1.Import necessary packages

In [54]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

2.Load the file

In [4]:

```
income_df=pd.read_csv(r"C:\Users\NAVEEN\OneDrive\Desktop\fds&ai\Sept clss\10th, 11th- Intro to Stats, Descriptive Sta
income_df
```

Out[4]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	Highest_Qualified_Member
0	5000	8000	3	2000	64200	Under-Graduate
1	6000	7000	2	3000	79920	Illiterate
2	10000	4500	2	0	112800	Under-Graduate
3	10000	2000	1	0	97200	Illiterate
4	12500	12000	2	3000	147000	Graduate
5	14000	8000	2	0	196560	Graduate
6	15000	16000	3	35000	167400	Post-Graduate
7	18000	20000	5	8000	216000	Graduate
8	19000	9000	2	0	218880	Under-Graduate
9	20000	9000	4	0	220800	Under-Graduate
10	20000	18000	4	8000	278400	Under-Graduate
11	22000	25000	6	12000	279840	Illiterate
12	23400	5000	3	0	292032	Illiterate
13	24000	10500	6	0	316800	Graduate
14	24000	10000	4	0	244800	Graduate
15	25000	12300	3	0	246000	Graduate
16	25000	20000	3	3500	261000	Graduate
17	25000	10000	6	0	258000	Under-Graduate
18	29000	6600	2	2000	348000	Graduate
19	30000	13000	4	0	385200	Graduate
20	30500	25000	5	5000	351360	Under-Graduate
21	32000	15000	4	0	445440	Professiona

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	Highest_Qualified_Membe
22	34000	19000	6	0	330480	Professiona
23	34000	25000	3	4000	469200	Professiona
24	35000	12000	3	0	466200	Graduate
25	35000	25000	4	0	449400	Professiona
26	39000	8000	4	0	556920	Under-Graduate
27	40000	10000	4	0	412800	Under-Graduate
28	42000	15000	4	0	488880	Graduate
29	43000	12000	4	0	619200	Graduate
30	45000	25000	6	0	523800	Graduate
31	45000	40000	6	3500	507600	Professiona
32	45000	10000	2	1000	437400	Post-Graduate
33	45000	22000	4	2500	610200	Post-Graduate
34	46000	25000	5	3500	596160	Graduate
35	47000	15000	7	0	456840	Professiona
36	50000	20000	4	0	570000	Professiona
37	50500	20000	3	0	581760	Professiona
38	55000	45000	6	12000	600600	Graduate
39	60000	10000	3	0	590400	Post-Graduate
40	60000	50000	6	10000	590400	Graduate
41	65000	20000	4	5000	647400	Illiterate
42	70000	9000	2	0	756000	Graduate
43	80000	20000	4	0	1075200	Graduate

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	Highest_Qualified_Member
44	85000	25000	5	0	1142400	Under-Graduate
45	90000	48000	7	0	885600	Post-Graduate
46	98000	25000	5	0	1152480	Professiona
47	100000	30000	6	0	1404000	Graduate
48	100000	50000	4	20000	1032000	Professiona
49	100000	40000	6	10000	1320000	Post-Graduate

In [5]:

```
income_df.head()
```

Out[5]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	Highest_Qualified_Member
0	5000	8000	3	2000	64200	Under-Graduate
1	6000	7000	2	3000	79920	Illiterate
2	10000	4500	2	0	112800	Under-Graduate
3	10000	2000	1	0	97200	Illiterate
4	12500	12000	2	3000	147000	Graduate

3.Analyze the data

In [7]:

```
income_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 7 columns):
 #   Column                  Non-Null Count  Dtype 
---  ------                  --------------  ----- 
 0   Mthly_HH_Income         50 non-null     int64 
 1   Mthly_HH_Expense        50 non-null     int64 
 2   No_of_Fly_Members       50 non-null     int64 
 3   Emi_or_Rent_Amt         50 non-null     int64 
 4   Annual_HH_Income        50 non-null     int64 
 5   Highest_Qualified_Member  50 non-null     object
 6   No_of_Earning_Members   50 non-null     int64 
dtypes: int64(6), object(1)
memory usage: 2.9+ KB
```

In [8]:
```
income_df.shape
```

Out[8]:
```
(50, 7)
```

In [9]:
```
income_df.describe().T
```

Out[9]:

	count	mean	std	min	25%	50%	75%	max
Mthly_HH_Income	50.0	41558.00	26097.908979	5000.0	23550.0	35000.0	50375.0	100000.0
Mthly_HH_Expense	50.0	18818.00	12090.216824	2000.0	10000.0	15500.0	25000.0	50000.0
No_of_Fly_Members	50.0	4.06	1.517382	1.0	3.0	4.0	5.0	7.0
Emi_or_Rent_Amt	50.0	3060.00	6241.434948	0.0	0.0	0.0	3500.0	35000.0
Annual_HH_Income	50.0	490019.04	320135.792123	64200.0	258750.0	447420.0	594720.0	1404000.0
No_of_Earning_Members	50.0	1.46	0.734291	1.0	1.0	1.0	2.0	4.0

In [10]:
```
income_df.isna().any()
```

Out[10]:

```
Mthly_HH_Income            False
Mthly_HH_Expense           False
No_of_Fly_Members          False
Emi_or_Rent_Amt            False
Annual_HH_Income           False
Highest_Qualified_Member   False
No_of_Earning_Members      False
dtype: bool
```

No null values in the dataset

4.What is the Mean Expense of a Household?

In [13]:

```
income_df["Mthly_HH_Expense"].mean()
```

Out[13]:

```
18818.0
```

5.What is the Median Household Expense?

In [15]:

```
income_df["Mthly_HH_Expense"].median()
```

Out[15]:

```
15500.0
```

6.What is the Monthly Expense for most of the Households?

In [17]:

```
mth_exp_tmp = pd.crosstab(index=income_df["Mthly_HH_Expense"], columns="count")
mth_exp_tmp.reset_index(inplace=True)
mth_exp_tmp[mth_exp_tmp['count'] == income_df.Mthly_HH_Expense.value_counts().max()]
```

Out[17]:

col_0	Mthly_HH_Expense	count
18	25000	8

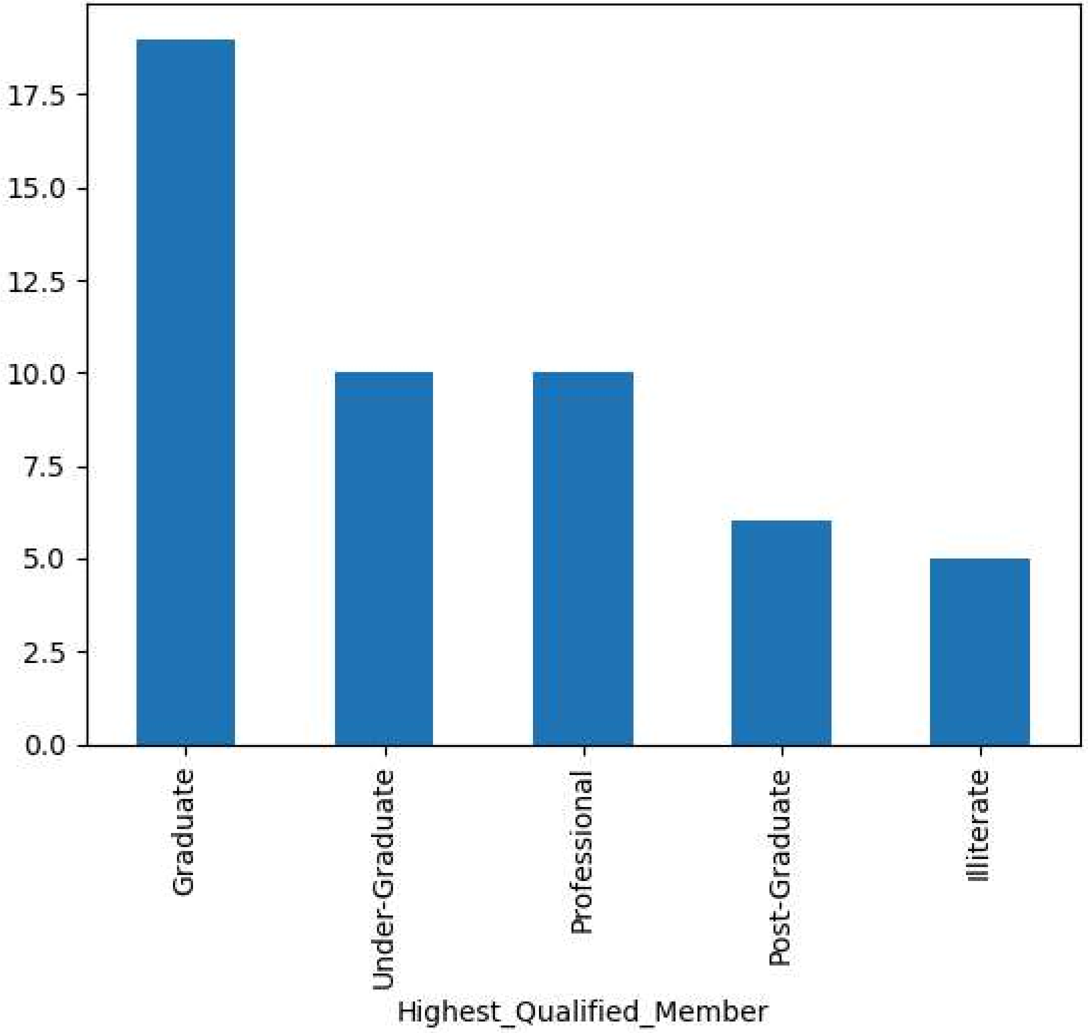
7.Plot the Histogram to count the Highest qualified member

In [19]:

```
income_df["Highest_Qualified_Member"].value_counts().plot(kind="bar")
```

Out[19]:

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



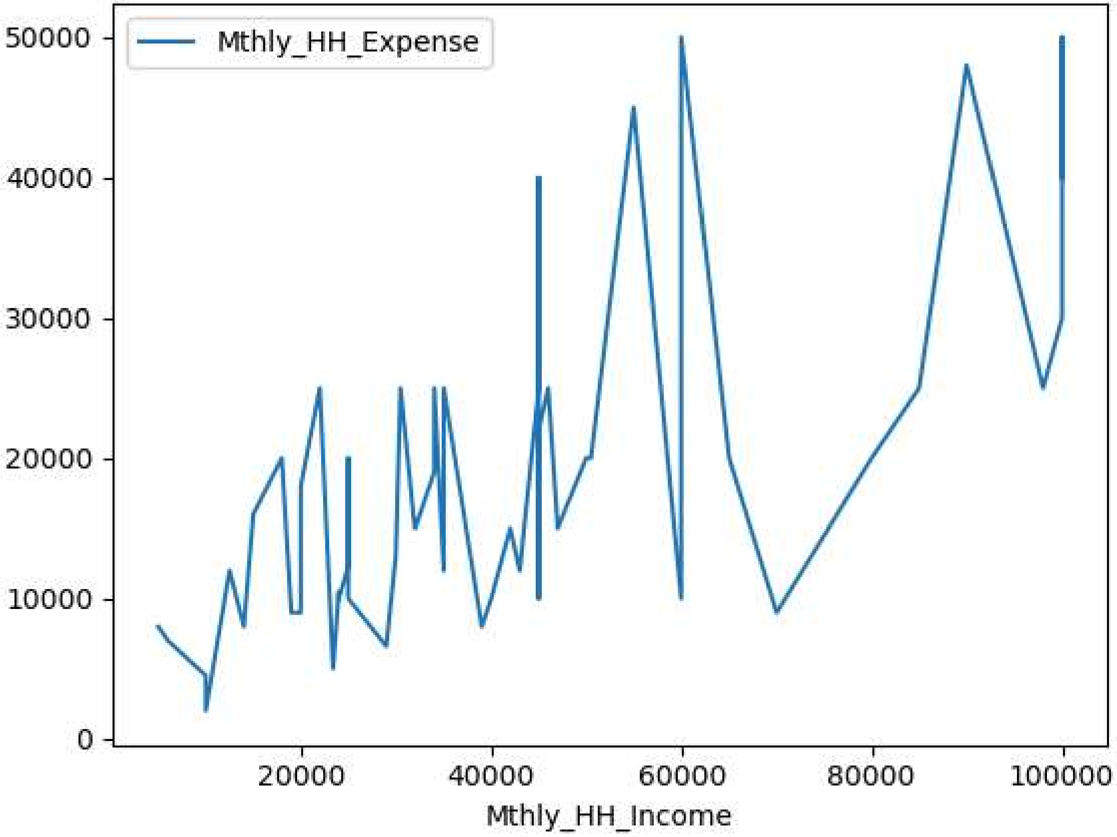
8.Calculate IQR(difference between 75% and 25% quartile)

In [21]:

```
income_df.plot(x="Mthly_HH_Income", y="Mthly_HH_Expense")
IQR=income_df["Mthly_HH_Expense"].quantile(0.75)-income_df["Mthly_HH_Expense"].quantile(0.25)
IQR
```

Out[21]:

```
15000.0
```



9.Calculate Standard Deviation for first 4 columns.

In [23]:

```
pd.DataFrame(income_df.iloc[:,0:5].std().to_frame()).T
```

Out[23]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income
0	26097.908979	12090.216824	1.517382	6241.434948	320135.792123

10.Calculate Variance for first 3 columns.

In [25]:

```
pd.DataFrame(income_df.iloc[:,0:4].var().to_frame()).T
```

Out[25]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt
0	6.811009e+08	1.461733e+08	2.302449	3.895551e+07

11.Calculate the count of Highest qualified member.

In [30]:

```
income_df["Highest_Qualified_Member"].value_counts().to_frame().T
```

Out[30]:

Highest_Qualified_Member	Graduate	Under-Graduate	Professional	Post-Graduate	Illiterate
count	19	10	10	6	5

11.Calculate the count of Highest qualified member.

In [39]:

```
income_df["Highest_Qualified_Member"].value_counts().to_frame().T
```

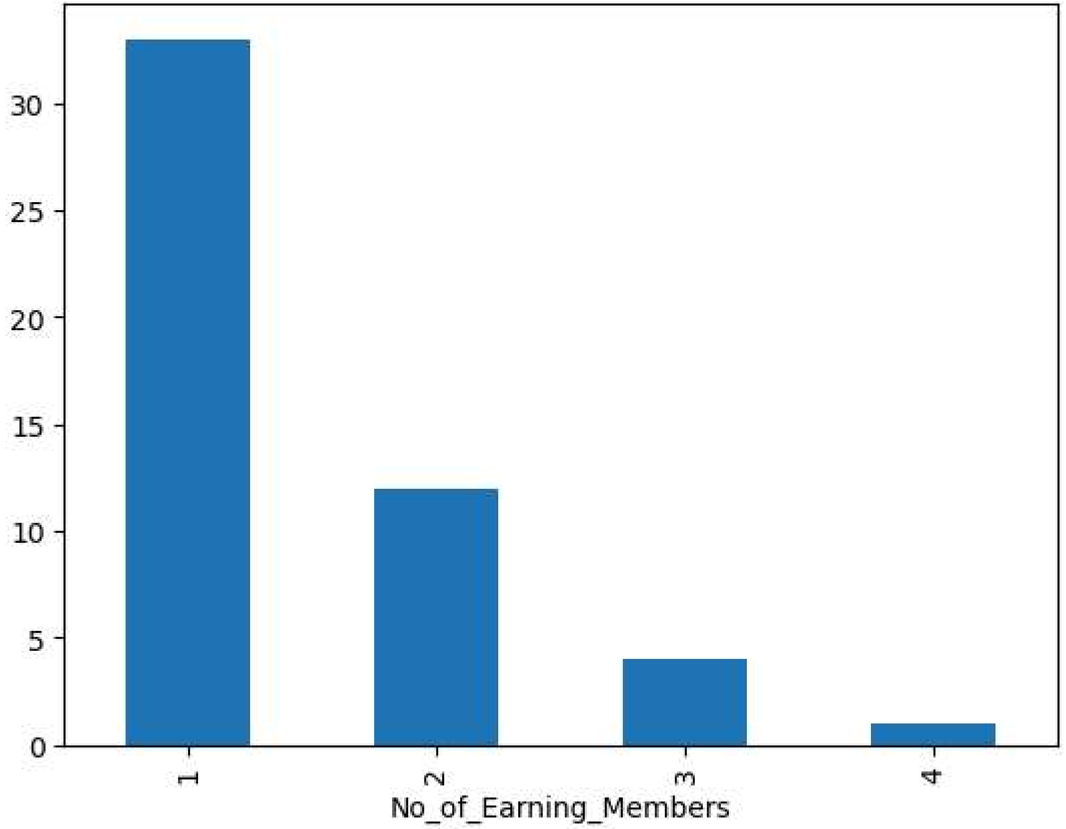
Out[39]:

Highest_Qualified_Member	Graduate	Under-Graduate	Professional	Post-Graduate	Illiterate
count	19	10	10	6	5

12.Plot the Histogram to count the No_of_Earning_Members

```
In [42]: income_df["No_of_Earning_Members"].value_counts().plot(kind="bar")
```

```
Out[42]: <Axes: xlabel='No_of_Earning_Members'>
```



13.Suppose you have option to invest in Stock A or Stock B. The stocks • have different expected returns and standard deviations. The expected return of Stock A is 15% and Stock B is 10%. Standard Deviation of the returns of these stocks is 10% and 5% respectively.

Which is better investment?

In [48]:

```python
#Here we need to calculate the coeff of variation

Coeff_of_var_StockA=10/15
print(Coeff_of_var_StockA)
Coeff_of_var_StockB=5/10
print(Coeff_of_var_StockB)
```

```
0.6666666666666666
0.5
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: