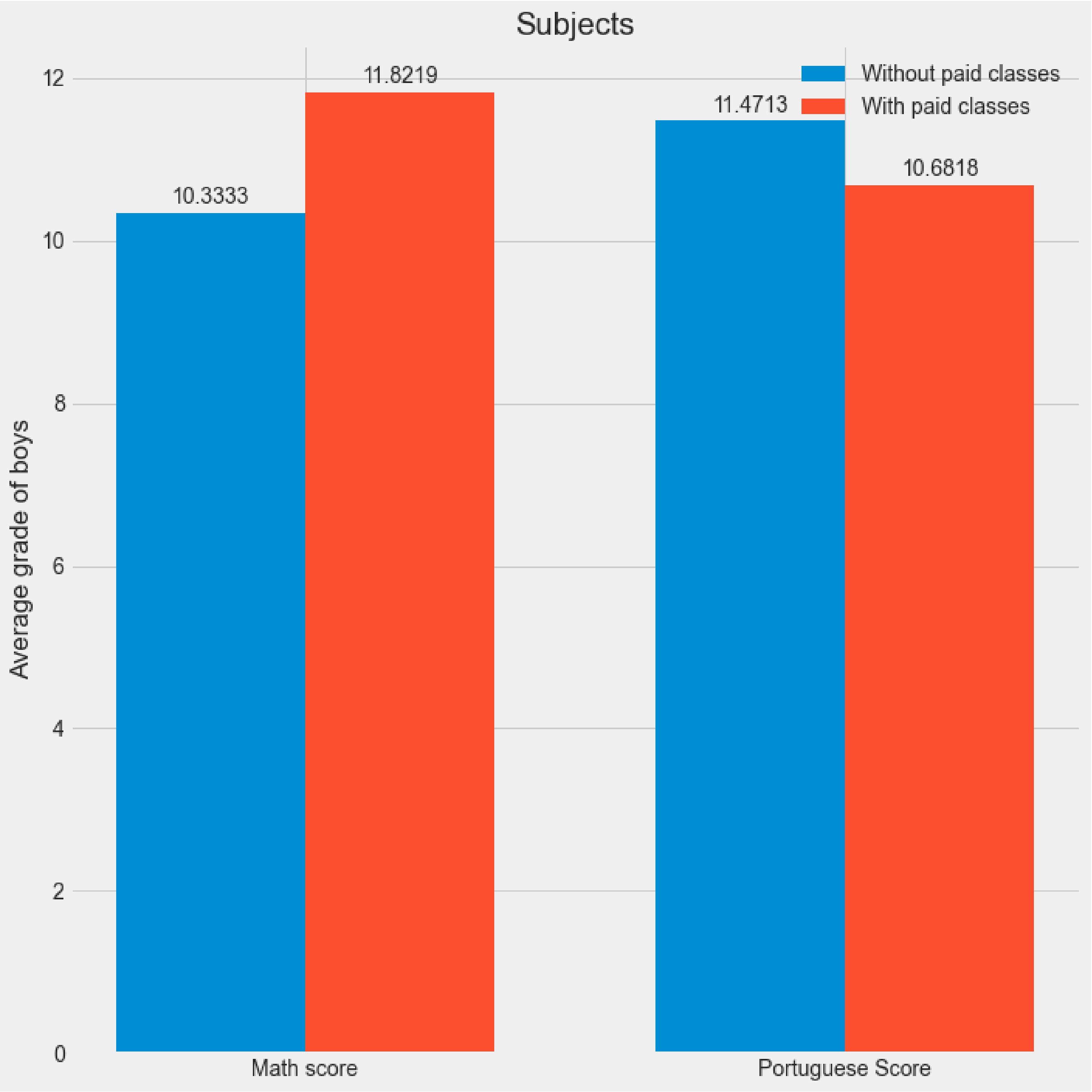


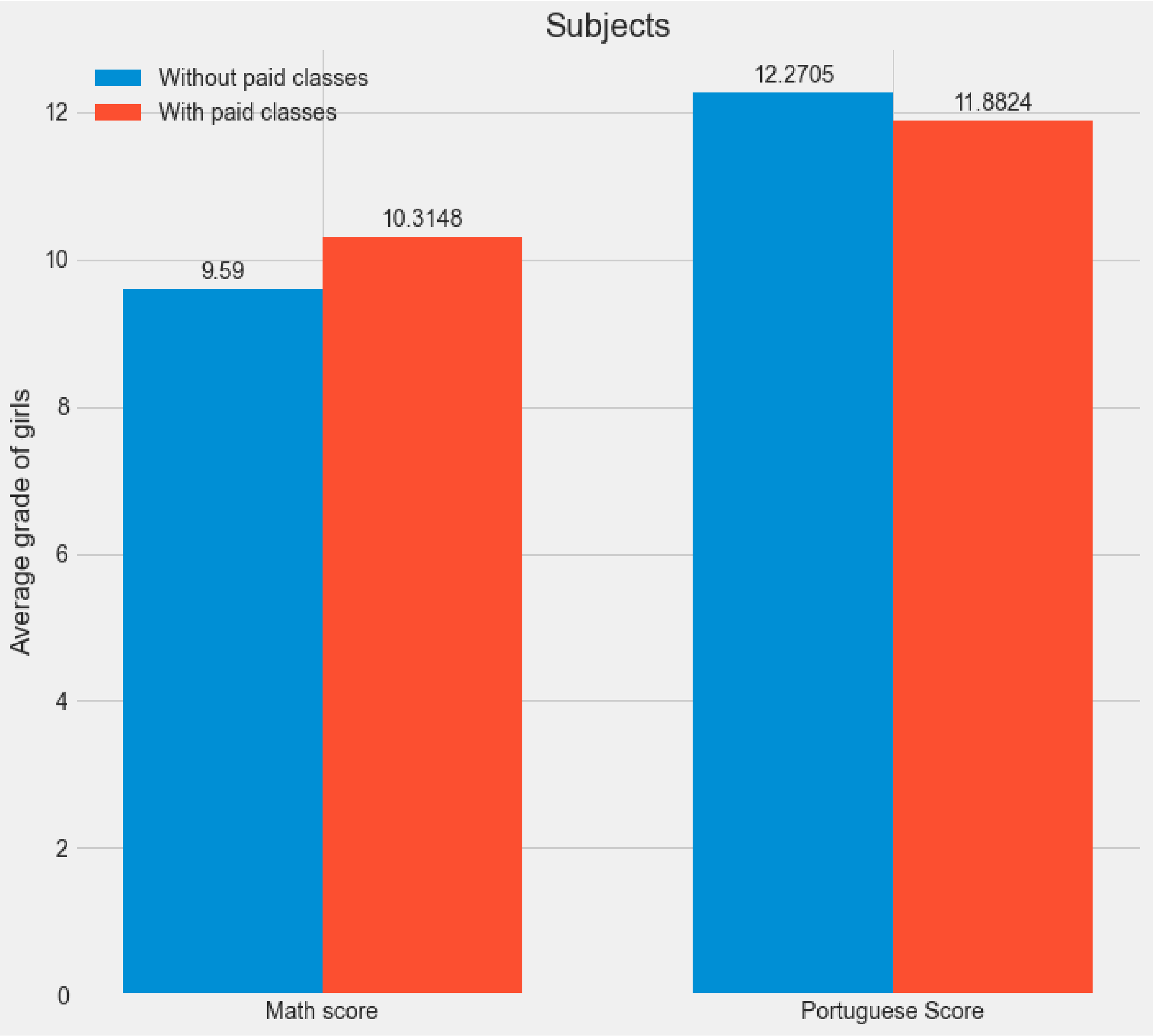


Joining paid classes seems to solve problem for low scores for both boys and girls

