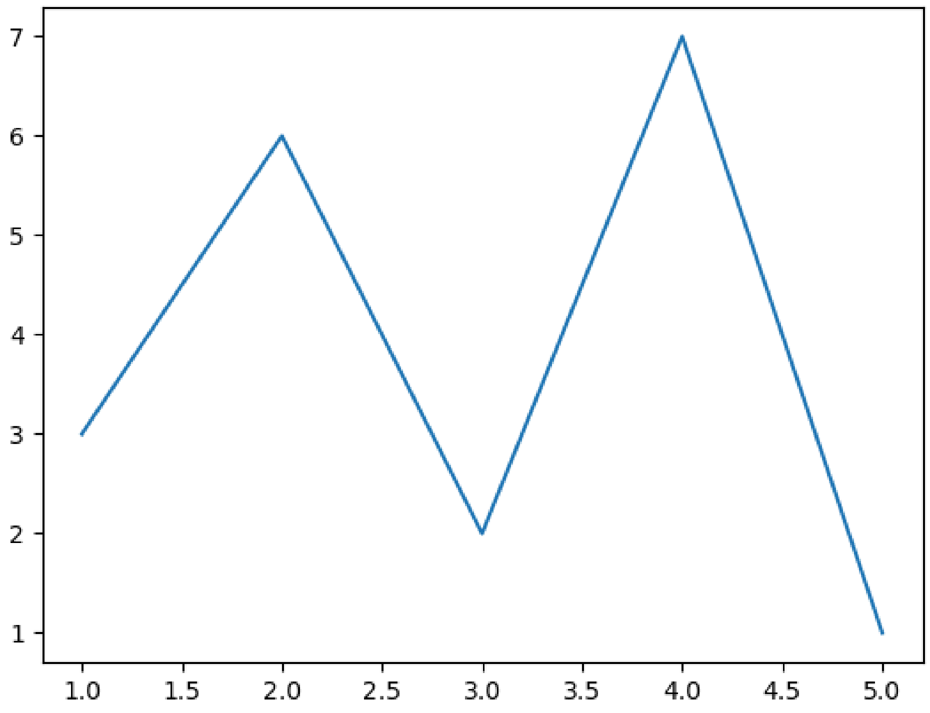
In [1]:
```python
import seaborn as sns
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5]
y = [3, 6, 2, 7, 1]
sns.lineplot(x=x, y=y)
```

Out[1]:
```
<Axes: >
```



In [3]:
```python
df = sns.load_dataset("tips")
df
```

Out[3]:

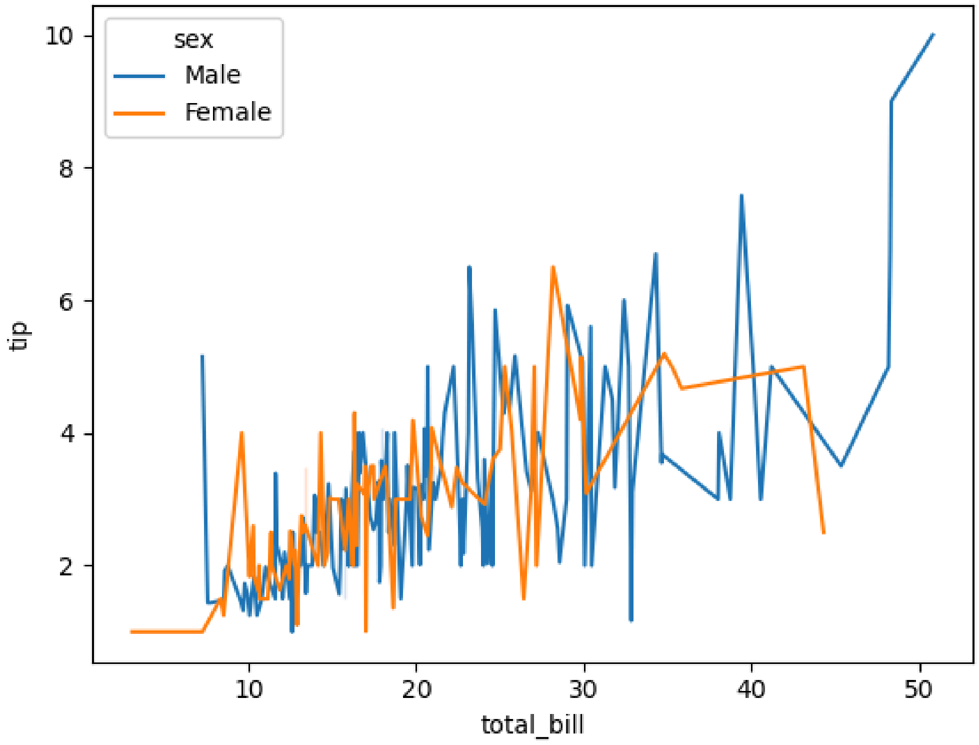
	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

In [4]:
```
sns.lineplot(x="total_bill",y="tip",data=df,hue="sex",linestyle="solid",legend="auto"

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[4], line 1
----> 1 sns.lineplot(x="total_bill",y="tips",data=df,hue="sex",linestyle="solid",legend
="auto")

File C:\ProgramData\anaconda3\lib\site-packages\seaborn\relational.py:618, in lineplot(d
ata, x, y, hue, size, style, units, palette, hue_order, hue_norm, sizes, size_order, siz
e_norm, dashes, markers, style_order, estimator, errorbar, n_boot, seed, orient, sort, e
rr_style, err_kws, legend, ci, ax, **kwargs)
    615 errorbar = _deprecate_ci(errorbar, ci)
    617 variables = _LinePlotter.get_semantics(locals())
--> 618 p = _LinePlotter(
    619     data=data, variables=variables,
    620     estimator=estimator, n_boot=n_boot, seed=seed, errorbar=errorbar,
    621     sort=sort, orient=orient, err_style=err_style, err_kws=err_kws,
    622     legend=legend,
    623 )
    625 p.map_hue(palette=palette, order=hue_order, norm=hue_norm)
    626 p.map_size(sizes=sizes, order=size_order, norm=size_norm)

File C:\ProgramData\anaconda3\lib\site-packages\seaborn\relational.py:365, in _LinePlott
er.__init__(self, data, variables, estimator, n_boot, seed, errorbar, sort, orient, err_
style, err_kws, legend)
    351 def __init__(
    352     self, *,
    353     data=None, variables={},
   (...)
    359     # the kind of plot to draw, but for the time being we need to set
    360     # this information so the SizeMapping can use it
    361     self._default_size_range = (
    362         np.r_[.5, 2] * mpl.rcParams["lines.linewidth"]
    363     )
--> 365     super().__init__(data=data, variables=variables)
    367     self.estimator = estimator
    368     self.errorbar = errorbar

File C:\ProgramData\anaconda3\lib\site-packages\seaborn\_oldcore.py:640, in VectorPlotte
r.__init__(self, data, variables)
    635 # var_ordered is relevant only for categorical axis variables, and may
    636 # be better handled by an internal axis information object that tracks
    637 # such information and is set up by the scale_* methods. The analogous
    638 # information for numeric axes would be information about log scales.
    639 self._var_ordered = {"x": False, "y": False}  # alt., used DefaultDict
--> 640 self.assign_variables(data, variables)
    642 for var, cls in self._semantic_mappings.items():
    643 
    644     # Create the mapping function
    645     map_func = partial(cls.map, plotter=self)

File C:\ProgramData\anaconda3\lib\site-packages\seaborn\_oldcore.py:701, in VectorPlotte
r.assign_variables(self, data, variables)
    699 else:
    700     self.input_format = "long"
--> 701     plot_data, variables = self._assign_variables_longform(
    702         data, **variables,
    703     )
    705 self.plot_data = plot_data
    706 self.variables = variables

File C:\ProgramData\anaconda3\lib\site-packages\seaborn\_oldcore.py:938, in VectorPlotte
r._assign_variables_longform(self, data, **kwargs)
    933 elif isinstance(val, (str, bytes)):
    934 
    935     # This looks like a column name but we don't know what it means!
    937     err = f"Could not interpret value `{val}` for parameter `{key}`"
--> 938     raise ValueError(err)
    940 else:
    941 
    # Otherwise, assume the value is itself data

    # Raise when data object is present and a vector can't matched
    if isinstance(data, pd.DataFrame) and not isinstance(val, pd.Series):

ValueError: Could not interpret value `tips` for parameter `y`
```

```
In [5]: sns.lineplot(x="total_bill",y="tip",data=df,hue="sex",linestyle="solid",legend="auto"
```

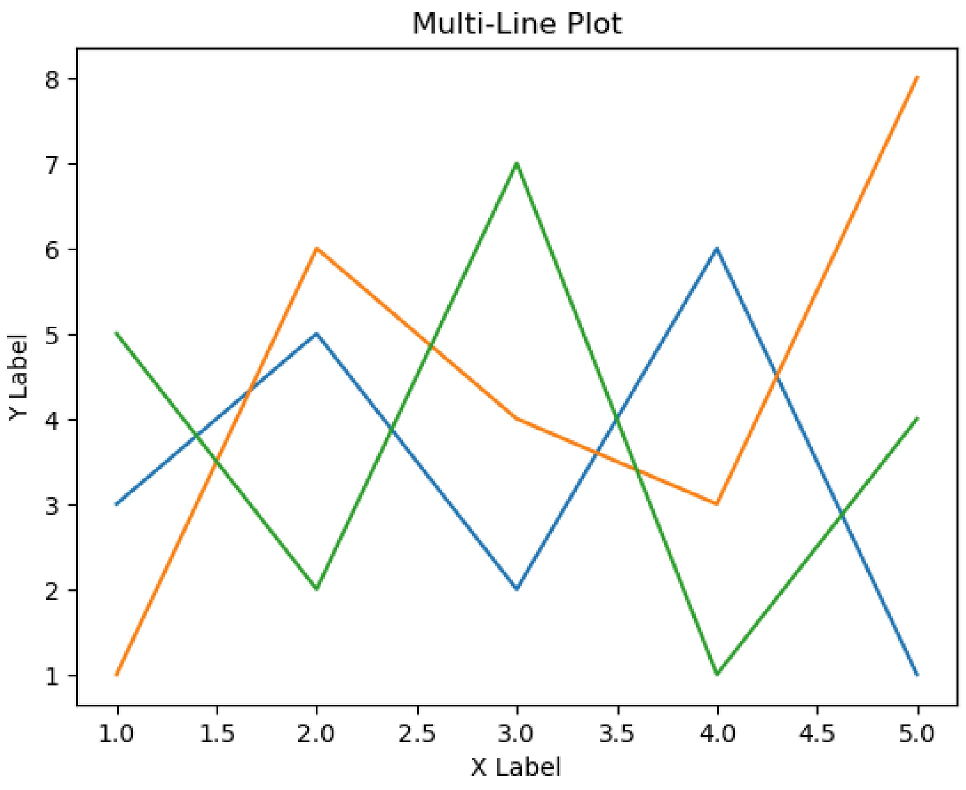
```
Out[5]: <Axes: xlabel='total_bill', ylabel='tip'>
```



In [7]:

```python
x = [1, 2, 3, 4, 5]
y1 = [3, 5, 2, 6, 1]
y2 = [1, 6, 4, 3, 8]
y3 = [5, 2, 7, 1, 4]
sns.lineplot(x=x, y=y1)
sns.lineplot(x=x, y=y2)
sns.lineplot(x=x, y=y3)
plt.title('Multi-Line Plot')
plt.xlabel('X Label')
plt.ylabel('Y Label')
text(0, 0.5, 'Y Label')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[7], line 11
      9 plt.xlabel('X Label')
     10 plt.ylabel('Y Label')
---> 11 text(0, 0.5, 'Y Label')

NameError: name 'text' is not defined
```



In [8]:
```
import seaborn as sns
import matplotlib.pyplot as pit
tips = sns. load _dataset ('tips')
avg_total bill = tips.groupby("day')['total_bill'].mean()
avg_tip = tips.groupby("day') ['tip']. mean()
plt. figure(figsize=(8, 6))
p1 = plt.bar(avg_total_bill.index, avg_total_bill, label='Total Bill')
p2 = plt.bar(avg_tip.index, avg_tip, bottom=avg_total_bill, label= 'Tip")
plt.xlabel('Day of the Week') plt.ylabel (' Amount')
plt.ylabel('Amount')
plt.title('Average Total Bill and Tip by Day')
plt.legend()
```

```
  Cell In[8], line 4
    avg_total bill = tips.groupby("day')['total_bill'].mean()
                                                       ^
SyntaxError: unterminated string literal (detected at line 4)
```

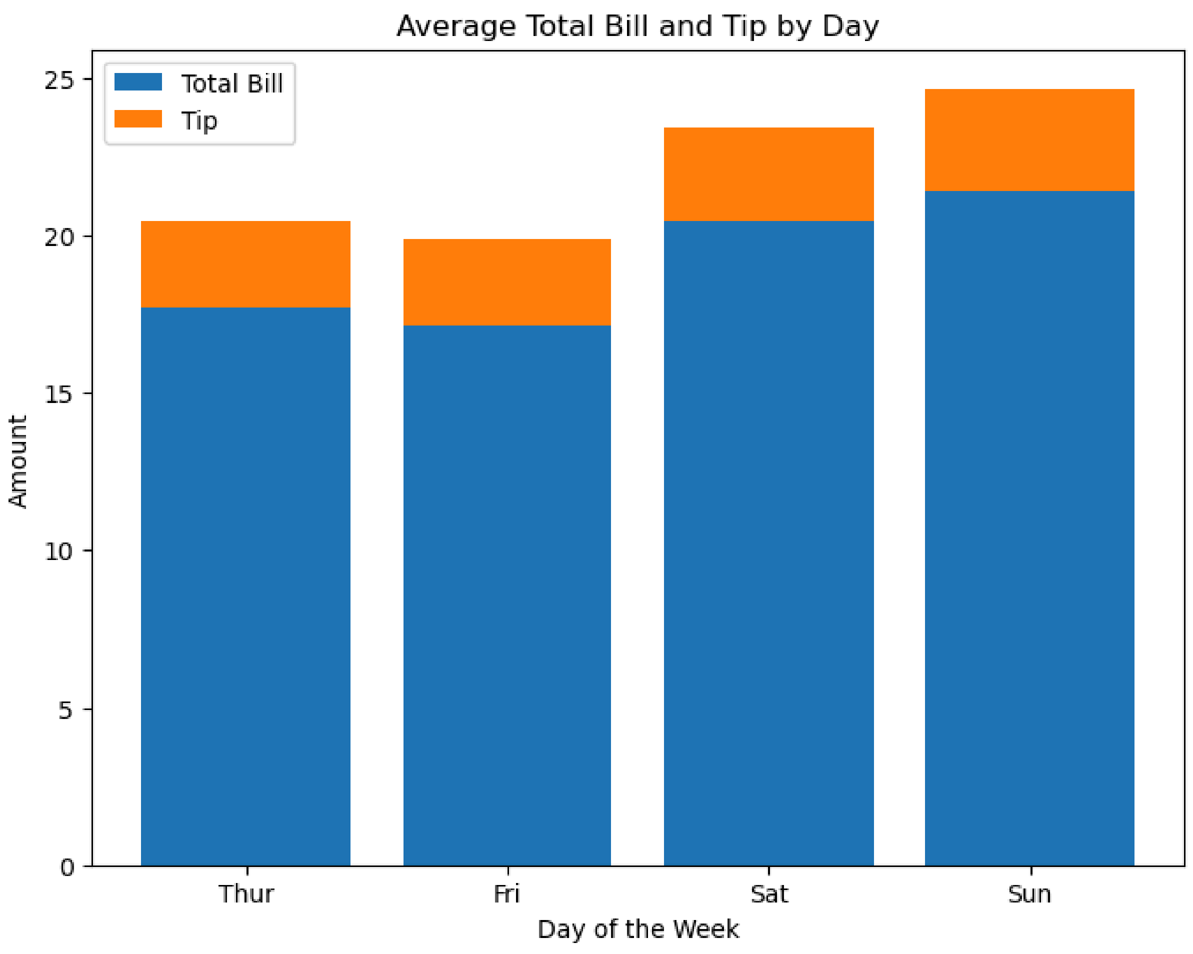
In [9]:
```
(matplotllb.legend.Legend at ex7eesf6fc6788>
```

```
  Cell In[9], line 1
    (matplotllb.legend.Legend at ex7eesf6fc6788>
     ^
SyntaxError: invalid syntax. Perhaps you forgot a comma?
```

In [ ]:
```

```

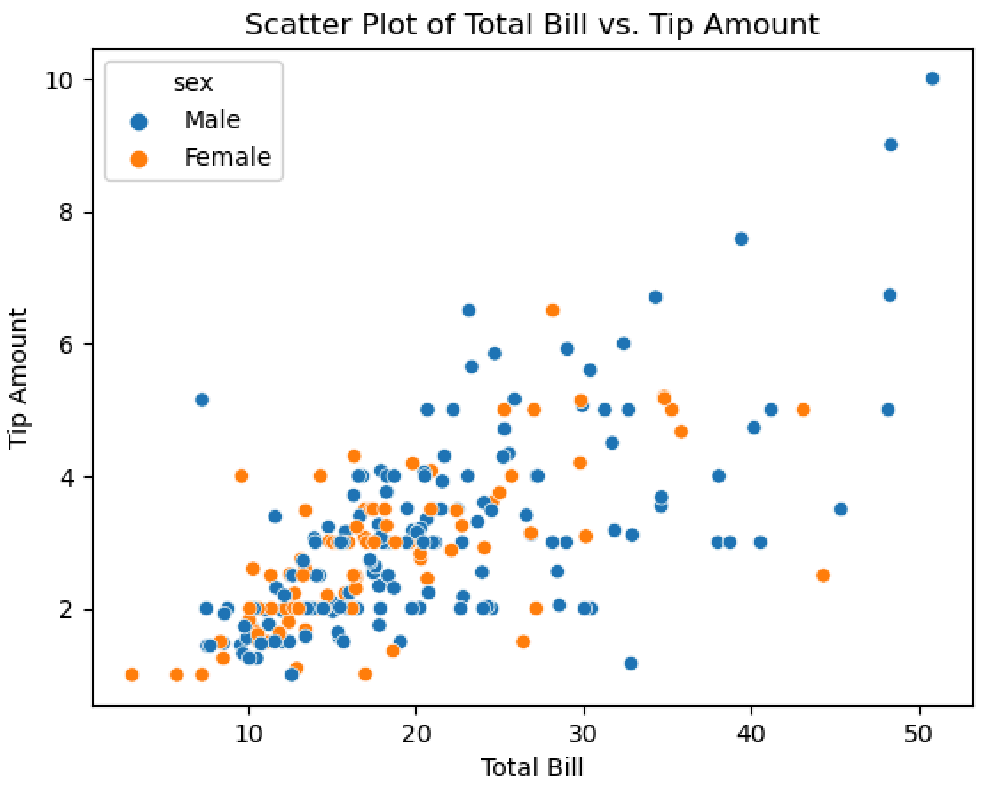
In [17]:
```
import seaborn as sns
import matplotlib.pyplot as pit
tips = sns.load_dataset ('tips')
avg_total_bill = tips.groupby('day')['total_bill'].mean()
avg_tip = tips.groupby('day') ['tip']. mean()
plt. figure(figsize=(8, 6))
p1 = plt.bar(avg_total_bill.index, avg_total_bill, label='Total Bill')
p2 = plt.bar(avg_tip.index, avg_tip, bottom=avg_total_bill, label= 'Tip')
plt.xlabel('Day of the Week')
plt.ylabel('Amount')
plt.title('Average Total Bill and Tip by Day')
plt.legend()
```

Out[17]:
```
<matplotlib.legend.Legend at 0x1f0b4f14dc0>
```



In [20]:
```python
import seaborn as sns
# Load the tips dataset
tips = sns. load_dataset( 'tips')
sns.scatterplot(x='total_bill', y='tip', hue='sex',data=tips)
plt.xlabel('Total Bill')
plt.ylabel('Tip Amount')
plt.title('Scatter Plot of Total Bill vs. Tip Amount')
```

Out[20]:
```
Text(0.5, 1.0, 'Scatter Plot of Total Bill vs.\xa0Tip\xa0Amount')
```



In [ ]:
```

```