Task 1: Creating a Pandas Dataframe

In [ ]:
```
import pandas as pd
import matplotlib.pyplot as plt

#Create the dataframe itself using the json file
df = pd.read_json(r'rain.json')
```

In [ ]:
```
df
```

Out[ ]:

	Month	Rainfall	Temperature
0	January	1.650	7
1	February	1.250	10
2	March	1.940	15
3	April	2.750	20
4	May	2.750	25
5	June	3.645	24
6	July	5.500	30
7	August	1.000	40
8	September	1.300	33
9	October	2.000	20
10	November	0.500	32
11	December	2.300	10

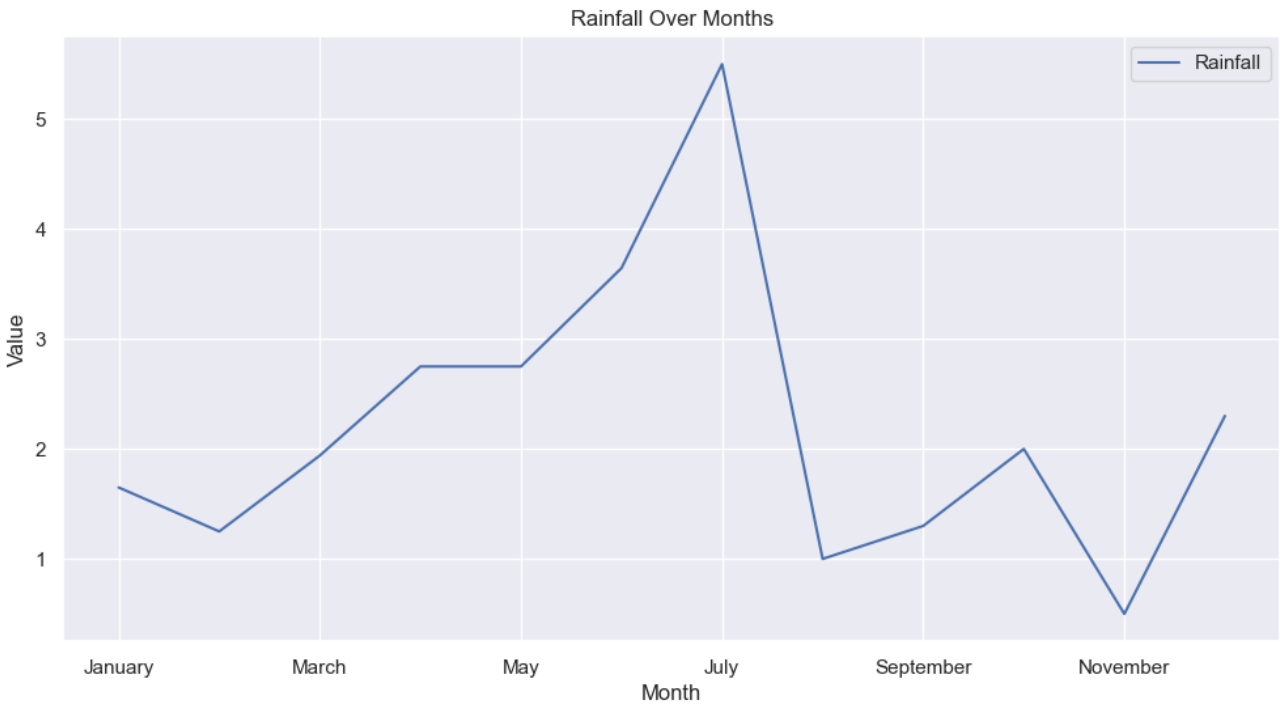
In [ ]:
```
print("df statistics: ")
df.describe()  #This shows the numerical only
```

```
df statistics:
```

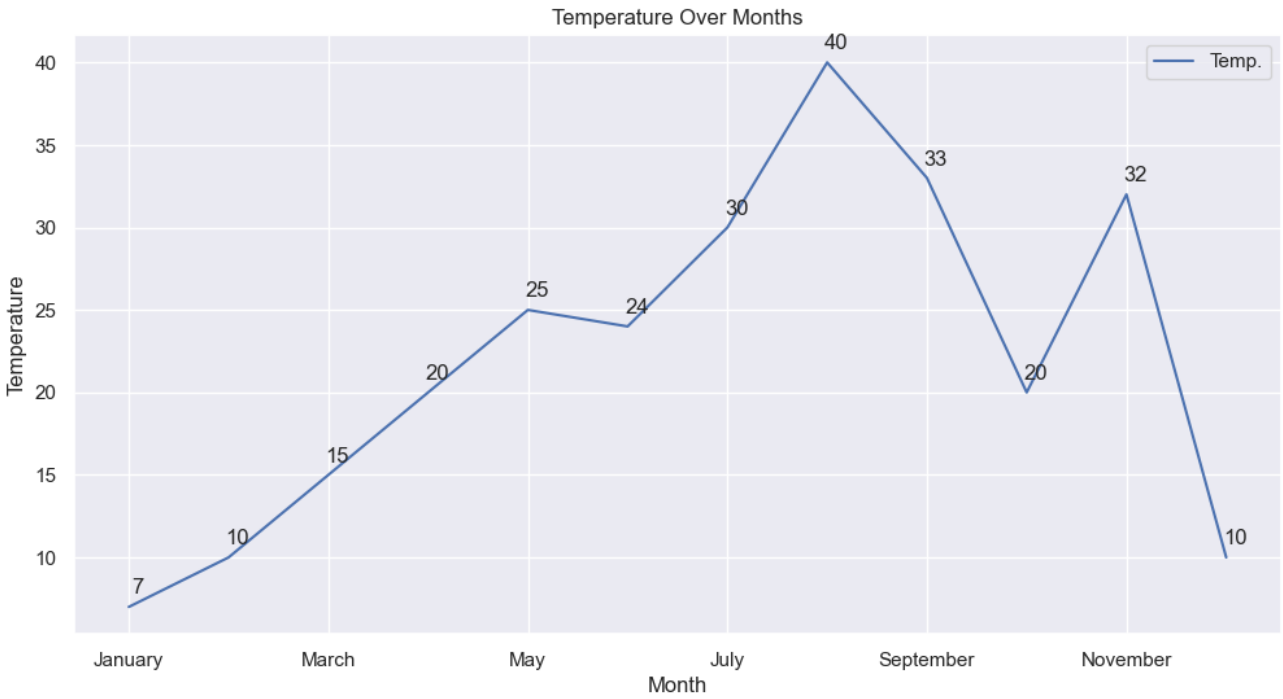
Out[ ]:

	Rainfall	Temperature
count	12.000000	12.000000
mean	2.215417	22.166667
std	1.349841	10.408330
min	0.500000	7.000000
25%	1.287500	13.750000
50%	1.970000	22.000000
75%	2.750000	30.500000
max	5.500000	40.000000

