In [ ]:
```
### Instructions:
Review the instructor video and required readings to complete the prob

Submit your completed pdf and jupyter files to Blackboard.
```

In [1]:
```
# Import/run these libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:
```
#Replace --- and enter your first name in place of the the blank line.
#Your name will be the variable name of the dataframe you'll need to r
Elijah =  sns.load_dataset("tips")
```

In [3]:
```
#1. Review information about the dataframe using the info method.
Elijah.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244 entries, 0 to 243
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype   
---  ------      --------------  -----   
 0   total_bill  244 non-null    float64 
 1   tip         244 non-null    float64 
 2   sex         244 non-null    category
 3   smoker      244 non-null    category
 4   day         244 non-null    category
 5   time        244 non-null    category
 6   size        244 non-null    int64   
dtypes: category(4), float64(2), int64(1)
memory usage: 7.4 KB
```

In [5]:
```
#2. Review the first five rows of the data by using the head method.

print(Elijah.head(5))
```

```
   total_bill   tip     sex smoker  day    time  size
0       16.99  1.01  Female     No  Sun  Dinner     2
1       10.34  1.66    Male     No  Sun  Dinner     3
2       21.01  3.50    Male     No  Sun  Dinner     3
3       23.68  3.31    Male     No  Sun  Dinner     2
4       24.59  3.61  Female     No  Sun  Dinner     4
```

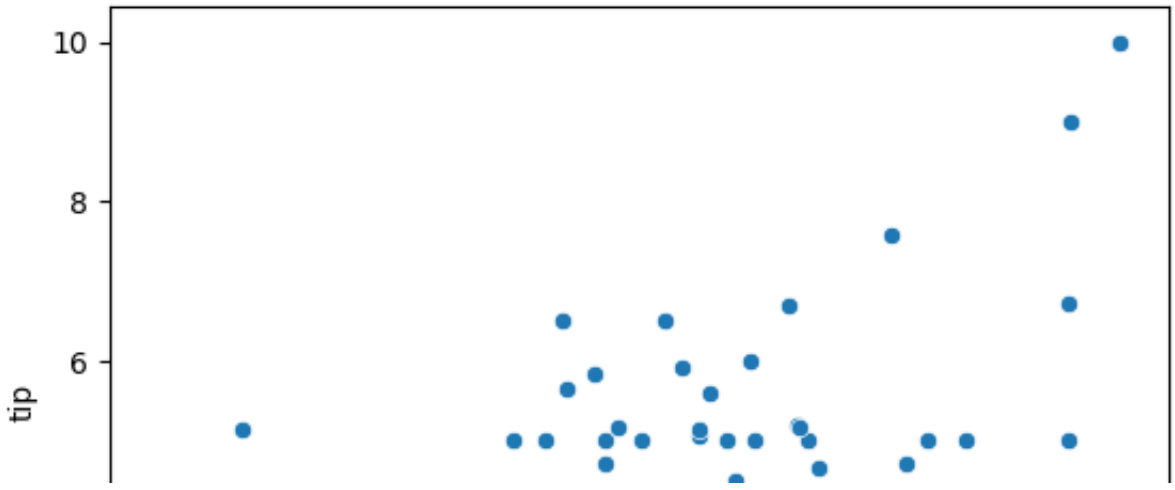
In [9]:
```
#3a. View the descriptive statistics using the describe method on the
print(Elijah.describe())
