Mini-Project on Big Data (MPBD)

Git:
https://github.com/wingyeung0317/EEE4463/blob/master/BD_Labs_V3/Lab6/YeungWing_MPBD.

...

Requirement 1

Consolidate 5 CSV files into a single CSV file

In [ ]:
```
import pandas as pd
```

In [ ]:
```
df1 = pd.read_csv("S1.csv").astype(float, errors='ignore')
df2 = pd.read_csv("S2.csv").astype(float, errors='ignore')
df3 = pd.read_csv("S3.csv").astype(float, errors='ignore')
df4 = pd.read_csv("S4.csv").astype(float, errors='ignore')
df5 = pd.read_csv("S5.csv").astype(float, errors='ignore')
```

In [ ]:
```
# if the students completed all 5 semesters, then their ID should be in every csv f
all5SemDF = pd.merge(df1, df2, on='Student', how='inner')
all5SemDF = pd.merge(all5SemDF, df3, on='Student', how='inner')
all5SemDF = pd.merge(all5SemDF, df4, on='Student', how='inner')
all5SemDF = pd.merge(all5SemDF, df5, on='Student', how='inner')

# drop rows with NaN values
all5SemDF_noNaN = all5SemDF.dropna(how='any')
all5SemDF_noNaN.reset_index(drop=True, inplace=True)
```

In [ ]:
```
all5SemDF_noNaN
```

Out[ ]:

	Student	L001	V001	V002	V003	V004	V005	V006	V007	L002	...	L004	V019	V020
0	S001	56.0	82.0	74.0	68.0	78.0	87.0	63.0	79.0	54.0	...	63.0	76.0	70.0
1	S003	52.0	83.0	88.0	79.0	85.0	66.0	62.0	74.0	56.0	...	67.0	74.0	69.0
2	S005	56.0	74.0	54.0	49.0	53.0	60.0	54.0	73.0	43.0	...	36.0	58.0	60.0
3	S007	61.0	85.0	76.0	79.0	79.0	64.0	66.0	82.0	66.0	...	46.0	11.0	62.0
4	S008	58.0	72.0	69.0	82.0	76.0	77.0	68.0	74.0	61.0	...	67.0	79.0	69.0
5	S010	59.0	56.0	42.0	60.0	33.0	65.0	60.0	67.0	50.0	...	50.0	56.0	61.0
6	S011	53.0	69.0	68.0	65.0	49.0	73.0	62.0	68.0	48.0	...	58.0	68.0	66.0
7	S012	58.0	72.0	65.0	60.0	65.0	69.0	63.0	64.0	51.0	...	55.0	74.0	71.0
8	S013	57.0	72.0	54.0	53.0	41.0	71.0	64.0	69.0	56.0	...	48.0	57.0	54.0
9	S014	50.0	87.0	92.0	80.0	75.0	72.0	71.0	82.0	59.0	...	49.0	74.0	65.0
10	S018	54.0	84.0	85.0	88.0	61.0	83.0	65.0	66.0	55.0	...	60.0	77.0	72.0
11	S019	61.0	80.0	81.0	79.0	79.0	61.0	70.0	71.0	63.0	...	68.0	82.0	72.0
12	S021	56.0	76.0	67.0	66.0	63.0	62.0	61.0	77.0	68.0	...	50.0	77.0	64.0
13	S022	55.0	55.0	59.0	55.0	46.0	63.0	63.0	52.0	53.0	...	63.0	72.0	57.0
14	S023	59.0	49.0	34.0	55.0	42.0	50.0	53.0	58.0	47.0	...	47.0	47.0	64.0
15	S026	60.0	80.0	65.0	74.0	58.0	66.0	61.0	77.0	46.0	...	58.0	57.0	73.0
16	S028	54.0	91.0	81.0	82.0	92.0	61.0	60.0	83.0	55.0	...	52.0	80.0	68.0
17	S029	52.0	64.0	52.0	61.0	57.0	59.0	46.0	61.0	44.0	...	49.0	50.0	58.0
18	S030	62.0	68.0	64.0	71.0	53.0	74.0	66.0	53.0	59.0	...	70.0	71.0	70.0
19	S033	62.0	91.0	86.0	79.0	91.0	71.0	71.0	71.0	61.0	...	64.0	74.0	63.0
20	S037	66.0	65.0	55.0	65.0	53.0	58.0	59.0	73.0	52.0	...	47.0	73.0	67.0
21	S040	58.0	57.0	50.0	59.0	39.0	63.0	56.0	74.0	53.0	...	41.0	64.0	53.0
22	S041	52.0	88.0	71.0	76.0	92.0	66.0	60.0	63.0	64.0	...	52.0	68.0	79.0
23	S044	58.0	82.0	75.0	69.0	64.0	73.0	73.0	69.0	52.0	...	63.0	76.0	72.0
24	S045	59.0	85.0	76.0	72.0	70.0	85.0	69.0	77.0	53.0	...	66.0	77.0	68.0
25	S046	62.0	88.0	77.0	74.0	92.0	55.0	74.0	74.0	58.0	...	52.0	83.0	66.0
26	S047	67.0	64.0	73.0	64.0	79.0	68.0	66.0	62.0	64.0	...	58.0	71.0	65.0
27	S050	58.0	76.0	73.0	73.0	77.0	68.0	65.0	59.0	57.0	...	62.0	57.0	62.0
28	S053	60.0	71.0	69.0	60.0	37.0	77.0	53.0	64.0	62.0	...	65.0	71.0	69.0
29	S054	53.0	69.0	62.0	64.0	62.0	56.0	63.0	66.0	50.0	...	53.0	59.0	68.0
30	S055	61.0	66.0	69.0	69.0	63.0	80.0	69.0	66.0	50.0	...	56.0	55.0	56.0
31	S056	63.0	82.0	66.0	60.0	84.0	61.0	66.0	63.0	61.0	...	50.0	53.0	66.0
32	S057	59.0	88.0	83.0	80.0	90.0	65.0	68.0	75.0	64.0	...	62.0	82.0	74.0
33	S058	57.0	66.0	76.0	72.0	64.0	75.0	56.0	70.0	58.0	...	58.0	80.0	70.0
34	S059	59.0	95.0	87.0	81.0	95.0	80.0	71.0	82.0	69.0	...	65.0	76.0	64.0
35	S060	73.0	83.0	73.0	76.0	86.0	69.0	72.0	76.0	67.0	...	68.0	74.0	77.0

	Student	L001	V001	V002	V003	V004	V005	V006	V007	L002	...	L004	V019	V020
36	S064	55.0	77.0	65.0	55.0	89.0	85.0	58.0	75.0	55.0	...	56.0	70.0	74.0
37	S067	62.0	70.0	79.0	78.0	74.0	64.0	67.0	74.0	60.0	...	62.0	72.0	71.0
38	S068	51.0	86.0	81.0	83.0	79.0	79.0	62.0	73.0	60.0	...	56.0	86.0	69.0
39	S069	63.0	66.0	51.0	53.0	84.0	67.0	64.0	72.0	59.0	...	60.0	64.0	57.0
40	S070	42.0	55.0	61.0	61.0	48.0	75.0	64.0	62.0	51.0	...	44.0	77.0	68.0
41	S073	64.0	83.0	88.0	80.0	82.0	55.0	69.0	60.0	62.0	...	69.0	61.0	65.0
42	S074	59.0	77.0	78.0	73.0	93.0	67.0	63.0	72.0	60.0	...	59.0	58.0	66.0
43	S075	58.0	83.0	56.0	59.0	78.0	69.0	64.0	73.0	49.0	...	47.0	64.0	68.0
44	S079	60.0	62.0	66.0	65.0	59.0	69.0	62.0	67.0	62.0	...	63.0	58.0	65.0
45	S081	62.0	82.0	70.0	72.0	81.0	73.0	55.0	72.0	65.0	...	64.0	70.0	70.0
46	S082	58.0	85.0	93.0	82.0	88.0	68.0	65.0	74.0	64.0	...	67.0	96.0	72.0
47	S083	69.0	57.0	65.0	58.0	56.0	63.0	67.0	68.0	54.0	...	66.0	58.0	64.0
48	S084	55.0	79.0	71.0	67.0	68.0	54.0	67.0	80.0	53.0	...	53.0	72.0	63.0
49	S085	61.0	85.0	81.0	82.0	88.0	70.0	67.0	61.0	65.0	...	59.0	68.0	71.0

In [ ]:
```
all5SemDF_noNaN.to_csv("Yeung_Wing.csv", index=False)
```

> # Requirement 1 is finished here. following are the DataFrames that are prepared for the relationship finding at the end.

In [ ]:
```
# all students include those who are not completed all 5 semesters
df = pd.merge(df1, df2, on='Student', how='left')
df = pd.merge(df, df3, on='Student', how='left')
df = pd.merge(df, df4, on='Student', how='left')
df = pd.merge(df, df5, on='Student', how='left')
```

In [ ]:
```
studentWithNANDF = df[~df['Student'].isin(all5SemDF_noNaN['Student'])]
studentQuitDF = df[~df['Student'].isin(all5SemDF['Student'])]
studentMaybeExecemption = studentWithNANDF[~studentWithNANDF['Student'].isin(studer
```

Requirement 2

Count the number of students who completed all 5 semesters

In [ ]:
```
print(f"Number of students who confirmed completed all 5 semesters: {len(all5SemDF_
print(f"Number of students who may completed all 5 semesters (Maybe NaN results due
print(f"Number of students who are 100% did not completed all 5 semesters: {len(stu
print(f"Number of students who may be exempted or not completed all 5 semesters: {l
print(f"Number of students who may not completed all 5 semesters: {len(all5SemDF)-l
print(f"Total number of students: {len(df)}")
```

```
Number of students who confirmed completed all 5 semesters: 50
Number of students who may completed all 5 semesters (Maybe NaN results due to exe
mption): 67
Number of students who are 100% did not completed all 5 semesters: 16
Number of students who may be exempted or not completed all 5 semesters: 17
Number of students who may not completed all 5 semesters: 17 + 16(100%) = 33
Total number of students: 83
```

> # Number of students who confirmed completed all 5 semesters: 50
>
> Number of students who may completed all 5 semesters (Maybe NaN results due to exemption): 67
>
> Number of students who are 100% did not completed all 5 semesters: 16
>
> Number of students who may be exempted or not completed all 5 semesters: 17
>
> Number of students who may not completed all 5 semesters: 17 + 16(100%) = 33
>
> Total number of students: 83

Requirement 3

Statistics (mean/ maximum/ minimum/ standard deviation of module marks) of the designated modules - V???, V???, V???

In [ ]:

```
all5SemDF_noNaN.describe().iloc[:,1]
```

Out[ ]:

```
count    50.000000
mean     75.240000
std      11.174972
min      49.000000
25%      66.500000
50%      77.000000
75%      83.750000
max      95.000000
Name: V001, dtype: float64
```

In [ ]:

```
all5SemDF_noNaN.describe()
```

Out[ ]:

	L001	V001	V002	V003	V004	V005	V006	V007	
count	50.000000	50.000000	50.00000	50.000000	50.000000	50.000000	50.000000	50.00000	50.00
mean	58.380000	75.240000	69.92000	69.340000	69.800000	68.220000	63.660000	69.94000	56.96
std	5.158271	11.174972	12.78047	9.908994	17.235464	8.514789	5.727164	7.38838	6.49
min	42.000000	49.000000	34.00000	49.000000	33.000000	50.000000	46.000000	52.00000	43.00
25%	55.250000	66.500000	64.25000	60.250000	57.250000	63.000000	61.000000	64.50000	52.25
50%	58.500000	77.000000	70.50000	70.000000	74.500000	68.000000	64.000000	71.50000	57.50
75%	61.000000	83.750000	78.75000	79.000000	84.000000	73.000000	67.000000	74.00000	62.00
max	73.000000	95.000000	93.00000	88.000000	95.000000	87.000000	74.000000	83.00000	69.00

8 rows × 31 columns

In [ ]:

```
pd.concat([all5SemDF_noNaN.describe().iloc[1:4], all5SemDF_noNaN.describe().iloc[7:
```

Out[ ]:

	L001	V001	V002	V003	V004	V005	V006	V007	
mean	58.380000	75.240000	69.92000	69.340000	69.800000	68.220000	63.660000	69.94000	56.96
std	5.158271	11.174972	12.78047	9.908994	17.235464	8.514789	5.727164	7.38838	6.49
min	42.000000	49.000000	34.00000	49.000000	33.000000	50.000000	46.000000	52.00000	43.00
max	73.000000	95.000000	93.00000	88.000000	95.000000	87.000000	74.000000	83.00000	69.00

4 rows × 31 columns

In [ ]:

```
df['L003'].describe()
```

Out[ ]:

```
count    73.000000
mean     48.849315
std      11.878635
min       0.000000
25%      46.000000
50%      52.000000
75%      56.000000
max      65.000000
Name: L003, dtype: float64
```

In [ ]:

```
# mean of V001
df['V001'].mean()
```

Out[ ]:

```
64.79518072289157
```

In [ ]:

```
# maximum value of V002
df['V002'].max()
```

Out[ ]:

```
93.0
```

In [ ]:

```
# minimum value of V003
df['V003'].min()
```

Out[ ]:

```
19.0
```

In [ ]:
```
# standard deviation of V004
df['V004'].std()
```

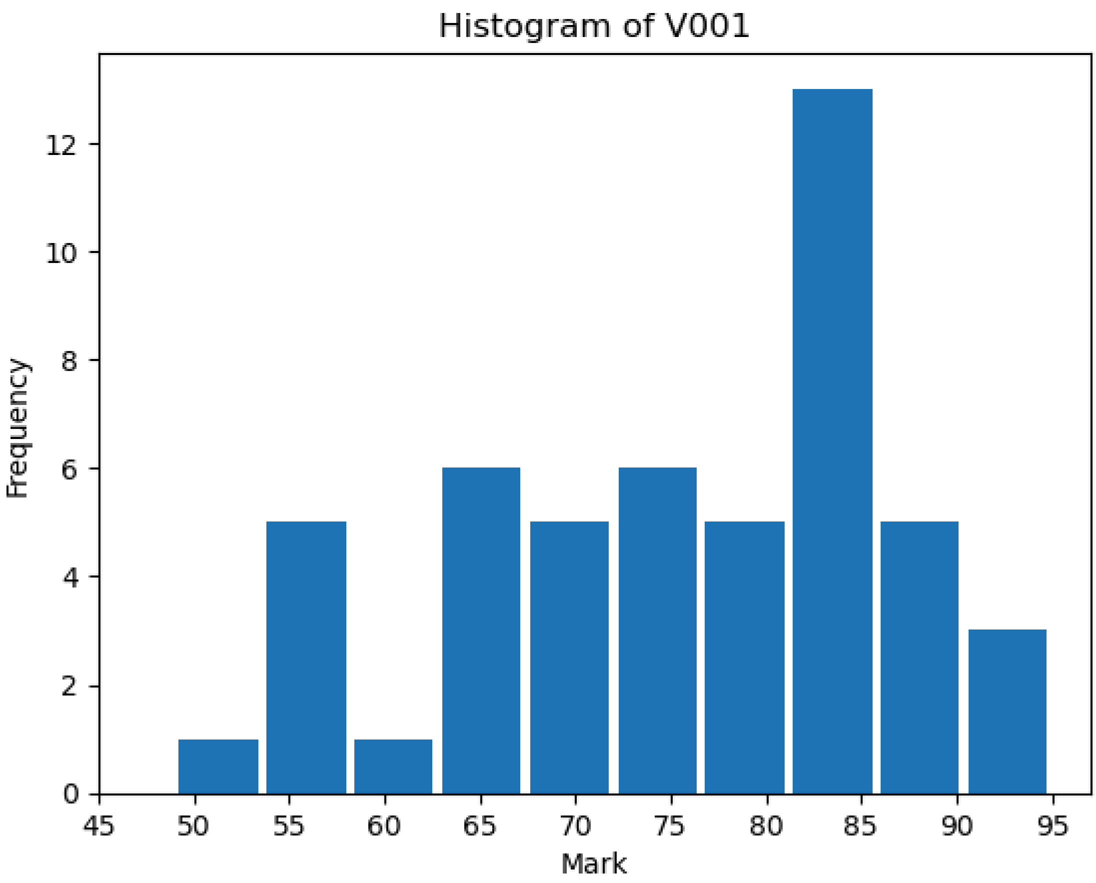
Out[ ]:
```
21.238891411158047
```

Requirement 4

Histogram of the designated module - V???

In [ ]:
```
import matplotlib.pyplot as plt
```

In [ ]:
```
plt.hist(all5SemDF_noNaN['V001'], rwidth=0.9) # rwidth is added for the space betwe
plt.xlabel('Mark')
plt.ylabel('Frequency')
#set x axis range and step
left, right = plt.xlim()
plt.xticks(range(int(left)-1, int(right)+1, 5))
plt.title('Histogram of V001')
plt.show()
```

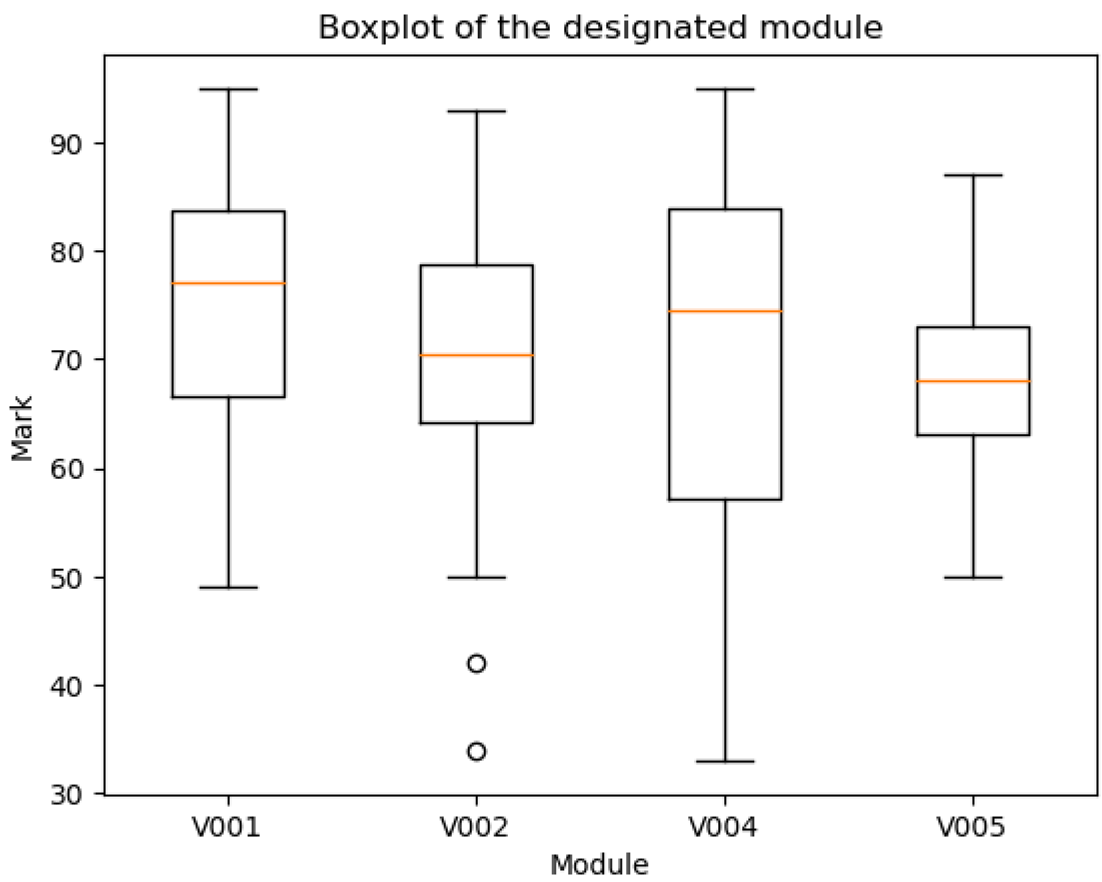


Requirement 5 (NOT needed for those who completed Requirement 1)

Box plot of the designated module - V???

In [ ]:
```
# import matplotlib.pyplot as plt
```

In [ ]:
```
plt.boxplot([all5SemDF_noNaN['V001'],all5SemDF_noNaN['V002'],all5SemDF_noNaN['V004'
plt.ylabel('Mark')
plt.xlabel('Module')
plt.title('Boxplot of the designated module')
plt.show()
```



Requirement 6

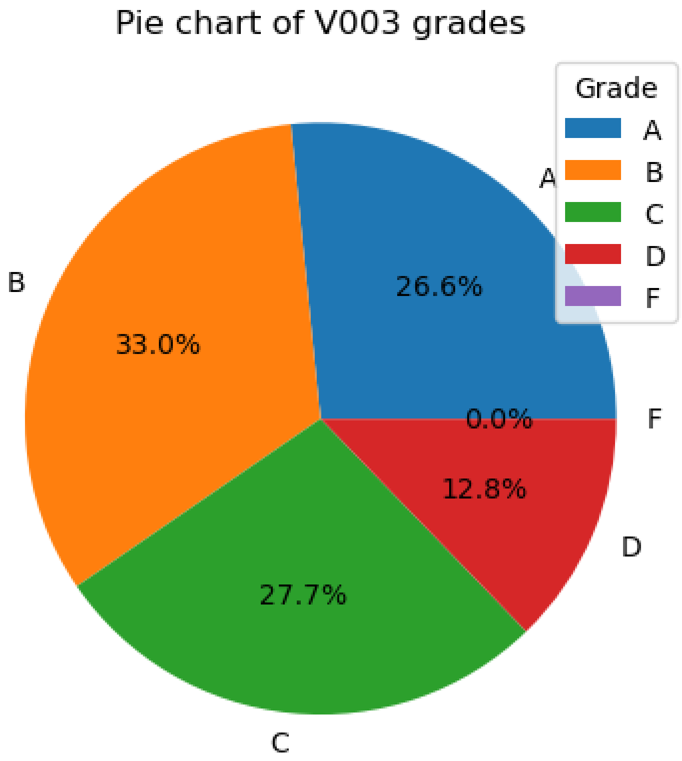
Pie chart of the designated module - V???

In [ ]:
```
# import matplotlib.pyplot as plt
```

In [ ]:
```
V003GradeA = len(all5SemDF_noNaN[all5SemDF_noNaN['V003']>=70]['V003'])
V003GradeB = len(all5SemDF_noNaN[(all5SemDF_noNaN['V003']>=60) & (df['V003']<70)]['
V003GradeC = len(all5SemDF_noNaN[(all5SemDF_noNaN['V003']>=50) & (df['V003']<60)]['
V003GradeD = len(all5SemDF_noNaN[(all5SemDF_noNaN['V003']>=40) & (df['V003']<50)]['
V003GradeF = len(all5SemDF_noNaN[all5SemDF_noNaN['V003']<40]['V003'])
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_10700\4009642701.py:2: UserWarning: Bo
olean Series key will be reindexed to match DataFrame index.
  V003GradeB = len(all5SemDF_noNaN[(all5SemDF_noNaN['V003']>=60) & (df['V003']<7
0)]['V003'])
C:\Users\admin\AppData\Local\Temp\ipykernel_10700\4009642701.py:3: UserWarning: Bo
olean Series key will be reindexed to match DataFrame index.
  V003GradeC = len(all5SemDF_noNaN[(all5SemDF_noNaN['V003']>=50) & (df['V003']<6
0)]['V003'])
C:\Users\admin\AppData\Local\Temp\ipykernel_10700\4009642701.py:4: UserWarning: Bo
olean Series key will be reindexed to match DataFrame index.
  V003GradeD = len(all5SemDF_noNaN[(all5SemDF_noNaN['V003']>=40) & (df['V003']<5
0)]['V003'])
```

In [ ]:
```
plt.pie([V003GradeA,V003GradeB,V003GradeC,V003GradeD,V003GradeF], labels=['A','B','
plt.title('Pie chart of V003 grades')
plt.legend(title='Grade')
plt.show()
```



Requirement 7A

Underlying relationship about project module to other modules

In [ ]:
```
all5SemDF_noNaN_stundentIndex = all5SemDF_noNaN.set_index('Student')

# grab all result of module start with letter L
lDF = all5SemDF_noNaN_stundentIndex.filter(regex='^L')
# grab all result of module start with letter V
vDF = all5SemDF_noNaN_stundentIndex.filter(regex='^V')
# grab all result of module start with letter P
pDF = all5SemDF_noNaN_stundentIndex.filter(regex='^P')
### Use all5SemDF_noNaN to avoid outliner that some students did not complete all t

print(f"Relationship between all language module: {lDF.corr().mean().mean()*100}%")
print(f"Relationship between all vocation module: {vDF.corr().mean().mean()*100}%")
print(f"Relationship between all project module: {pDF.corr().mean().mean()*100}% (1
```

```
Relationship between all language module: 57.835510276836324%
Relationship between all vocation module: 46.097306542476524%
Relationship between all project module: 100.0% (100% because there is only one pr
oject module)
```

In [ ]:
```
meanDF = pd.DataFrame(lDF.mean(axis=1), columns=['LAN'])
meanDF = pd.merge(meanDF, pd.DataFrame(vDF.mean(axis=1), columns=['EEE']), on='Stud
meanDF = pd.merge(meanDF, pd.DataFrame(pDF.mean(axis=1), columns=['PROJ']), on='Stu

display(meanDF.corr())
```

	LAN	EEE	PROJ
LAN	1.000000	0.575835	0.300924
EEE	0.575835	1.000000	0.449187
PROJ	0.300924	0.449187	1.000000

In [ ]:

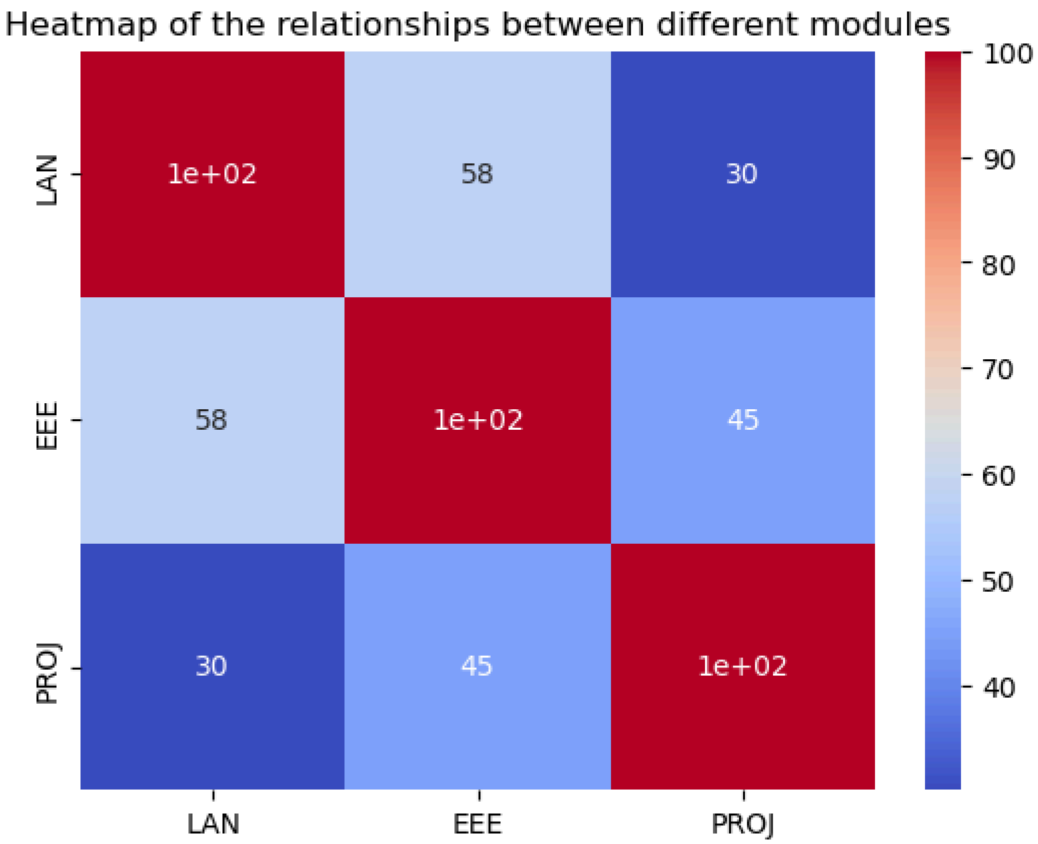
```
lvCorr = meanDF.corr().loc["LAN", "EEE"]
lpCorr = meanDF.corr().loc["LAN", "PROJ"]
vpCorr = meanDF.corr().loc["EEE", "PROJ"]

print(f"Relationship between language and vocation module: {lvCorr*100}%")
print(f"Relationship between language and project module: {lpCorr*100}%")
print(f"Relationship between vocation and project module: {vpCorr*100}%")
print("")
print("We can see that in term of project module, the grade of vocation course is m
```

```
Relationship between language and vocation module: 57.583506636907224%
Relationship between language and project module: 30.092426414939265%
Relationship between vocation and project module: 44.91872964027379%

We can see that in term of project module, the grade of vocation course is more re
lated then language module.
```

> # We can see that in term of project module, the grade of vocation course is more related then language module.

In [ ]:

```
import seaborn as sns
sns.heatmap(meanDF.corr()*100, annot=True, cmap='coolwarm')
plt.title("Heatmap of the relationships between different modules")
plt.show()
```



Requirement 7B

Underlying relationship between the student who didn't finished all 5 sems to other students

In [ ]:

```
# only compare the module studied by the withdrawn students
studentQuitNoNADF = studentQuitDF.dropna(how='any', axis=1)
studentQuitNoNADF
```

Out[ ]:

	Student	L001	V001	V002	V003	V004	V005	V006	V007
8	S009	58.0	82.0	47.0	46.0	69.0	64.0	58.0	64.0
14	S015	45.0	24.0	16.0	34.0	41.0	58.0	32.0	70.0
30	S031	58.0	20.0	25.0	23.0	19.0	59.0	46.0	47.0
32	S034	50.0	51.0	44.0	43.0	31.0	64.0	44.0	71.0
33	S035	42.0	28.0	36.0	22.0	40.0	58.0	33.0	60.0
34	S036	49.0	18.0	32.0	33.0	17.0	50.0	57.0	54.0
45	S048	49.0	42.0	45.0	35.0	56.0	45.0	41.0	65.0
46	S049	42.0	43.0	41.0	47.0	10.0	49.0	68.0	59.0
49	S052	40.0	50.0	0.0	24.0	70.0	40.0	28.0	0.0
60	S063	61.0	21.0	36.0	29.0	17.0	65.0	69.0	40.0
62	S065	66.0	84.0	79.0	83.0	85.0	78.0	78.0	81.0
68	S071	56.0	59.0	72.0	68.0	50.0	74.0	58.0	69.0
69	S072	14.0	20.0	28.0	19.0	16.0	35.0	20.0	37.0
74	S077	56.0	58.0	54.0	60.0	51.0	63.0	56.0	61.0
75	S078	50.0	51.0	50.0	63.0	58.0	58.0	49.0	55.0
77	S080	61.0	70.0	55.0	73.0	56.0	63.0	63.0	67.0

In [ ]:

```
all5SemComparableDF = all5SemDF_noNaN[studentQuitNoNADF.columns]
all5SemComparableDF['quit'] = False
all5SemComparableDF
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_10700\2722242388.py:2: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  all5SemComparableDF['quit'] = False
```

Out[ ]:

	Student	L001	V001	V002	V003	V004	V005	V006	V007	quit
0	S001	56.0	82.0	74.0	68.0	78.0	87.0	63.0	79.0	False
1	S003	52.0	83.0	88.0	79.0	85.0	66.0	62.0	74.0	False
2	S005	56.0	74.0	54.0	49.0	53.0	60.0	54.0	73.0	False
3	S007	61.0	85.0	76.0	79.0	79.0	64.0	66.0	82.0	False
4	S008	58.0	72.0	69.0	82.0	76.0	77.0	68.0	74.0	False
5	S010	59.0	56.0	42.0	60.0	33.0	65.0	60.0	67.0	False
6	S011	53.0	69.0	68.0	65.0	49.0	73.0	62.0	68.0	False
7	S012	58.0	72.0	65.0	60.0	65.0	69.0	63.0	64.0	False
8	S013	57.0	72.0	54.0	53.0	41.0	71.0	64.0	69.0	False
9	S014	50.0	87.0	92.0	80.0	75.0	72.0	71.0	82.0	False
10	S018	54.0	84.0	85.0	88.0	61.0	83.0	65.0	66.0	False
11	S019	61.0	80.0	81.0	79.0	79.0	61.0	70.0	71.0	False
12	S021	56.0	76.0	67.0	66.0	63.0	62.0	61.0	77.0	False
13	S022	55.0	55.0	59.0	55.0	46.0	63.0	63.0	52.0	False
14	S023	59.0	49.0	34.0	55.0	42.0	50.0	53.0	58.0	False
15	S026	60.0	80.0	65.0	74.0	58.0	66.0	61.0	77.0	False
16	S028	54.0	91.0	81.0	82.0	92.0	61.0	60.0	83.0	False
17	S029	52.0	64.0	52.0	61.0	57.0	59.0	46.0	61.0	False
18	S030	62.0	68.0	64.0	71.0	53.0	74.0	66.0	53.0	False
19	S033	62.0	91.0	86.0	79.0	91.0	71.0	71.0	71.0	False
20	S037	66.0	65.0	55.0	65.0	53.0	58.0	59.0	73.0	False
21	S040	58.0	57.0	50.0	59.0	39.0	63.0	56.0	74.0	False
22	S041	52.0	88.0	71.0	76.0	92.0	66.0	60.0	63.0	False
23	S044	58.0	82.0	75.0	69.0	64.0	73.0	73.0	69.0	False
24	S045	59.0	85.0	76.0	72.0	70.0	85.0	69.0	77.0	False
25	S046	62.0	88.0	77.0	74.0	92.0	55.0	74.0	74.0	False
26	S047	67.0	64.0	73.0	64.0	79.0	68.0	66.0	62.0	False
27	S050	58.0	76.0	73.0	73.0	77.0	68.0	65.0	59.0	False
28	S053	60.0	71.0	69.0	60.0	37.0	77.0	53.0	64.0	False
29	S054	53.0	69.0	62.0	64.0	62.0	56.0	63.0	66.0	False
30	S055	61.0	66.0	69.0	69.0	63.0	80.0	69.0	66.0	False
31	S056	63.0	82.0	66.0	60.0	84.0	61.0	66.0	63.0	False
32	S057	59.0	88.0	83.0	80.0	90.0	65.0	68.0	75.0	False
33	S058	57.0	66.0	76.0	72.0	64.0	75.0	56.0	70.0	False
34	S059	59.0	95.0	87.0	81.0	95.0	80.0	71.0	82.0	False
35	S060	73.0	83.0	73.0	76.0	86.0	69.0	72.0	76.0	False

	Student	L001	V001	V002	V003	V004	V005	V006	V007	quit
36	S064	55.0	77.0	65.0	55.0	89.0	85.0	58.0	75.0	False
37	S067	62.0	70.0	79.0	78.0	74.0	64.0	67.0	74.0	False
38	S068	51.0	86.0	81.0	83.0	79.0	79.0	62.0	73.0	False
39	S069	63.0	66.0	51.0	53.0	84.0	67.0	64.0	72.0	False
40	S070	42.0	55.0	61.0	61.0	48.0	75.0	64.0	62.0	False
41	S073	64.0	83.0	88.0	80.0	82.0	55.0	69.0	60.0	False
42	S074	59.0	77.0	78.0	73.0	93.0	67.0	63.0	72.0	False
43	S075	58.0	83.0	56.0	59.0	78.0	69.0	64.0	73.0	False
44	S079	60.0	62.0	66.0	65.0	59.0	69.0	62.0	67.0	False
45	S081	62.0	82.0	70.0	72.0	81.0	73.0	55.0	72.0	False
46	S082	58.0	85.0	93.0	82.0	88.0	68.0	65.0	74.0	False
47	S083	69.0	57.0	65.0	58.0	56.0	63.0	67.0	68.0	False
48	S084	55.0	79.0	71.0	67.0	68.0	54.0	67.0	80.0	False
49	S085	61.0	85.0	81.0	82.0	88.0	70.0	67.0	61.0	False

In [ ]:

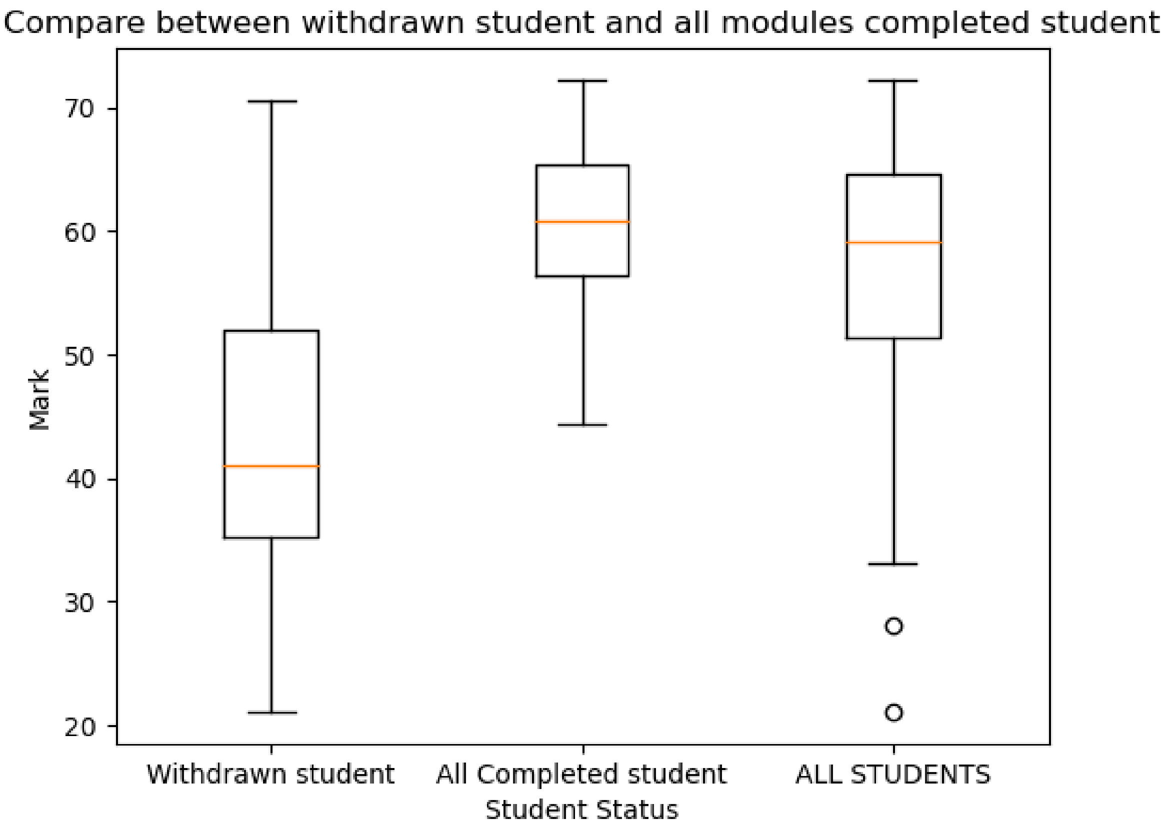
```
studentQuitNoNADF['quit'] = True
relationshipDF = pd.concat([studentQuitNoNADF, all5SemComparableDF]).reset_index(dr
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_10700\2532102351.py:1: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  studentQuitNoNADF['quit'] = True
```

In [ ]:

```
plt.boxplot([relationshipDF[relationshipDF["quit"]==True].mean(axis=1), relationshi
plt.xlabel("Student Status")
plt.ylabel("Mark")
plt.title("Compare between withdrawn student and all modules completed student")
plt.show()

print("We can see although the mean of grade of the withdrawn students is much lowe
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_10700\2361699887.py:1: FutureWarning:
Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is
deprecated; in a future version this will raise TypeError.  Select only valid colu
mns before calling the reduction.
  plt.boxplot([relationshipDF[relationshipDF["quit"]==True].mean(axis=1), relation
shipDF[relationshipDF["quit"]==False].mean(axis=1), relationshipDF.mean(axis=1)],
labels=["Withdrawn student", "All Completed student", "ALL STUDENTS"])
```




```
We can see although the mean of grade of the withdrawn students is much lower then
the students completed all modules, there are still some examples that they get hi
gh marks.
```

> # We can see although the mean of grade of the withdrawn students is much lower then the students completed all modules, there are still some examples that they get high marks.

In [ ]:
```
modules = []
for col, foo in enumerate(relationshipDF.columns.str.startswith(('L', 'V', 'P'))):
    if foo == True:
        modules.append(relationshipDF.columns[col])

passDF = relationshipDF.copy().set_index("Student")

for i in modules:
    passDF[i] = relationshipDF.set_index("Student")[i].apply(lambda x: True if x<4(

passDF
```

Out[ ]:

Student	L001	V001	V002	V003	V004	V005	V006	V007	quit
S009	False	False	False	False	False	False	False	False	True
S015	False	True	True	True	False	False	True	False	True
S031	False	True	True	True	True	False	False	False	True
S034	False	False	False	False	True	False	False	False	True
S035	False	True	True	True	False	False	True	False	True
...	...	...	...	...	...	...	...	...	...
S081	False	False	False	False	False	False	False	False	False
S082	False	False	False	False	False	False	False	False	False
S083	False	False	False	False	False	False	False	False	False
S084	False	False	False	False	False	False	False	False	False
S085	False	False	False	False	False	False	False	False	False

66 rows × 9 columns

> # In Python, False = 0, True = 1, so we set false to be pass and true to be failed to see the relationship between withdrawn rate and pass rate to designated modules

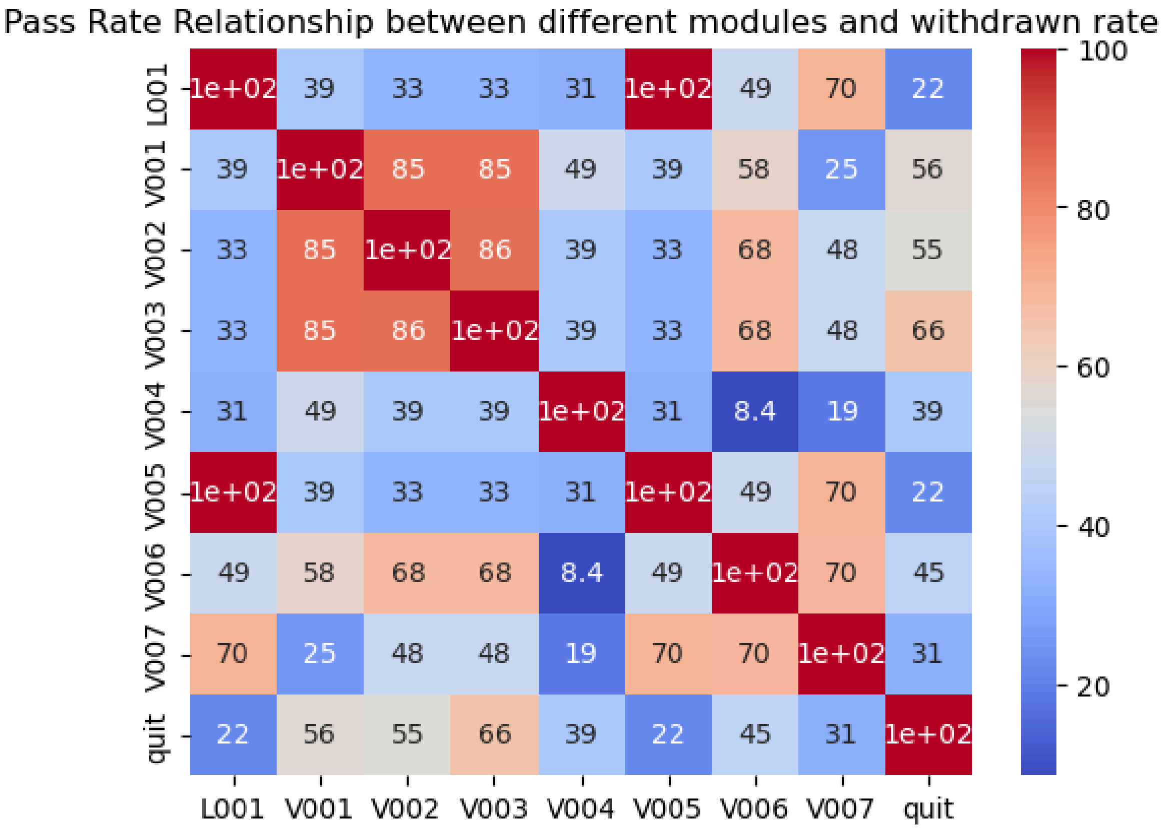
In [ ]:
```
passDF.corr()
```

Out[ ]:

	L001	V001	V002	V003	V004	V005	V006	V007	quit
L001	1.000000	0.392232	0.333974	0.333974	0.312147	1.000000	0.488325	0.701646	0.219265
V001	0.392232	1.000000	0.851469	0.851469	0.488663	0.392232	0.582334	0.251558	0.559017
V002	0.333974	0.851469	1.000000	0.857759	0.393535	0.333974	0.683917	0.475986	0.548204
V003	0.333974	0.851469	0.857759	1.000000	0.393535	0.333974	0.683917	0.475986	0.656532
V004	0.312147	0.488663	0.393535	0.393535	1.000000	0.312147	0.084108	0.187317	0.393366
V005	1.000000	0.392232	0.333974	0.333974	0.312147	1.000000	0.488325	0.701646	0.219265
V006	0.488325	0.582334	0.683917	0.683917	0.084108	0.488325	1.000000	0.695971	0.449013
V007	0.701646	0.251558	0.475986	0.475986	0.187317	0.701646	0.695971	1.000000	0.312500
quit	0.219265	0.559017	0.548204	0.656532	0.393366	0.219265	0.449013	0.312500	1.000000

In [ ]:
```
sns.heatmap(passDF.corr()*100, annot=True, cmap='coolwarm')
plt.title("Pass Rate Relationship between different modules and withdrawn rate")
plt.show()
```

```
display(passDF.corr()["quit"]*100)

print("We can see the pass rate between V001, V002 and V003 are highly related, and
```




```
L001      21.926450
V001      55.901699
V002      54.820436
V003      65.653216
V004      39.336604
V005      21.926450
V006      44.901326
V007      31.250000
quit     100.000000
Name: quit, dtype: float64
We can see the pass rate between V001, V002 and V003 are highly related, and most
withdrawn students are also failed to these modules.
```

> # We can see the pass rate between V001, V002 and V003 are highly related, and most withdrawn students are also failed to these modules.