In [53]:

```
# Finding for boys

def g_math(stra):
    return student_M['G3'].loc[(student_M['paid']==stra) & (student_M['sex']=='M')].mean

def g_por(stra):
    return student_P['G3'].loc[(student_P['paid']==stra)& (student_P['sex']=='M')].mean(

N = 3

without_paid = (g_math('no'), g_por('no'))
with_paid = (g_math('yes'), g_por('yes'))

labels = ['Math score', 'Portuguese Score']

x = np.arange(len(labels))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots()
fig.set_size_inches(10, 10)
rects1 = ax.bar(x - width/2, without_paid, width, label='Without paid classes')
rects2 = ax.bar(x + width/2, with_paid, width, label='With paid classes')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Average grade of boys')
ax.set_title('Subjects')
ax.set_xticks(x, labels)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)

fig.tight_layout()

plt.show()

# Using internet services seems to improve performance of the student

# Finding for girls

def g_math(stra):
    return student_M['G3'].loc[(student_M['paid']==stra) & (student_M['sex']=='F')].mean

def g_por(stra):
    return student_P['G3'].loc[(student_P['paid']==stra)& (student_P['sex']=='F')].mean(

N = 3

without_paid = (g_math('no'), g_por('no'))
with_paid = (g_math('yes'), g_por('yes'))

labels = ['Math score', 'Portuguese Score']

x = np.arange(len(labels))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots()
fig.set_size_inches(10, 9)
rects1 = ax.bar(x - width/2, without_paid, width, label='Without paid classes')
rects2 = ax.bar(x + width/2, with_paid, width, label='With paid classes')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Average grade of girls')
ax.set_title('Subjects')
ax.set_xticks(x, labels)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)

fig.tight_layout()

plt.show()

# Joining maths paid classes seems to help the student to improve their performnance but
```