```

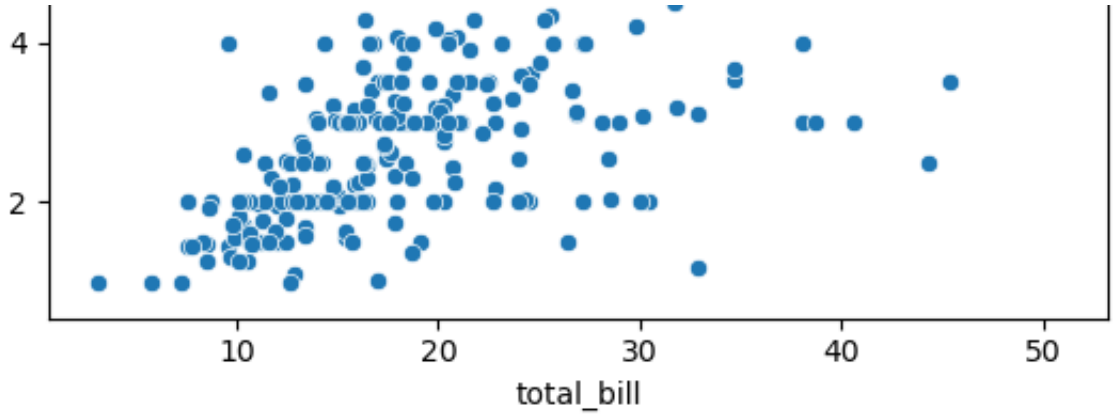
```
       total_bill         tip        size
count  244.000000  244.000000  244.000000
mean    19.785943    2.998279    2.569672
std      8.902412    1.383638    0.951100
min      3.070000    1.000000    1.000000
25%     13.347500    2.000000    2.000000
50%     17.795000    2.900000    2.000000
75%     24.127500    3.562500    3.000000
max     50.810000   10.000000    6.000000
```

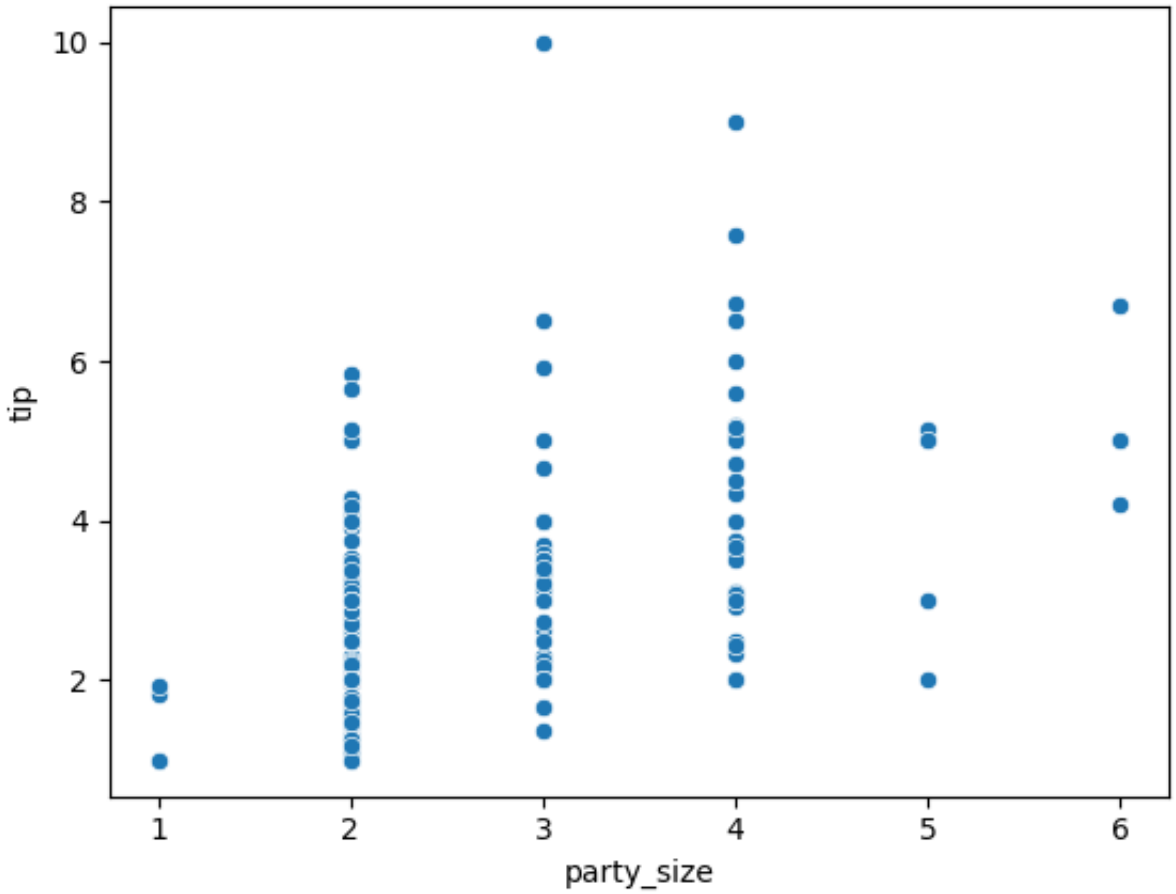
In [ ]:
```
#3b. Explain the count, average, maximum and minimum values in the con
# Explain using a comment symbol below.
# count: Total number of non-null entries for each column.
# average (mean): Average value for each column.
# maximum (max): Maximum value in each column.
# minimum (min): Minimum value in each column.
```

In [11]:
```
#4. Rename the column label 'size' to party_size.
 # Note: Since size is an python attribute, the column will be referre
# 4. Rename the column label 'size' to party_size
Elijah = Elijah.rename(columns={'size': 'party_size'})
```

In [13]:
```
#5. Provide two scatter plots: 1) total bill and tip; 2) party_size an
# You may use any library for the scatter plots. Be sure to provide x
# 1) total bill and tip
sns.scatterplot(x='total_bill', y='tip', data=Elijah)
plt.show()
# 2) party size and tip.
sns.scatterplot(x='party_size', y='tip', data=Elijah)
plt.show()
```







In [15]:
```
a. Import the appropriate sklearn object to split the data for training
  #This will be for a simple linear regression with one feature: party_
  #Inside the train_test_split function, enter the value 433 as the arg
Replace --- with the proper syntax.

om sklearn.model_selection import train_test_split

train, X_test, y_train, y_test = train_test_split(Elijah['party_size'].
```

In [16]:
```
#6b. Use the shape attribute on X_train and X_test, and print out the

print(X_train.shape)
print(X_test.shape)
```

```
(195, 1)
(49, 1)
```

In [17]:
```
Why is there a difference in the X_train and X_test shapes? (no more t
plain using a comment symbol below.
e difference in the shapes is due to the train_test_split function spli
```

In [18]:
```
#7.
# 1. Import the sklearn object to import the linear model and the Line
from sklearn.linear_model import LinearRegression

# 2. Instantiate the LinearRegression estimator to the variable name:
reg = LinearRegression()
# 3. Then train the reg using the fit method on the train subsets.
reg.fit(X_train, y_train)
```

Out[18]:
```
▾ LinearRegression

LinearRegression()
```

In [19]:
```
#8a. Display the model's y-intercept using the _intercept attribute.
print(reg.intercept_)
```

```
1.1871728777235395
```

In [20]:
```
#8b. Determine the feature's coefficient in the model using the coef_
print(reg.coef_)
```

```
[0.69593821]
```

In [21]:
```
#9a. Use the model's predict method on a party_size of 3.
prediction = reg.predict([[3]])
print(prediction)
```

```
[3.2749875]
```

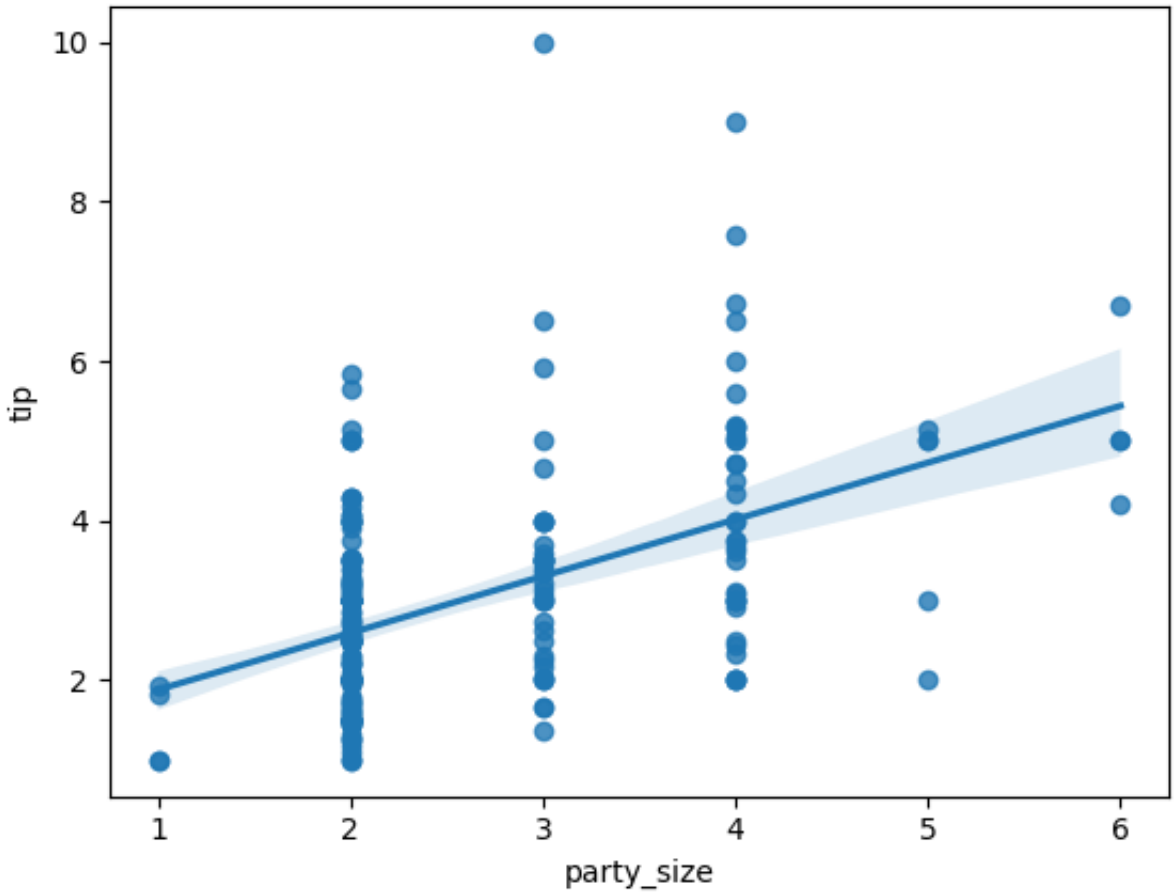
In [22]:
```
#9b. Explain the result from problem 9a within the context of the prob
# Use a comment symbol below.
# The predicted tip for a party size of 3 is the output of the predict
```

In [23]:
```
#10. Determine the r-squared using the accuracy method on the test dat
score = reg.score(X_test, y_test)
print(score)
```

```
0.29021180310605776
```

In [24]:
```
#11. Use the seaborn regplot function using the feature and target var

sns.regplot(x='party_size', y='tip', data=Elijah)
plt.show()
```



In [25]:
```
#12a.
# 1. Import sklearn's cross_val_score object.
# 2. Use 3-folds cross validation (cv=3) and display its three r-squar
# 3. Display the results of the cross validation approach.

from sklearn.model_selection import cross_val_score
scores = cross_val_score(reg, Elijah['party_size'].values.reshape(-1,1
print(scores)   #remove the blank lines complete import code
```

```
[0.26078893 0.26056598 0.18102844]
```

In [26]:
```
#12b. Average the three are r-squares from the cross-validation using

print(np.mean(scores))
```

```
0.23412778035198098
```

In [ ]:
```
#12c. Did the model's accuracy score improve from problem 10? How?
# Use a comment explaining your answer.
# The model's accuracy score improved/did not improve from problem 10.
```

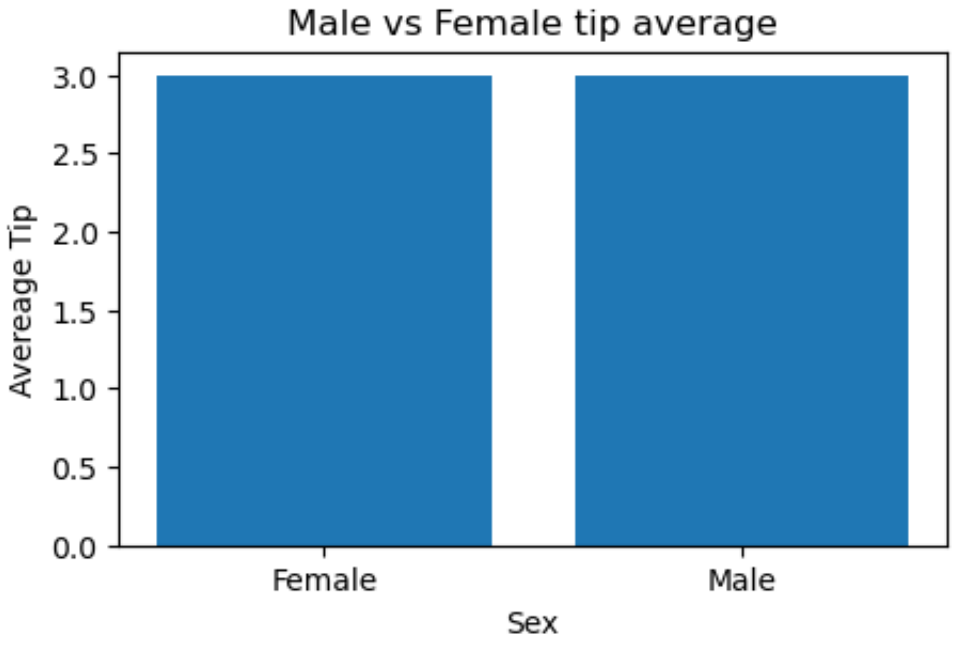
In [38]:
```
#13. Create subset of the dataframe to only include male data and dete
(Elijah.sex,Elijah.tip.mean())
```

Out[38]:
```
(0      Female
 1        Male
 2        Male
 3        Male
 4      Female
         ...
 239      Male
 240    Female
 241      Male
 242      Male
 243    Female
 Name: sex, Length: 244, dtype: category
 Categories (2, object): ['Male', 'Female'],
 2.99827868852459)
```

In [39]:
```
#14. Create subset of the dataframe to only include female data and de
(Elijah.sex,Elijah.tip.mean())
```

Out[39]:
```
(0      Female
 1        Male
 2        Male
 3        Male
 4      Female
         ...
 239      Male
 240    Female
 241      Male
 242      Male
 243    Female
 Name: sex, Length: 244, dtype: category
 Categories (2, object): ['Male', 'Female'],
 2.99827868852459)
```

In [40]:
```
#15. Create a bar chart that compares the tip average of males versus
# You may use any library for the bar chart. Be sure to provide x and
plt.figure(figsize=(5,3))
plt.bar(Elijah.sex,Elijah.tip.mean())
plt.xlabel('Sex')
plt.ylabel('Avereage Tip')
plt.title('Male vs Female tip average')
plt.show()
```



In [ ]: