M Koushik

2211CS010343

Group 4

DATASET DESCRIPTION:

The data set, which is obtained from an Excel file (MIDMARKS.xlsx), consists of student marks in various subjects like Digital Variance (DV), Mathematics-II (M2), Programming Principles (PP), Basic Electrical and Electronics Engineering (BEEE), Formal Languages (FL), and Fundamentals of Information Management Systems (FIMS). There were inconsistencies in the data set in the form of non-numeric data in certain columns, which was made numeric for correct analysis. Missing values were also detected and replaced by 0 to preserve data integrity.

To evaluate overall student performance, a Total column was added by adding marks of all subjects, and a Percentage column was added by calculating total marks into percentage points on a scale of 120 maximum marks. To further classify student performance, a grading system was used to classify students into various grade ranges: A (90%+), B+ (80-89%), B (70-79%), C+ (60-69%), C (50-59%), and D (less than 50%). This system of grading facilitated an understanding of the overall pattern of scores in an organized manner.

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df = pd.read_excel(r"C:\Users\Saiko\OneDrive\Desktop\3-2\Data Analytics\MIDMARKS.xlsx"
```

In [2]:
```
pip install seaborn
```

```
Requirement already satisfied: seaborn in c:\users\saiko\anaconda3\lib\site-packages
(0.13.2)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in c:\users\saiko\anaconda3\lib
\site-packages (from seaborn) (1.26.4)
Requirement already satisfied: pandas>=1.2 in c:\users\saiko\anaconda3\lib\site-pack
ages (from seaborn) (2.2.2)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in c:\users\saiko\anaconda3\l
ib\site-packages (from seaborn) (3.9.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\saiko\anaconda3\lib\site
-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\saiko\anaconda3\lib\site-pac
kages (from matplotlib!=3.6.1,>=3.4->seaborn) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\saiko\anaconda3\lib\sit
e-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\saiko\anaconda3\lib\sit
e-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\saiko\anaconda3\lib\site-
packages (from matplotlib!=3.6.1,>=3.4->seaborn) (24.1)
Requirement already satisfied: pillow>=8 in c:\users\saiko\anaconda3\lib\site-packag
es (from matplotlib!=3.6.1,>=3.4->seaborn) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\saiko\anaconda3\lib\site
-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\saiko\anaconda3\lib
\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\saiko\anaconda3\lib\site-pac
kages (from pandas>=1.2->seaborn) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\saiko\anaconda3\lib\site-p
ackages (from pandas>=1.2->seaborn) (2023.3)
Requirement already satisfied: six>=1.5 in c:\users\saiko\anaconda3\lib\site-package
s (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.4->seaborn) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [3]:
```
df
```

Out[3]:

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage
0	1	1	12	0	17	9	19.0	15	value	NaN
1	2	2	19	12	16	16	18.0	3	value	NaN
2	3	3	18	14	18	18	18.0	16	value	NaN
3	4	4	15	9	19	17	19.0	15	value	NaN
4	5	5	18	17	19	19	20.0	18	value	NaN
...	...	...	...	...	...	...	...	...	...	...
711	712	712	19	8	8	19	17.0	18	value	NaN
712	713	713	12	1	7	10	20.0	8	value	NaN
713	714	714	17	6	14	14	17.0	18	value	NaN
714	715	715	12	1	6	7	15.0	12	value	NaN
715	716	716	19	14	17	16	20.0	19	value	NaN

716 rows × 10 columns

Mid Marks Data

In [4]:
```
df.rename(columns={'M-II':'M2'},inplace=True)
```

Renaming M-II as M2

In [5]:
```
df
```

Out[5]:

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage
0	1	1	12	0	17	9	19.0	15	value	NaN
1	2	2	19	12	16	16	18.0	3	value	NaN
2	3	3	18	14	18	18	18.0	16	value	NaN
3	4	4	15	9	19	17	19.0	15	value	NaN
4	5	5	18	17	19	19	20.0	18	value	NaN
...	...	...	...	...	...	...	...	...	...	...
711	712	712	19	8	8	19	17.0	18	value	NaN
712	713	713	12	1	7	10	20.0	8	value	NaN
713	714	714	17	6	14	14	17.0	18	value	NaN
714	715	715	12	1	6	7	15.0	12	value	NaN
715	716	716	19	14	17	16	20.0	19	value	NaN

716 rows × 10 columns

In [6]:
```
df['DV'] = pd.to_numeric(df['DV'], errors='coerce')
df['M2'] = pd.to_numeric(df['M2'], errors='coerce')
df['PP'] = pd.to_numeric(df['PP'], errors='coerce')
df['BEEE'] = pd.to_numeric(df['BEEE'], errors='coerce')
df['FL'] = pd.to_numeric(df['FL'], errors='coerce')
df['FIMS'] = pd.to_numeric(df['FIMS'], errors='coerce')
df.fillna(0, inplace=True)
```

Converting into numeric

In [7]:
```
df['Total'] = df['DV'] + df['M2'] + df['PP'] + df['BEEE'] + df['FL'] + df['FIMS']

df
```

Out[7]:

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total
0	1	1	12.0	0	17.0	9	19.0	15	value	0.0	72.0
1	2	2	19.0	12	16.0	16	18.0	3	value	0.0	84.0
2	3	3	18.0	14	18.0	18	18.0	16	value	0.0	102.0
3	4	4	15.0	9	19.0	17	19.0	15	value	0.0	94.0
4	5	5	18.0	17	19.0	19	20.0	18	value	0.0	111.0
...	...	...	...	...	...	...	...	...	...	...	...
711	712	712	19.0	8	8.0	19	17.0	18	value	0.0	89.0
712	713	713	12.0	1	7.0	10	20.0	8	value	0.0	58.0
713	714	714	17.0	6	14.0	14	17.0	18	value	0.0	86.0
714	715	715	12.0	1	6.0	7	15.0	12	value	0.0	53.0
715	716	716	19.0	14	17.0	16	20.0	19	value	0.0	105.0

716 rows × 11 columns

Calculating total

In [8]:
```
df["Percentage"] = (df['Total']/120)*100
```

In [9]:
```
df
```

Out[9]:

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentage
0	1	1	12.0	0	17.0	9	19.0	15	value	0.0	72.0	60.000000
1	2	2	19.0	12	16.0	16	18.0	3	value	0.0	84.0	70.000000
2	3	3	18.0	14	18.0	18	18.0	16	value	0.0	102.0	85.000000
3	4	4	15.0	9	19.0	17	19.0	15	value	0.0	94.0	78.333333
4	5	5	18.0	17	19.0	19	20.0	18	value	0.0	111.0	92.500000
...	...	...	...	...	...	...	...	...	...	...	...	...
711	712	712	19.0	8	8.0	19	17.0	18	value	0.0	89.0	74.166667
712	713	713	12.0	1	7.0	10	20.0	8	value	0.0	58.0	48.333333
713	714	714	17.0	6	14.0	14	17.0	18	value	0.0	86.0	71.666667
714	715	715	12.0	1	6.0	7	15.0	12	value	0.0	53.0	44.166667
715	716	716	19.0	14	17.0	16	20.0	19	value	0.0	105.0	87.500000

716 rows × 12 columns

In [10]:
```
df['Percentage'] = df['Percentage'].round().astype(int)
df
```

Out[10]:

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentage
0	1	1	12.0	0	17.0	9	19.0	15	value	0.0	72.0	60
1	2	2	19.0	12	16.0	16	18.0	3	value	0.0	84.0	70
2	3	3	18.0	14	18.0	18	18.0	16	value	0.0	102.0	85
3	4	4	15.0	9	19.0	17	19.0	15	value	0.0	94.0	78
4	5	5	18.0	17	19.0	19	20.0	18	value	0.0	111.0	92
...	...	...	...	...	...	...	...	...	...	...	...	...
711	712	712	19.0	8	8.0	19	17.0	18	value	0.0	89.0	74
712	713	713	12.0	1	7.0	10	20.0	8	value	0.0	58.0	48
713	714	714	17.0	6	14.0	14	17.0	18	value	0.0	86.0	72
714	715	715	12.0	1	6.0	7	15.0	12	value	0.0	53.0	44
715	716	716	19.0	14	17.0	16	20.0	19	value	0.0	105.0	88

716 rows × 12 columns

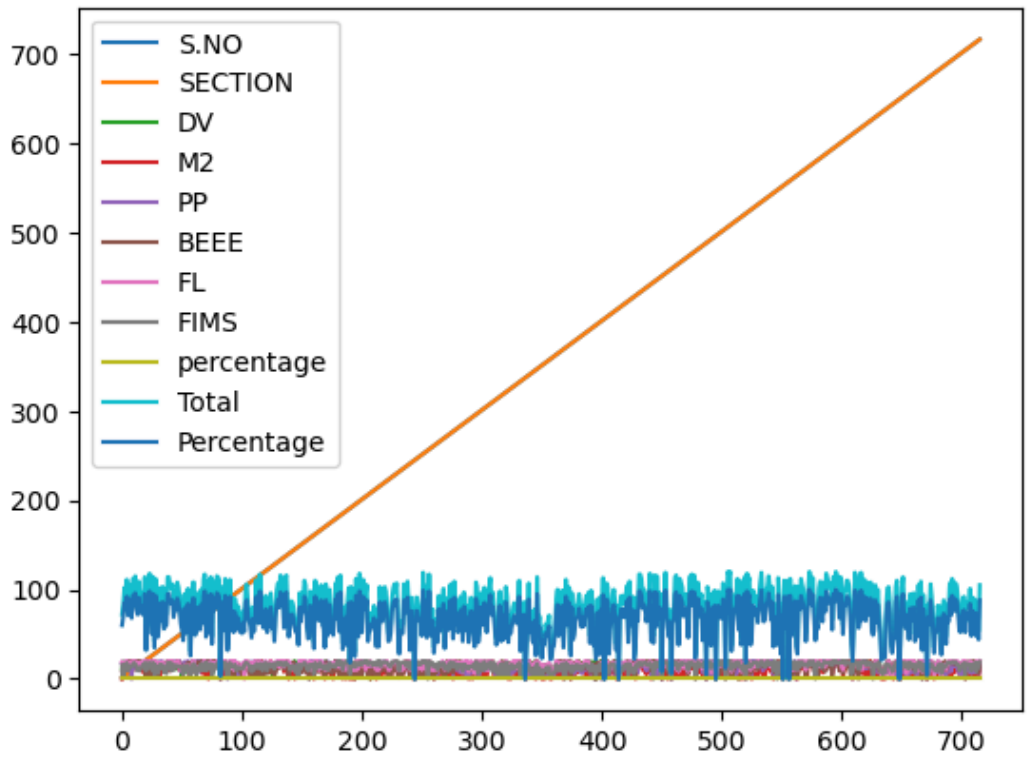
In [11]:
```
def assign_grade(percentage):
    if percentage >= 90:
        return 'A'
    elif percentage >= 80:
        return 'B+'
    elif percentage >= 70:
        return 'B'
    elif percentage >= 60:
        return 'C+'
    elif percentage >=50:
        return 'C'
    elif percentage >=40:
        return 'D'
    else:
        return 'F'
df['Grade'] = df['Percentage'].apply(assign_grade)
df
```

Out[11]:

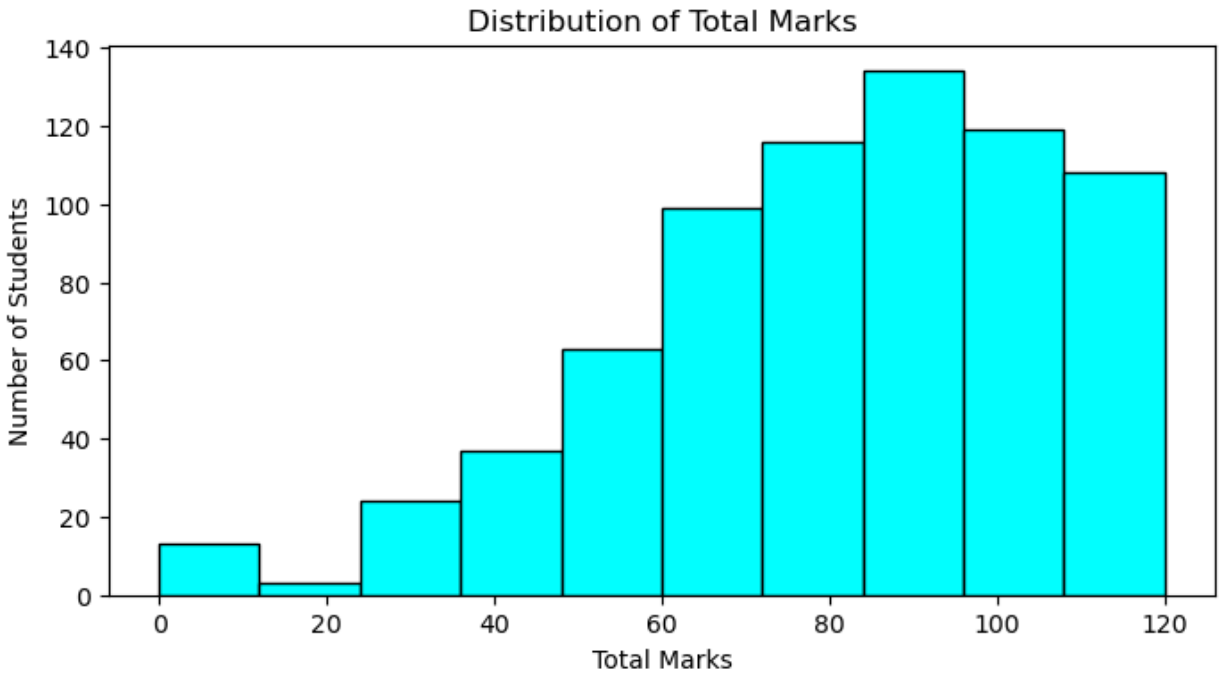
	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentage	Gr
0	1	1	12.0	0	17.0	9	19.0	15	value	0.0	72.0	60	
1	2	2	19.0	12	16.0	16	18.0	3	value	0.0	84.0	70	
2	3	3	18.0	14	18.0	18	18.0	16	value	0.0	102.0	85	
3	4	4	15.0	9	19.0	17	19.0	15	value	0.0	94.0	78	
4	5	5	18.0	17	19.0	19	20.0	18	value	0.0	111.0	92	
...	...	...	...	...	...	...	...	...	...	...	...	...	
711	712	712	19.0	8	8.0	19	17.0	18	value	0.0	89.0	74	
712	713	713	12.0	1	7.0	10	20.0	8	value	0.0	58.0	48	
713	714	714	17.0	6	14.0	14	17.0	18	value	0.0	86.0	72	
714	715	715	12.0	1	6.0	7	15.0	12	value	0.0	53.0	44	
715	716	716	19.0	14	17.0	16	20.0	19	value	0.0	105.0	88	

716 rows × 13 columns

In [12]:
```
df.plot()
plt.show()
```

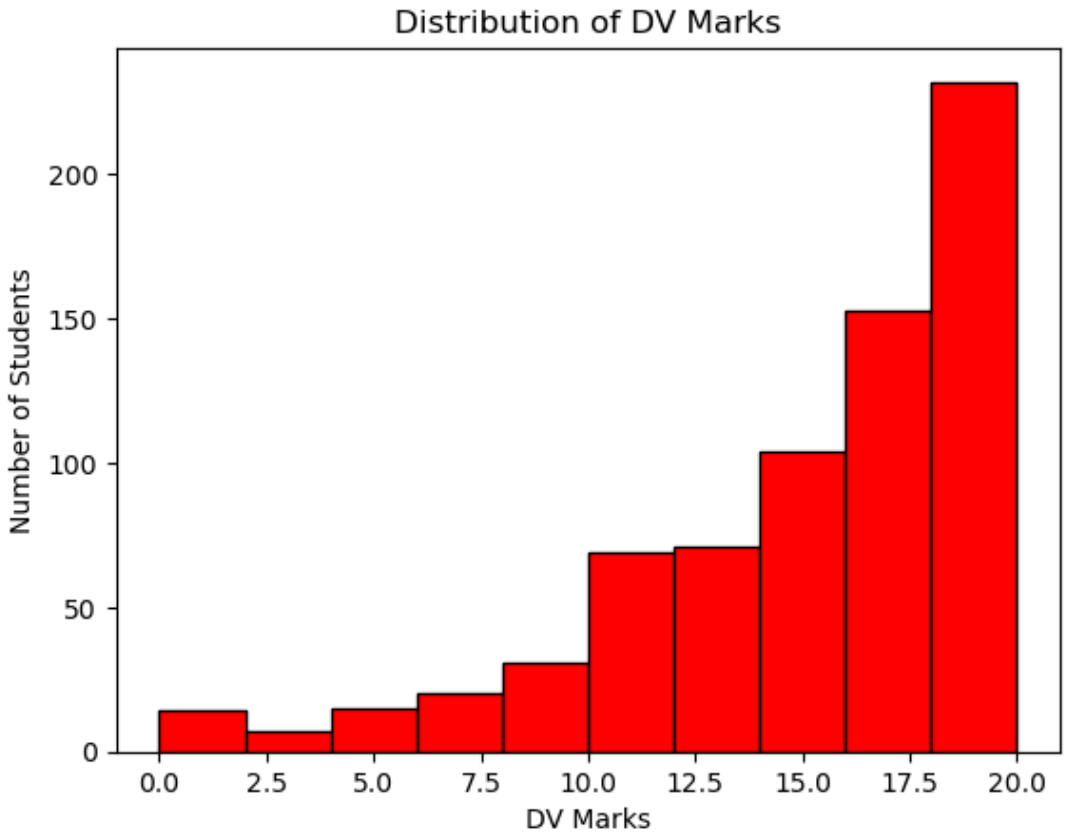


In [13]:
```
plt.figure(figsize=[8, 4])
plt.hist(df['Total'], color='cyan', bins=10, edgecolor='black')
plt.title("Distribution of Total Marks")
plt.xlabel("Total Marks")
plt.ylabel("Number of Students")
plt.show()
```



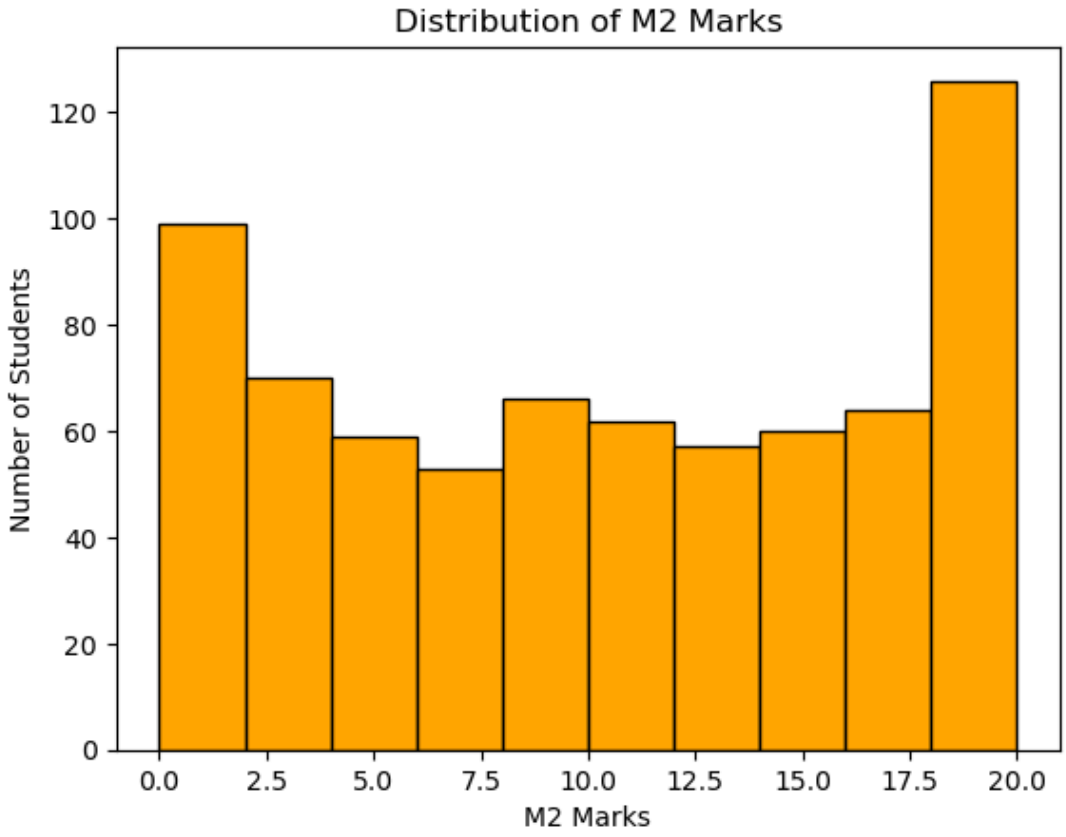
Distribution of Total marks in Histogram

In [14]:
```
df['DV'] = pd.to_numeric(df['DV'], errors='coerce')
df = df.dropna(subset=['DV'])
plt.hist(df['DV'], bins=10, color='red', edgecolor='black')
plt.title("Distribution of DV Marks")
plt.xlabel("DV Marks")
plt.ylabel("Number of Students")
plt.show()
```



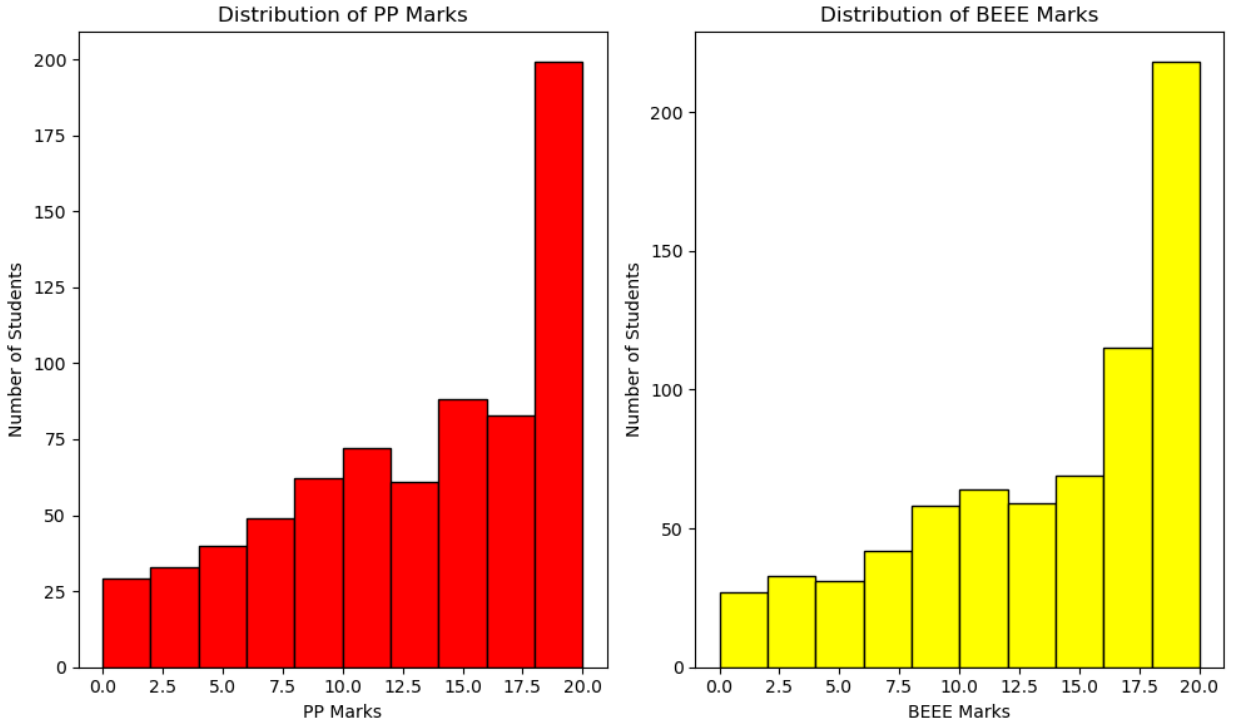
Distribution of DV Marks

In [15]:
```
df['M2'] = pd.to_numeric(df['M2'], errors='coerce')
df = df.dropna(subset=['M2'])
plt.hist(df['M2'], bins=10, color='orange', edgecolor='black')
plt.title("Distribution of M2 Marks")
plt.xlabel("M2 Marks")
plt.ylabel("Number of Students")
plt.show()
```



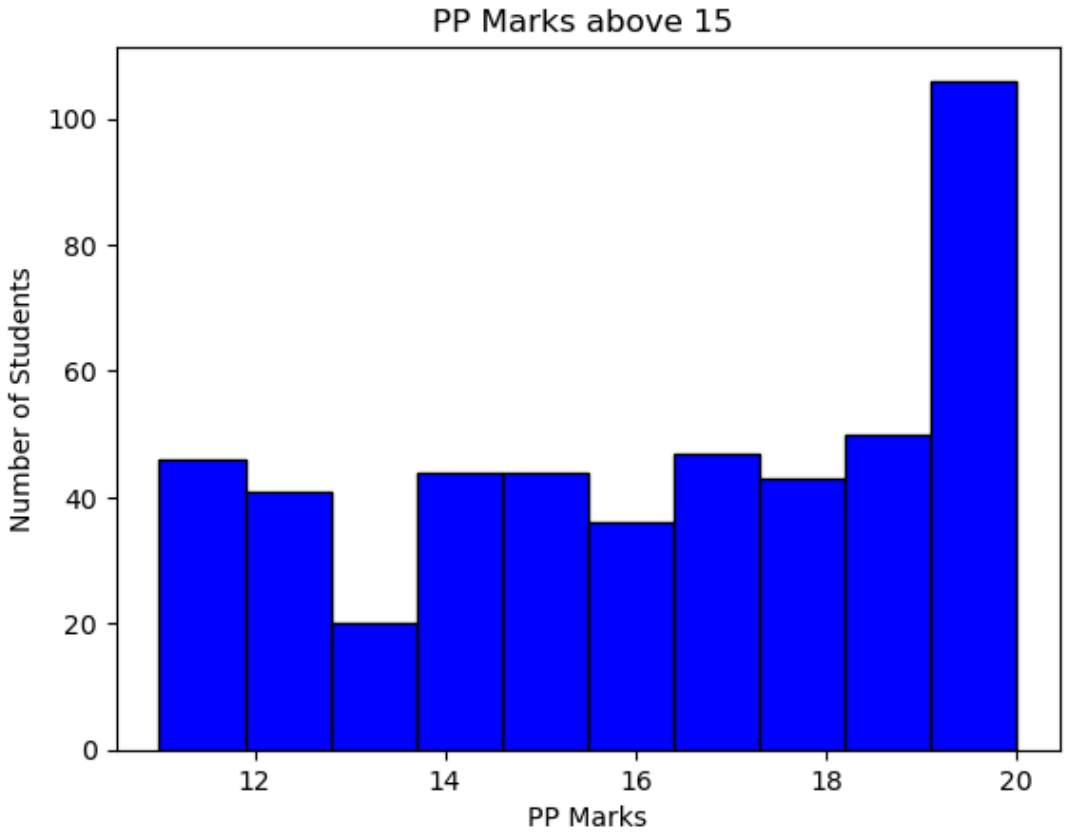
Distribution of M2 Marks

In [16]:
```
df['PP'] = pd.to_numeric(df['PP'], errors='coerce')
df['BEEE'] = pd.to_numeric(df['BEEE'], errors='coerce')
df_clean = df.dropna(subset=['PP', 'BEEE'])

plt.figure(figsize=[10, 6])

plt.subplot(1, 2, 1)
plt.hist(df_clean['PP'], bins=10, color='red', edgecolor='black')
plt.title("Distribution of PP Marks")
plt.xlabel("PP Marks")
plt.ylabel("Number of Students")

plt.subplot(1, 2, 2)
plt.hist(df_clean['BEEE'], bins=10, color='yellow', edgecolor='black')
plt.title("Distribution of BEEE Marks")
plt.xlabel("BEEE Marks")
plt.ylabel("Number of Students")

plt.tight_layout()
plt.show()
```



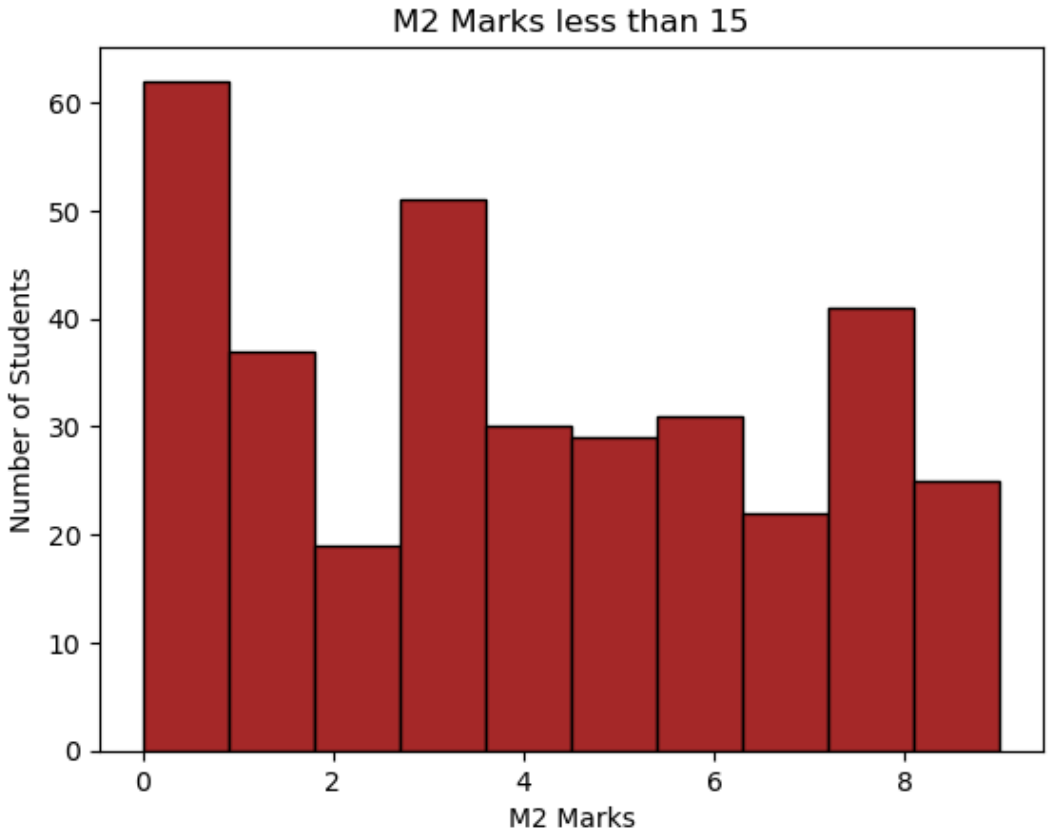
Comparision of PP marks and BEEE marks

In [17]:
```
filtered_df = df[df['PP'] > 10]
plt.hist(filtered_df['PP'], bins=10, color='blue', edgecolor='black')
plt.title("PP Marks above 15 ")
plt.xlabel("PP Marks")
plt.ylabel("Number of Students")
plt.show()
```



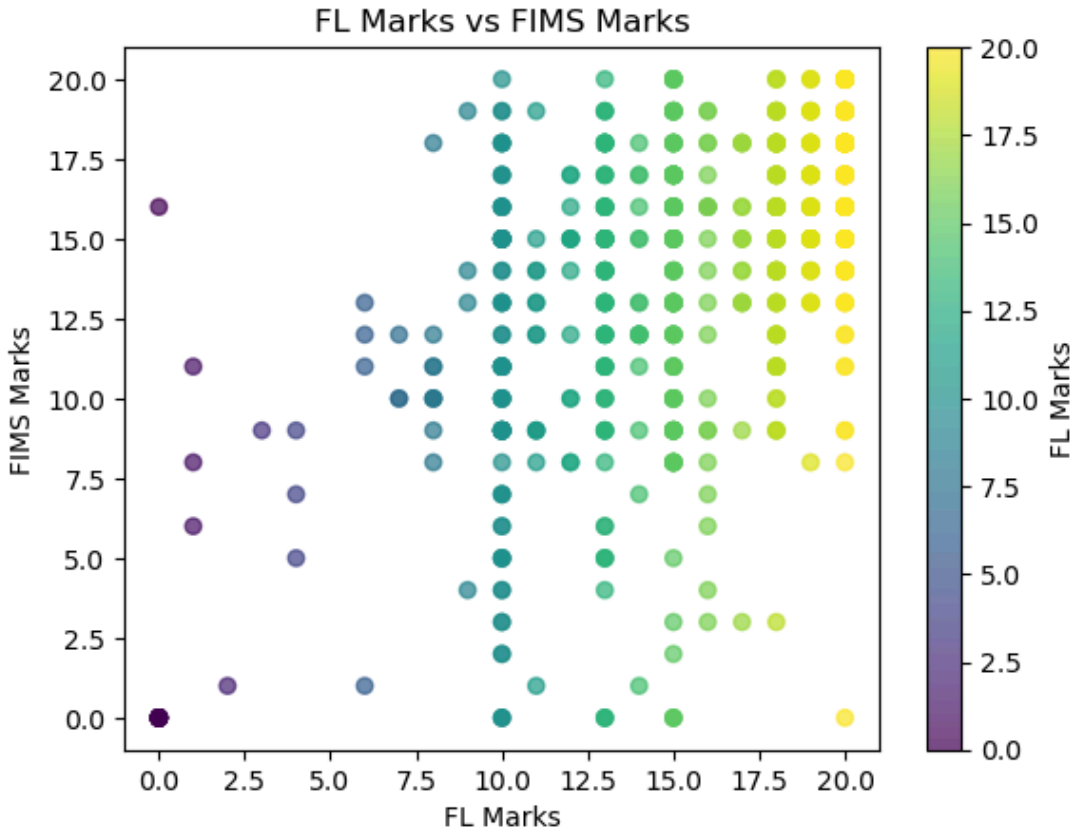
PP Marks who got more than 10

In [18]:
```
filtered_df = df[df['M2'] < 10]

plt.hist(filtered_df['M2'], bins=10, color='brown', edgecolor='black')
plt.title("M2 Marks less than 15")
plt.xlabel("M2 Marks")
plt.ylabel("Number of Students")
plt.show()
```



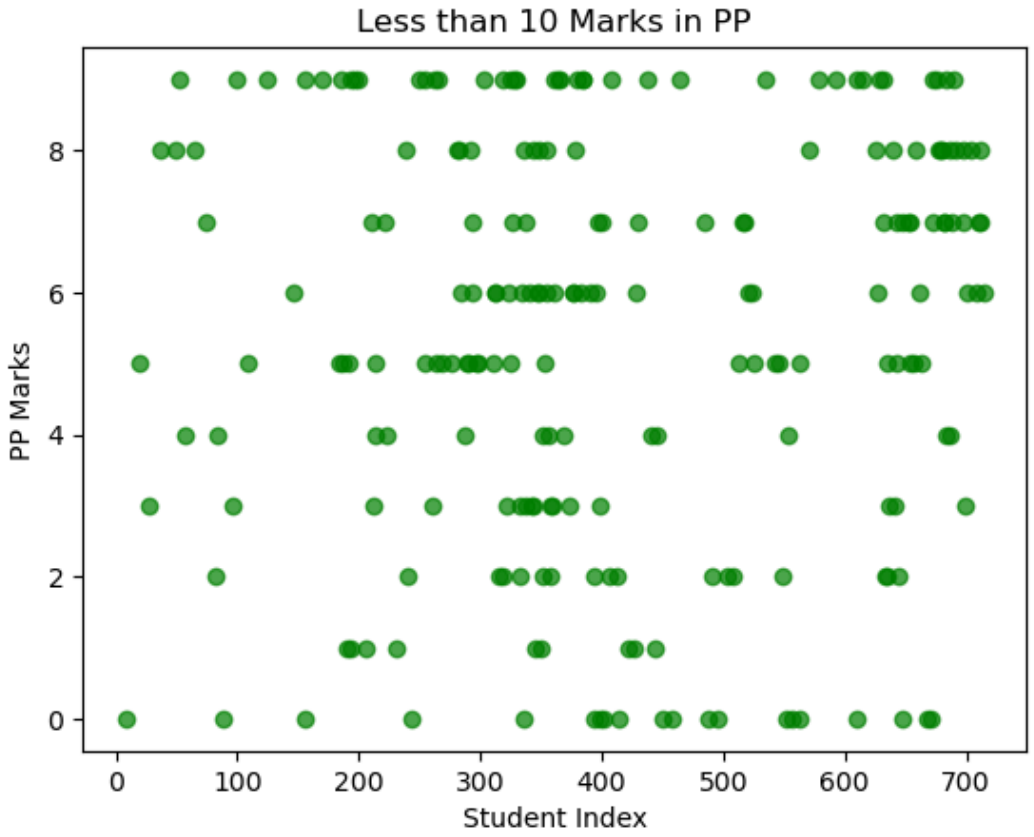
M2 Marks Who got less than 10

In [19]:
```
plt.scatter(df['FL'], df['FIMS'], c=df['FL'], cmap='viridis', alpha=0.7)
plt.title("FL Marks vs FIMS Marks")
plt.xlabel("FL Marks")
plt.ylabel("FIMS Marks")
plt.colorbar(label='FL Marks')
plt.show()
```



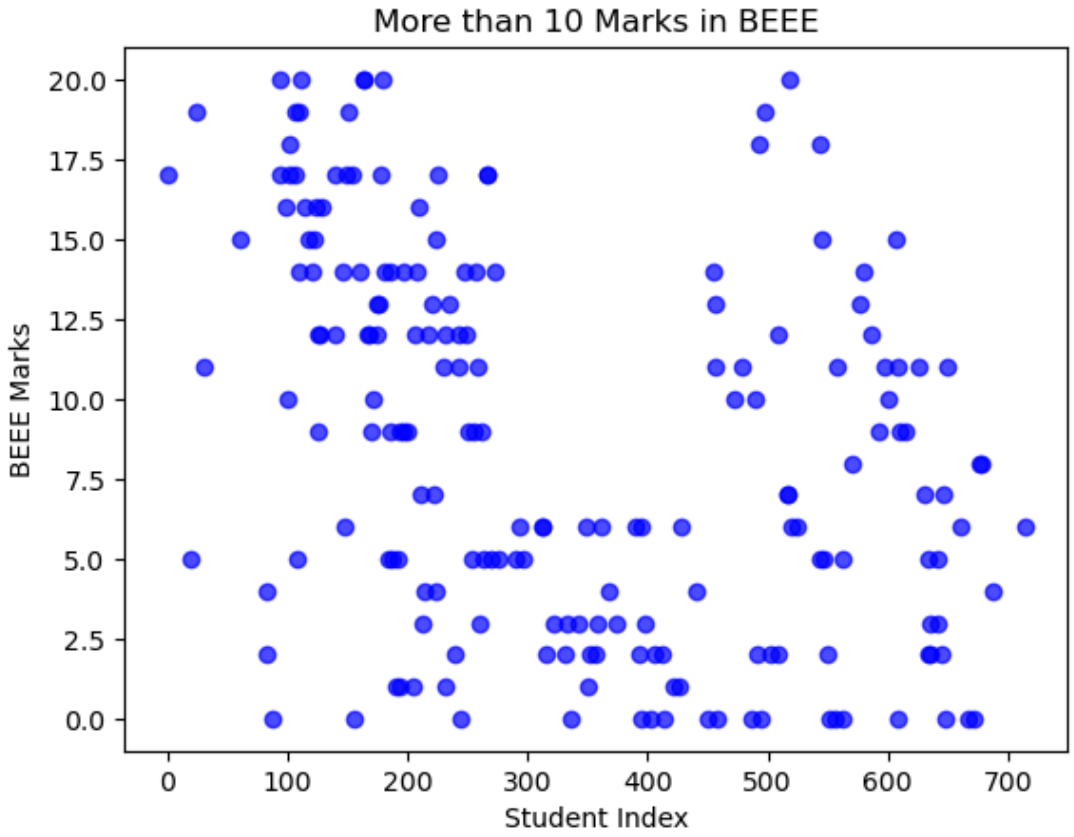
Scatter plot of FL VS FIMS Marks

In [20]:
```
filtered_df = df[df['PP'] < 10]

plt.scatter(filtered_df.index, filtered_df['PP'], alpha=0.7, color='green')
plt.title("Less than 10 Marks in PP")
plt.xlabel("Student Index")
plt.ylabel("PP Marks")
plt.show()
```



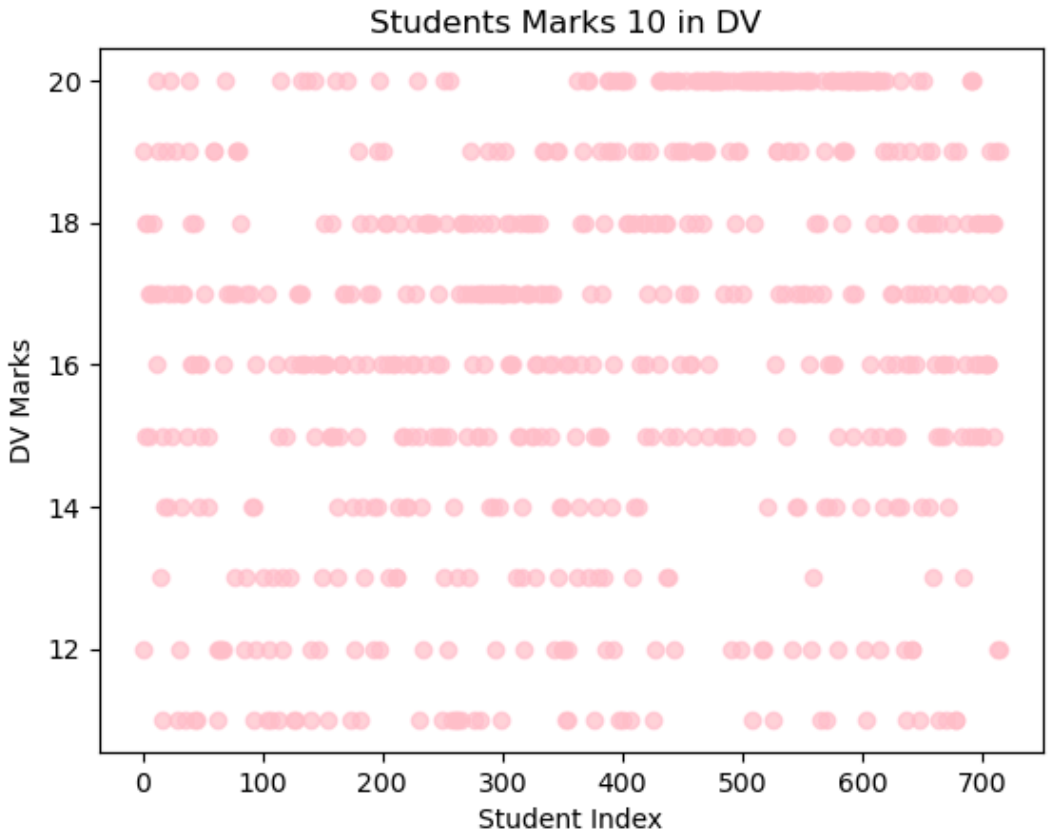
Scoring of PP Marks Less than 10

In [21]:
```
filtered_df = df[df['BEEE'] < 10]

plt.scatter(filtered_df.index, filtered_df['PP'], alpha=0.7, color='blue')
plt.title("More than 10 Marks in BEEE")
plt.xlabel("Student Index")
plt.ylabel("BEEE Marks")
plt.show()
```



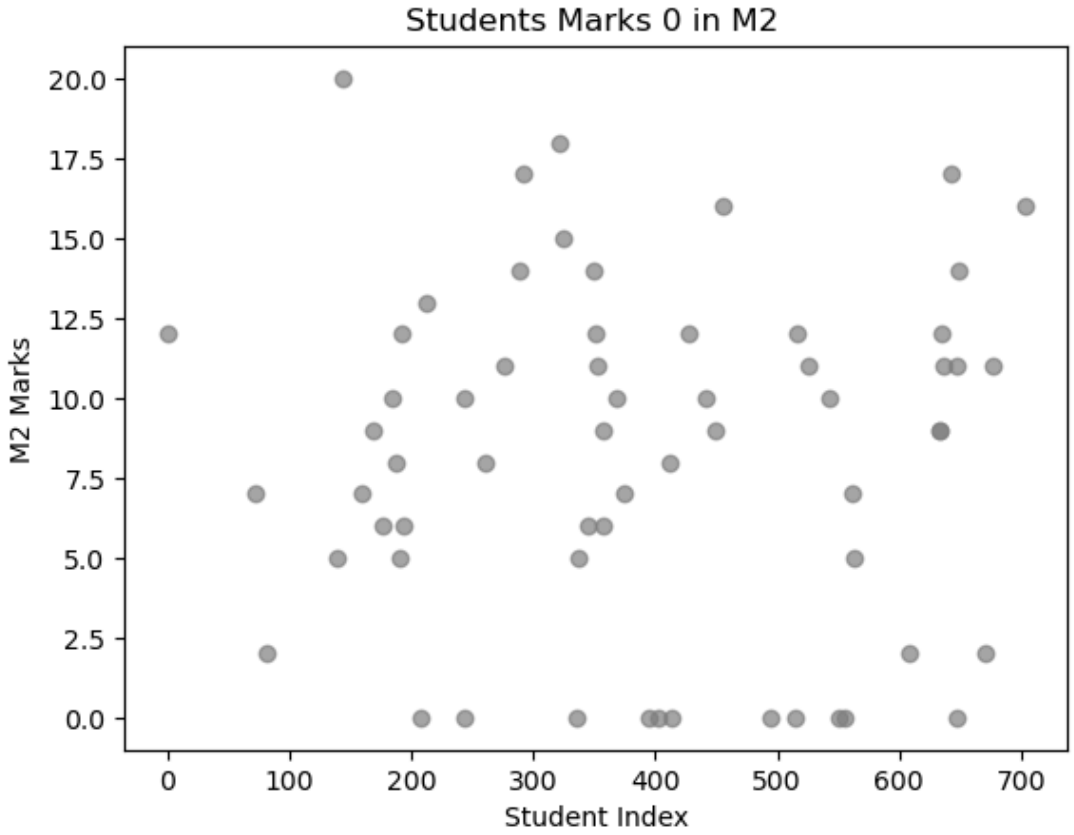
Scoring of BEEE Marks More than 15

In [22]:
```
filtered_df = df[df['DV'] > 10]
plt.scatter(filtered_df.index, filtered_df['DV'], alpha=0.7, color='pink')
plt.title("Students Marks 10 in DV ")
plt.xlabel("Student Index")
plt.ylabel("DV Marks")
plt.show()
```



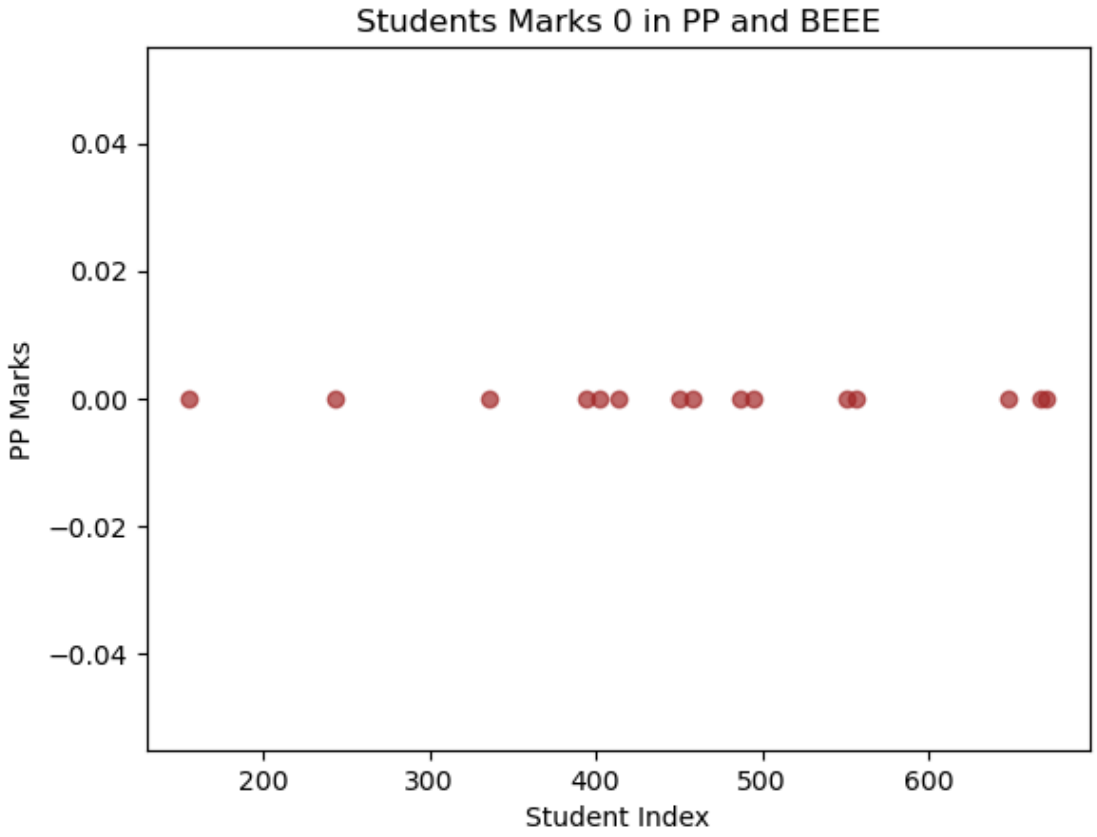
Students who scored 10 marks in DV

In [23]:
```
filtered_df = df[df['M2'] == 0]
plt.scatter(filtered_df.index, filtered_df['DV'], alpha=0.7, color='grey')
plt.title("Students Marks 0 in M2 ")
plt.xlabel("Student Index")
plt.ylabel("M2 Marks")
plt.show()
```



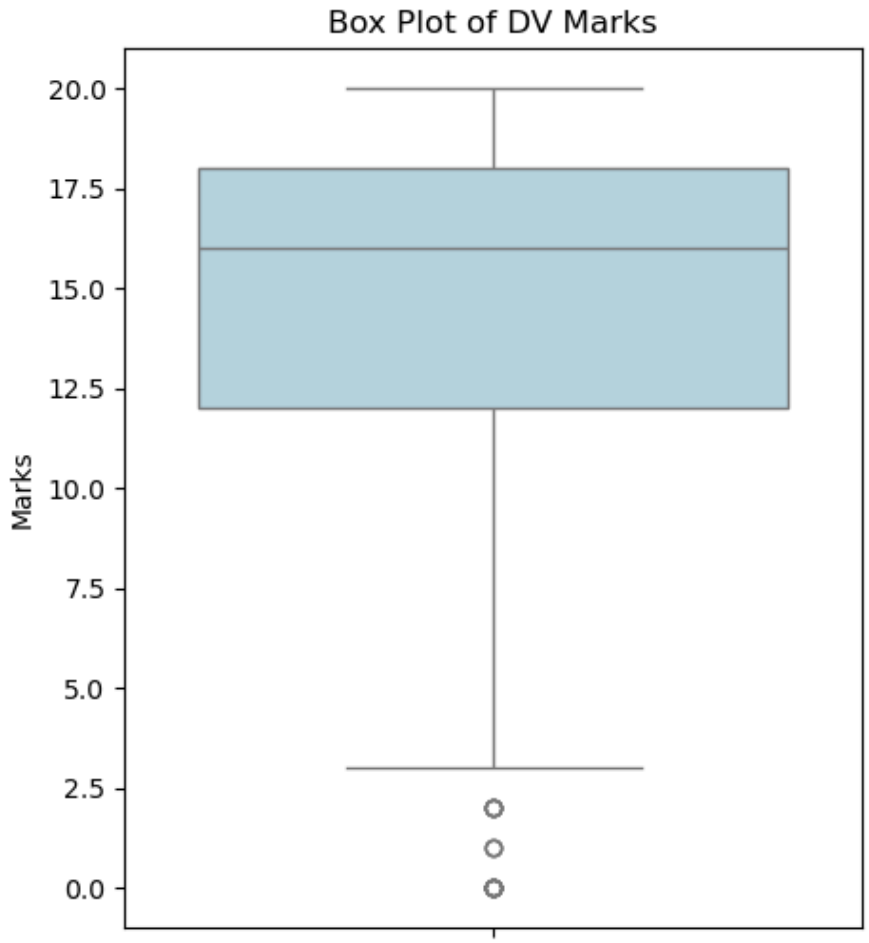
Students who scored 0 Marks in M2

In [24]:
```
filtered_df = df[(df['PP'] == 0) & (df['BEEE'] == 0)]
plt.scatter(filtered_df.index, filtered_df['PP'], alpha=0.7, color='brown')
plt.title("Students Marks 0 in PP and BEEE")
plt.xlabel("Student Index")
plt.ylabel("PP Marks")
plt.show()
```



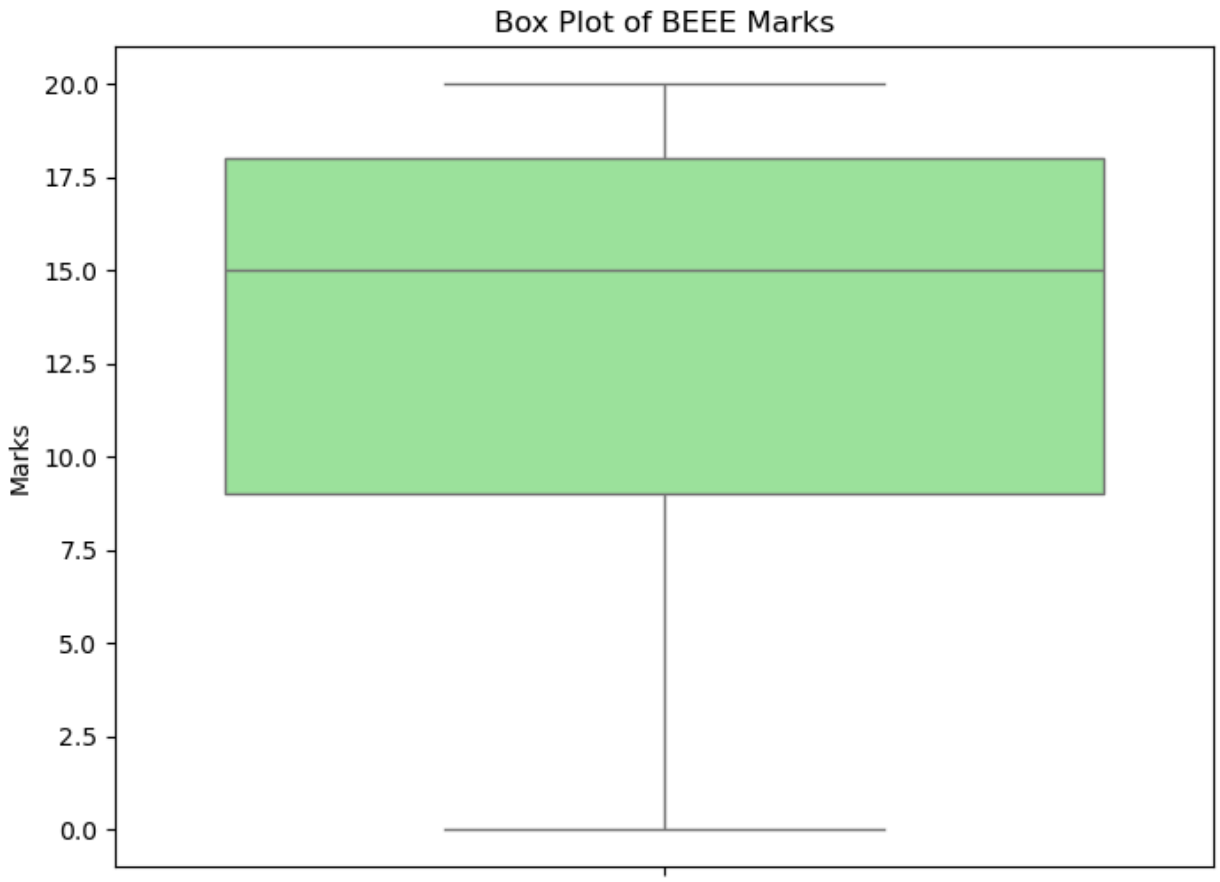
Students scored 0 Marks in PP and BEEE

In [25]:
```
plt.figure(figsize=(5, 6))
sns.boxplot(data=df['DV'], color='lightblue')
plt.title("Box Plot of DV Marks")
plt.ylabel("Marks")
plt.show()
```



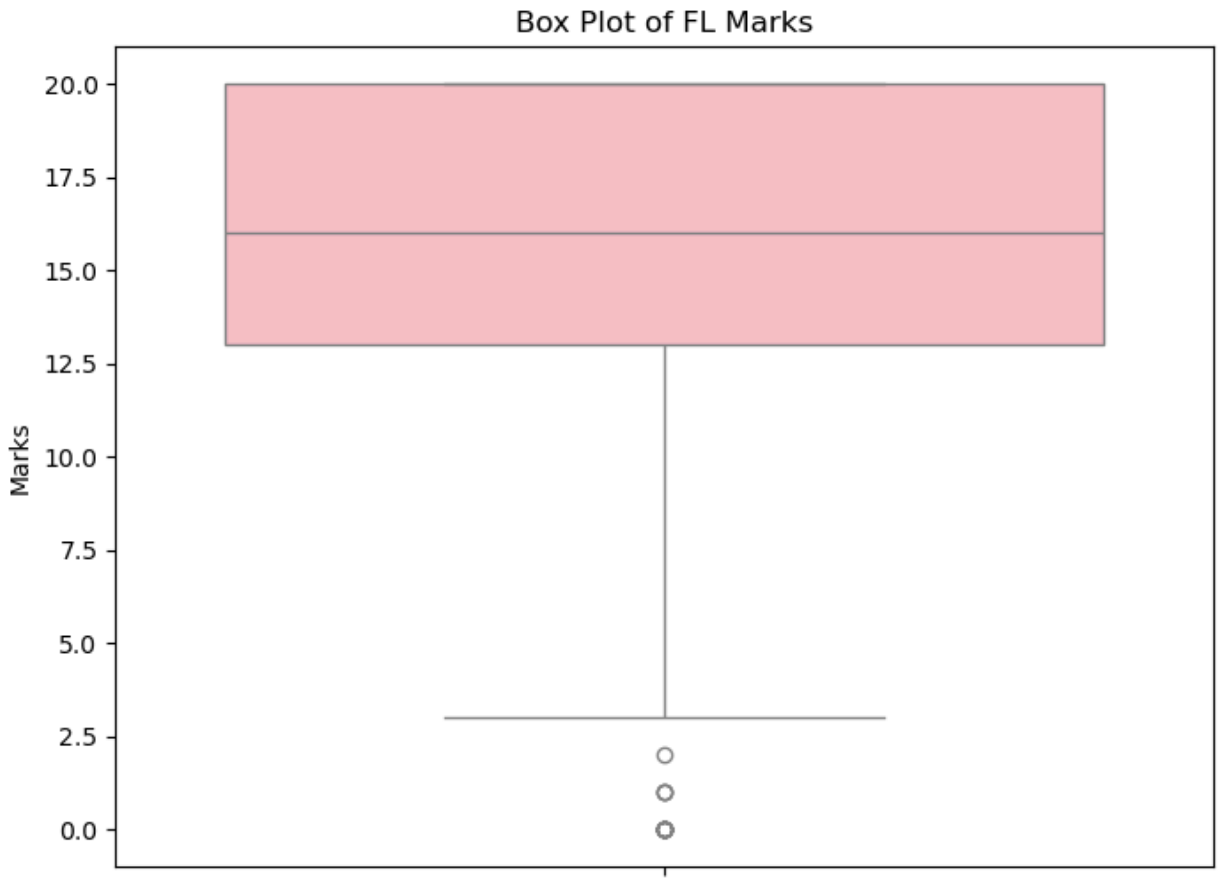
Box plot for the DV Marks

In [26]:
```
plt.figure(figsize=(8, 6))
sns.boxplot(data=df['BEEE'], color='lightgreen')
plt.title("Box Plot of BEEE Marks")
plt.ylabel("Marks")
plt.show()
```



Box plot of BEEE Marks

In [27]:
```
plt.figure(figsize=(8, 6))
sns.boxplot(data=df['FL'], color='lightpink')
plt.title("Box Plot of FL Marks")
plt.ylabel("Marks")
plt.show()
```



Box plot for the FL Marks

In [28]:
```
a=df.loc[(df['Total'] >= 75) & (df['Total'] <= 80)]
a=a.reset_index()
a
```

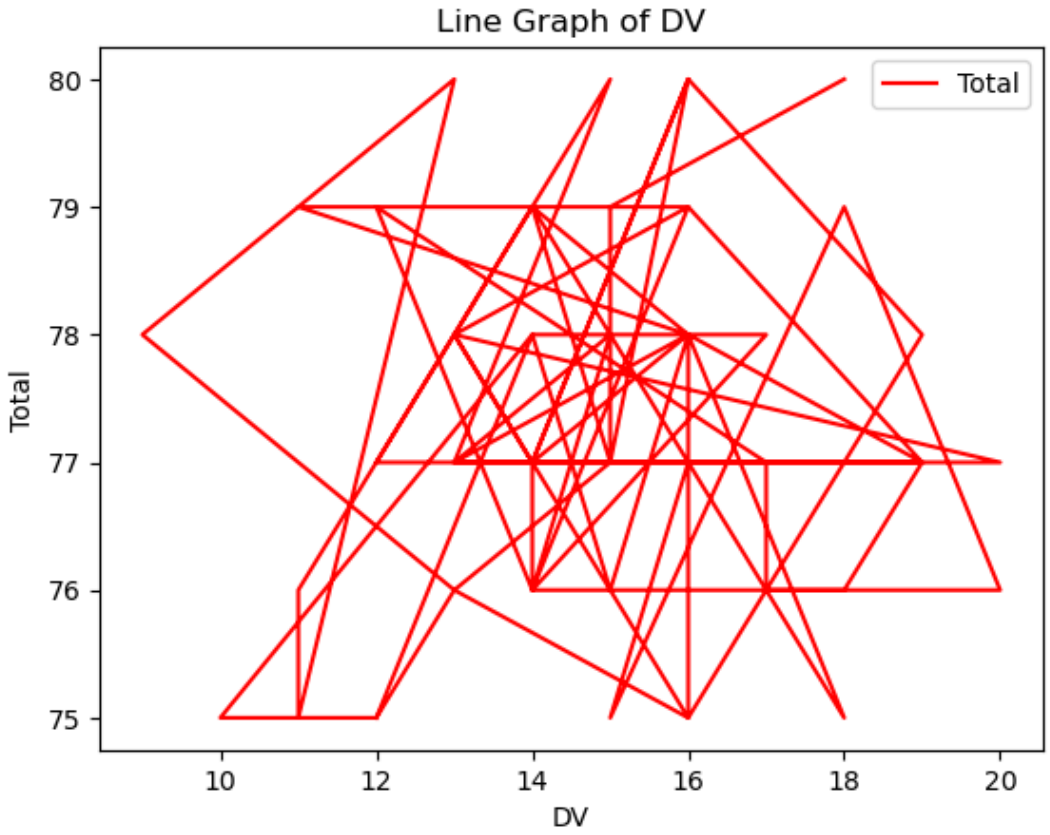
Out[28]:

	index	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentag
0	31	32	32	12.0	2	17.0	11	18.0	15	value	0.0	75.0	6
1	33	34	34	14.0	10	17.0	12	13.0	12	value	0.0	78.0	6
2	56	57	57	10.0	17	12.0	17	10.0	9	value	0.0	75.0	6
3	67	68	68	12.0	6	13.0	20	15.0	9	value	0.0	75.0	6
4	101	102	102	13.0	12	18.0	4	18.0	11	value	0.0	76.0	6
5	106	107	107	9.0	13	17.0	6	18.0	15	value	0.0	78.0	6
6	109	110	110	13.0	12	19.0	4	18.0	14	value	0.0	80.0	6
7	114	115	115	11.0	14	16.0	4	18.0	12	value	0.0	75.0	6
8	126	127	127	11.0	14	12.0	7	15.0	17	value	0.0	76.0	6
9	143	144	144	15.0	5	16.0	19	10.0	15	value	0.0	80.0	6
10	149	150	150	13.0	16	17.0	7	13.0	11	value	0.0	77.0	6
11	164	165	165	15.0	10	20.0	8	10.0	15	value	0.0	78.0	6
12	193	194	194	14.0	14	9.0	7	15.0	17	value	0.0	76.0	6
13	213	214	214	14.0	13	5.0	12	13.0	20	value	0.0	77.0	6
14	216	217	217	16.0	8	18.0	11	13.0	14	value	0.0	80.0	6
15	217	218	218	15.0	5	12.0	9	18.0	18	value	0.0	77.0	6
16	221	222	222	14.0	5	13.0	9	20.0	18	value	0.0	79.0	6
17	242	243	243	18.0	6	12.0	8	15.0	16	value	0.0	75.0	6
18	248	249	249	16.0	6	14.0	13	13.0	16	value	0.0	78.0	6
19	249	250	250	15.0	6	12.0	9	15.0	19	value	0.0	76.0	6
20	259	260	260	14.0	8	12.0	13	13.0	18	value	0.0	78.0	6
21	278	279	279	17.0	8	11.0	15	15.0	12	value	0.0	78.0	6
22	293	294	294	14.0	9	7.0	14	16.0	16	value	0.0	76.0	6
23	330	331	331	18.0	4	9.0	10	20.0	15	value	0.0	76.0	6
24	334	335	335	19.0	2	6.0	20	17.0	13	value	0.0	77.0	6
25	338	339	339	16.0	3	7.0	16	20.0	16	value	0.0	78.0	6
26	341	342	342	16.0	4	12.0	17	15.0	11	value	0.0	75.0	6
27	344	345	345	19.0	1	8.0	19	18.0	13	value	0.0	78.0	6
28	366	367	367	16.0	5	10.0	20	18.0	11	value	0.0	80.0	6
29	378	379	379	14.0	9	8.0	18	13.0	15	value	0.0	77.0	6
30	408	409	409	13.0	8	11.0	17	13.0	16	value	0.0	78.0	6
31	415	416	416	16.0	14	11.0	18	15.0	5	value	0.0	79.0	6
32	425	426	426	11.0	15	11.0	19	15.0	8	value	0.0	79.0	6
33	430	431	431	16.0	11	7.0	17	13.0	14	value	0.0	78.0	6
34	437	438	438	13.0	5	9.0	18	15.0	17	value	0.0	77.0	6
35	457	458	458	16.0	12	13.0	4	15.0	17	value	0.0	77.0	6
36	459	460	460	15.0	4	15.0	11	11.0	19	value	0.0	75.0	6
37	460	461	461	18.0	2	12.0	18	12.0	17	value	0.0	79.0	6

	index	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentag
38	518	519	519	20.0	1	20.0	6	14.0	15	value	0.0	76.0	6
39	530	531	531	17.0	4	15.0	10	17.0	13	value	0.0	76.0	6
40	535	536	536	17.0	2	11.0	13	20.0	14	value	0.0	77.0	6
41	541	542	542	12.0	8	14.0	14	13.0	18	value	0.0	79.0	6
42	571	572	572	14.0	1	14.0	10	20.0	17	value	0.0	76.0	6
43	577	578	578	16.0	11	13.0	8	20.0	11	value	0.0	79.0	6
44	584	585	585	19.0	7	11.0	12	15.0	13	value	0.0	77.0	6
45	615	616	616	12.0	12	15.0	14	13.0	11	value	0.0	77.0	6
46	617	618	618	14.0	6	17.0	14	10.0	18	value	0.0	79.0	6
47	620	621	621	16.0	3	11.0	17	15.0	16	value	0.0	78.0	6
48	629	630	630	14.0	4	9.0	16	19.0	15	value	0.0	77.0	6
49	631	632	632	20.0	6	7.0	9	16.0	19	value	0.0	77.0	6
50	658	659	659	13.0	2	8.0	18	20.0	17	value	0.0	78.0	6
51	673	674	674	16.0	2	9.0	17	17.0	14	value	0.0	75.0	6
52	684	685	685	13.0	9	9.0	15	14.0	16	value	0.0	76.0	6
53	689	690	690	15.0	3	9.0	16	16.0	18	value	0.0	77.0	6
54	700	701	701	15.0	8	6.0	18	17.0	15	value	0.0	79.0	6
55	706	707	707	18.0	1	10.0	19	16.0	16	value	0.0	80.0	6

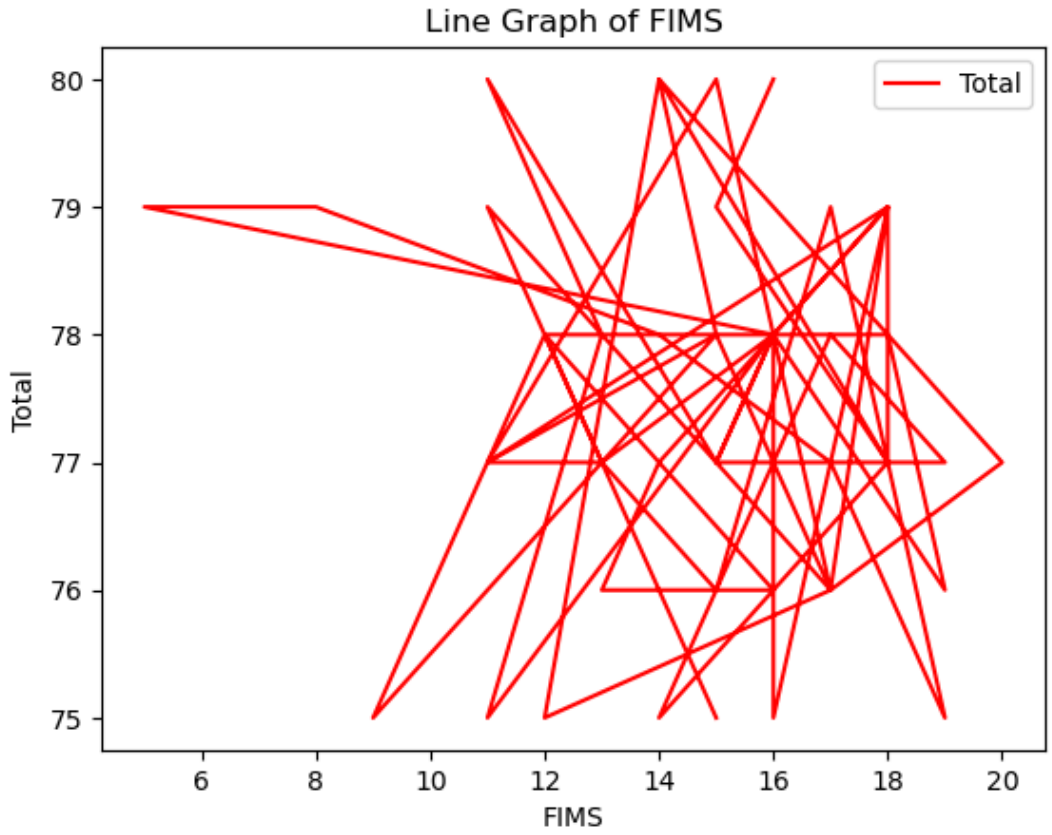
Total marks between 75 to 80

In [29]:
```
a.plot.line(x='DV',y='Total',color='red')
plt.title("Line Graph of DV")
plt.ylabel("Total")
plt.show()
```



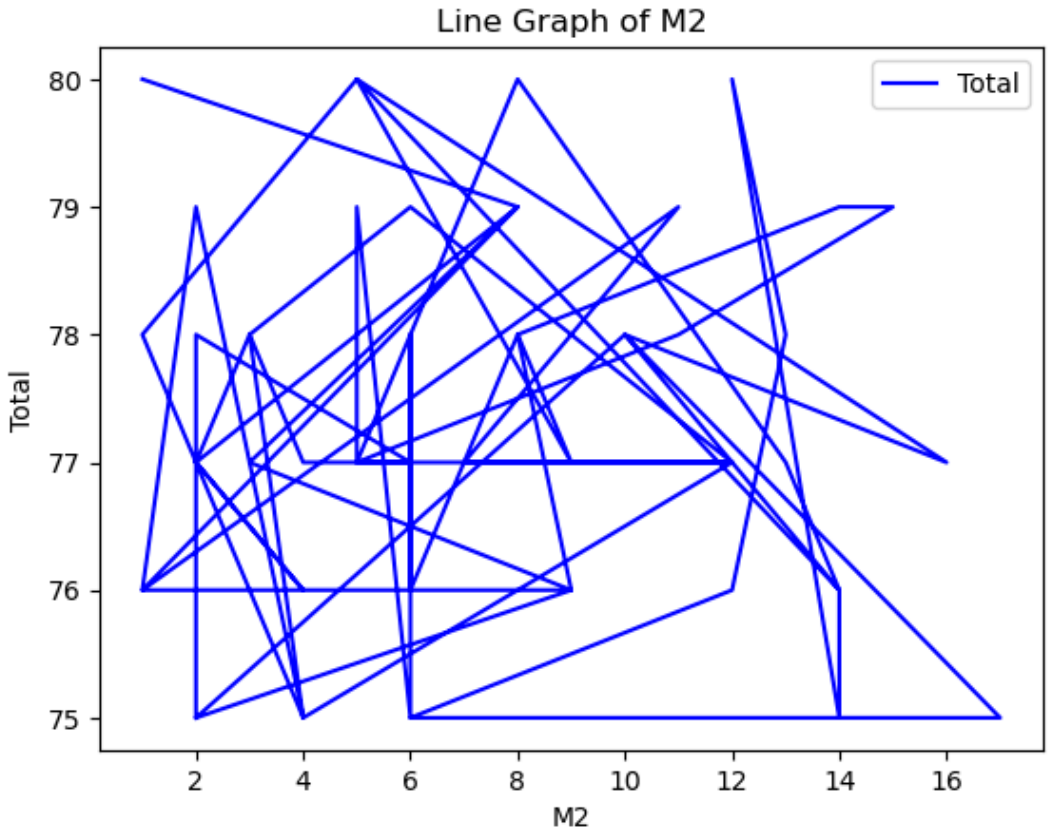
Lineplot for total and DV

In [30]:
```
a.plot.line(x='FIMS',y='Total',color='red')
plt.title("Line Graph of FIMS")
plt.ylabel("Total")
plt.show()
```



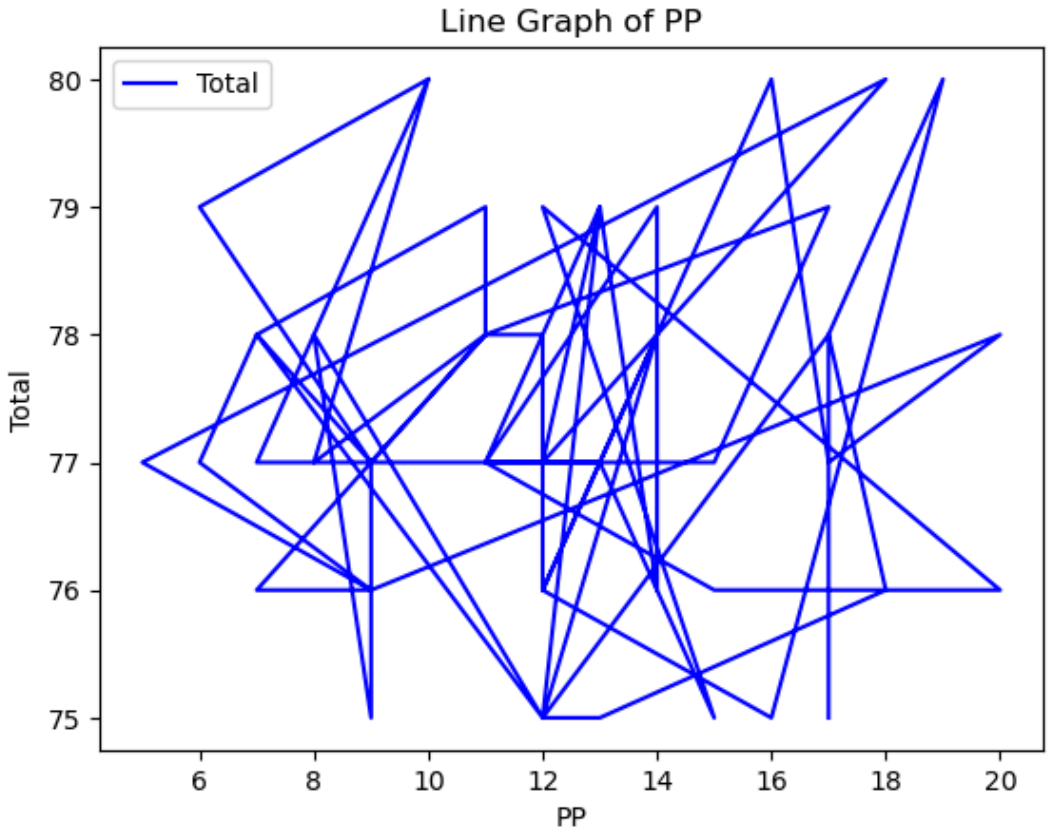
FIMS Line plot for Total marks

In [31]:
```
a.plot.line(x='M2',y='Total',color='blue')
plt.title("Line Graph of M2")
plt.ylabel("Total")
plt.show()
```



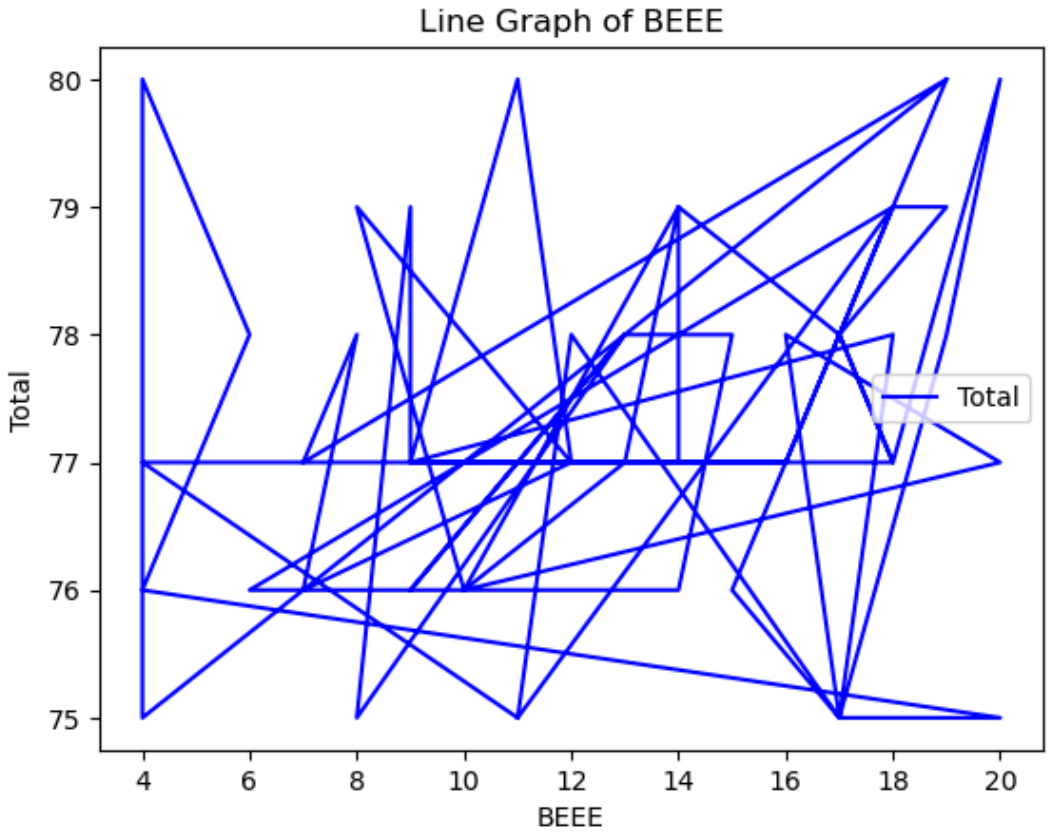
Lineplot for total and M2

In [32]:
```
a.plot.line(x='PP',y='Total',color='blue')
plt.title("Line Graph of PP")
plt.ylabel("Total")
plt.show()
```



PP line plot for Total marks

In [33]:
```
a.plot.line(x='BEEE',y='Total',color='blue')
plt.title("Line Graph of BEEE")
plt.ylabel("Total")
plt.show()
```



Total BEEE marks line plot

In [34]:
```
b=df.loc[(df['Total'] >= 115) & (df['Total'] <= 120)]
b=b.reset_index()
b
```

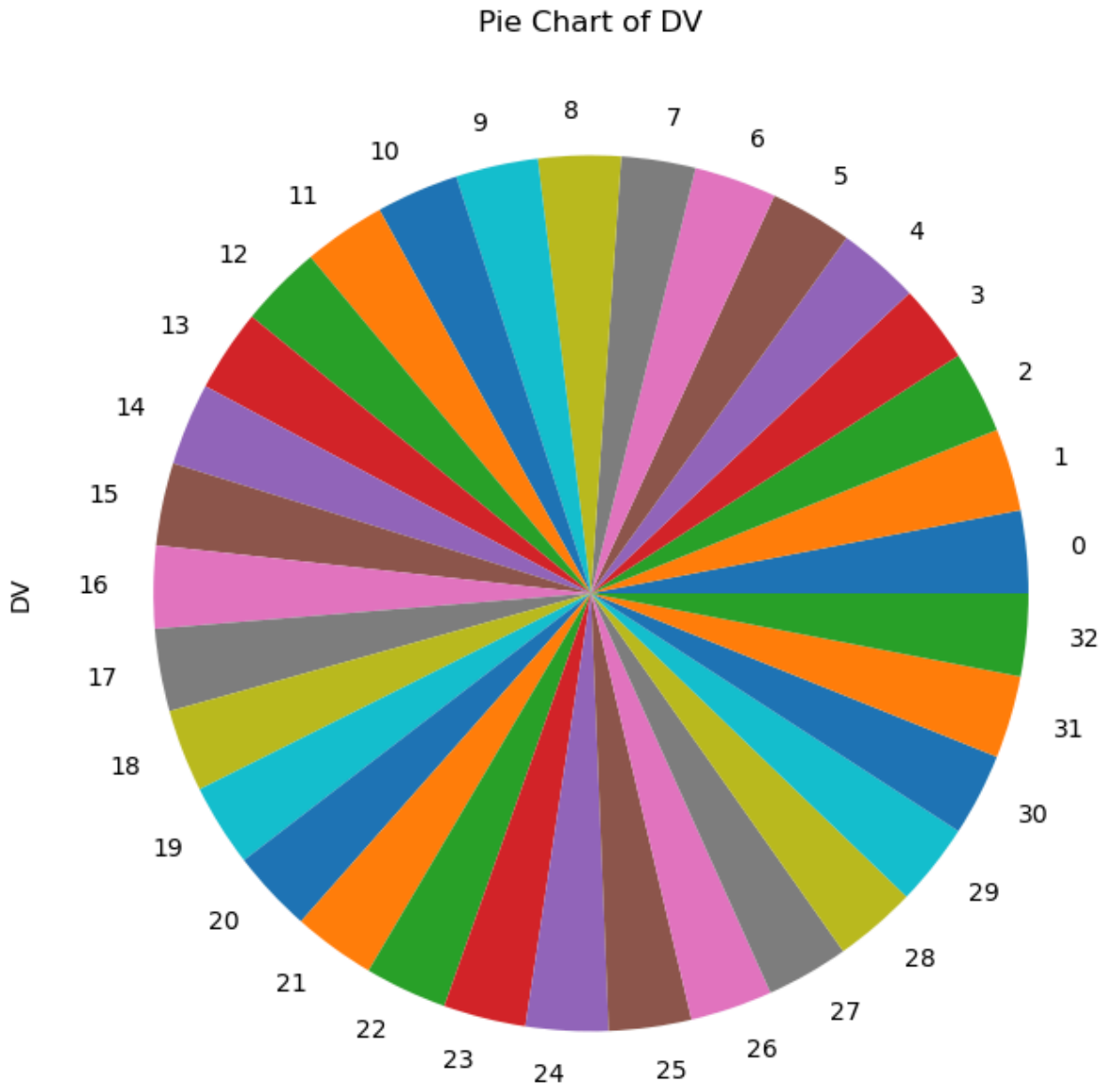
Out[34]:

	index	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentag
0	11	12	12	20.0	20	20.0	20	19.0	16	value	0.0	115.0	9
1	23	24	24	20.0	20	20.0	20	20.0	18	value	0.0	118.0	9
2	69	70	70	20.0	20	20.0	19	20.0	18	value	0.0	117.0	9
3	79	80	80	19.0	20	20.0	19	20.0	17	value	0.0	115.0	9
4	115	116	116	20.0	20	20.0	20	20.0	17	value	0.0	117.0	9
5	132	133	133	20.0	18	20.0	20	20.0	18	value	0.0	116.0	9
6	137	138	138	20.0	20	20.0	20	18.0	18	value	0.0	116.0	9
7	182	183	183	18.0	19	19.0	20	20.0	20	value	0.0	116.0	9
8	198	199	199	20.0	20	20.0	20	20.0	18	value	0.0	118.0	9
9	251	252	252	20.0	20	20.0	19	20.0	20	value	0.0	119.0	9
10	256	257	257	20.0	20	20.0	17	20.0	20	value	0.0	117.0	9
11	431	432	432	20.0	20	19.0	20	20.0	18	value	0.0	117.0	9
12	432	433	433	20.0	20	19.0	20	20.0	18	value	0.0	117.0	9
13	440	441	441	20.0	20	20.0	20	20.0	18	value	0.0	118.0	9
14	444	445	445	20.0	20	20.0	20	18.0	18	value	0.0	116.0	9
15	446	447	447	20.0	20	17.0	20	20.0	18	value	0.0	115.0	9
16	453	454	454	20.0	20	20.0	20	20.0	19	value	0.0	119.0	9
17	462	463	463	20.0	17	20.0	20	20.0	19	value	0.0	116.0	9
18	474	475	475	20.0	18	20.0	19	20.0	18	value	0.0	115.0	9
19	475	476	476	20.0	19	20.0	19	20.0	19	value	0.0	117.0	9
20	477	478	478	20.0	20	20.0	20	20.0	16	value	0.0	116.0	9
21	505	506	506	20.0	20	20.0	20	20.0	20	value	0.0	120.0	10
22	506	507	507	20.0	18	20.0	20	20.0	19	value	0.0	117.0	9
23	507	508	508	20.0	20	20.0	20	20.0	20	value	0.0	120.0	10
24	521	522	522	20.0	19	20.0	19	20.0	20	value	0.0	118.0	9
25	533	534	534	20.0	19	20.0	20	20.0	20	value	0.0	119.0	9
26	539	540	540	20.0	18	20.0	18	20.0	19	value	0.0	115.0	9
27	573	574	574	20.0	20	20.0	20	20.0	20	value	0.0	120.0	10
28	587	588	588	20.0	20	20.0	20	20.0	18	value	0.0	118.0	9
29	595	596	596	20.0	20	20.0	20	20.0	20	value	0.0	120.0	10
30	596	597	597	20.0	20	20.0	19	19.0	18	value	0.0	116.0	9
31	611	612	612	20.0	20	19.0	20	20.0	20	value	0.0	119.0	9
32	616	617	617	20.0	20	20.0	20	20.0	19	value	0.0	119.0	9

Students who scores max marks 115 to 120

In [35]:
```
b['DV'].plot(kind='pie',subplots=True,figsize=(8,8))
plt.title("Pie Chart of DV")
```

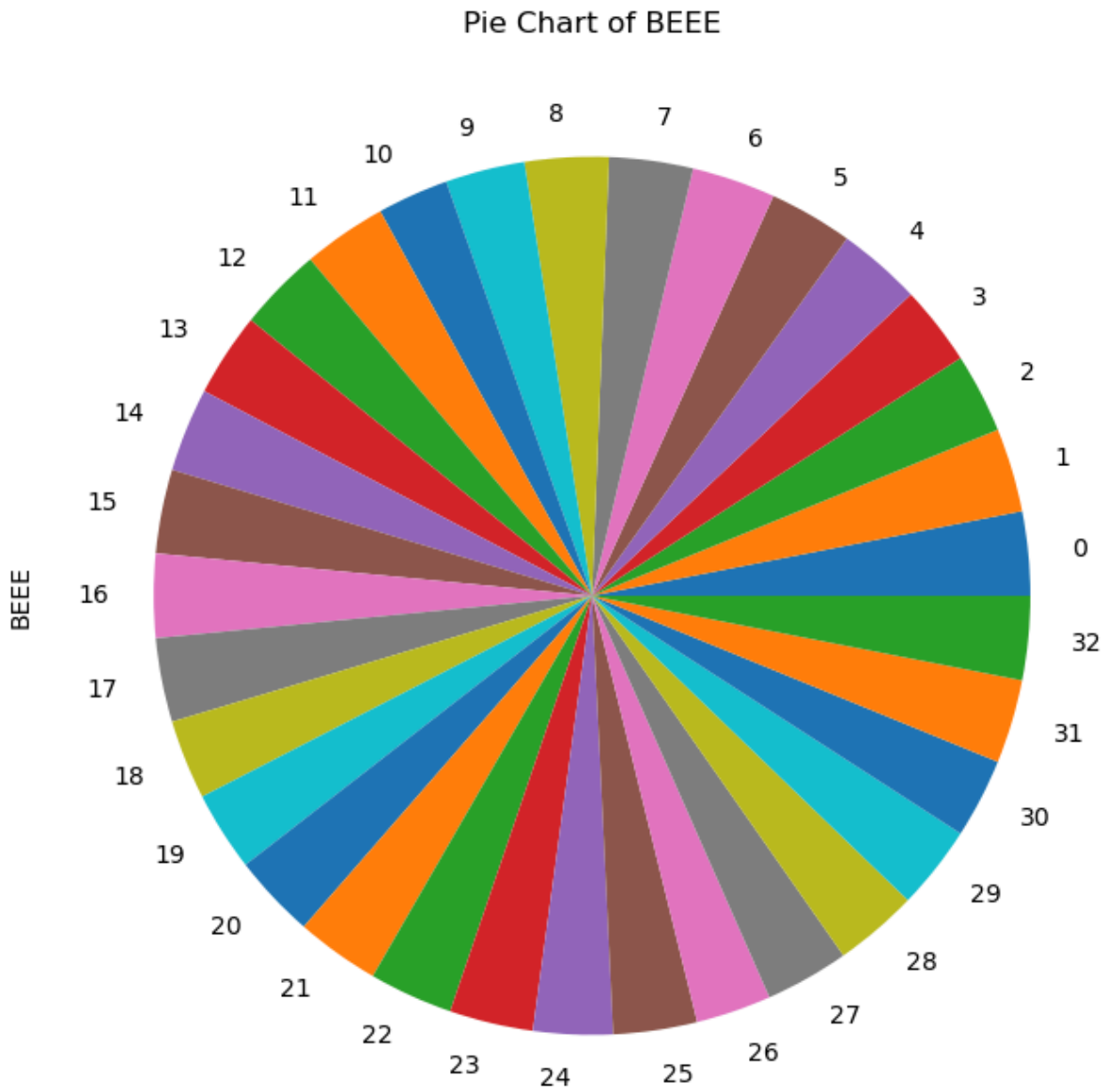
Out[35]:
```
Text(0.5, 1.0, 'Pie Chart of DV')
```



PIE Chart of DV for 32 students

In [36]:
```
b['BEEE'].plot(kind='pie',subplots=True,figsize=(8,8))
plt.title("Pie Chart of BEEE")
```

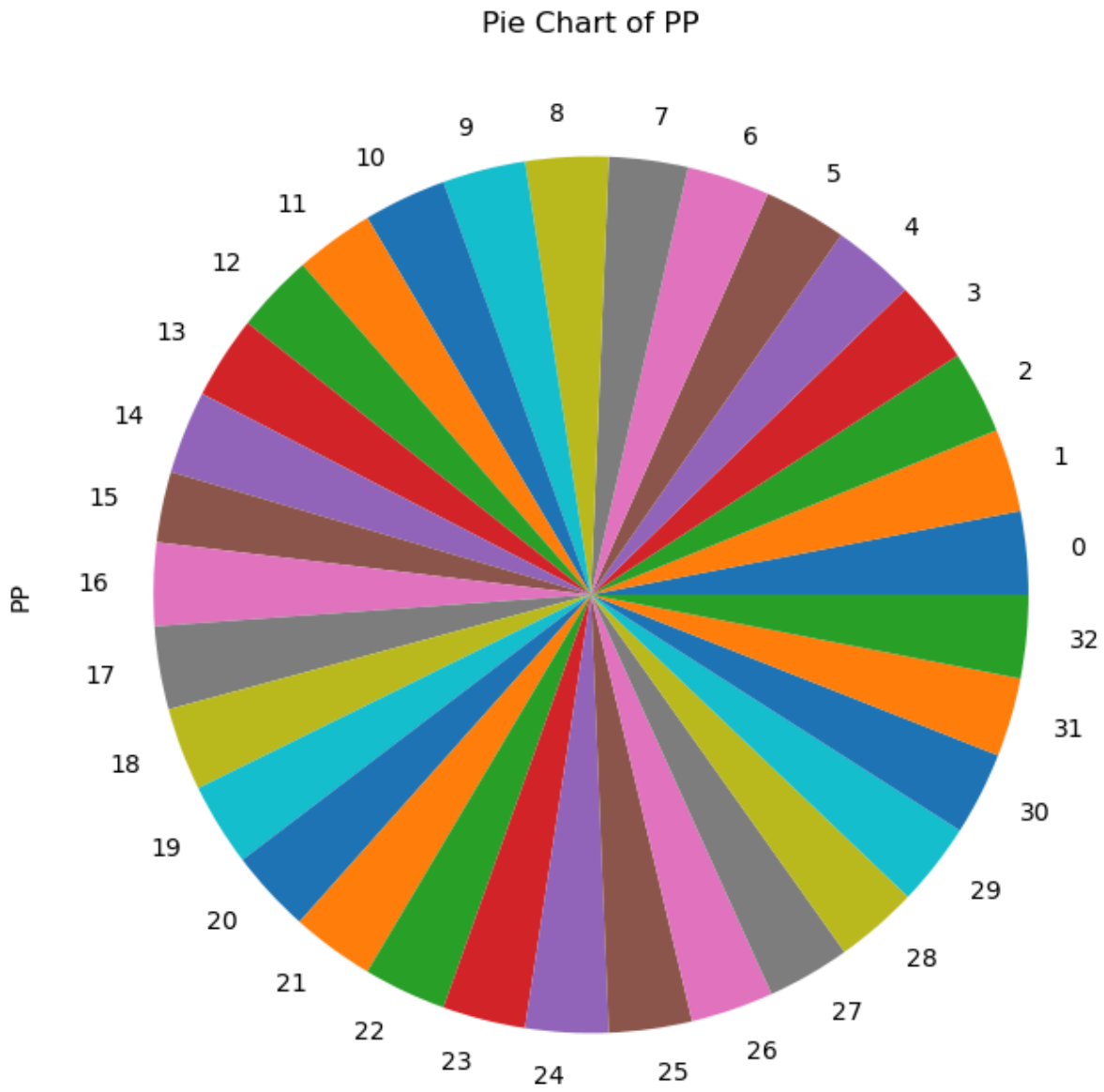
Out[36]: Text(0.5, 1.0, 'Pie Chart of BEEE')



32 students BEEE in PIE chart representation

In [37]:
```
b['PP'].plot(kind='pie',subplots=True,figsize=(8,8))
plt.title("Pie Chart of PP")
```

Out[37]:
```
Text(0.5, 1.0, 'Pie Chart of PP')
```



PP pie chart for 32 students

In [38]:
```
df.sort_values('Total').tail(10)
```

Out[38]:

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentage	Gr
198	199	199	20.0	20	20.0	20	20.0	18	value	0.0	118.0	98	
533	534	534	20.0	19	20.0	20	20.0	20	value	0.0	119.0	99	
616	617	617	20.0	20	20.0	20	20.0	19	value	0.0	119.0	99	
611	612	612	20.0	20	19.0	20	20.0	20	value	0.0	119.0	99	
453	454	454	20.0	20	20.0	20	20.0	19	value	0.0	119.0	99	
251	252	252	20.0	20	20.0	19	20.0	20	value	0.0	119.0	99	
505	506	506	20.0	20	20.0	20	20.0	20	value	0.0	120.0	100	
573	574	574	20.0	20	20.0	20	20.0	20	value	0.0	120.0	100	
595	596	596	20.0	20	20.0	20	20.0	20	value	0.0	120.0	100	
507	508	508	20.0	20	20.0	20	20.0	20	value	0.0	120.0	100	

In [39]:
```
df.sort_values('DV').tail(20)
```

Out[39]:

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentage	Gr
521	522	522	20.0	19	20.0	19	20.0	20	value	0.0	118.0	98	
522	523	523	20.0	14	19.0	19	20.0	18	value	0.0	110.0	92	
523	524	524	20.0	18	20.0	19	20.0	16	value	0.0	113.0	94	
389	390	390	20.0	15	14.0	16	20.0	12	value	0.0	97.0	81	
526	527	527	20.0	6	20.0	18	17.0	18	value	0.0	99.0	82	
388	389	389	20.0	4	16.0	18	19.0	14	value	0.0	91.0	76	
631	632	632	20.0	6	7.0	9	16.0	19	value	0.0	77.0	64	
531	532	532	20.0	6	14.0	19	20.0	15	value	0.0	94.0	78	
532	533	533	20.0	12	20.0	19	20.0	17	value	0.0	108.0	90	
533	534	534	20.0	19	20.0	20	20.0	20	value	0.0	119.0	99	
534	535	535	20.0	12	9.0	18	20.0	19	value	0.0	98.0	82	
537	538	538	20.0	11	17.0	18	20.0	14	value	0.0	100.0	83	
539	540	540	20.0	18	20.0	18	20.0	19	value	0.0	115.0	96	
542	543	543	20.0	11	20.0	17	18.0	17	value	0.0	103.0	86	
372	373	373	20.0	14	18.0	17	19.0	15	value	0.0	103.0	86	
619	620	620	20.0	20	20.0	16	18.0	17	value	0.0	111.0	92	
547	548	548	20.0	20	17.0	18	19.0	19	value	0.0	113.0	94	
616	617	617	20.0	20	20.0	20	20.0	19	value	0.0	119.0	99	
552	553	553	20.0	19	20.0	20	13.0	18	value	0.0	110.0	92	
646	647	647	20.0	20	11.0	20	20.0	19	value	0.0	110.0	92	

In [40]:
```
df.sort_values('DV').head(50)
```

Out[40]:

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentage	Grad
495	496	496	0.0	0	0.0	0	0.0	0	value	0.0	0.0	0	
244	245	245	0.0	0	0.0	0	0.0	0	value	0.0	0.0	0	
514	515	515	0.0	0	12.0	16	20.0	18	value	0.0	66.0	55	
336	337	337	0.0	0	0.0	0	0.0	0	value	0.0	0.0	0	
402	403	403	0.0	0	0.0	0	0.0	0	value	0.0	0.0	0	
414	415	415	0.0	0	0.0	0	0.0	0	value	0.0	0.0	0	
556	557	557	0.0	0	0.0	0	0.0	0	value	0.0	0.0	0	
395	396	396	0.0	0	6.0	7	13.0	0	value	0.0	26.0	22	
648	649	649	0.0	0	0.0	0	0.0	0	value	0.0	0.0	0	
551	552	552	0.0	0	0.0	0	0.0	0	value	0.0	0.0	0	
207	208	208	0.0	0	16.0	10	20.0	19	value	0.0	65.0	54	
50	51	51	1.0	16	15.0	13	10.0	11	value	0.0	66.0	55	
487	488	488	1.0	5	0.0	0	0.0	0	value	0.0	6.0	5	
503	504	504	1.0	1	2.0	0	10.0	0	value	0.0	14.0	12	
82	83	83	2.0	0	2.0	0	0.0	0	value	0.0	4.0	3	
88	89	89	2.0	17	0.0	3	15.0	2	value	0.0	39.0	32	
427	428	428	2.0	5	1.0	2	10.0	6	value	0.0	26.0	22	
671	672	672	2.0	0	0.0	0	2.0	1	value	0.0	5.0	4	
57	58	58	2.0	2	4.0	10	10.0	3	value	0.0	31.0	26	
609	610	610	2.0	0	0.0	3	10.0	9	value	0.0	24.0	20	
85	86	86	3.0	4	14.0	13	18.0	13	value	0.0	65.0	54	
549	550	550	4.0	3	2.0	6	10.0	8	value	0.0	33.0	28	
70	71	71	4.0	2	16.0	10	15.0	9	value	0.0	56.0	47	
20	21	21	4.0	2	5.0	3	16.0	9	value	0.0	39.0	32	
223	224	224	4.0	15	4.0	5	10.0	14	value	0.0	52.0	43	
337	338	338	5.0	0	3.0	11	7.0	10	value	0.0	36.0	30	
125	126	126	5.0	16	9.0	7	18.0	14	value	0.0	69.0	57	
75	76	76	5.0	8	7.0	15	10.0	2	value	0.0	47.0	39	
139	140	140	5.0	0	12.0	4	20.0	15	value	0.0	56.0	47	
360	361	361	5.0	3	9.0	10	10.0	7	value	0.0	44.0	37	
398	399	399	5.0	3	3.0	2	10.0	9	value	0.0	32.0	27	
190	191	191	5.0	0	1.0	1	10.0	5	value	0.0	22.0	18	
563	564	564	5.0	0	5.0	4	10.0	10	value	0.0	34.0	28	
683	684	684	5.0	1	4.0	15	1.0	6	value	0.0	32.0	27	
478	479	479	5.0	2	11.0	0	10.0	0	value	0.0	28.0	23	
27	28	28	5.0	4	3.0	12	13.0	5	value	0.0	42.0	35	
687	688	688	6.0	3	4.0	9	10.0	15	value	0.0	47.0	39	
458	459	459	6.0	1	0.0	0	0.0	0	value	0.0	7.0	6	

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentage	Grad
194	195	195	6.0	0	1.0	0	10.0	13	value	0.0	30.0	25	
345	346	346	6.0	0	1.0	11	9.0	4	value	0.0	31.0	26	
176	177	177	6.0	0	13.0	8	18.0	14	value	0.0	59.0	49	
51	52	52	6.0	12	10.0	11	10.0	3	value	0.0	52.0	43	
464	465	465	6.0	1	9.0	11	8.0	10	value	0.0	45.0	38	
25	26	26	6.0	10	10.0	11	13.0	10	value	0.0	60.0	50	
484	485	485	6.0	3	7.0	11	11.0	14	value	0.0	52.0	43	
358	359	359	6.0	0	3.0	3	10.0	4	value	0.0	26.0	22	
240	241	241	6.0	6	2.0	3	10.0	11	value	0.0	38.0	32	
98	99	99	6.0	7	16.0	9	13.0	13	value	0.0	64.0	53	
406	407	407	7.0	2	2.0	6	10.0	7	value	0.0	34.0	28	
127	128	128	7.0	4	12.0	4	13.0	11	value	0.0	51.0	42	

In [41]:
```
h = df[
    (df['DV'] < 10.0) |
    (df['PP'] < 10.0) |
    (df['M2'] < 10.0) |
    (df['BEEE'] < 10.0) |
    (df['FL'] < 10.0) |
    (df['FIMS'] < 10.0)
]
h['SECTION'].value_counts()
```

Out[41]:
```
SECTION
1      1
471    1
490    1
489    1
488    1
      ..
262    1
261    1
260    1
259    1
715    1
Name: count, Length: 443, dtype: int64
```

In [42]:
```
df['backlogs'] = (df[['DV', 'M2', 'PP', 'BEEE', 'FL', 'FIMS']] < 10).sum(axis=1)
df
```

Out[42]:

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentage	Gr
0	1	1	12.0	0	17.0	9	19.0	15	value	0.0	72.0	60	
1	2	2	19.0	12	16.0	16	18.0	3	value	0.0	84.0	70	
2	3	3	18.0	14	18.0	18	18.0	16	value	0.0	102.0	85	
3	4	4	15.0	9	19.0	17	19.0	15	value	0.0	94.0	78	
4	5	5	18.0	17	19.0	19	20.0	18	value	0.0	111.0	92	
...	...	...	...	...	...	...	...	...	...	...	...	...	
711	712	712	19.0	8	8.0	19	17.0	18	value	0.0	89.0	74	
712	713	713	12.0	1	7.0	10	20.0	8	value	0.0	58.0	48	
713	714	714	17.0	6	14.0	14	17.0	18	value	0.0	86.0	72	
714	715	715	12.0	1	6.0	7	15.0	12	value	0.0	53.0	44	
715	716	716	19.0	14	17.0	16	20.0	19	value	0.0	105.0	88	

716 rows × 14 columns

In [43]:
```
j=df.sort_values('backlogs')
j
```

Out[43]:

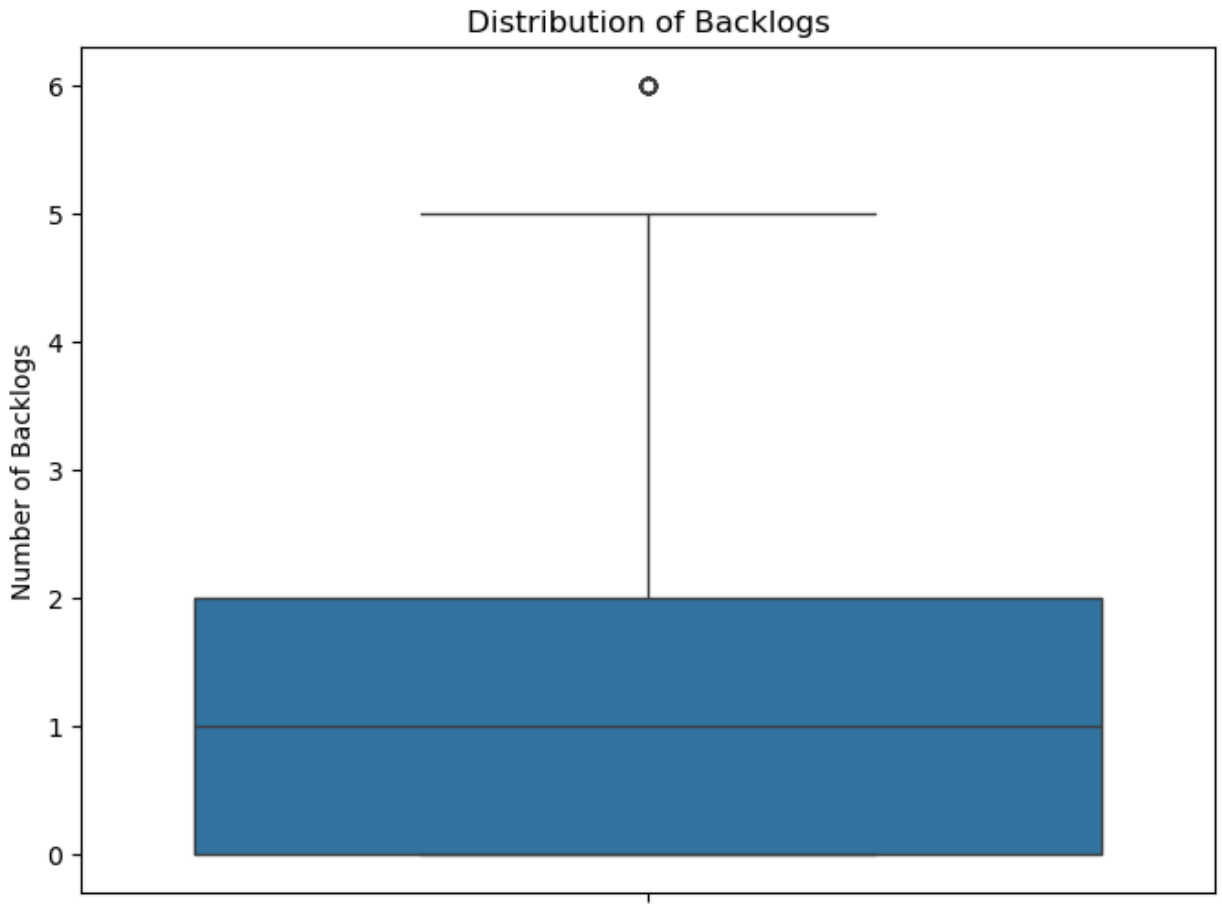
	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentage	Gr
715	716	716	19.0	14	17.0	16	20.0	19	value	0.0	105.0	88	
590	591	591	17.0	20	16.0	16	20.0	19	value	0.0	108.0	90	
264	265	265	17.0	15	12.0	16	15.0	18	value	0.0	93.0	78	
591	592	592	20.0	18	19.0	14	19.0	16	value	0.0	106.0	88	
593	594	594	20.0	17	18.0	20	19.0	20	value	0.0	114.0	95	
...	...	...	...	...	...	...	...	...	...	...	...	...	
648	649	649	0.0	0	0.0	0	0.0	0	value	0.0	0.0	0	
244	245	245	0.0	0	0.0	0	0.0	0	value	0.0	0.0	0	
633	634	634	9.0	0	2.0	2	3.0	9	value	0.0	25.0	21	
556	557	557	0.0	0	0.0	0	0.0	0	value	0.0	0.0	0	
414	415	415	0.0	0	0.0	0	0.0	0	value	0.0	0.0	0	

716 rows × 14 columns

In [44]:
```
j.value_counts('backlogs')
```

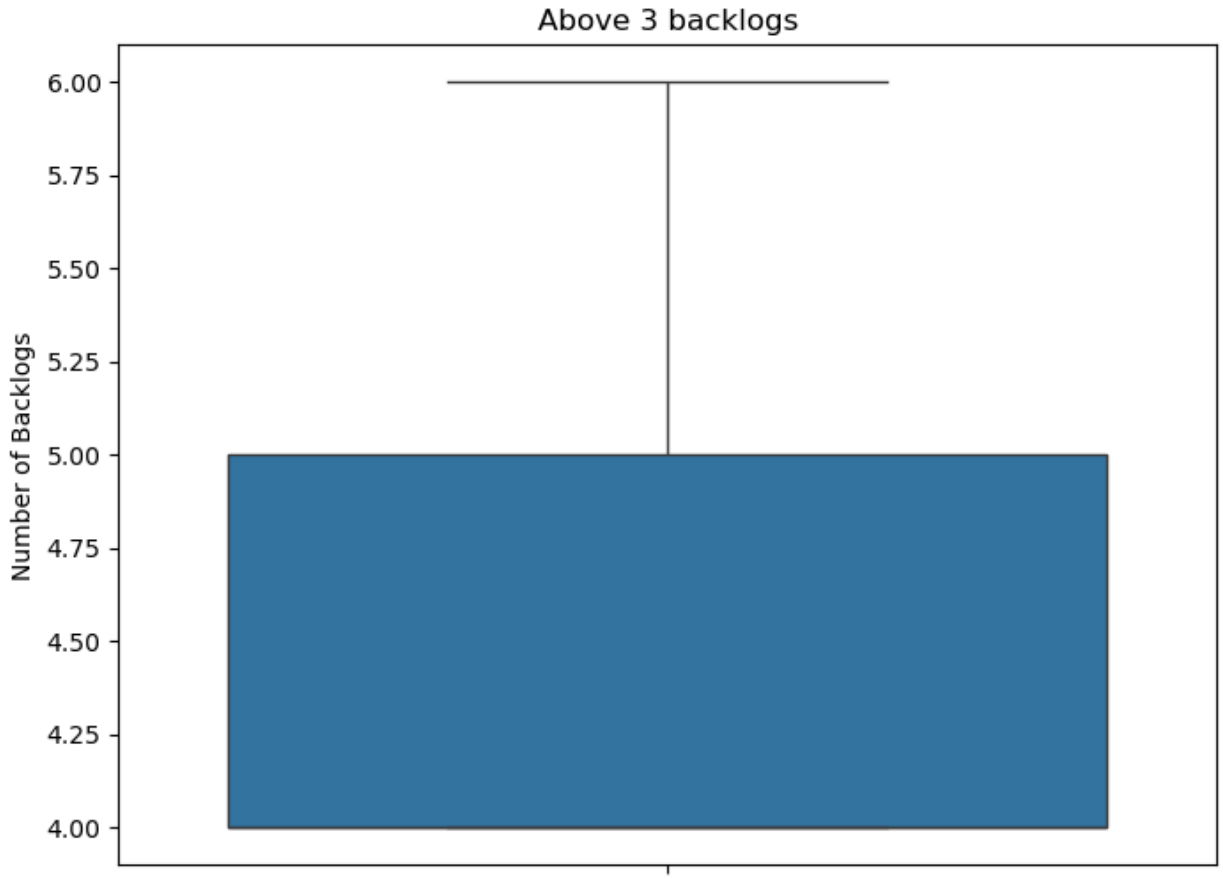
Out[44]:
```
backlogs
0    273
1    172
2    121
3     69
4     43
5     23
6     15
Name: count, dtype: int64
```

In [45]:
```
plt.figure(figsize=(8, 6))
sns.boxplot(y=df['backlogs'])
plt.title("Distribution of Backlogs")
plt.ylabel("Number of Backlogs")
plt.show()
```

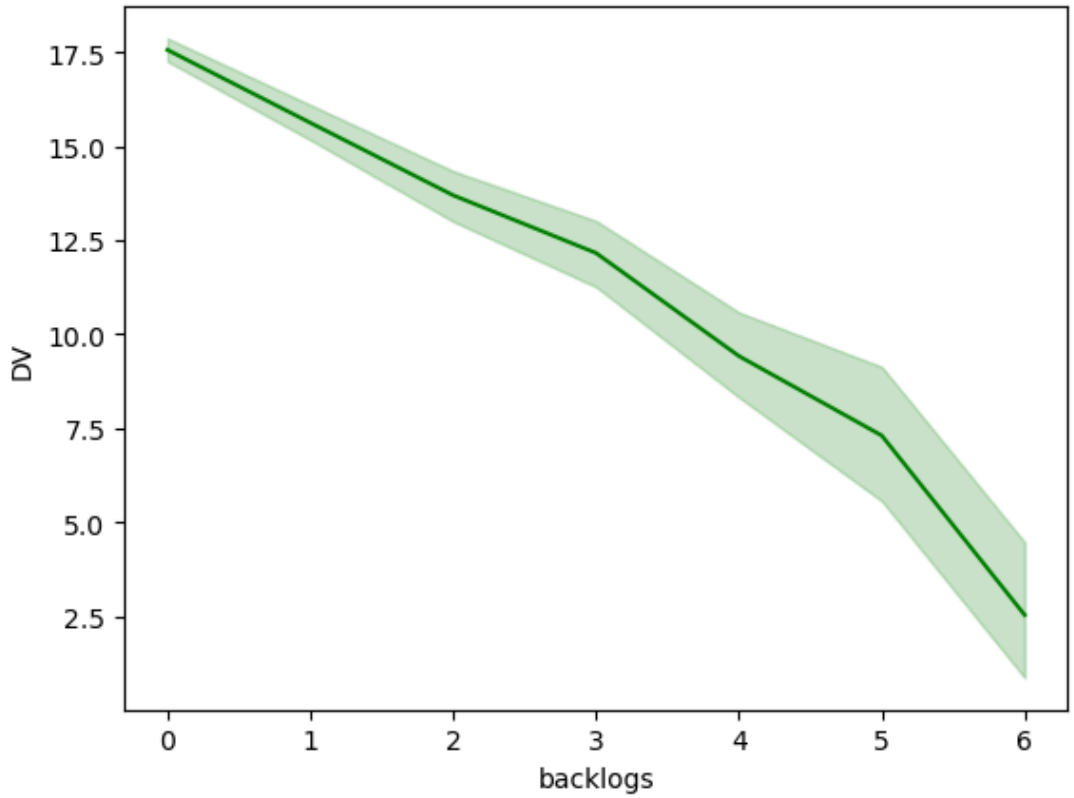


Students having backlogs in all subjects count

In [46]:
```
filtered_df = df[df['backlogs'] > 3]
plt.figure(figsize=(8, 6))
sns.boxplot(y=filtered_df['backlogs'])
plt.title("Above 3 backlogs")
plt.ylabel("Number of Backlogs")
plt.show()
```

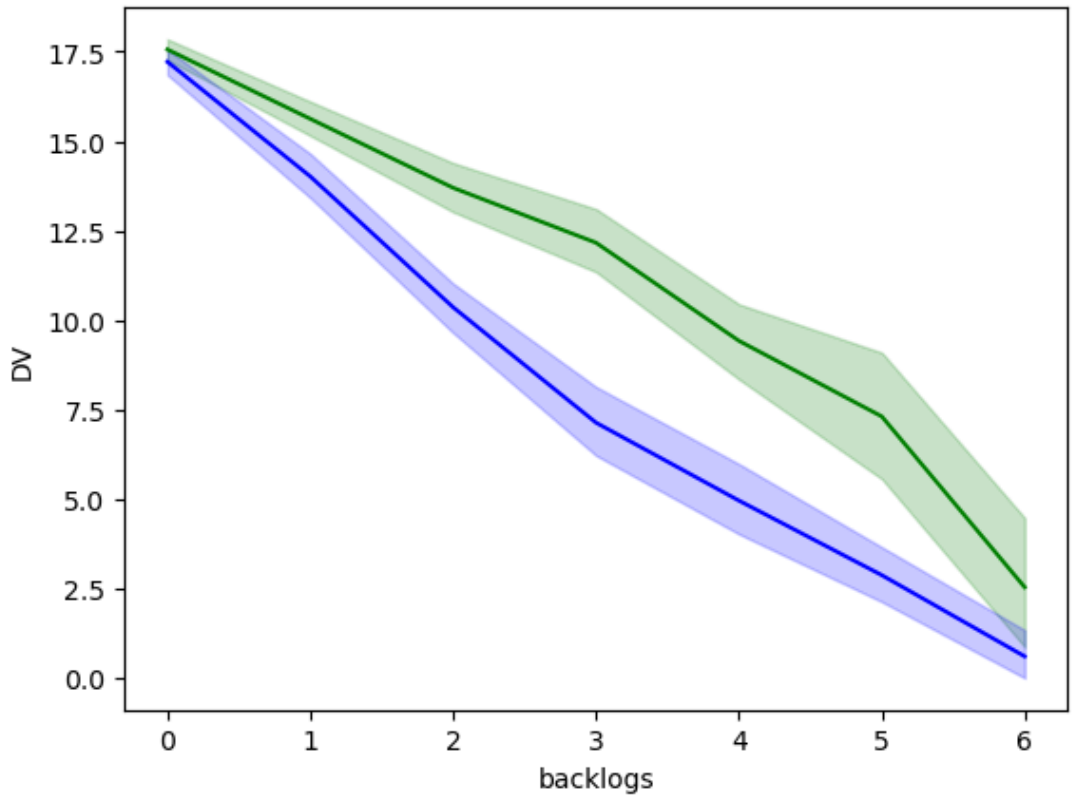


In [47]:
```
sns.lineplot(x='backlogs', y='DV', data=df, color = 'green')
plt.show()
```



Backlogs in DV

In [48]:
```
sns.lineplot(x='backlogs', y='DV', data=df, color='green')
sns.lineplot(x='backlogs', y='PP', data=df, color='blue')
plt.show()
```



backlogs who has in dv and aslo in pp

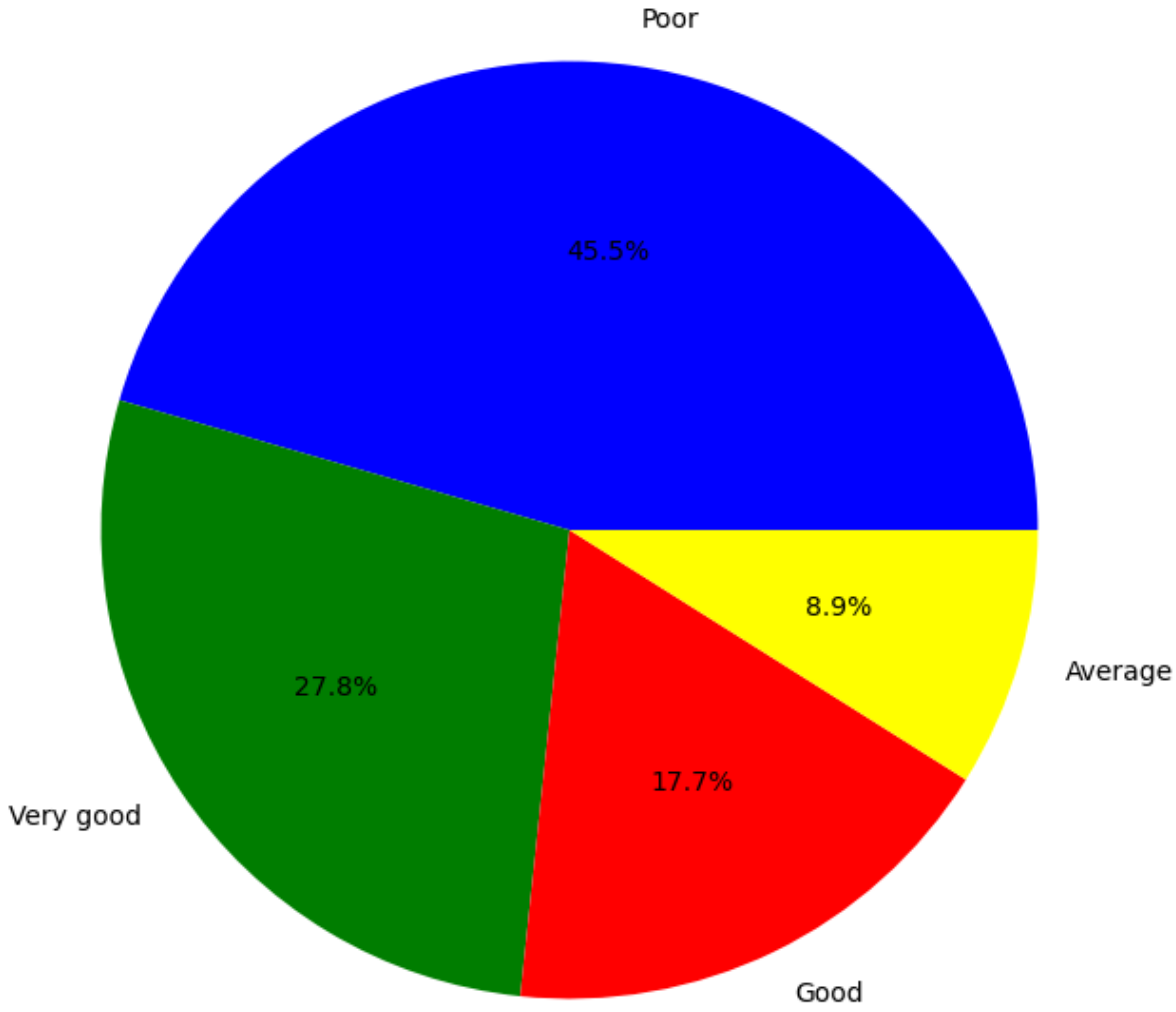
In [49]:
```
def assign_grade(PP):
    if 18 <= PP <= 20:
        return 'Very good'
    elif 15 <= PP <= 17:
        return 'Good'
    elif 13 <= PP <= 14:
        return 'Average'
    else:
        return 'Poor'
df['skills'] = df['PP'].apply(assign_grade)
df
```

Out[49]:

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentage	Gr
0	1	1	12.0	0	17.0	9	19.0	15	value	0.0	72.0	60	
1	2	2	19.0	12	16.0	16	18.0	3	value	0.0	84.0	70	
2	3	3	18.0	14	18.0	18	18.0	16	value	0.0	102.0	85	
3	4	4	15.0	9	19.0	17	19.0	15	value	0.0	94.0	78	
4	5	5	18.0	17	19.0	19	20.0	18	value	0.0	111.0	92	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
711	712	712	19.0	8	8.0	19	17.0	18	value	0.0	89.0	74	
712	713	713	12.0	1	7.0	10	20.0	8	value	0.0	58.0	48	
713	714	714	17.0	6	14.0	14	17.0	18	value	0.0	86.0	72	
714	715	715	12.0	1	6.0	7	15.0	12	value	0.0	53.0	44	
715	716	716	19.0	14	17.0	16	20.0	19	value	0.0	105.0	88	

716 rows × 15 columns

In [50]:
```
df['skills'] = df['PP'].apply(assign_grade)
skill_counts = df['skills'].value_counts()
plt.figure(figsize=(8, 8))
plt.pie(skill_counts, labels=skill_counts.index, autopct='%1.1f%%', colors=['blue','gr
plt.show()
df.skills.value_counts()
```



Out[50]:
```
skills
Poor         326
Very good    199
Good         127
Average       64
Name: count, dtype: int64
```

In [51]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 716 entries, 0 to 715
Data columns (total 15 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   S.NO        716 non-null    int64  
 1   SECTION     716 non-null    int64  
 2   DV          716 non-null    float64
 3   M2          716 non-null    int64  
 4   PP          716 non-null    float64
 5   BEEE        716 non-null    int64  
 6   FL          716 non-null    float64
 7   FIMS        716 non-null    int64  
 8   new_column  716 non-null    object 
 9   percentage  716 non-null    float64
 10  Total       716 non-null    float64
 11  Percentage  716 non-null    int32  
 12  Grade       716 non-null    object 
 13  backlogs    716 non-null    int64  
 14  skills      716 non-null    object 
dtypes: float64(5), int32(1), int64(6), object(3)
memory usage: 81.2+ KB
```

In [52]:
```
skills_count = df['skills'].value_counts()
print(skills_count)
```

```
skills
Poor         326
Very good    199
Good         127
Average       64
Name: count, dtype: int64
```

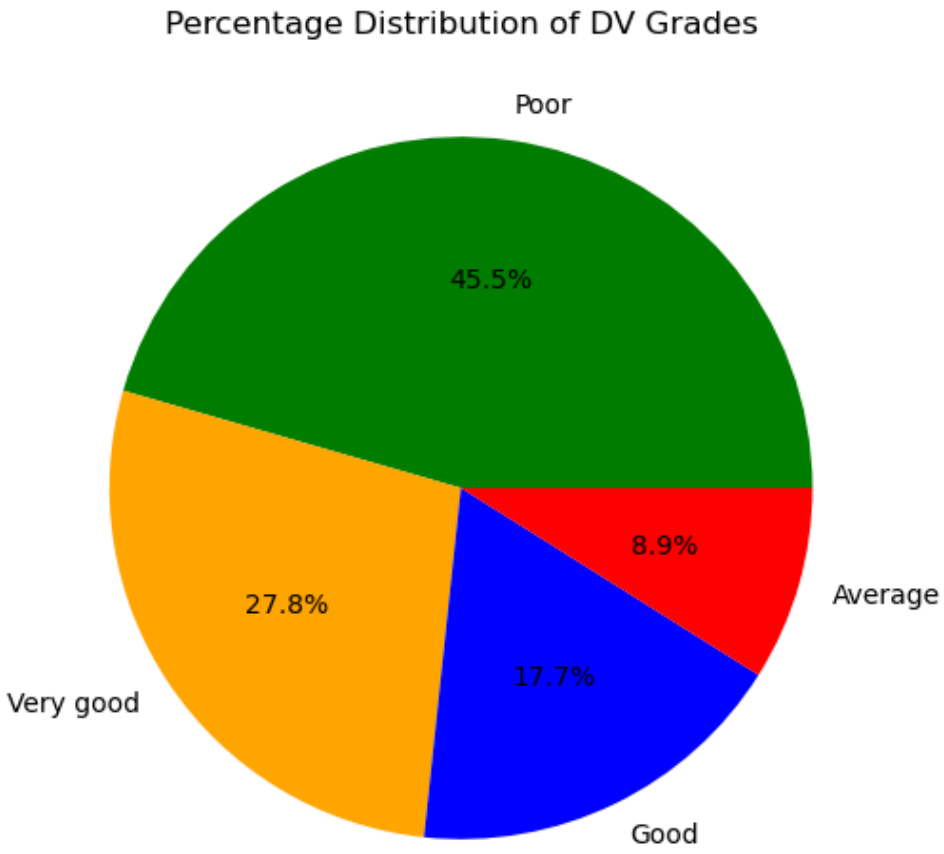
In [53]:
```
def assign_grade(DV):
    if 18 <= DV <= 20:
        return 'Very good'
    elif 15 <= DV <= 17:
        return 'Good'
    elif 13 <= DV <= 14:
        return 'Average'
    else:
        return 'Poor'
df['DV_skills'] = df['DV'].apply(assign_grade)
df
```

Out[53]:

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	new_column	percentage	Total	Percentage	Gr
0	1	1	12.0	0	17.0	9	19.0	15	value	0.0	72.0	60	
1	2	2	19.0	12	16.0	16	18.0	3	value	0.0	84.0	70	
2	3	3	18.0	14	18.0	18	18.0	16	value	0.0	102.0	85	
3	4	4	15.0	9	19.0	17	19.0	15	value	0.0	94.0	78	
4	5	5	18.0	17	19.0	19	20.0	18	value	0.0	111.0	92	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
711	712	712	19.0	8	8.0	19	17.0	18	value	0.0	89.0	74	
712	713	713	12.0	1	7.0	10	20.0	8	value	0.0	58.0	48	
713	714	714	17.0	6	14.0	14	17.0	18	value	0.0	86.0	72	
714	715	715	12.0	1	6.0	7	15.0	12	value	0.0	53.0	44	
715	716	716	19.0	14	17.0	16	20.0	19	value	0.0	105.0	88	

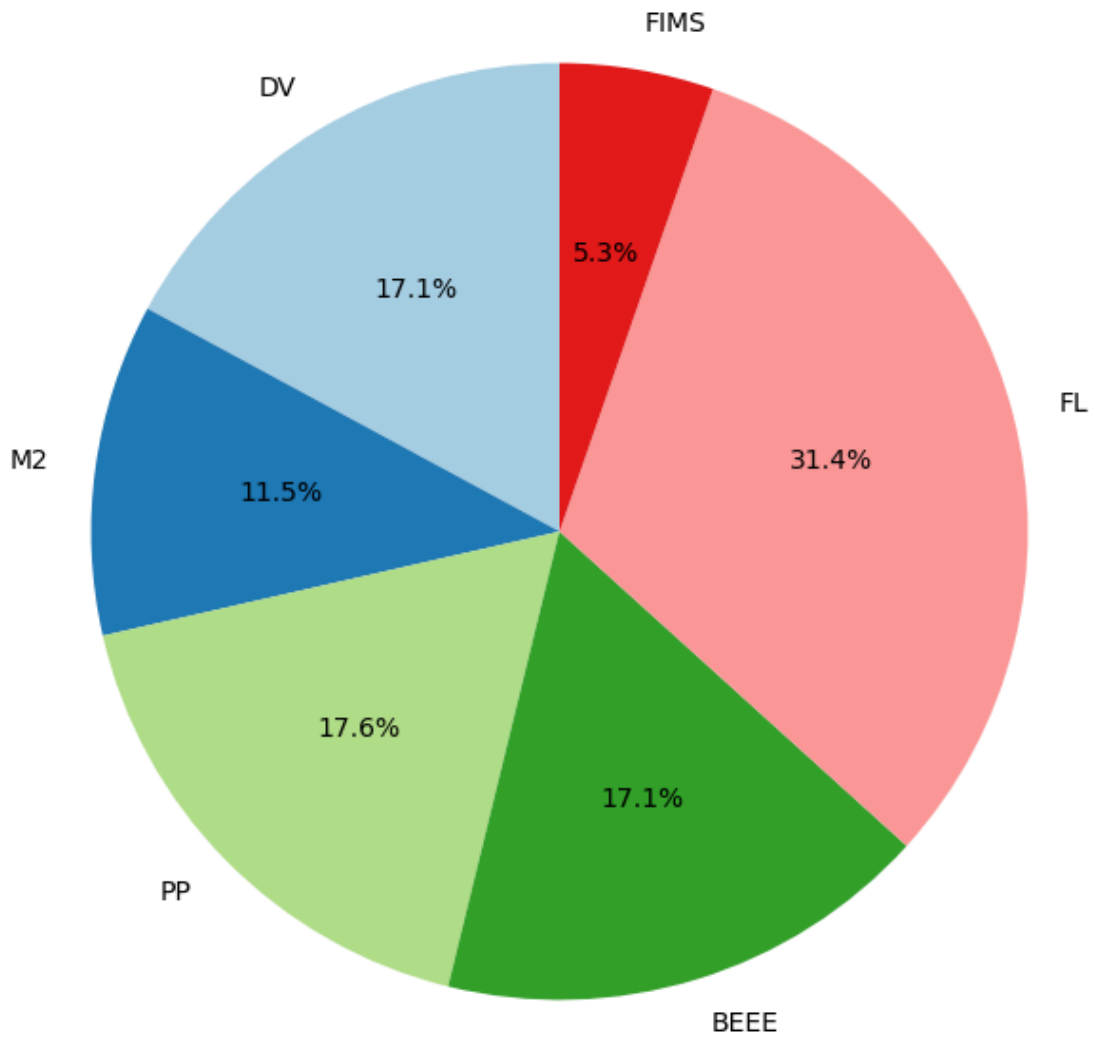
716 rows × 16 columns

In [54]:
```
df['DV_Grade'] = df['PP'].apply(assign_grade)
grade_counts = df['DV_Grade'].value_counts()
plt.figure(figsize=(6, 6))
grade_counts.plot(kind='pie', autopct='%1.1f%%', colors=['green', 'orange', 'blue', 'r
plt.title("Percentage Distribution of DV Grades")
plt.ylabel("")
plt.show()
```



In [55]:
```
subjects = ['DV', 'M2', 'PP', 'BEEE', 'FL', 'FIMS']
counts = [(df[subject] == 20).sum() for subject in subjects]
plt.figure(figsize=(8, 8))
plt.pie(counts, labels=subjects, autopct='%1.1f%%', startangle=90, colors=plt.cm.Paire
```

Out[55]:
```
([<matplotlib.patches.Wedge at 0x1d0c39e2ed0>,
  <matplotlib.patches.Wedge at 0x1d0c39e3290>,
  <matplotlib.patches.Wedge at 0x1d0c39e3f50>,
  <matplotlib.patches.Wedge at 0x1d0c39e3920>,
  <matplotlib.patches.Wedge at 0x1d0c39e0f50>,
  <matplotlib.patches.Wedge at 0x1d0c39e0170>],
 [Text(-0.563203304360446, 0.9448820233010442, 'DV'),
  Text(-1.0898900286816453, 0.14879423839760325, 'M2'),
  Text(-0.783882243055931, -0.7717050142519501, 'PP'),
  Text(0.322401702380501, -1.051692513191073, 'BEEE'),
  Text(1.0656695317720752, 0.2726691200937249, 'FL'),
  Text(0.18284189946149537, 1.084697579881744, 'FIMS')],
 [Text(-0.30720180237842504, 0.5153901945278422, '17.1%'),
  Text(-0.5944854701899882, 0.08116049367141995, '11.5%'),
  Text(-0.4275721325759623, -0.4209300077737909, '17.6%'),
  Text(0.1758554740257278, -0.5736504617405852, '17.1%'),
  Text(0.5812742900574955, 0.14872861096021356, '31.4%'),
  Text(0.09973194516081564, 0.5916532253900422, '5.3%')])
```

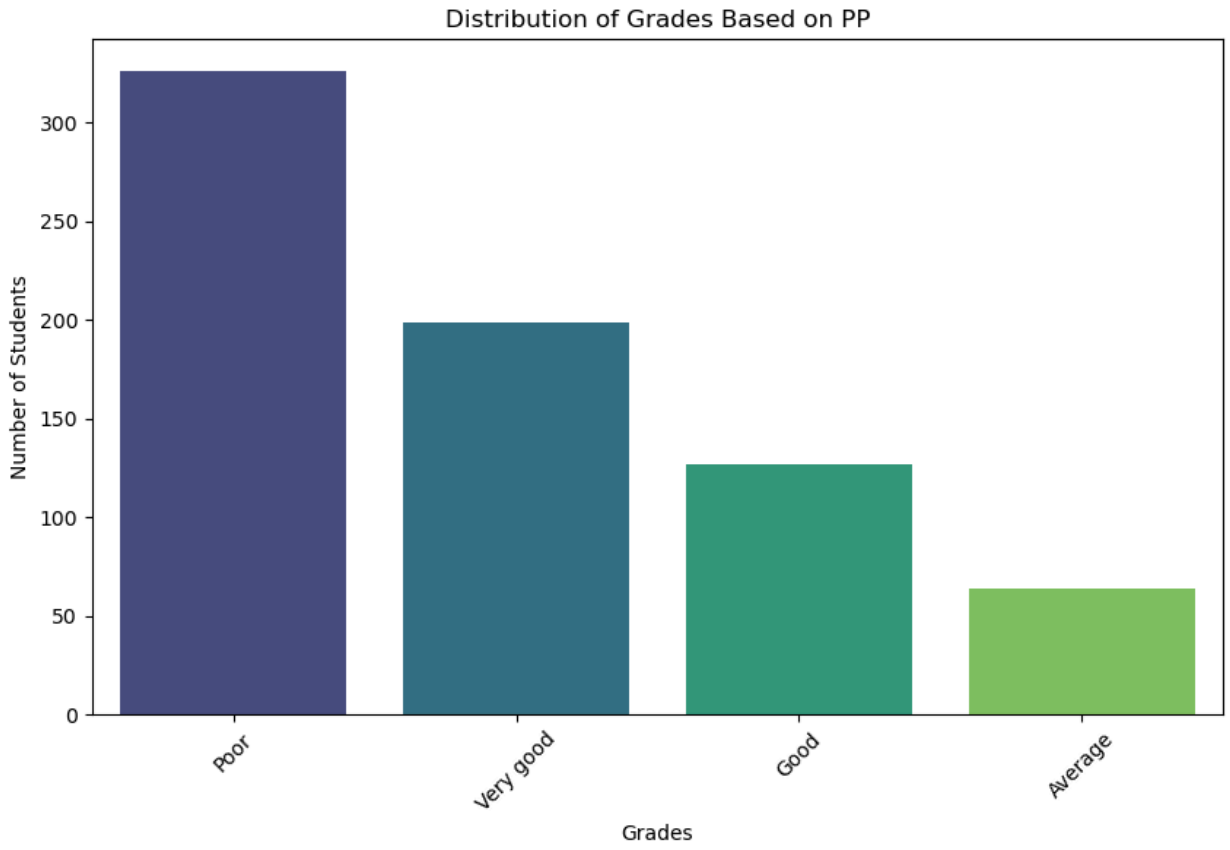


In [56]:
```
skills_count = df['DV_skills'].value_counts()
print(skills_count)
```

```
DV_skills
Very good    232
Good         216
Poor         197
Average       71
Name: count, dtype: int64
```

In [57]:
```
grade_counts = df['skills'].value_counts()
plt.figure(figsize=(10, 6))
sns.barplot(x=grade_counts.index, y=grade_counts.values, palette='viridis')
plt.title('Distribution of Grades Based on PP')
plt.xlabel('Grades')
plt.ylabel('Number of Students')
plt.xticks(rotation=45)
plt.show()
```

```
C:\Users\Saiko\AppData\Local\Temp\ipykernel_18268\3268027734.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=grade_counts.index, y=grade_counts.values, palette='viridis')
```



In [58]:
```
subset = df[df.iloc[:, 1:].eq(20).any(axis=1)]
print(subset)
```

```
     S.NO  SECTION    DV  M2    PP  BEEE    FL  FIMS new_column  percentage  \
4       5        5  18.0  17  19.0    19  20.0    18      value         0.0
6       7        7  15.0  10  20.0    20  15.0    14      value         0.0
7       8        8  17.0  17  19.0    20  19.0    13      value         0.0
8       9        9  10.0  18   0.0    20  19.0    15      value         0.0
9      10       10  18.0  19  20.0    20  20.0    15      value         0.0
..    ...      ...   ...  ..   ...   ...   ...   ...        ...         ...
703   704      704  16.0   0  11.0    16  20.0     0      value         0.0
707   708      708  19.0  17  12.0    17  20.0    16      value         0.0
709   710      710  18.0   9  12.0    20  16.0    16      value         0.0
712   713      713  12.0   1   7.0    10  20.0     8      value         0.0
715   716      716  19.0  14  17.0    16  20.0    19      value         0.0

     Total  Percentage Grade  backlogs     skills  DV_skills   DV_Grade
4    111.0          92     A         0  Very good  Very good  Very good
6     94.0          78     B         0  Very good       Good  Very good
7    105.0          88    B+         0  Very good       Good  Very good
8     82.0          68    C+         1       Poor       Poor       Poor
9    112.0          93     A         0  Very good  Very good  Very good
..     ...         ...   ...       ...        ...        ...        ...
703   63.0          52     C         2       Poor       Good       Poor
707  101.0          84    B+         0       Poor  Very good       Poor
709   91.0          76     B         1       Poor  Very good       Poor
712   58.0          48     D         3       Poor       Poor       Poor
715  105.0          88    B+         0       Good  Very good       Good

[301 rows x 17 columns]
```

In [59]:
```
subset = []
for index, row in df.iterrows():
    if (row.iloc[1:] == 20).any():
        subset.append(row)
subset_df = pd.DataFrame(subset)
print(subset_df)
```

```
     S.NO  SECTION    DV  M2    PP  BEEE    FL  FIMS new_column  percentage  \
4       5        5  18.0  17  19.0    19  20.0    18      value         0.0
6       7        7  15.0  10  20.0    20  15.0    14      value         0.0
7       8        8  17.0  17  19.0    20  19.0    13      value         0.0
8       9        9  10.0  18   0.0    20  19.0    15      value         0.0
9      10       10  18.0  19  20.0    20  20.0    15      value         0.0
..    ...      ...   ...  ..   ...   ...   ...   ...        ...         ...
703   704      704  16.0   0  11.0    16  20.0     0      value         0.0
707   708      708  19.0  17  12.0    17  20.0    16      value         0.0
709   710      710  18.0   9  12.0    20  16.0    16      value         0.0
712   713      713  12.0   1   7.0    10  20.0     8      value         0.0
715   716      716  19.0  14  17.0    16  20.0    19      value         0.0

     Total  Percentage Grade  backlogs     skills  DV_skills   DV_Grade
4    111.0          92     A         0  Very good  Very good  Very good
6     94.0          78     B         0  Very good       Good  Very good
7    105.0          88    B+         0  Very good       Good  Very good
8     82.0          68    C+         1       Poor       Poor       Poor
9    112.0          93     A         0  Very good  Very good  Very good
..     ...         ...   ...       ...        ...        ...        ...
703   63.0          52     C         2       Poor       Good       Poor
707  101.0          84    B+         0       Poor  Very good       Poor
709   91.0          76     B         1       Poor  Very good       Poor
712   58.0          48     D         3       Poor       Poor       Poor
715  105.0          88    B+         0       Good  Very good       Good

[301 rows x 17 columns]
```

In [60]:
```
subjects = ['DV', 'M2', 'PP', 'BEEE', 'FL', 'FIMS']
subset = []
for index, row in df.iterrows():
    for subject in subjects:
        if row[subject] == 20:
            subset.append(row)
            break
subset_df = pd.DataFrame(subset)
print(subset_df)
```

```
     S.NO  SECTION    DV  M2    PP  BEEE    FL  FIMS new_column  percentage  \
4       5        5  18.0  17  19.0    19  20.0    18      value         0.0
6       7        7  15.0  10  20.0    20  15.0    14      value         0.0
7       8        8  17.0  17  19.0    20  19.0    13      value         0.0
8       9        9  10.0  18   0.0    20  19.0    15      value         0.0
9      10       10  18.0  19  20.0    20  20.0    15      value         0.0
..    ...      ...   ...  ..   ...   ...   ...   ...        ...         ...
703   704      704  16.0   0  11.0    16  20.0     0      value         0.0
707   708      708  19.0  17  12.0    17  20.0    16      value         0.0
709   710      710  18.0   9  12.0    20  16.0    16      value         0.0
712   713      713  12.0   1   7.0    10  20.0     8      value         0.0
715   716      716  19.0  14  17.0    16  20.0    19      value         0.0

     Total  Percentage Grade  backlogs     skills  DV_skills   DV_Grade
4    111.0          92     A         0  Very good  Very good  Very good
6     94.0          78     B         0  Very good       Good  Very good
7    105.0          88    B+         0  Very good       Good  Very good
8     82.0          68    C+         1       Poor       Poor       Poor
9    112.0          93     A         0  Very good  Very good  Very good
..     ...         ...   ...       ...        ...        ...        ...
703   63.0          52     C         2       Poor       Good       Poor
707  101.0          84    B+         0       Poor  Very good       Poor
709   91.0          76     B         1       Poor  Very good       Poor
712   58.0          48     D         3       Poor       Poor       Poor
715  105.0          88    B+         0       Good  Very good       Good

[300 rows x 17 columns]
```

In [61]:
```
subjects = ['DV', 'M2', 'PP', 'BEEE', 'FL', 'FIMS']
subset = df[df[subjects].eq(20).any(axis=1)]
print("Subset of students who scored 20 in any subject:")
print(subset)
for subject in subjects:
    count_20 = (df[subject] == 20).sum()
    print(f"Students who scored 20 in {subject}: {count_20}")
```

```
Subset of students who scored 20 in any subject:
     S.NO  SECTION    DV  M2    PP  BEEE    FL  FIMS new_column  percentage  \
4       5        5  18.0  17  19.0    19  20.0    18      value         0.0
6       7        7  15.0  10  20.0    20  15.0    14      value         0.0
7       8        8  17.0  17  19.0    20  19.0    13      value         0.0
8       9        9  10.0  18   0.0    20  19.0    15      value         0.0
9      10       10  18.0  19  20.0    20  20.0    15      value         0.0
..    ...      ...   ...  ..   ...   ...   ...   ...        ...         ...
703   704      704  16.0   0  11.0    16  20.0     0      value         0.0
707   708      708  19.0  17  12.0    17  20.0    16      value         0.0
709   710      710  18.0   9  12.0    20  16.0    16      value         0.0
712   713      713  12.0   1   7.0    10  20.0     8      value         0.0
715   716      716  19.0  14  17.0    16  20.0    19      value         0.0

     Total  Percentage Grade  backlogs     skills  DV_skills   DV_Grade
4    111.0          92     A         0  Very good  Very good  Very good
6     94.0          78     B         0  Very good       Good  Very good
7    105.0          88    B+         0  Very good       Good  Very good
8     82.0          68    C+         1       Poor       Poor       Poor
9    112.0          93     A         0  Very good  Very good  Very good
..     ...         ...   ...       ...        ...        ...        ...
703   63.0          52     C         2       Poor       Good       Poor
707  101.0          84    B+         0       Poor  Very good       Poor
709   91.0          76     B         1       Poor  Very good       Poor
712   58.0          48     D         3       Poor       Poor       Poor
715  105.0          88    B+         0       Good  Very good       Good

[300 rows x 17 columns]
Students who scored 20 in DV: 103
Students who scored 20 in M2: 69
Students who scored 20 in PP: 106
Students who scored 20 in BEEE: 103
Students who scored 20 in FL: 189
Students who scored 20 in FIMS: 32
```

In [62]:
```
print(df.shape)
```

```
(716, 17)
```

In [63]:
```
print(df.describe(include='all'))
```

```
              S.NO     SECTION          DV          M2          PP  \
count   716.000000  716.000000  716.000000  716.000000  716.000000
unique         NaN         NaN         NaN         NaN         NaN
top            NaN         NaN         NaN         NaN         NaN
freq           NaN         NaN         NaN         NaN         NaN
mean    358.500000  358.500000   14.787709    9.949721   12.769553
std     206.835684  206.835684    4.569545    6.599236    5.817381
min       1.000000    1.000000    0.000000    0.000000    0.000000
25%     179.750000  179.750000   12.000000    4.000000    9.000000
50%     358.500000  358.500000   16.000000   10.000000   14.000000
75%     537.250000  537.250000   18.000000   16.000000   18.000000
max     716.000000  716.000000   20.000000   20.000000   20.000000

              BEEE          FL        FIMS new_column  percentage       Total  \
count   716.000000  716.000000  716.000000        716       716.0  716.000000
unique         NaN         NaN         NaN          1         NaN         NaN
top            NaN         NaN         NaN      value         NaN         NaN
freq           NaN         NaN         NaN        716         NaN         NaN
mean     13.287709   15.544693   14.047486        NaN         0.0   80.386872
std       5.783112    4.476132    4.709815        NaN         0.0   25.335341
min       0.000000    0.000000    0.000000        NaN         0.0    0.000000
25%       9.000000   13.000000   12.000000        NaN         0.0   64.000000
50%      15.000000   16.000000   15.000000        NaN         0.0   84.000000
75%      18.000000   20.000000   18.000000        NaN         0.0  101.000000
max      20.000000   20.000000   20.000000        NaN         0.0  120.000000

        Percentage Grade    backlogs skills  DV_skills DV_Grade
count   716.000000   716  716.000000    716         716      716
unique         NaN     7         NaN      4           4        4
top            NaN     B         NaN   Poor  Very good     Poor
freq           NaN   134         NaN    326         232      326
mean     66.979050   NaN    1.393855    NaN         NaN      NaN
std      21.120457   NaN    1.536927    NaN         NaN      NaN
min       0.000000   NaN    0.000000    NaN         NaN      NaN
25%      53.000000   NaN    0.000000    NaN         NaN      NaN
50%      70.000000   NaN    1.000000    NaN         NaN      NaN
75%      84.000000   NaN    2.000000    NaN         NaN      NaN
max     100.000000   NaN    6.000000    NaN         NaN      NaN
```

In [64]:
```
df.describe()
```

Out[64]:

	S.NO	SECTION	DV	M2	PP	BEEE	FL	FIMS	p
count	716.000000	716.000000	716.000000	716.000000	716.000000	716.000000	716.000000	716.000000	
mean	358.500000	358.500000	14.787709	9.949721	12.769553	13.287709	15.544693	14.047486	
std	206.835684	206.835684	4.569545	6.599236	5.817381	5.783112	4.476132	4.709815	
min	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	179.750000	179.750000	12.000000	4.000000	9.000000	9.000000	13.000000	12.000000	
50%	358.500000	358.500000	16.000000	10.000000	14.000000	15.000000	16.000000	15.000000	
75%	537.250000	537.250000	18.000000	16.000000	18.000000	18.000000	20.000000	18.000000	
max	716.000000	716.000000	20.000000	20.000000	20.000000	20.000000	20.000000	20.000000	

DATASET OBSERVATION:

After cleaning and organizing the dataset, different statistical and graphical analyses were performed to get deeper insights. Histograms and distribution plots were created using Seaborn and Matplotlib to understand the distribution of student marks over different subjects. Total marks distribution gave a clear indication of how students performed overall, whereas the percentage distribution indicated if most of the students were concentrated in a particular range or there was a high variation in scores. Further, a grade-wise distribution analysis was done to see the number of students in each grade category to understand overall trends of performance.

Aside from individual marks analysis, a correlation study was performed to analyze the relationship between various subjects. This assisted in the identification of those subjects that correlated well with each other, such that students who excelled in one subject tended to excel in another. This information could prove beneficial in terms of determining if there are particular subjects that need more focus or if there is a pattern to student strengths.

Overall, the dataset was reformed from raw, inconsistent records into a properly structured, insightful dataset that offered a clear view of student performance, grading distributions, and subject-wise analysis. The study not only identified score distributions and trends but also gave valuable insights into areas of potential academic improvement.

In [ ]: