```
import numpy as np
import pandas as pd


import os




import matplotlib.pyplot as plt
import seaborn as sns
```

Load File

```
df = pd.read_csv('/content/Datsset.csv')
df.head()
```

	Certification Course	Gender	Department	Height(CM)	Weight(KG)	10th Mark	12th Mark	colle ma
0	No	Male	BCA	100.0	58.0	79.0	64.0	80
1	No	Female	BCA	90.0	40.0	70.0	80.0	70
2	Yes	Male	BCA	159.0	78.0	71.0	61.0	55
3	Yes	Female	BCA	147.0	20.0	70.0	59.0	58
4	No	Male	BCA	170.0	54.0	40.0	65.0	30

Next steps: Generate code with df View recommended plots

Explore Data

```
df.shape
```

```
(235, 19)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 235 entries, 0 to 234
Data columns (total 19 columns):
 #   Column                                            Non-Null Count  Dtype  
---  ------                                            --------------  -----  
 0   Certification Course                              235 non-null    object 
 1   Gender                                            235 non-null    object 
 2   Department                                        235 non-null    object 
 3   Height(CM)                                        235 non-null    float64
 4   Weight(KG)                                        235 non-null    float64
 5   10th Mark                                         235 non-null    float64
 6   12th Mark                                         235 non-null    float64
 7   college mark                                      235 non-null    float64
 8   hobbies                                           235 non-null    object 
 9   daily studing time                                235 non-null    object 
 10  prefer to study in                                235 non-null    object 
 11  salary expectation                                235 non-null    int64  
 12  Do you like your degree?                          235 non-null    object 
 13  willingness to pursue a career based on their degree  235 non-null    object 
 14  social medai & video                              235 non-null    object 
 15  Travelling Time                                   235 non-null    object 
 16  Stress Level                                      235 non-null    object 
 17  Financial Status                                  235 non-null    object 
 18  JoinedProgram                                     235 non-null    object 
dtypes: float64(5), int64(1), object(13)
memory usage: 35.0+ KB
```

```
df.describe()
```

	Height(CM)	Weight(KG)	10th Mark	12th Mark	college mark	salary expectation
count	235.000000	235.000000	235.000000	235.000000	235.000000	2.350000e+02
mean	157.402128	60.803830	76.848511	68.775872	70.660553	3.248168e+04
std	21.510805	14.895844	13.047560	11.018192	15.727446	1.113146e+05
min	4.500000	20.000000	7.400000	45.000000	1.000000	0.000000e+00
25%	152.000000	50.000000	70.000000	60.000000	60.000000	1.500000e+04
50%	160.000000	60.000000	80.000000	69.000000	70.000000	2.000000e+04
75%	170.000000	70.000000	86.250000	76.000000	80.000000	2.500000e+04
max	192.000000	106.000000	98.000000	94.000000	100.000000	1.500000e+06

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

```
Certification Course                                          0
Gender                                                        0
Department                                                    0
Height(CM)                                                    0
Weight(KG)                                                    0
10th Mark                                                     0
12th Mark                                                     0
college mark                                                  0
hobbies                                                       0
daily studing time                                            0
prefer to study in                                            0
salary expectation                                            0
Do you like your degree?                                      0
willingness to pursue a career based on their degree          0
social medai & video                                          0
Travelling Time                                               0
Stress Level                                                  0
Financial Status                                              0
JoinedProgram                                                 0
dtype: int64
```

Let's tidy up some columns

```
df.columns
```

```
Index(['Certification Course', 'Gender', 'Department', 'Height(CM)',
       'Weight(KG)', '10th Mark', '12th Mark', 'college mark', 'hobbies',
       'daily studing time', 'prefer to study in', 'salary expectation',
       'Do you like your degree?',
       'willingness to pursue a career based on their degree  ',
       'social medai & video', 'Travelling Time ', 'Stress Level ',
       'Financial Status', 'JoinedProgram'],
      dtype='object')
```

```
old_names = ['Certification Course', 'Gender', 'Department', 'Height(CM)',
       'Weight(KG)', '10th Mark', '12th Mark', 'college mark', 'hobbies',
       'daily studing time', 'prefer to study in', 'salary expectation',
       'Do you like your degree?',
       'willingness to pursue a career based on their degree  ',
       'social medai & video', 'Travelling Time ', 'Stress Level ',
       'Financial Status', 'JoinedProgram']

new_names = ['Certification_Course', 'Gender', 'Department', 'Height(CM)',
       'Weight(KG)', '10th_Mark', '12th_Mark', 'college_mark', 'hobbies',
       'daily_studing_time', 'prefer_to_study_in', 'salary_expectation',
       'Do_you_like_your_degree?',
       'willingness_to_pursue_a_career_based_on_their_degree',
       'social_medai_&_video', 'Travelling_Time', 'Stress_Level',
       'Financial_Status', 'JoinedProgram']

df.rename(columns=dict(zip(old_names, new_names)), inplace=True)
```

```
df.columns
```

```
Index(['Certification_Course', 'Gender', 'Department', 'Height(CM)',
       'Weight(KG)', '10th_Mark', '12th_Mark', 'college_mark', 'hobbies',
       'daily_studing_time', 'prefer_to_study_in', 'salary_expectation',
       'Do_you_like_your_degree?',
       'willingness_to_pursue_a_career_based_on_their_degree',
       'social_medai_&_video', 'Travelling_Time', 'Stress_Level',
       'Financial_Status', 'JoinedProgram'],
      dtype='object')
```

Explore with some visuals

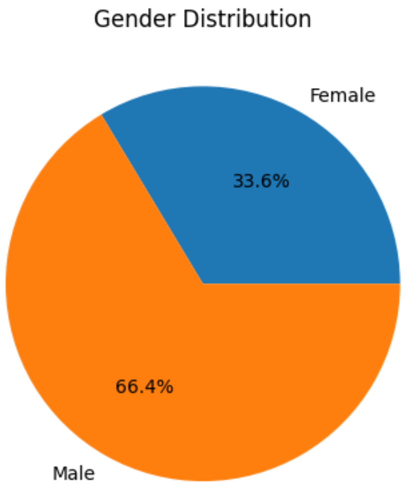
- females are generally doing better than their male counterparts

```
df.groupby('Gender')[['Height(CM)','Weight(KG)',
                      '10th_Mark', '12th_Mark', 'college_mark']].mean()
```

Gender	Height(CM)	Weight(KG)	10th_Mark	12th_Mark	college_mark
Female	150.348101	50.468354	78.974684	71.715190	76.870886
Male	160.974359	66.037821	75.771795	67.287372	67.515577

```
df.groupby(['Gender'])[['Gender']].value_counts().plot(kind='pie',autopct='%1.1f%%',ylabel="")
plt.title('Gender Distribution')
```

```
    Text(0.5, 1.0, 'Gender Distribution')
```



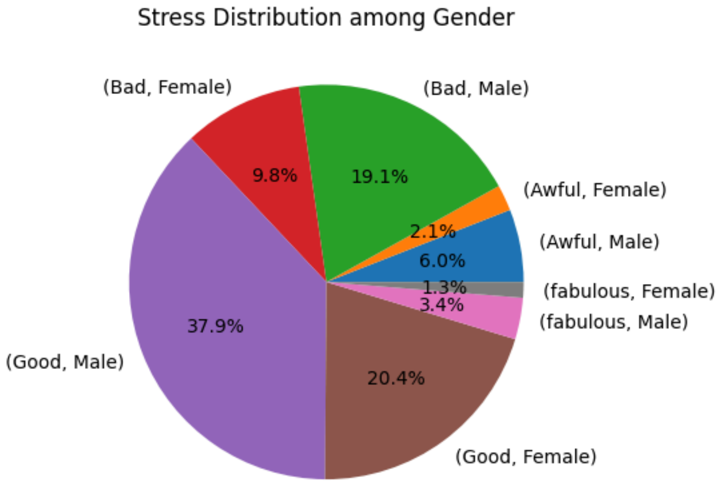
How is stress imparting performance?

```
df.groupby('Stress_Level')[['Gender']].value_counts()
```

```
    Stress_Level  Gender
    Awful         Male      14
                  Female     5
    Bad           Male      45
                  Female    23
    Good          Male      89
                  Female    48
    fabulous      Male       8
                  Female     3
    Name: count, dtype: int64
```

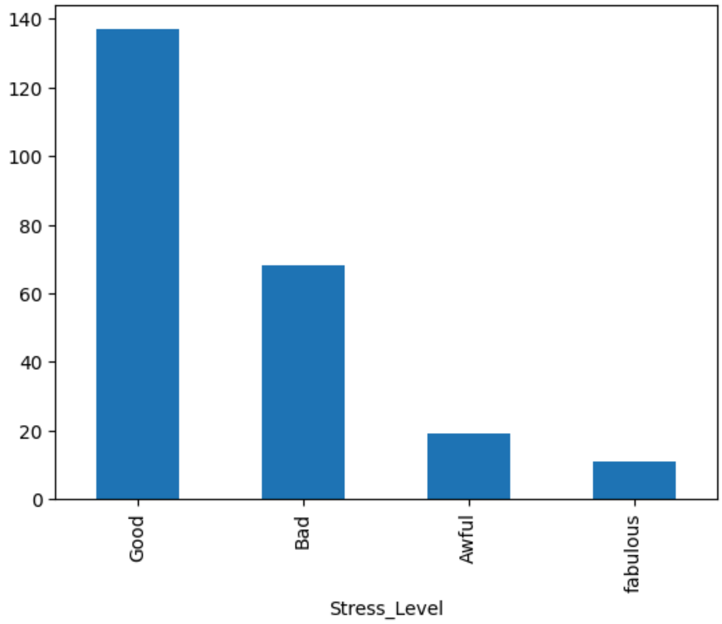
```
df.groupby('Stress_Level')[['Gender']].value_counts().plot(kind='pie',autopct='%1.1f%%',ylabel="")
plt.title('Stress Distribution among Gender')
```

```
Text(0.5, 1.0, 'Stress Distribution among Gender')
```



```
df['Stress_Level'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='Stress_Level'>
```

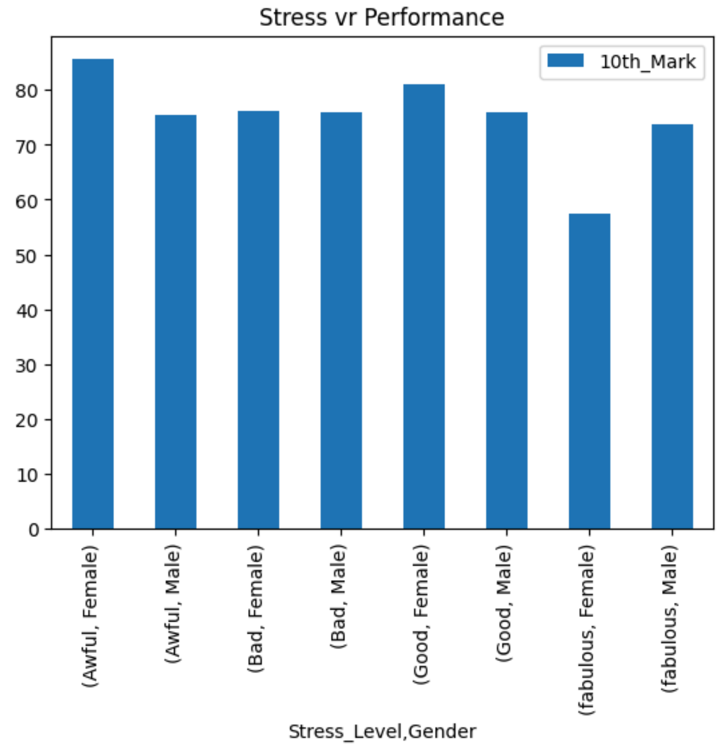


```
df.groupby(['Stress_Level', 'Gender'])[['Weight(KG)',
                    '10th_Mark', '12th_Mark', 'college_mark']].mean()
```

		Weight(KG)	10th_Mark	12th_Mark	college_mark
Stress_Level	Gender				
Awful	Female	48.600000	85.600000	75.200000	75.000000
	Male	67.571429	75.400000	66.000000	68.571429
Bad	Female	48.391304	76.139130	65.004348	73.260870
	Male	65.133333	75.933333	65.346667	66.451111
Good	Female	52.104167	80.995833	75.716667	79.850000
	Male	66.482022	75.930337	68.238539	67.057640
fabulous	Female	43.333333	57.333333	53.333333	60.000000
	Male	63.500000	73.750000	69.875000	76.750000

```
df.groupby(['Stress_Level', 'Gender'])[['10th_Mark']].mean().plot(kind='bar')
plt.title('Stress vr Performance')
```

```
    Text(0.5, 1.0, 'Stress vr Performance')
```



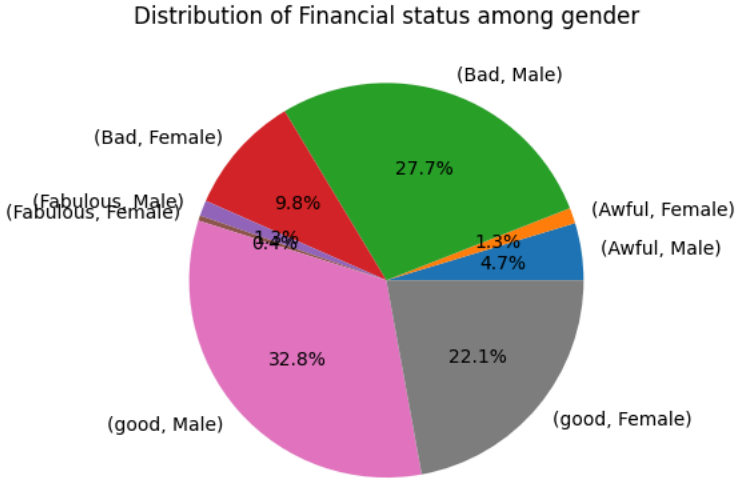
Financial status of participants

```
df.Financial_Status.unique()
```

```
    array(['Bad', 'good', 'Awful', 'Fabulous'], dtype=object)
```

```
df.groupby('Financial_Status')[['Gender']].value_counts().plot(kind='pie',autopct='%1.1f%%',ylabel="")
plt.title('Distribution of Financial status among gender')
```

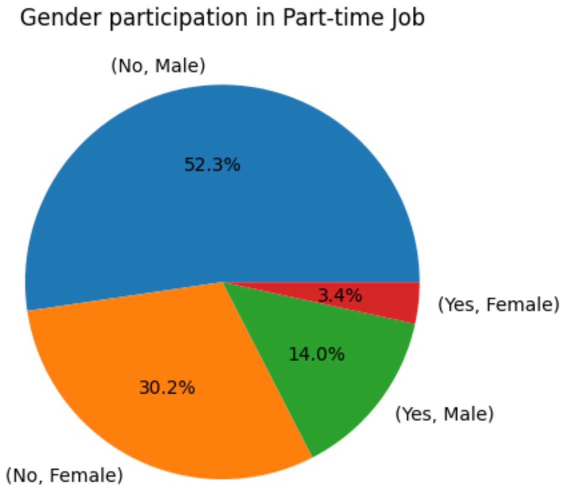
```
    Text(0.5, 1.0, 'Distribution of Financial status among gender')
```



More Males are engaged in part-time jobs

```
df.groupby('JoinedProgram')[['Gender']].value_counts().plot(kind='pie',autopct='%1.1f%%',ylabel="")
plt.title('Gender participation in Part-time Job')
```

```
Text(0.5, 1.0, 'Gender participation in Part-time Job')
```



```
df.daily_studing_time.unique()
```

```
array(['0 - 30 minute', '30 - 60 minute', '1 - 2 Hour', '2 - 3 hour',
       '3 - 4 hour', 'More Than 4 hour'], dtype=object)
```

```
def convert_to_minutes(time_range):
    if 'More Than' in time_range:
        return 240  # Assuming "More Than 4 hour" is equivalent to 240 minutes
    else:
        hours, _, _ = time_range.partition('-')
        hours = int(hours.strip())
        return hours * 60


# Apply the function to the column and create a new column
df['daily_studing_time_minutes'] = df['daily_studing_time'].apply(convert_to_minutes)
```

```
df.Travelling_Time.unique()
```

```
array(['30 - 60 minutes', '0 - 30 minutes', '1 - 1.30 hour',
       '2 - 2.30 hour', '1.30 - 2 hour', 'more than 3 hour',
       '2.30 - 3 hour'], dtype=object)
```

```
df['social_medai_&_video'].unique()
```

```
array(['1.30 - 2 hour', '1 - 1.30 hour', 'More than 2 hour',
       '30 - 60 Minute', '1 - 30 Minute', '0 Minute'], dtype=object)
```

```
def convert_social_medai_video_to_minutes(time_range):
    if 'More than' in time_range:
        return 180  # Assuming "more than 3 hour" is equivalent to 180 minutes
    else:
        parts = time_range.split('-')
        if len(parts) == 2:
            start, end = parts
            start_time, end_time = map(float, [start.strip().split()[0], end.strip().split()[0]])
            if 'hour' in time_range:
                return (end_time - start_time) * 60
            else:
                return end_time - start_time
        else:
            return float(time_range.split()[0])


# Apply the function to the column and create a new column with minutes
df['social_media_Minutes'] = df['social_medai_&_video'].apply(convert_social_medai_video_to_minutes)
```

```
def convert_travelling_time_to_minutes(time_range):
    if 'more than' in time_range:
        return 180  # Assuming "more than 3 hour" is equivalent to 180 minutes
    else:
        parts = time_range.split('-')
        if len(parts) == 2:
            start, end = parts
            start_time, end_time = map(float, [start.strip().split()[0], end.strip().split()[0]])
            if 'hour' in time_range:
```

```
            return (end_time - start_time) * 60
        else:
            return end_time - start_time
    else:
        return float(time_range.split()[0])

# Apply the function to the column and create a new column with minutes
df['Travelling_Time_Minutes'] = df['Travelling_Time'].apply(convert_travelling_time_to_minutes)
```

```
df.drop(columns=['daily_studing_time','social_medai_&_video', 'Travelling_Time'], inplace=True)
```

```
df.groupby('Gender')[['daily_studing_time_minutes','social_media_Minutes','Travelling_Time_Minutes']].mean()
```

Gender	daily_studing_time_minutes	social_media_Minutes	Travelling_Time_Minutes
Female	540.000000	46.075949	30.075949
Male	732.692308	49.211538	36.500000

Students turn to social media and videos when they are stressed.

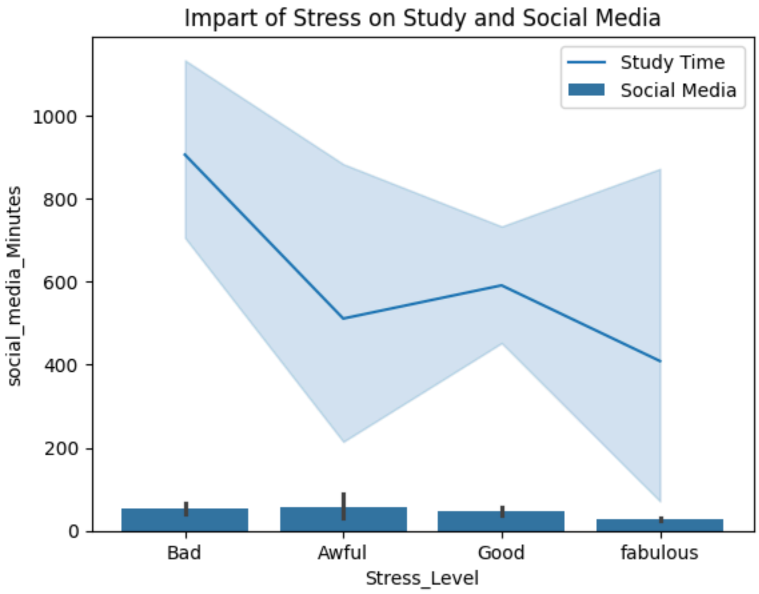
- One can notice that, when stress level is worst, students spend more time on social media than studying

```
df.groupby(['Stress_Level','Gender'])[['daily_studing_time_minutes','social_media_Minutes','Travelling_Time_Minutes']].mean(
```

Stress_Level	Gender	daily_studing_time_minutes	social_media_Minutes	Travelling_Time_Minutes
Awful	**Female**	372.000000	84.000000	30.000000
	Male	561.428571	47.428571	29.142857
Bad	**Female**	751.304348	44.956522	27.913043
	Male	986.666667	57.022222	36.266667
Good	**Female**	480.000000	43.687500	30.875000
	Male	651.910112	47.426966	38.089888
fabulous	**Female**	160.000000	29.666667	34.000000
	Male	502.500000	28.250000	33.000000

```
ax=sns.barplot(df[['Stress_Level', 'daily_studing_time_minutes','social_media_Minutes']],x='Stress_Level', y='social_media_M
sns.lineplot(df[['Stress_Level', 'daily_studing_time_minutes','social_media_Minutes']],x='Stress_Level', y='daily_studing_ti
plt.title('Impart of Stress on Study and Social Media')
```

```
Text(0.5, 1.0, 'Impart of Stress on Study and Social Media')
```



```
catCols = [col for col in df.columns if df[col].dtypes=='object']
catCols
```

```
['Certification_Course',
 'Gender',
 'Department',
 'hobbies',
 'prefer_to_study_in',
 'Do_you_like_your_degree?',
 'willingness_to_pursue_a_career_based_on_their_degree',
 'Stress_Level',
 'Financial_Status',
 'JoinedProgram']
```

```
numCols = df.select_dtypes(include=['int64', 'float64']).columns.tolist()
numCols
```

```
['Height(CM)',
 'Weight(KG)',
 '10th_Mark',
 '12th_Mark',
 'college_mark',
 'salary_expectation',
 'daily_studing_time_minutes',
 'social_media_Minutes',
 'Travelling_Time_Minutes']
```

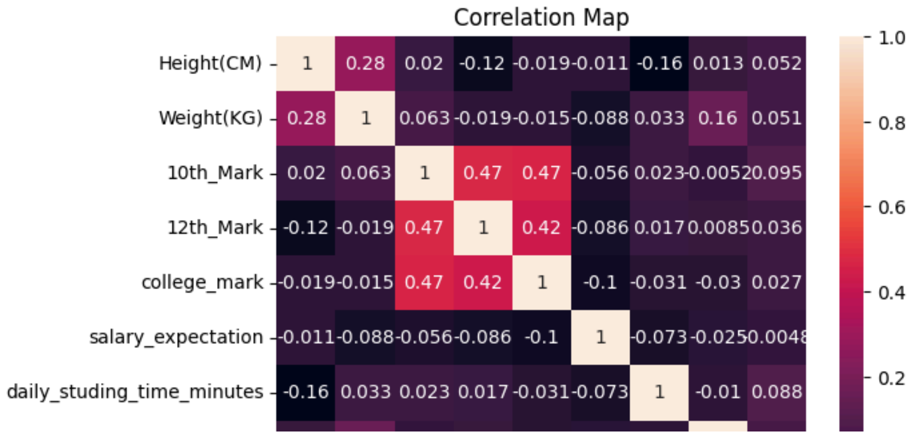
how do the numerical columns corelate?

```
df[numCols].corr()
```

	Height(CM)	Weight(KG)	10th_Mark	12th_Mark	college_mark	salary_expectation	daily_studing_tim
Height(CM)	1.000000	0.275948	0.019938	-0.119618	-0.018535	-0.010708	
Weight(KG)	0.275948	1.000000	0.062977	-0.019133	-0.014901	-0.087787	
10th_Mark	0.019938	0.062977	1.000000	0.473254	0.465861	-0.055794	
12th_Mark	-0.119618	-0.019133	0.473254	1.000000	0.424828	-0.085623	
college_mark	-0.018535	-0.014901	0.465861	0.424828	1.000000	-0.103034	
salary_expectation	-0.010708	-0.087787	-0.055794	-0.085623	-0.103034	1.000000	
daily_studing_time_minutes	-0.160902	0.032652	0.023114	0.016854	-0.031191	-0.073046	
social_media_Minutes	0.012591	0.158480	-0.005240	0.008511	-0.029986	-0.025266	
Travelling_Time_Minutes	0.051755	0.051249	0.095286	0.035975	0.027092	-0.004790	

```
sns.heatmap(data=df[numCols].corr(), annot=True)
plt.title('Correlation Map')
```

```
Text(0.5, 1.0, 'Correlation Map')
```



Correlation Map
Height(CM)
Weight(KG)
10th_Mark
12th_Mark
college_mark
salary_expectation
daily_studing_time_minutes