Forward
School

Program Code: J620-002-4:2020

Program Name: FRONT-END SOFTWARE DEVELOPMENT

Title : Case Study - Data Analysis of Student Performance

Name: Chong Mun Chen

IC Number: 960327-07-5097

Date : 7/7/2023

Introduction : Practising more with Pandas DataFrame and Matplotlib.

Conclusion : I am getting a lot better at constructing DataFrames and plotting graphs with Matplotlib with this exercise.

Guideline EDA link: https://medium.com/dataseries/an-eda-checklist-800beeaee555 (https://medium.com/dataseries/an-eda-checklist-800beeaee555)

Sample Exercise:

High Student students academic performance

I'll do the dataset in Excel

Randomizers in Excel (dont shoot me)

I like to "visualize my simulated data"

=RANDBETWEEN(0,100)

=CHOOSE(RANDBETWEEN(1,3),"B40","M40","T20")

What data is needed?

Describe the data

```
Student demography
Subjects taken
Trial exam results
attendance, contact Hours
Final results
Others? Sports activities
```

"Correlation is not causation"

Means that just because two things correlate does not necessarily mean that one causes the other

Case Study Exercise

Plot the Student Results table

Some basic stats

Look for Average, Min, Max

Exploratory Data Analysis (EDA) Check list

- Domain knowledge
  - What is this dataset about?
- Check if the data is intuitive
- Find out how the data was generated
- Understand the process

- Select a smaller dataset
  - depending on the data size, If what to go big bang, make sure enough resources.
- Explore individual features
- Explore pairs and groups

- Clean up features
- Selecting features of interest
- Generating derived feature(s)
- Extract , Transform and Load (the whole dataset)
- Sampling the data (in ML)

In [2]:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

1. Import Data from CSV

In [143]:

```python
df = pd.read_csv('student_performance.csv')
df
```

Out[143]:

	Student ID	Name	Term	IncomeGroup	NonsenseData	School	Tuisyen	Attendance	BM	BI
0	7	Psy	2	B40	xvxc	SK 8estari	No	60	24	NaN
1	8	Edward	2	M40	sf	SK 8estari	Yes	30	43	28.0
2	6	Mei Lin	2	M40	dsf	SK 8estari	Yes	78	0	20.0
3	9	Miyazawa	2	T20	df	SK 8estari	No	100	32	94.0
4	4	Letchumi	2	T20	xvxc	SK 8estari	No	80	97	52.0
5	3	Muthu	2	T20	sf	SK 8estari	Yes	58	31	65.0
6	5	Ah Chong	2	B40	dsf	SK 8estari	Yes	64	16	84.0
7	2	Siti	2	M40	df	SK 8estari	Yes	57	35	68.0
8	1	Ali	2	B40	xvxc	SK 8estari	No	100	16	89.0
9	10	Ah Beng	2	T20	sf	SK 8estari	No	100	43	100.0
10	7	Psy	1	B40	dsf	SK 8estari	No	60	14	-10.0
11	8	Edward	1	M40	df	SK 8estari	Yes	30	33	18.0
12	6	Mei Lin	1	M40	xvxc	SK 8estari	Yes	78	10	10.0
13	9	Miyazawa	1	T20	sf	SK 8estari	No	100	22	84.0
14	4	Letchumi	1	T20	dsf	SK 8estari	No	80	87	42.0
15	3	Muthu	1	T20	df	SK 8estari	Yes	58	21	55.0
16	5	Ah Chong	1	B40	xvxc	SK 8estari	Yes	64	6	74.0
17	2	Siti	1	M40	sf	SK 8estari	Yes	57	25	58.0
18	1	Ali	1	B40	dsf	SK 8estari	No	100	6	79.0
19	10	Ah Beng	1	T20	df	SK 8estari	No	100	33	90.0

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 13 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Student ID    20 non-null     int64  
 1   Name          20 non-null     object 
 2   Term          20 non-null     int64  
 3   IncomeGroup   20 non-null     object 
 4   NonsenseData  20 non-null     object 
 5   School        20 non-null     object 
 6   Tuisyen       20 non-null     object 
 7   Attendance    20 non-null     int64  
 8   BM            20 non-null     int64  
 9   BI            19 non-null     float64
 10  Maths         20 non-null     int64  
 11  Sejarah       20 non-null     int64  
 12  Total         20 non-null     int64  
dtypes: float64(1), int64(7), object(5)
memory usage: 2.2+ KB
```

In [144]:

```
df.isnull().sum()
```

Out[144]:

```
Student ID      0
Name            0
Term            0
IncomeGroup     0
NonsenseData    0
School          0
Tuisyen         0
Attendance      0
BM              0
BI              1
Maths           0
Sejarah         0
Total           0
dtype: int64
```

2. Data Cleaning - Remove Useless Data

In [289]:

```
new_df = df
new_df = new_df.drop(columns=['NonsenseData', 'School'])
new_df = new_df.rename(columns={'Student ID':'StudentID'})
new_df = new_df.set_index('StudentID').sort_values('StudentID')
new_df
```

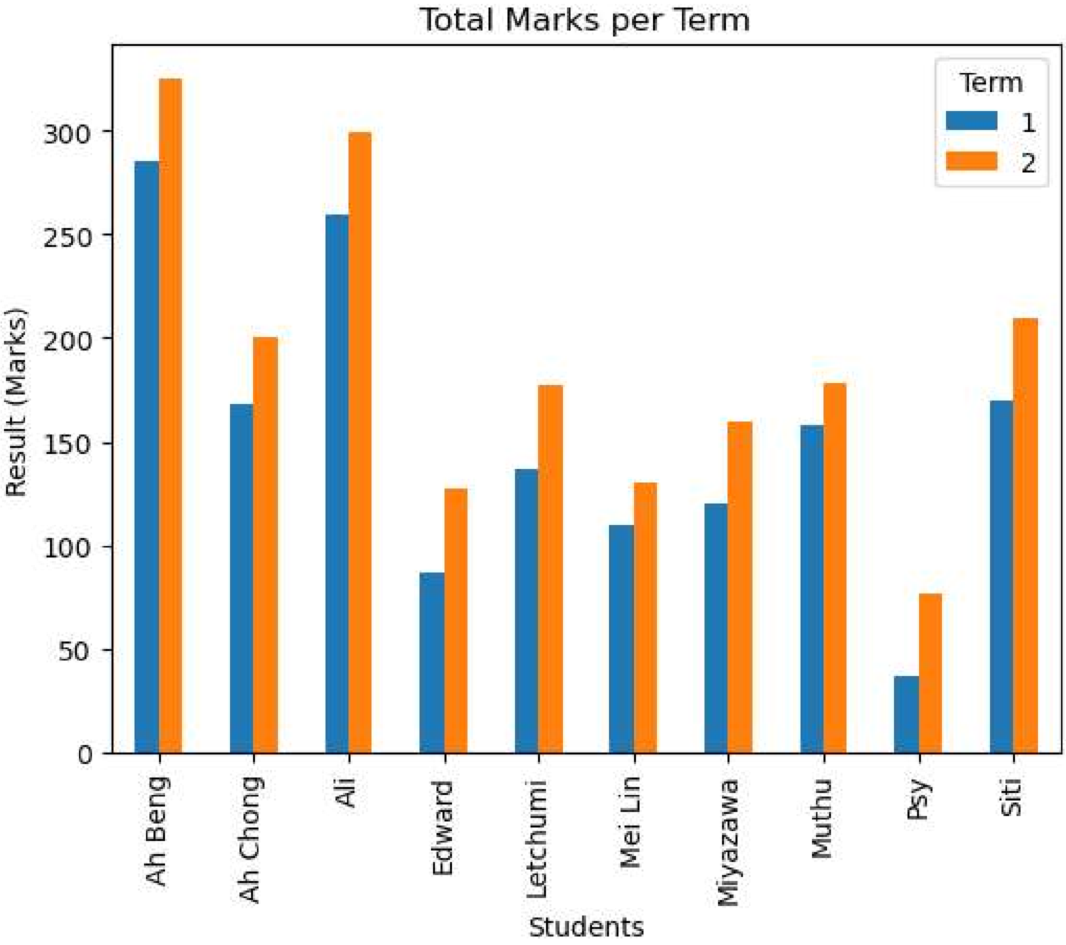
Out[289]:

	Name	Term	IncomeGroup	Tuisyen	Attendance	BM	BI	Maths	Sejarah	Total
StudentID										
1	Ali	2	B40	No	100	16	89.0	97	97	299
1	Ali	1	B40	No	100	6	79.0	87	87	259
2	Siti	2	M40	Yes	57	35	68.0	35	72	210
2	Siti	1	M40	Yes	57	25	58.0	25	62	170
3	Muthu	2	T20	Yes	58	31	65.0	0	82	178
3	Muthu	1	T20	Yes	58	21	55.0	10	72	158
4	Letchumi	2	T20	No	80	97	52.0	17	11	177
4	Letchumi	1	T20	No	80	87	42.0	7	1	137
5	Ah Chong	2	B40	Yes	64	16	84.0	6	94	200
5	Ah Chong	1	B40	Yes	64	6	74.0	4	84	168
6	Mei Lin	2	M40	Yes	78	0	20.0	47	63	130
6	Mei Lin	1	M40	Yes	78	10	10.0	37	53	110
7	Psy	1	B40	No	60	14	-10.0	27	6	37
7	Psy	2	B40	No	60	24	NaN	37	16	77
8	Edward	2	M40	Yes	30	43	28.0	22	34	127
8	Edward	1	M40	Yes	30	33	18.0	12	24	87
9	Miyazawa	1	T20	No	100	22	84.0	23	-9	120
9	Miyazawa	2	T20	No	100	32	94.0	33	1	160
10	Ah Beng	2	T20	No	100	43	100.0	90	92	325
10	Ah Beng	1	T20	No	100	33	90.0	80	82	285

In [310]:

```
# Method 1
# new_df = new_df.set_index('Name').sort_values('Name')
# term_one = new_df[new_df['Term'] == 1]
# term_two = new_df[new_df['Term'] == 2]
# new_df = pd.DataFrame({'Term 1': term_one['Total'], 'Term 2': term_two['Total'],
# ax = new_df.plot.bar(rot=45)

# Method 2
grouped_df = new_df[['Name', 'Term', 'Total']]
pivot_df = grouped_df.pivot(index='Name', columns='Term', values='Total')
pivot_df.plot.bar()
plt.title('Total Marks per Term')
plt.xlabel('Students')
plt.ylabel('Result (Marks)')
plt.show()
```



3. Basic Statistics of Table

In [216]:

```
new_df.describe()
```

Out[216]:

	Term	Attendance	BM	BI	Maths	Sejarah	Total
count	20.000000	20.000000	20.000000	19.000000	20.000000	20.000000	20.000000
mean	1.500000	72.700000	29.700000	57.894737	34.800000	51.200000	170.700000
std	0.512989	22.571757	24.525175	31.985925	30.365493	36.685864	75.577844
min	1.000000	30.000000	0.000000	-10.000000	0.000000	-9.000000	37.000000
25%	1.000000	58.000000	15.500000	35.000000	11.500000	14.750000	125.250000
50%	1.500000	71.000000	24.500000	65.000000	26.000000	62.500000	164.000000
75%	2.000000	100.000000	33.500000	84.000000	39.500000	82.500000	202.500000
max	2.000000	100.000000	97.000000	100.000000	97.000000	97.000000	325.000000

The top 3 and last 3 students each term

In [312]:

```
student_placing = new_df[['Name', 'Total', 'Term']]
term_one_top = student_placing[student_placing['Term'] == 1].sort_values(by=['Tot
term_one_last = student_placing[student_placing['Term'] == 1].sort_values(by=['To
term_two_top = student_placing[student_placing['Term'] == 2].sort_values(by=['Tot
term_two_last = student_placing[student_placing['Term'] == 2].sort_values(by=['To

print('Top 3 students for Term 1')
print(term_one_top.head(3))
print()

print('Last 3 students for Term 1')
print(term_one_last.tail(3))
print()

print('Top 3 students for Term 2')
print(term_two_top.head(3))
print()

print('Last 3 students for Term 2')
print(term_two_last.tail(3))
print()
```

```
Top 3 students for Term 1
              Name  Total  Term
StudentID
10         Ah Beng    285     1
1              Ali    259     1
2             Siti    170     1

Last 3 students for Term 1
              Name  Total  Term
StudentID
6          Mei Lin    110     1
8           Edward     87     1
7              Psy     37     1

Top 3 students for Term 2
              Name  Total  Term
StudentID
10         Ah Beng    325     2
1              Ali    299     2
2             Siti    210     2

Last 3 students for Term 2
              Name  Total  Term
StudentID
6          Mei Lin    130     2
8           Edward    127     2
7              Psy     77     2
```

Average Scores for each term

In [154]:

```
# Average for each subject and total scores
new_df.groupby(['Term'])[['BI', 'BM', 'Maths', 'Sejarah', 'Total']].mean()
```

Out[154]:

	BI	BM	Maths	Sejarah	Total
Term					
1	50.000000	25.7	31.2	46.2	153.1
2	66.666667	33.7	38.4	56.2	188.3

Max score for each subject

In [313]:

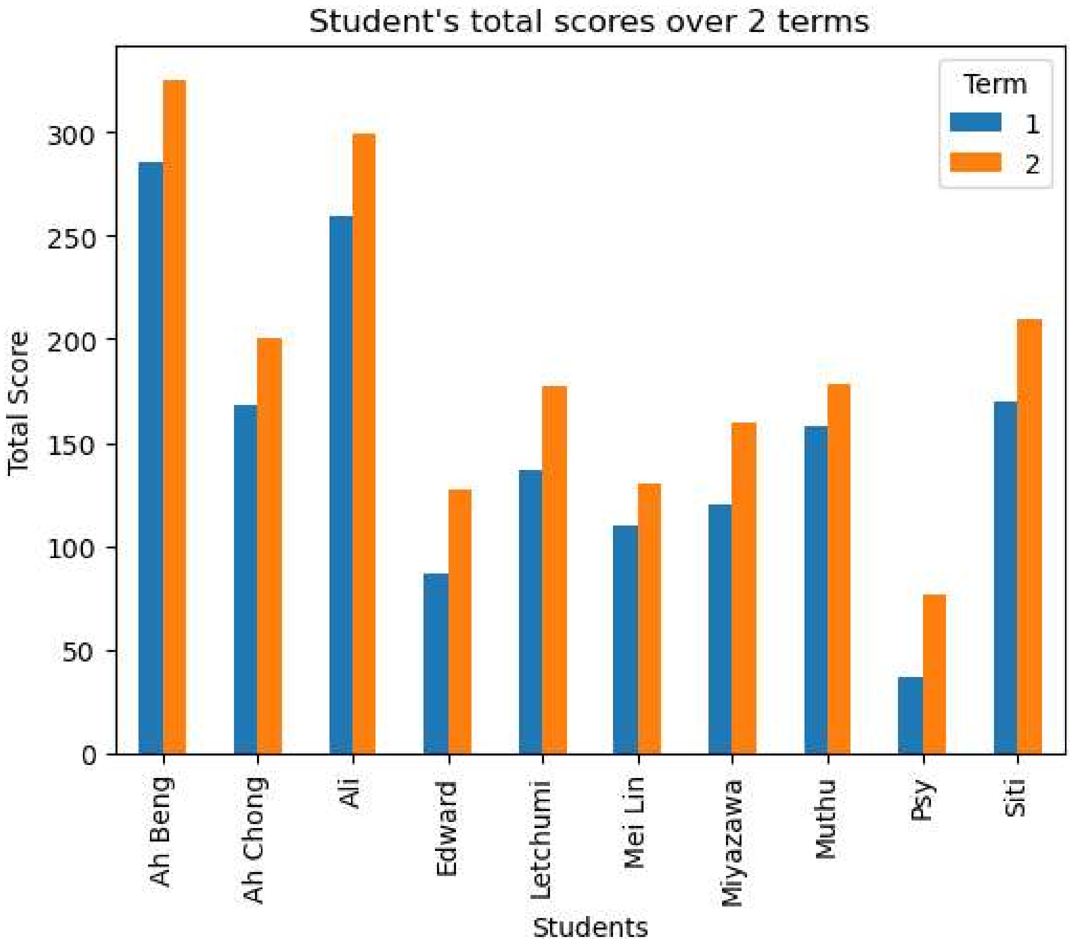
```
new_df[['BI', 'BM', 'Maths', 'Sejarah']].max()
```

Out[313]:

```
BI         100.0
BM          97.0
Maths       97.0
Sejarah     97.0
dtype: float64
```

In [311]:

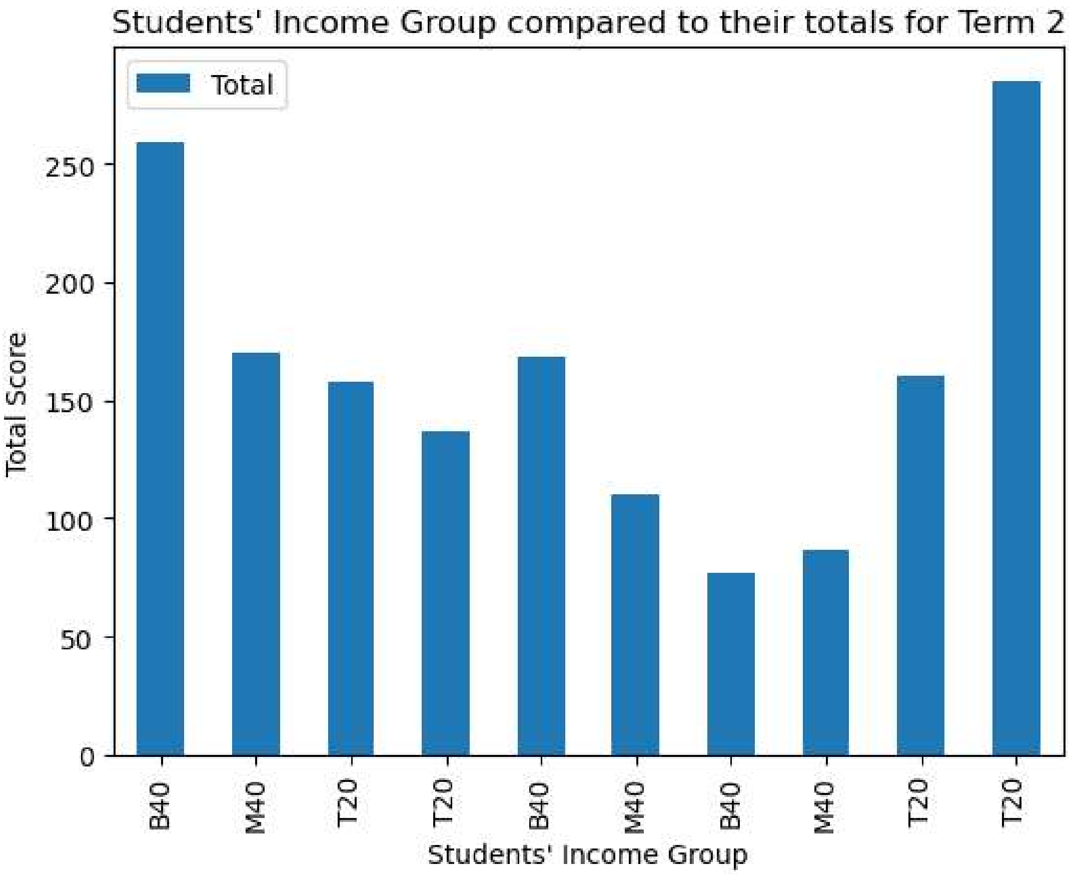
```
grouped_df = new_df[['Name', 'Term', 'Total']]
pivot_df = grouped_df.pivot(index='Name', columns='Term', values='Total')
pivot_df.plot.bar()
plt.title("Student's total scores over 2 terms")
plt.xlabel('Students')
plt.ylabel('Total Score')
plt.show()
```



Type Markdown and LaTeX: $\alpha 2$

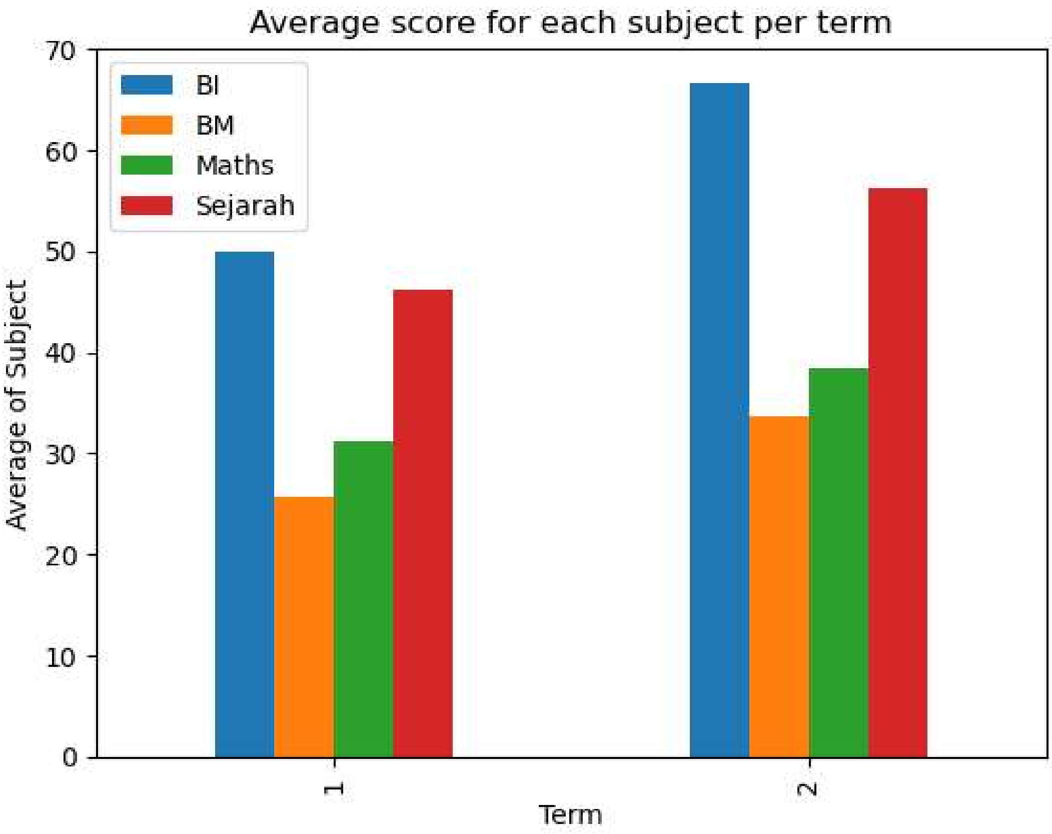
In [315]:

```
income_group_df = new_df.drop_duplicates(subset=['Name'], keep='last')
income_group_df.plot.bar(x='IncomeGroup', y='Total')
plt.title("Students' Income Group compared to their totals for Term 2")
plt.xlabel("Students' Income Group")
plt.ylabel('Total Score')
plt.show()
```



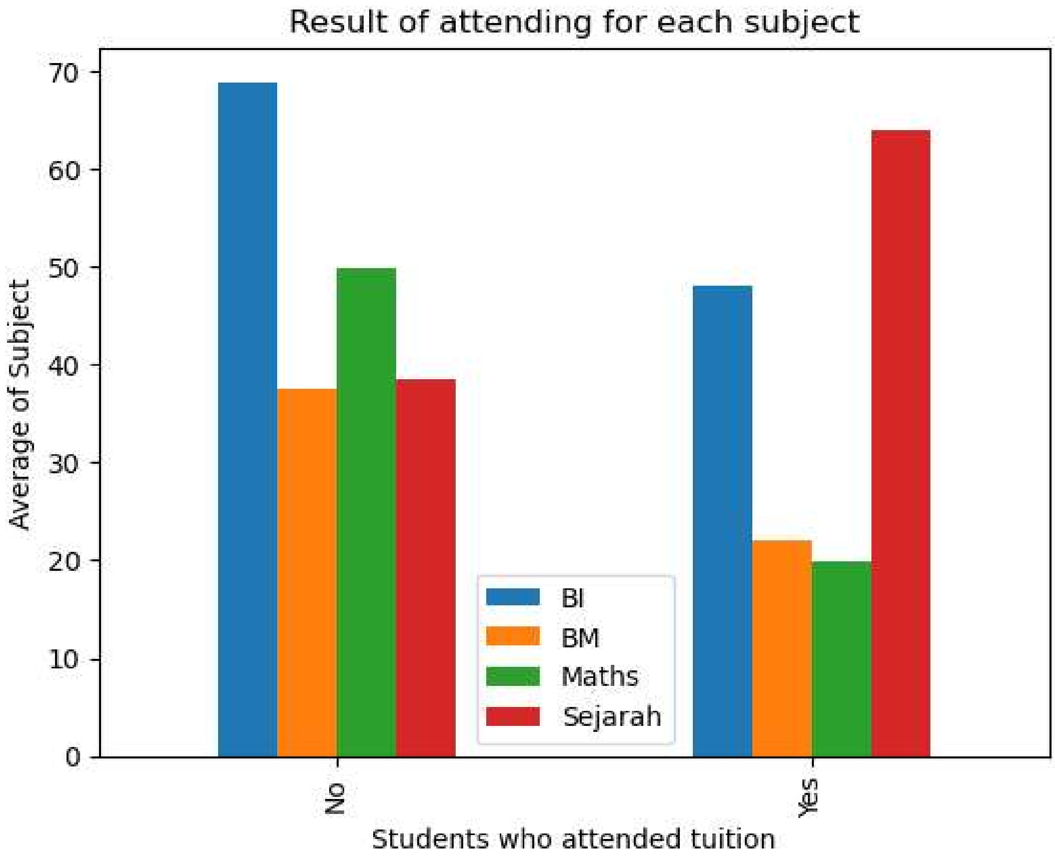
In [316]:

```python
term_df = new_df.groupby(['Term'])[['BI','BM','Maths','Sejarah']].mean()
term_df.plot.bar()
plt.title("Average score for each subject per term")
plt.xlabel('Term')
plt.ylabel('Average of Subject')
plt.show()
```



In [317]:

```
tuisyen_df = new_df.groupby(['Tuisyen'])[['BI','BM','Maths','Sejarah']].mean()
tuisyen_df.plot.bar()
plt.title("Result of attending for each subject")
plt.xlabel('Students who attended tuition')
plt.ylabel('Average of Subject')
plt.show()
```



4. Replace IncomeGroup and Tuisyen to Numerical Value

In [318]:

```
new_df.loc[new_df['IncomeGroup'] == 'B40', 'IncomeGroup'] = 1
new_df.loc[new_df['IncomeGroup'] == 'M40', 'IncomeGroup'] = 2
new_df.loc[new_df['IncomeGroup'] == 'T20', 'IncomeGroup'] = 3

new_df.loc[new_df['Tuisyen'] == 'No', 'Tuisyen'] = 1
new_df.loc[new_df['Tuisyen'] == 'Yes', 'Tuisyen'] = 0

new_df
```

Out[318]:

	Name	Term	IncomeGroup	Tuisyen	Attendance	BM	BI	Maths	Sejarah	Total
StudentID										
1	Ali	2	1	1	100	16	89.0	97	97	299
1	Ali	1	1	1	100	6	79.0	87	87	259
2	Siti	2	2	0	57	35	68.0	35	72	210
2	Siti	1	2	0	57	25	58.0	25	62	170
3	Muthu	2	3	0	58	31	65.0	0	82	178
3	Muthu	1	3	0	58	21	55.0	10	72	158
4	Letchumi	2	3	1	80	97	52.0	17	11	177
4	Letchumi	1	3	1	80	87	42.0	7	1	137
5	Ah Chong	2	1	0	64	16	84.0	6	94	200
5	Ah Chong	1	1	0	64	6	74.0	4	84	168
6	Mei Lin	2	2	0	78	0	20.0	47	63	130
6	Mei Lin	1	2	0	78	10	10.0	37	53	110
7	Psy	1	1	1	60	14	-10.0	27	6	37
7	Psy	2	1	1	60	24	NaN	37	16	77
8	Edward	2	2	0	30	43	28.0	22	34	127
8	Edward	1	2	0	30	33	18.0	12	24	87
9	Miyazawa	1	3	1	100	22	84.0	23	-9	120
9	Miyazawa	2	3	1	100	32	94.0	33	1	160
10	Ah Beng	2	3	1	100	43	100.0	90	92	325
10	Ah Beng	1	3	1	100	33	90.0	80	82	285

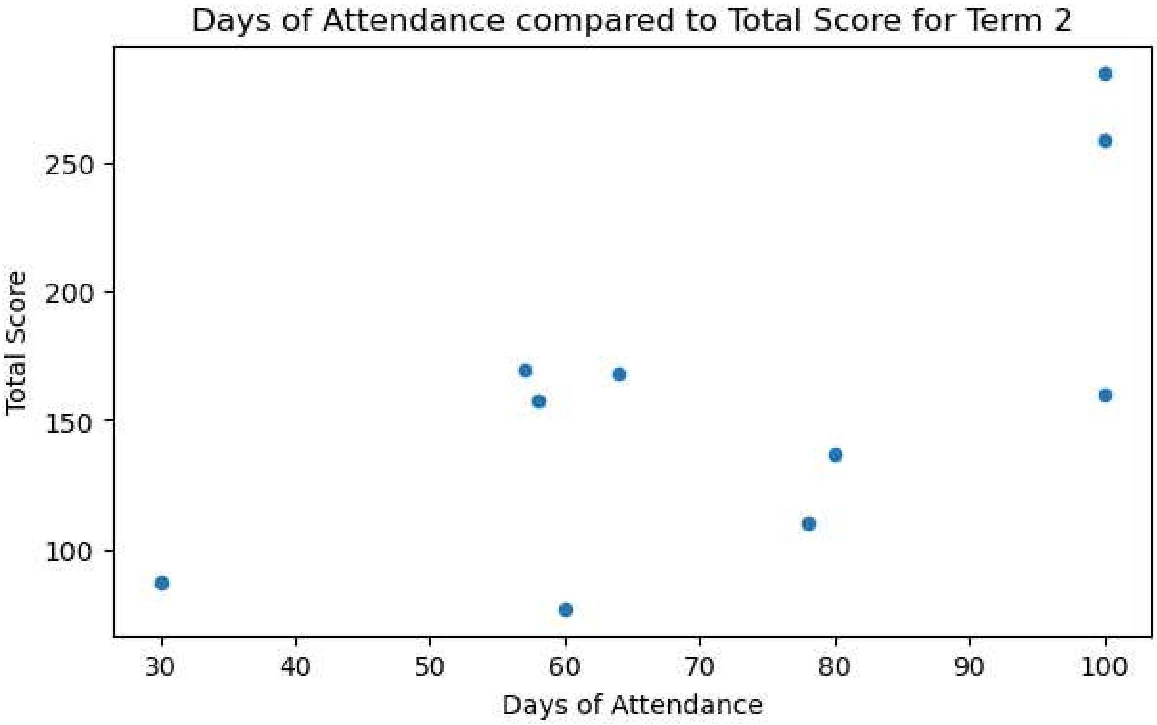
5. Check the correlation between income group, tuisyen and result

In [319]:
```
new_df[['IncomeGroup', 'Tuisyen', 'BM', 'BI', 'Maths', 'Sejarah', 'Total']].corr(
```

Out[319]:

	IncomeGroup	Tuisyen	BM	BI	Maths	Sejarah	Total
IncomeGroup	1.000000	0.120386	0.565563	0.192207	-0.137483	-0.259916	0.129596
Tuisyen	0.120386	1.000000	0.322119	0.335015	0.506815	-0.357972	0.229420
BM	0.565563	0.322119	1.000000	0.018501	-0.219737	-0.427311	0.045778
BI	0.192207	0.335015	0.018501	1.000000	0.379601	0.410982	0.799557
Maths	-0.137483	0.506815	-0.219737	0.379601	1.000000	0.406545	0.681237
Sejarah	-0.259916	-0.357972	-0.427311	0.410982	0.406545	1.000000	0.713688
Total	0.129596	0.229420	0.045778	0.799557	0.681237	0.713688	1.000000

In [321]:
```
filtered_df = new_df.drop_duplicates(subset=['Name'], keep='last')
filtered_df.plot(x='Attendance', y='Total', kind='scatter', figsize=(7,4))
plt.title("Days of Attendance compared to Total Score for Term 2")
plt.xlabel('Days of Attendance')
plt.ylabel('Total Score')
plt.show()
```



6. Conclusion

What is your finding?

I find that students who attended Tuition has performed better overall than the ones who did not attend. Besides that, every student has improved in Term 2 when comparing Term 1 results and Term 2 results.