

```
In [13]: import pandas as pd;
import numpy as np;
```

2(a) Write a python program to load data into pandas DataFrame

```
In [2]: #df is dataframe where we store all imported data from csv
df = pd.read_csv('data/scienceSalaries_90157 (3).csv');
df.head()
```

```
Out[2]:
```

	work_year	experience_level	employment_type	job_title	salary	salary_currency	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2023	SE	FT	Principal Data Scientist	80000	EUR	85847	ES	100	ES	L
1	2023	MI	CT	ML Engineer	30000	USD	30000	US	100	US	S
2	2023	MI	CT	ML Engineer	25500	USD	25500	US	100	US	S
3	2023	SE	FT	Data Scientist	175000	USD	175000	CA	100	CA	M
4	2023	SE	FT	Data Scientist	120000	USD	120000	CA	100	CA	M

2(b) Write a python program to remove unnecessary columns i.e., salary and salary currency

```
In [3]: # we are using DataFrame salary.drop() to remove multiple columns salary and salary currency
#the pandas allow us to use key word like columns to drop columns and index to drop row.
#inplace=True induce change in our original DataFrame
df.drop(columns=['salary','salary_currency'],inplace=True);
```

```
In [4]: df.head(10)
```

```
Out[4]:
```

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
0	2023	SE	FT	Principal Data Scientist	85847	ES	100	ES	L
1	2023	MI	CT	ML Engineer	30000	US	100	US	S
2	2023	MI	CT	ML Engineer	25500	US	100	US	S
3	2023	SE	FT	Data Scientist	175000	CA	100	CA	M
4	2023	SE	FT	Data Scientist	120000	CA	100	CA	M
5	2023	SE	FT	Applied Scientist	222200	US	0	US	L
6	2023	SE	FT	Applied Scientist	136000	US	0	US	L
7	2023	SE	FT	Data Scientist	219000	CA	0	CA	M
8	2023	SE	FT	Data Scientist	141000	CA	0	CA	M
9	2023	SE	FT	Data Scientist	147100	US	0	US	M

2(c) Write a python program to check duplicates value in the dataframe.

```
In [5]: # duplicated() method will give boolean series showing true if a row have duplicate values.
df(df.duplicated())
```

```
Out[5]:
```

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
115	2023	SE	FT	Data Scientist	150000	US	0	US	M
123	2023	SE	FT	Analytics Engineer	289800	US	0	US	M
153	2023	MI	FT	Data Engineer	100000	US	100	US	M
164	2023	MI	FT	Data Engineer	70000	US	100	US	M
160	2023	SE	FT	Data Engineer	115000	US	0	US	M
...
3439	2022	MI	FT	Data Scientist	78000	US	100	US	M
3440	2022	SE	FT	Data Engineer	135000	US	100	US	M
3441	2022	SE	FT	Data Engineer	115000	US	100	US	M
3566	2021	MI	FT	Data Engineer	200000	US	100	US	L
3709	2021	MI	FT	Data Scientist	90734	DE	50	DE	L

1171 rows x 9 columns

2(d)Write a python program to remove the NaN missing values from updated dataframe.

```
In [6]: # dropna removes row having nan value.
df.dropna(inplace=True)
```

```
In [7]: df.isna().sum()
```

```
Out[7]:
```

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
work_year	0								
experience_level	0								
employment_type	0								
job_title	0								
salary_in_usd	0								
employee_residence	0								
remote_ratio	0								
company_location	0								
company_size	0								
dtype:	int64								

2(e) Write a python program to see the unique values from all the columns in the dataframe

```
In [9]: for column in df.columns: #df.columns give us column in array so we looping to get each column name inside of array.
all_unique = df[column].unique() # we get all unique value of columns that are in DataFrame df.
print(f" unique value in {column} column is {all_unique}") # displaying unique value of each column in array
```

unique value in work_year column is [2023 2022 2020 2021]

unique value in experience_level column is ['SE' 'MI' 'EN' 'EX']

unique value in employment_type column is ['FT' 'CT' 'FL' 'PT']

unique value in job_title column is ['Principal Data Scientist' 'ML Engineer' 'Data Scientist' 'Applied Scientist' 'Data Analyst' 'Data Modeler' 'Research Engineer' 'Analytics Engineer' 'Business Intelligence Engineer' 'Machine Learning Engineer' 'Data Strategist' 'Data Engineer' 'Computer Vision Engineer' 'Data Quality Analyst' 'Compliance Data Analyst' 'Data Architect' 'Applied Machine Learning Engineer' 'AI Developer' 'Research Scientist' 'Autonomous Vehicle Technician' 'Applied Machine Learning Scientist' 'Lead Data Scientist' 'Cloud Database Engineer' 'Financial Data Analyst' 'Data Infrastructure Engineer' 'Software Data Engineer' 'AI Programmer' 'Data Operations Engineer' 'BI Developer' 'Data Science Lead' 'Deep Learning Researcher' 'BI Analyst' 'Data Science Consultant' 'Data Analytics Specialist' 'Machine Learning Infrastructure Engineer' 'BI Data Analyst' 'Head of Data Science' 'Insight Analyst' 'Deep Learning Engineer' 'Machine Learning Software Engineer' 'Big Data Architect' 'Product Data Analyst' 'Computer Vision Software Engineer' 'Azure Data Engineer' 'Marketing Data Engineer' 'Data Analytics Lead' 'Data Lead' 'Data Science Engineer' 'Machine Learning Research Engineer' 'ML Engineer' 'Manager Data Management' 'Machine Learning Developer' '3D Computer Vision Researcher' 'Principal Machine Learning Engineer' 'Data Analytics Engineer' 'Data Analytics Consultant' 'Data Management Specialist' 'Data Science Tech Lead' 'Data Scientist Lead' 'Cloud Data Engineer' 'Data Operations Analyst' 'Marketing Data Analyst' 'Power BI Developer' 'Product Data Scientist' 'Principal Data Architect' 'Machine Learning Manager' 'Lead Machine Learning Engineer' 'ETL Developer' 'Cloud Data Architect' 'Data Data Engineer' 'Head of Machine Learning' 'Principal Data Analyst' 'Principal Data Engineer' 'Staff Data Scientist' 'Finance Data Analyst']

unique value in salary_in_usd column is [85847 30000 25500 ... 28309 432000 94665]

unique value in employee_residence column is ['ES' 'US' 'CA' 'DE' 'GB' 'NG' 'IN' 'HK' 'PT' 'NL' 'CH' 'CF' 'FR' 'AU' 'FI' 'UA' 'IE' 'IL' 'GH' 'AT' 'CO' 'SG' 'SE' 'SI' 'MX' 'UZ' 'BR' 'TH' 'HR' 'PL' 'KW' 'VN' 'CY' 'AR' 'AM' 'BA' 'KE' 'GR' 'MK' 'LV' 'RO' 'PK' 'ID' 'MA' 'LT' 'BG' 'AS' 'IP' 'HU' 'GM' 'CZ' 'CR' 'TR' 'CL' 'PE' 'DK' 'BO' 'PH' 'DO' 'EG' 'ID' 'AE' 'MY' 'JP' 'EE' 'HN' 'TN' 'RU' 'DZ' 'IQ' 'BG' 'AE' 'BS' 'NZ' 'MO' 'LU' 'MT']

unique value in remote_ratio column is [100 0 50]

unique value in company_location column is ['ES' 'US' 'CA' 'DE' 'GB' 'NG' 'IN' 'HK' 'NL' 'CH' 'CF' 'FR' 'FI' 'UA' 'IE' 'IL' 'GH' 'CO' 'SG' 'AU' 'SE' 'SI' 'MX' 'BR' 'PT' 'RU' 'TH' 'HR' 'VN' 'EE' 'AM' 'BA' 'KE' 'GR' 'MK' 'LV' 'RO' 'PK' 'ID' 'MA' 'PL' 'AL' 'AR' 'LT' 'AS' 'CR' 'IR' 'BS' 'HU' 'AT' 'SK' 'CZ' 'TR' 'PR' 'DK' 'BO' 'PH' 'BE' 'ID' 'EG' 'AE' 'LU' 'MY' 'HN' 'JP' 'DZ' 'IQ' 'CN' 'NZ' 'CL' 'MO' 'MT']

unique value in company_size column is ['L' 'S' 'M']

2(f) Rename the experience level columns as below. SE – Senior Level/Expert,MI – Medium Level/Intermediate,EN – Entry Level,EX – Executive Level

```
In [9]: df["experience_level"]
```

```
Out[9]:
```

	experience_level
0	SE
1	MI
2	MI
3	SE
4	SE
...	...
3750	MI
3751	MI
3752	EN
3753	EN
3754	SE

Name: experience_level, Length: 3755, dtype: object

```
In [10]: #we use df["column"].replace method to replace value
df["experience_level"].replace(to_replace=["SE","MI","EN","EX"],value=["Senior Level/Expert","Medium Level/Intermediate","Entry Level","Executive Level"],inplace=True);
```

```
In [11]: df["experience_level"]
```

```
Out[11]:
```

	experience_level
0	Senior Level/Expert
1	Medium Level/Intermediate
2	Medium Level/Intermediate
3	Senior Level/Expert
4	Senior Level/Expert
...	...
3750	Senior Level/Expert
3751	Medium Level/Intermediate
3752	Entry Level
3753	Entry Level
3754	Senior Level/Expert

Name: experience_level, Length: 3755, dtype: object

3(a) Write a Python program to show summary statistics of sum, mean,standard deviation, skewness, and kurtosis of any chosen variable.

```
In [12]: #describe() method gives basic statistical details like mean,std,min,max etc of DataFrame Column.
sum_statistics=df["salary_in_usd"].describe()
```

sum_kurtosis = df["salary_in_usd"].kurtosis() # calculating kurtosis of column salary_in_usd

sum_skew = df["salary_in_usd"].skew() # calculating skewness of column salary_in_usd

sum_statistics["kurtosis"]=sum_kurtosis # creating kurtosis column in DataFrame sum_statistics

sum_statistics["skewness"]=sum_skew # creating skewness column in DataFrame sum_statistics

```
In [13]: sum_statistics
```

```
Out[13]:
```

	count	mean	std	min	25%	50%	75%	max	kurtosis	skewness
salary_in_usd	3755.000000	137579.399890	63955.625278	5132.000000	95000.000000	135000.000000	175000.000000	450000.000000	0.834005	0.530401

Name: salary_in_usd, dtype: float64

3(b) Write a Python program to calculate and show correlation of all variables.

```
In [14]: df.dtypes # we are using dtype to get datatypes of columns with in DataFrame df.
```

```
Out[14]:
```

	work_year	experience_level	employment_type	job_title	salary_in_usd	employee_residence	remote_ratio	company_location	company_size
work_year	int64								
experience_level	object								
employment_type	object								
job_title	object								
salary_in_usd	int64								
employee_residence	object								
remote_ratio	int64								
company_location	object								
company_size	object								
dtype:	object								

```
In [15]: nums=df.select_dtypes(["int64"]) # select_dtypes() selects the column according to its datatypes.
nums.columns
```

Index(['work_year', 'salary_in_usd', 'remote_ratio'], dtype='object')

```
Out[15]:
```

show_corr = nums.corr() #shows correlation between diffrent variables of numeric datatypes.

show_corr

```
Out[16]:
```

	work_year	salary_in_usd	remote_ratio
work_year	1.00000	0.228290	-0.236430
salary_in_usd	0.22829	1.000000	-0.064171
remote_ratio	-0.23643	-0.064171	1.000000

4(a) Write a python program to find out top 15 jobs. Make a bar graph of sales as well

```
In [17]: import matplotlib.pyplot as plt
```

```
In [18]: # value_count() is same as groupby.counts()
# we are selecting 15 greatest number of jobs in table according its frequency in column
top_15 =df["job_title"].value_counts().nlargest(15)
top_15
```

```
Out[18]:
```

job_title	count, dtype: int64
Data Engineer	1040
Data Scientist	840
Data Analyst	632
Machine Learning Engineer	280
Analytics Engineer	103
Data Architect	101
Research Scientist	82
Data Science Manager	58
Applied Scientist	58
Research Engineer	37
ML Engineer	34
Data Manager	29
Machine Learning Scientist	26
Data Science Consultant	24
Data Analytics Manager	22

Name: count, dtype: int64

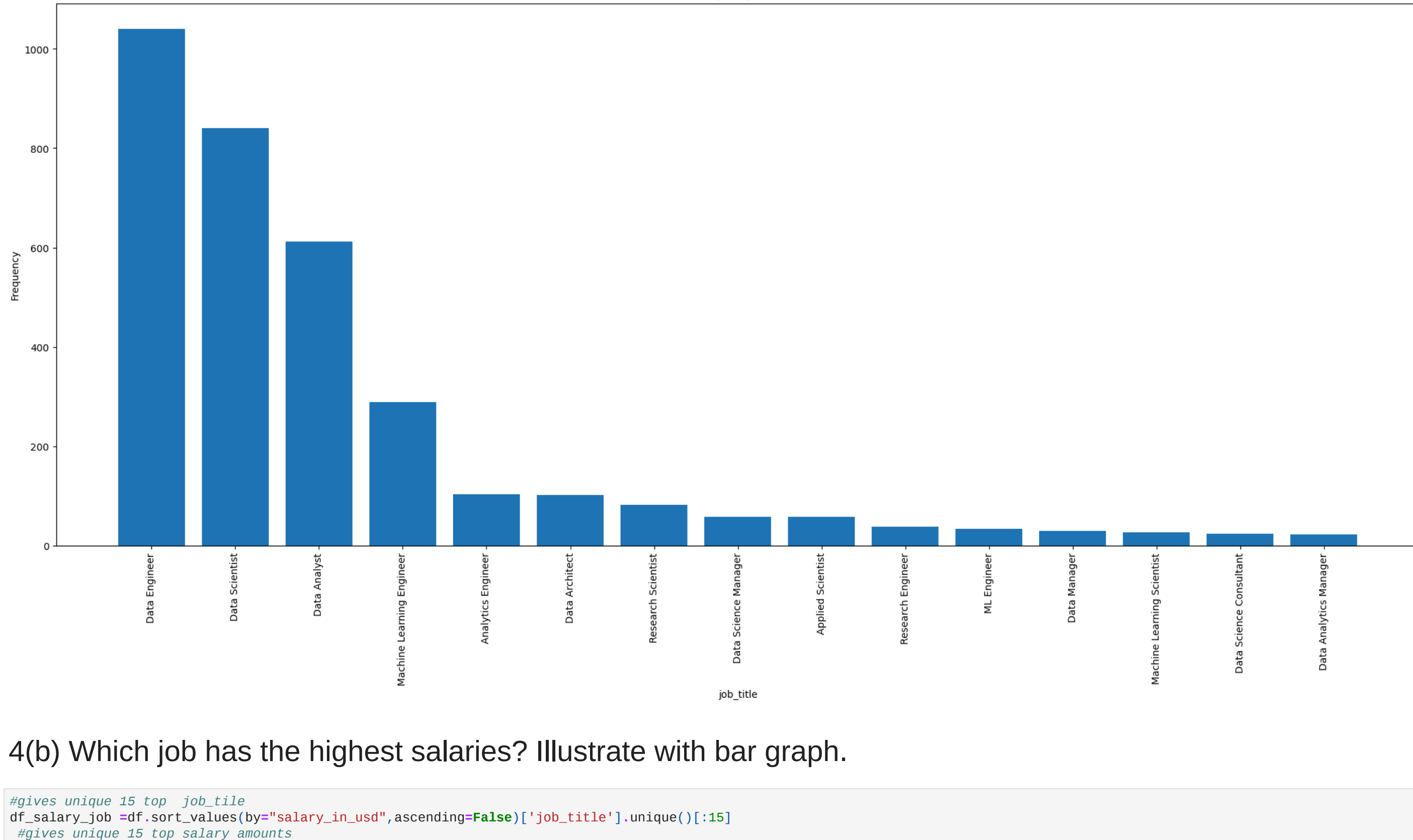
```
In [19]: #job_title is our index and we converting it in list for bar plot
x=top_15.index.tolist()
```

#values are no of count of jobs in table

#top_15.values.tolist()

#bar plot code

```
plt.figure(figsize=(25,10)) #
plt.xticks(rotation=90)
plt.xlabel("job_title") # Giving X-axis label
plt.ylabel("Frequency") # Giving Y-axis label
plt.title("top 15 jobs") # Giving title in top of figure
plt.bar(x,h,align="center") # using bar graph method of matplotlib to create bar graph.
plt.show()
```



4(b) Which job has the highest salaries? Illustrate with bar graph.

```
In [20]: #Gives unique 15 top job_title
df.salary_job =df.sort_values(by="salary_in_usd",ascending=False)["job_title"].unique()[1:15]
df.salary_job
```

#Gives unique 15 top salary amounts

df.salary_amount = df.sort_values(by="salary_in_usd",ascending=False)["salary_in_usd"].unique()[1:15]

```
In [21]: x=df.salary_job
#df.salary_amount
plt.figure(figsize=(20,8)) #help to change size of figure plot using matplotlib
plt.xlabel("Job Titles") # gives label in x axis of bargraph
plt.ylabel("Salary") # gives label in y axis of bargraph
plt.title("job with highest salaries") # gives title in top of figure
plt.xticks(rotation=90)
plt.bar(x,h); # Plotting Bar Graph
plt.show()
```



4(c) Write a python program to find out salaries based on experience level. Illustrate it through bar graph.

```
In [22]: # we use groupby function on 'experience_level' and then apply describe()
# and we select salary_column which provides summary statistics for the 'salary_in_usd' column for each 'experience_level'

# Next, we use .loc[""] to access statistics for all unique values with in column experience level
# Specifically, we're interested in the mean salary for all unique values with in column experience level so we use mean()

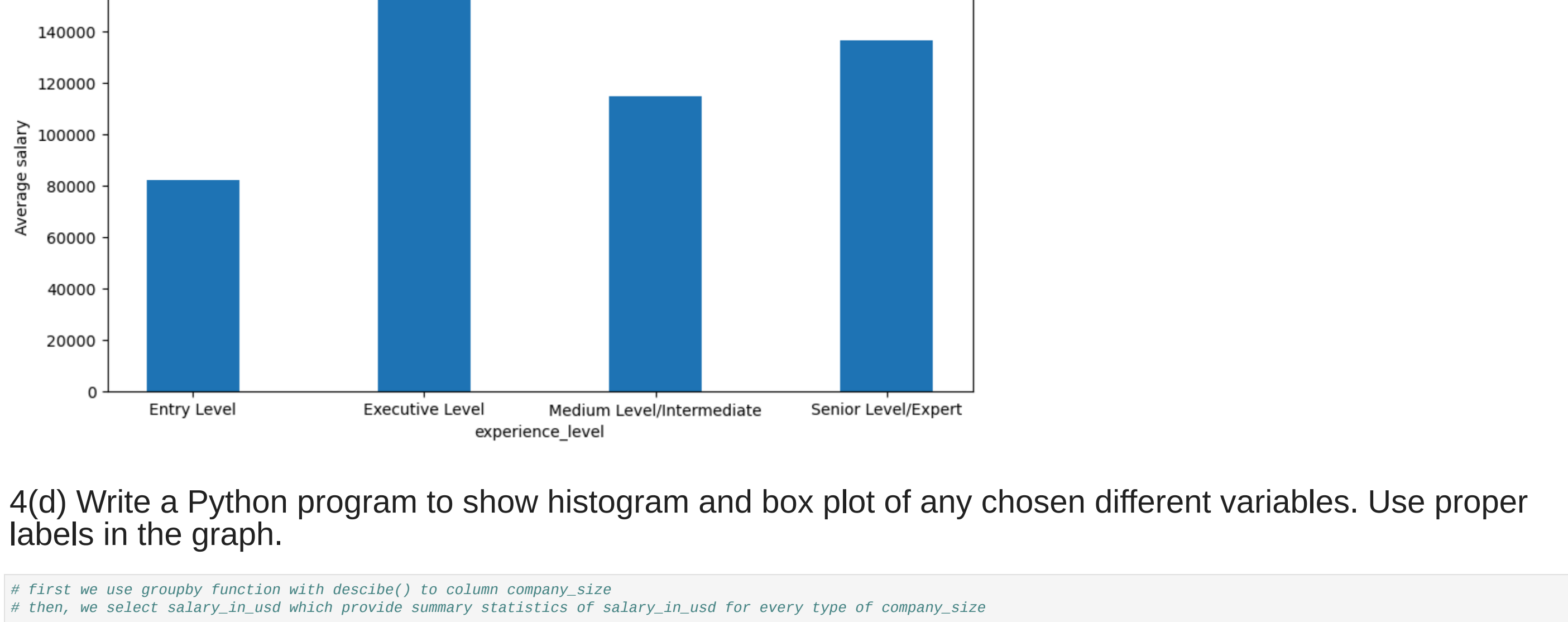
Entry_Level = df.groupby("experience_level").describe()["salary_in_usd"].loc["Entry Level"].mean()
Medium_Level = df.groupby("experience_level").describe()["salary_in_usd"].loc["Executive Level"].mean()
Medium_Intermediate_Level = df.groupby("experience_level").describe()["salary_in_usd"].loc["Medium Level/Intermediate"].mean()
Senior_Expert_Level = df.groupby("experience_level").describe()["salary_in_usd"].loc["Senior Level/Expert"].mean()
```

```
In [23]: #we use bar function that we get from matplotlib we imported earlier.

x=["Entry Level","Executive Level","Medium Level/Intermediate","Senior Level/Expert"]
#inserting all mean of different level of experience column
h=[Entry_Level,Medium_Level,Medium_Intermediate_Level,Senior_Expert_Level]

plt.figure(figsize=(10,5)) #help us to give height and width of figure
plt.xlabel("Average salary") # gives label in x-axis of bargraph
plt.ylabel("experience_level") # gives label in y axis of bargraph
plt.title("salaries based on experience level")
# we are giving name of each bar of experience levels as x-axis
# we are giving array of means 'h' as y-axis value
plt.bar(x,h,0.4)

plt.show()#Display the plot
```



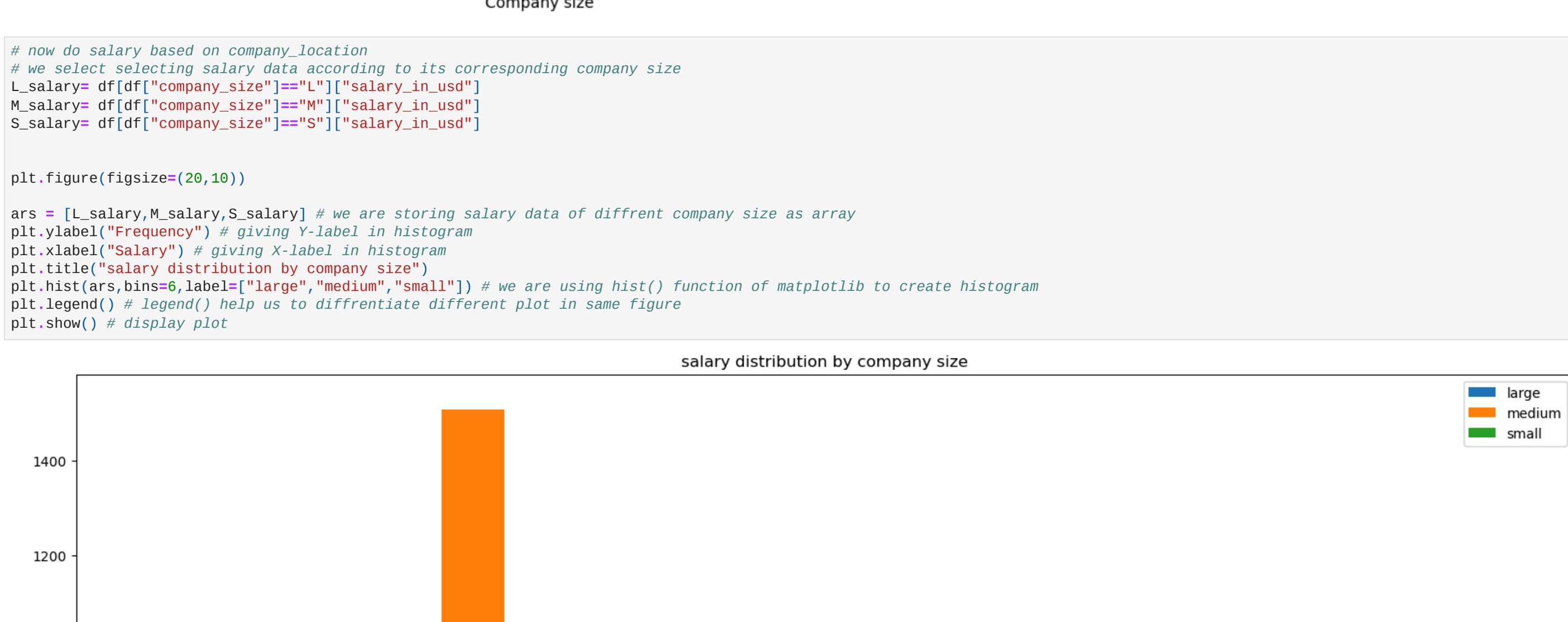
4(d) Write a Python program to show histogram and box plot of any chosen different variables. Use proper labels in the graph.

```
In [24]: # first we use groupby function with describe() to column company_size
# then, we select salary_in_usd which provide summary statistics of salary_in_usd for every type of company_size

largeCompany = df.groupby("company_size").describe()["salary_in_usd"].loc["L"]
mediumCompany = df.groupby("company_size").salary_in_usd.describe().loc["M"]
smallCompany = df.groupby("company_size").salary_in_usd.describe().loc["S"]

#we creating array size data for summary statistics of every type of company_size
size_data = [largeCompany,mediumCompany,smallCompany]

plt.figure(figsize=(10,5))
plt.ylabel("salary") # gives label in Y-axis
plt.xlabel("Company size") # gives label in X-axis
plt.title("salary based on company size") # gives title in figure
# we use matplotlib boxplot to create box plot for different type of company size.
plt.boxplot(size_data,labels=["Large","Medium","Small"],patch_artist=True)
plt.show() #displays the plot
```



```
In [25]: # now do salary based on company_location
# we select selecting salary data according to its corresponding company size
M_salary = df[df["company_size"]=="M"]["salary_in_usd"]
S_salary = df[df["company_size"]=="S"]["salary_in_usd"]

plt.figure(figsize=(20,10))

ars = [L_salary,M_salary,S_salary] # we are storing salary data of different company size as array
plt.ylabel("Frequency") # giving Y-label in histogram
plt.xlabel("Salary") # giving X-label in histogram
plt.title("salary distribution by company size")
# we are using hist() function of matplotlib to create histogram
plt.legend() # legend() help us to differentiate different plot in same figure
plt.show()
```