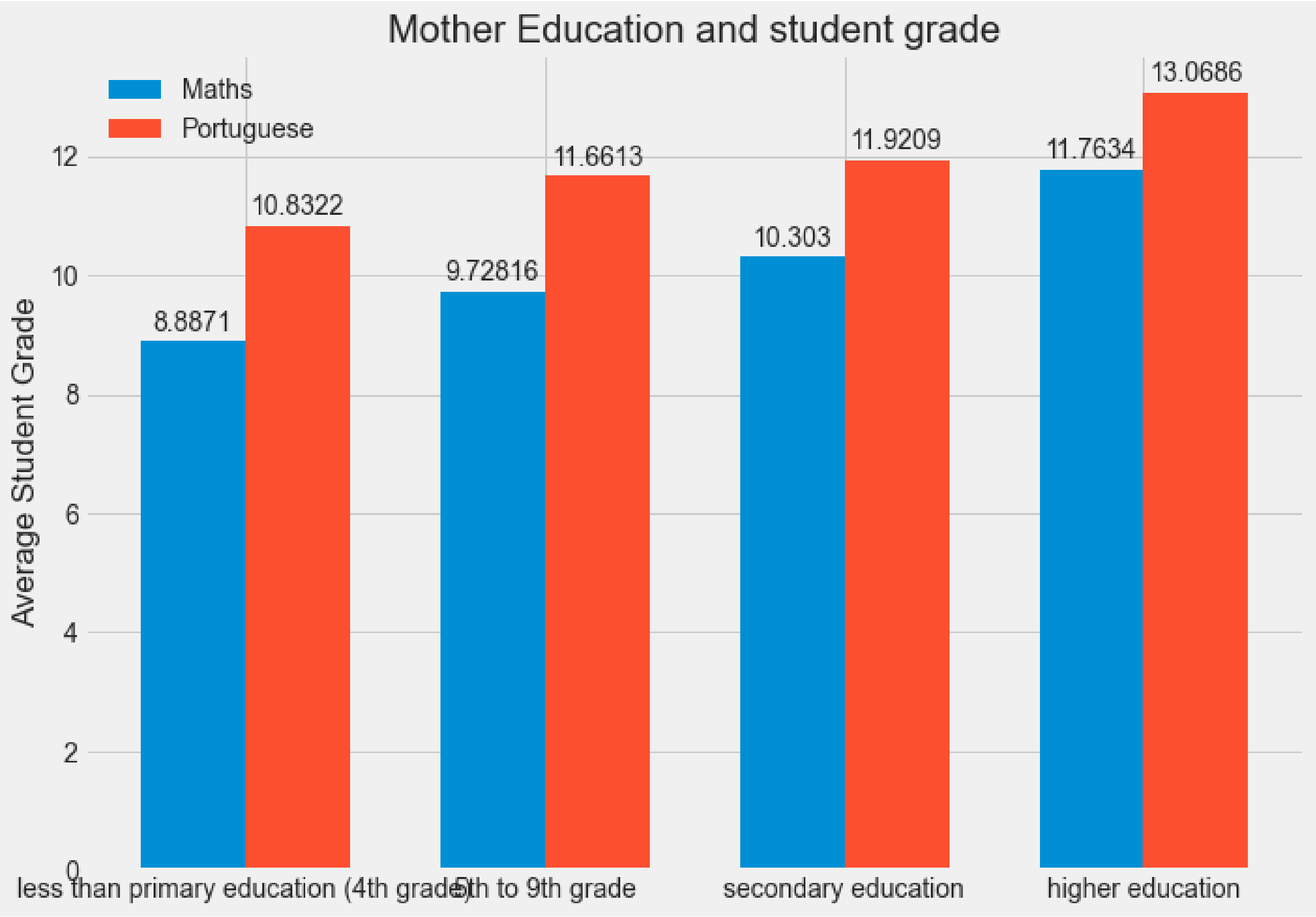


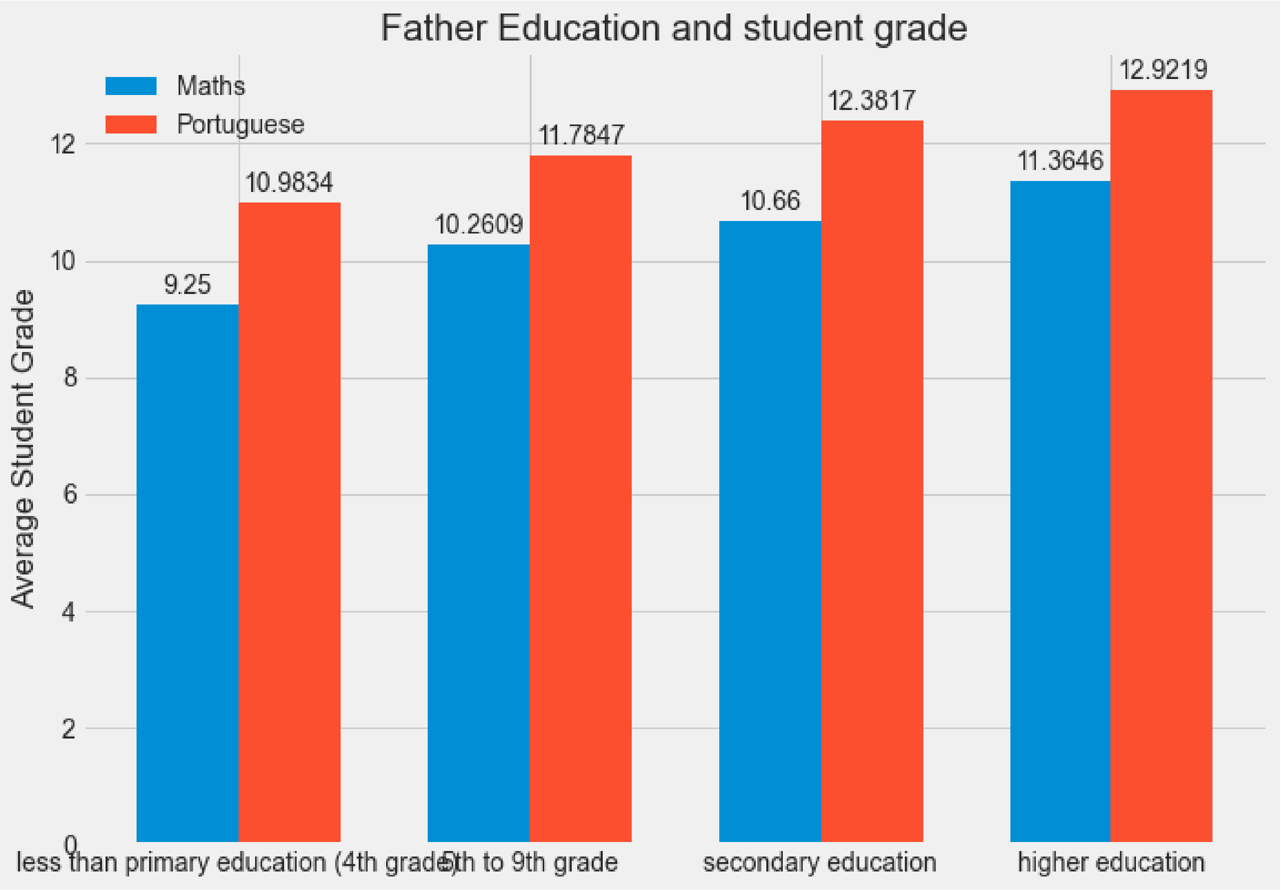


2.Does parents education have any effect on student performance ?

```
In [54]: # Finding for Mother education

         def g_edu(stra):
             return student_M['G3'].loc[(student_M['Medu']<=stra)].mean()
         def g_edu1(stra):
             return student_M['G3'].loc[(student_M['Medu']==stra)].mean()

         def g_edu2(stra):
             return student_P['G3'].loc[(student_P['Medu']<=stra)].mean()
         def g_edu3(stra):
             return student_P['G3'].loc[(student_P['Medu']==stra)].mean()

         N = 3

         math_means = (g_edu(1), g_edu1(2), g_edu1(3),g_edu1(4))
         por_means = (g_edu2(1), g_edu3(2), g_edu3(3),g_edu3(4))

         labels = ['less than primary education (4th grade)', '5th to 9th grade', 'secondary edu

         x = np.arange(len(labels))  # the label locations
         width = 0.35  # the width of the bars

         fig, ax = plt.subplots()
         fig.set_size_inches(10, 7)
         rects1 = ax.bar(x - width/2, math_means, width, label='Maths')
         rects2 = ax.bar(x + width/2, por_means, width, label='Portuguese')

         # Add some text for labels, title and custom x-axis tick labels, etc.
         ax.set_ylabel('Average Student Grade')
         ax.set_title('Mother Education and student grade')
         ax.set_xticks(x, labels)
         ax.legend()

         ax.bar_label(rects1, padding=3)
         ax.bar_label(rects2, padding=3)

         fig.tight_layout()

         plt.show()

         # Finding for father education

         def g_edu(stra):
             return student_M['G3'].loc[(student_M['Fedu']<=stra)].mean()
         def g_edu1(stra):
             return student_M['G3'].loc[(student_M['Fedu']==stra)].mean()

         def g_edu2(stra):
             return student_P['G3'].loc[(student_P['Fedu']<=stra)].mean()
         def g_edu3(stra):
             return student_P['G3'].loc[(student_P['Fedu']==stra)].mean()

         N = 3

         math_means = (g_edu(1), g_edu1(2), g_edu1(3),g_edu1(4))
         por_means = (g_edu2(1), g_edu3(2), g_edu3(3),g_edu3(4))

         labels = ['less than primary education (4th grade)', '5th to 9th grade', 'secondary edu

         x = np.arange(len(labels))  # the label locations
         width = 0.35  # the width of the bars

         fig, ax = plt.subplots()
         fig.set_size_inches(10, 7)
```

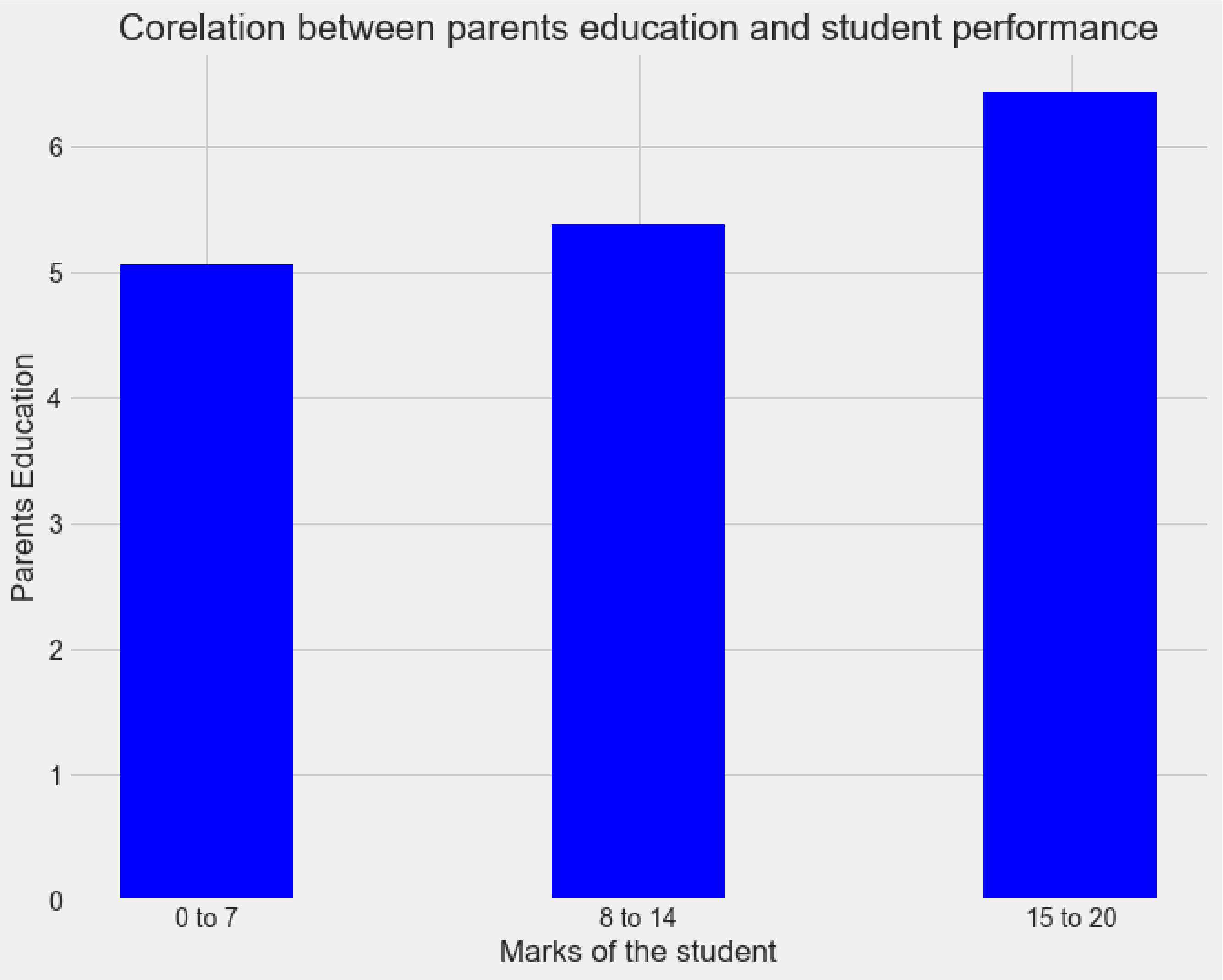
```
rects1 = ax.bar(x - width/2, math_means, width, label='Maths')
rects2 = ax.bar(x + width/2, por_means, width, label='Portuguese')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Average Student Grade')
ax.set_title('Father Education and student grade')
ax.set_xticks(x, labels)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)

fig.tight_layout()

plt.show()
```





Parents education seems to affect the student performance

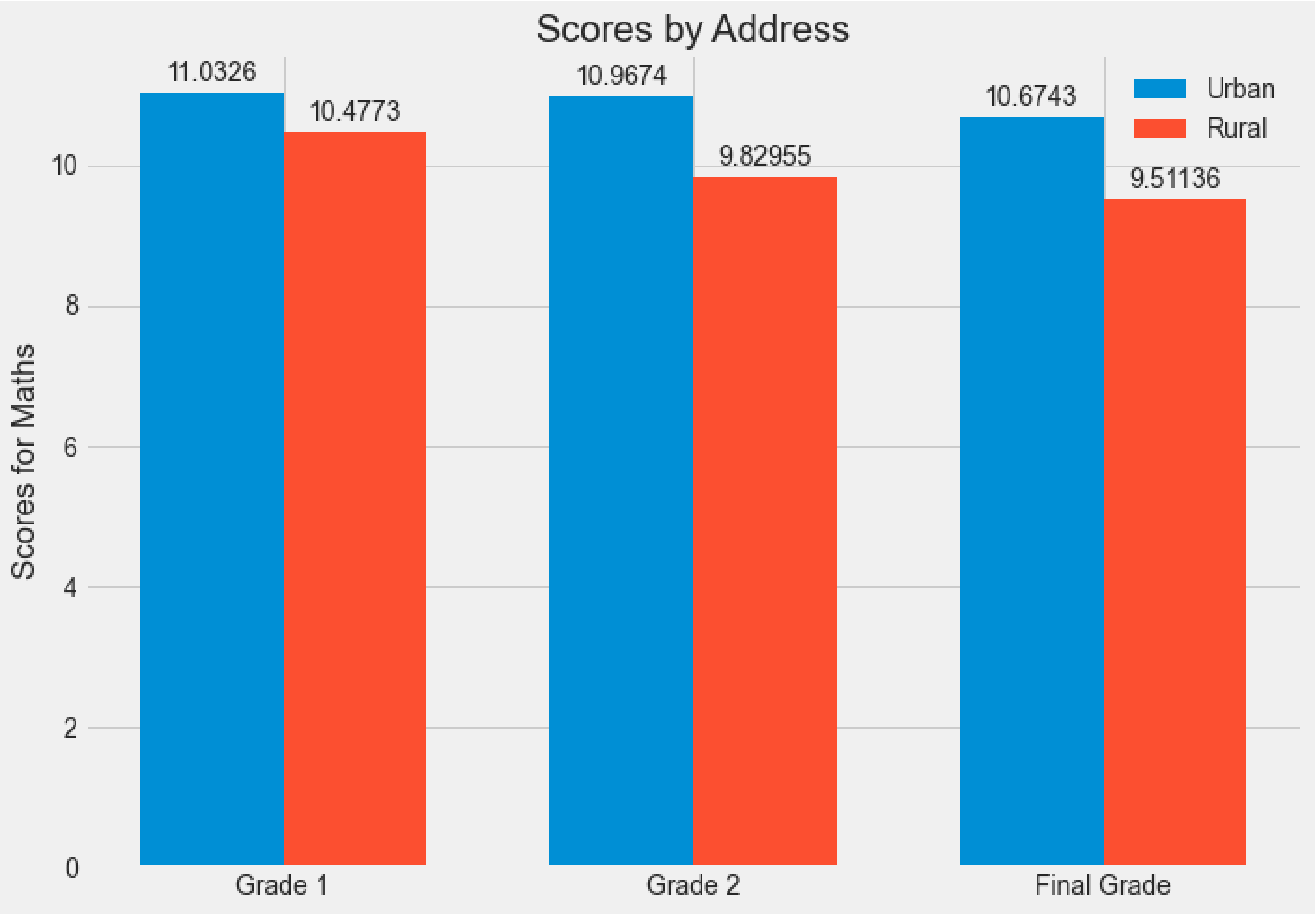
In [26]:
```
student_M['Parents_education'] = student_M['Medu'] + student_M['Medu'] # calculating par

def fjob(stra):
    return student_M['Parents_education'].loc[(student_M['G3']<=stra)].mean()

def fjob1(stra , stra1):
    return student_M['Parents_education'].loc[(student_M['G3']>=stra) & (student_M['mear


Fjob = {'0 to 7':fjob(7), '8 to 14':fjob1(8,14),'15 to 20':fjob1(15,20) }

Father_job = list(Fjob.keys())
Student_grade = list(Fjob.values())

fig = plt.figure(figsize = (10, 8))

# creating the bar plot
plt.bar(Father_job, Student_grade, color ='blue',width = 0.4)

plt.xlabel("Marks of the student")
plt.ylabel("Parents Education")
plt.title("Corelation between parents education and student performance")
plt.show()

# The higher the student scoring chances of his parents is educated is getting higher
```



3. Rural Vs Urban analysis

For maths subject

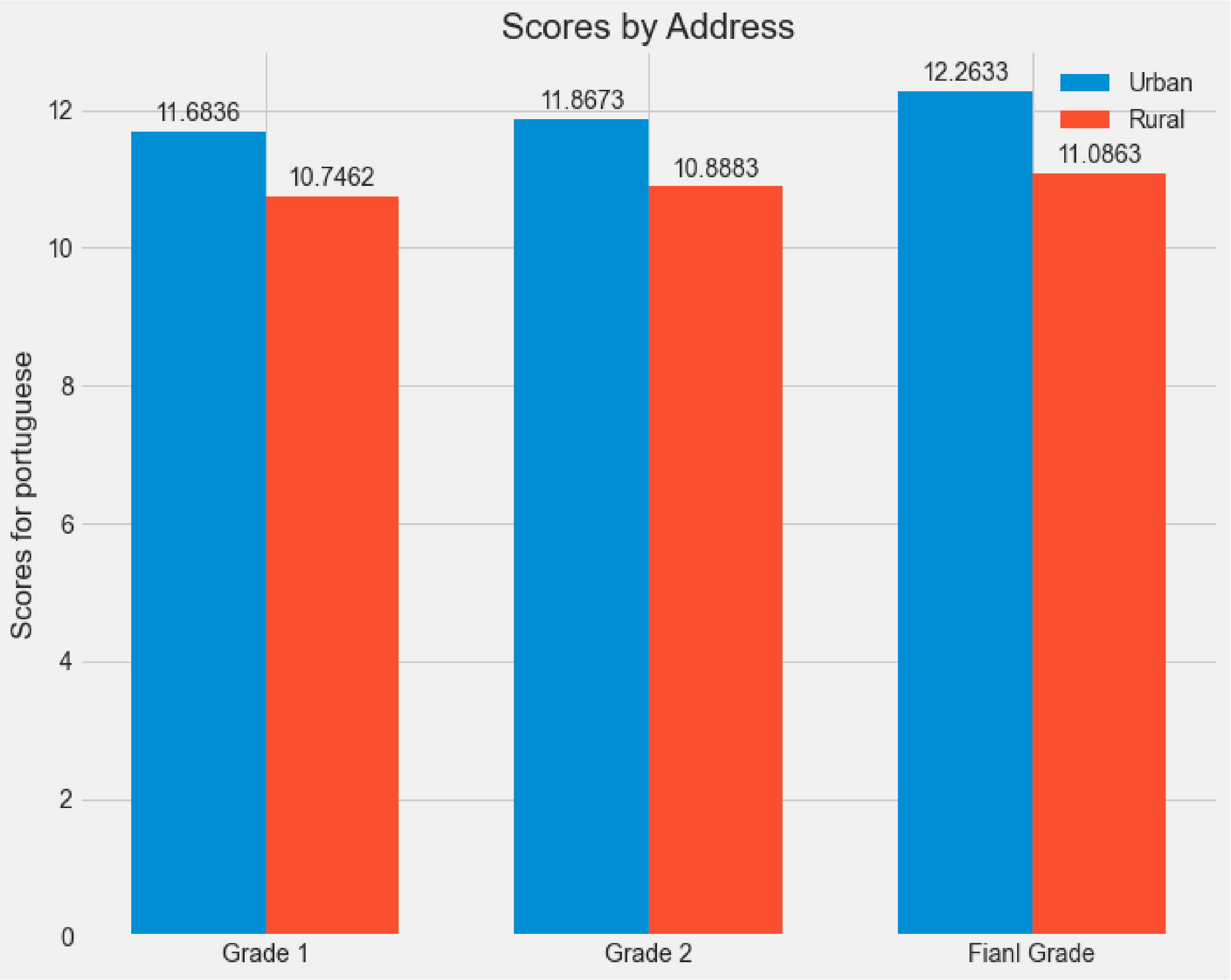
In [55]:
```python
def g1(stra):
    return student_M['G1'].loc[(student_M['address']==stra)].mean()
def g2(stra):
    return student_M['G2'].loc[(student_M['address']==stra)].mean()
def g3(stra):
    return student_M['G3'].loc[(student_M['address']==stra)].mean()
N = 3

men_means = (g1('U'), g2('U'), g3('U'))
women_means = (g1('R'), g2('R'), g3('R'))

labels = ['Grade 1', 'Grade 2', 'Final Grade']

x = np.arange(len(labels))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots()
fig.set_size_inches(10, 7)
rects1 = ax.bar(x - width/2, men_means, width, label='Urban')
rects2 = ax.bar(x + width/2, women_means, width, label='Rural')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Scores for Maths')
ax.set_title('Scores by Address')

ax.set_xticks(x, labels)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)

fig.tight_layout()

plt.show()
```



For portuguese subject

In [56]:

```
def g1(stra):
    return student_P['G1'].loc[(student_P['address']==stra)].mean()
def g2(stra):
    return student_P['G2'].loc[(student_P['address']==stra)].mean()
def g3(stra):
    return student_P['G3'].loc[(student_P['address']==stra)].mean()
N = 3

men_means = (g1('U'), g2('U'), g3('U'))
women_means = (g1('R'), g2('R'), g3('R'))

labels = ['Grade 1', 'Grade 2', 'Fianl Grade']

x = np.arange(len(labels))  # the label locations
width = 0.35  # the width of the bars

fig, ax = plt.subplots()
fig.set_size_inches(10, 8)
rects1 = ax.bar(x - width/2, men_means, width, label='Urban')
rects2 = ax.bar(x + width/2, women_means, width, label='Rural')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Scores for portuguese')
ax.set_title('Scores by Address')
ax.set_xticks(x, labels)
ax.legend()

ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)

fig.tight_layout()

plt.show()

# Student in rural area seems to score lower than student in urban area in both subjects
```


Scores by Address
Urban
Rural
Scores for portuguese
0
2
4
6
8
10
12
11.6836
10.7462
11.8673
10.8883
12.2633
11.0863
Grade 1
Grade 2
Fianl Grade