In [ ]:
```
# Create the second line plot for 'Rainfall'
df.plot(x='Month', y='Rainfall', label='Rainfall')
plt.title('Rainfall Over Months') # Set the title

# Set the labels for the axes
plt.xlabel('Month')
plt.ylabel('Value')

# Display the legend
plt.legend()

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

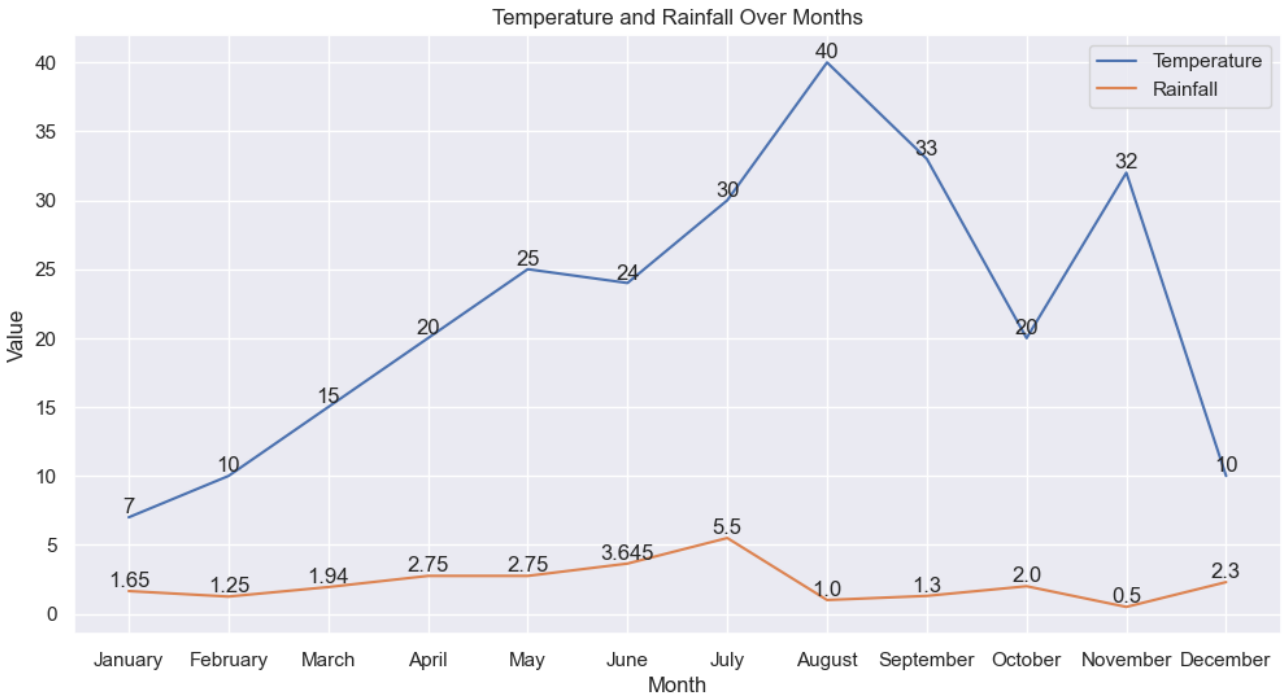


In [ ]:
```
# Create the first line plot for 'Temperature'
ax = df.plot(x='Month', y='Temperature', label='Temp.')

# Add value labels to 'Temperature' data points
for i, temp in enumerate(df['Temperature']):
    ax.annotate(str(temp), xy=(i, temp), xytext=(5, 5), textcoords='offset points', ha='

# Set the title
plt.title('Temperature Over Months') # Set the title

# Set the labels for the axes
plt.xlabel('Month')
plt.ylabel('Temperature')

# Display the legend
plt.legend()

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



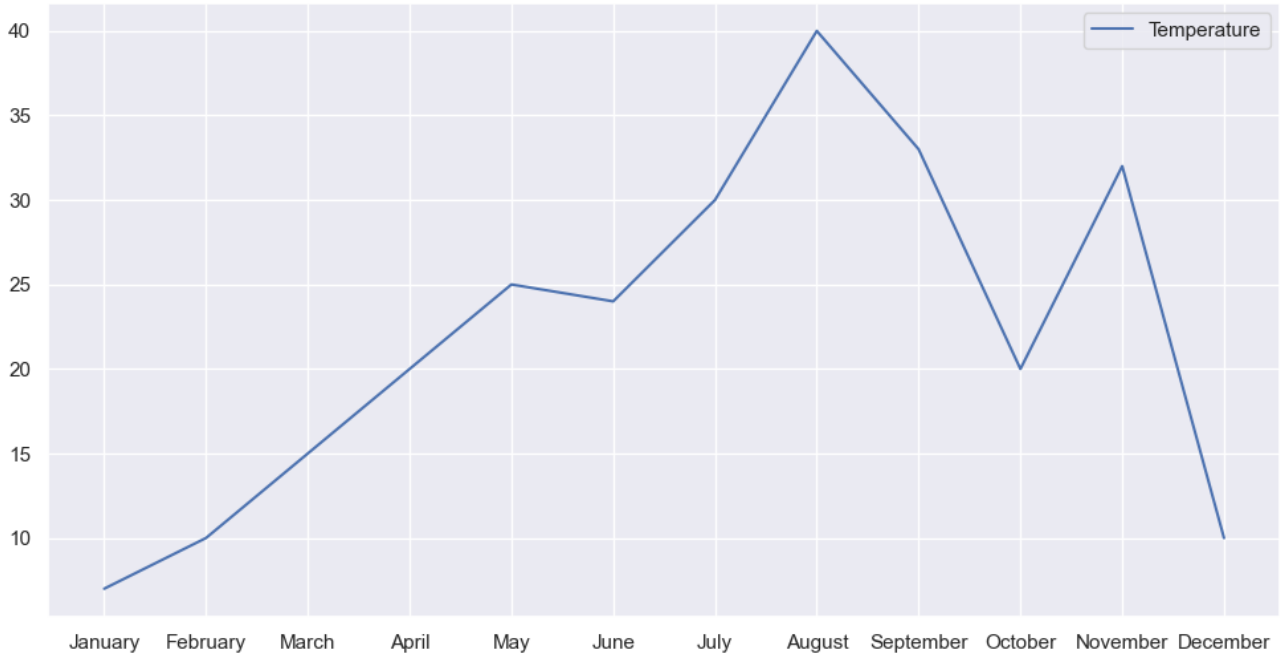
In [ ]:
```
#  to plot 'Rainfall' and 'Temperature' on the same figure
# Create a figure and axis
fig, ax = plt.subplots()

# Plot 'Temperature' with label values
ax.plot(df['Month'], df['Temperature'], label='Temperature')
for i, temp in enumerate(df['Temperature']):
    ax.text(df['Month'][i], temp, str(temp), ha='center', va='bottom')

# Plot 'Rainfall' with label values
ax.plot(df['Month'], df['Rainfall'], label='Rainfall')
for i, rainfall in enumerate(df['Rainfall']):
    ax.text(df['Month'][i], rainfall, str(rainfall), ha='center', va='bottom')


# Set the title and labels for the axes
plt.title('Temperature and Rainfall Over Months')
plt.xlabel('Month')
plt.ylabel('Value')

# Display the legend
plt.legend()
```

Out[ ]:
```
<matplotlib.legend.Legend at 0x2555fd3e710>
```



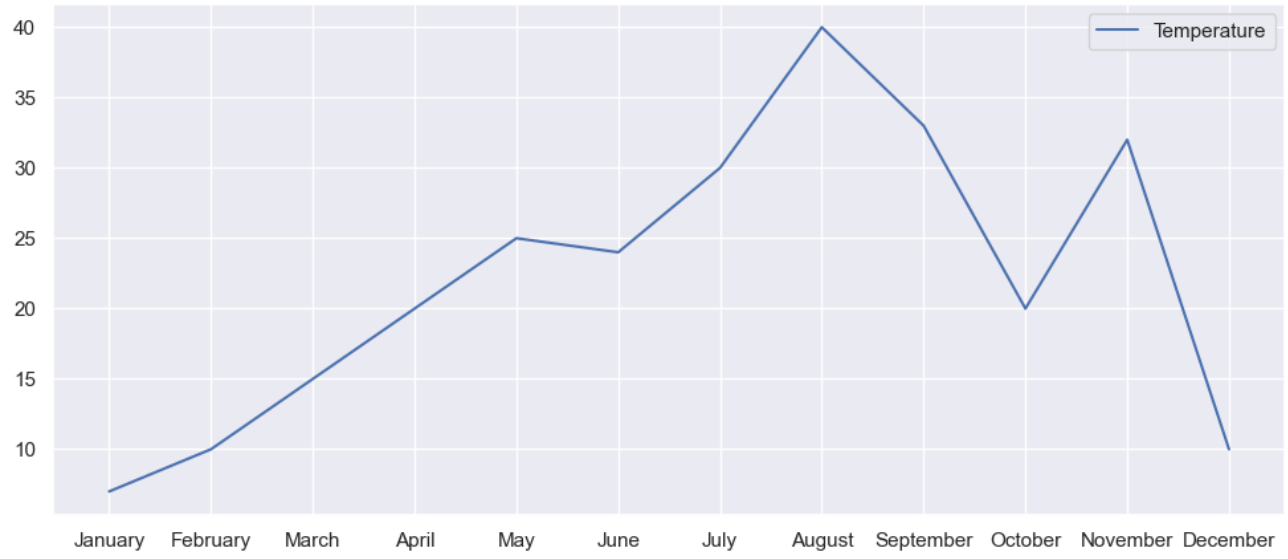
Task 2: MatplotLib PyPlot

In [ ]:
```
plt.plot( df['Month'], df['Temperature'], label = 'Temperature')
# Display the legend
plt.legend()

plt.show()
#Here the months are overlapping, so we need to fix this
```

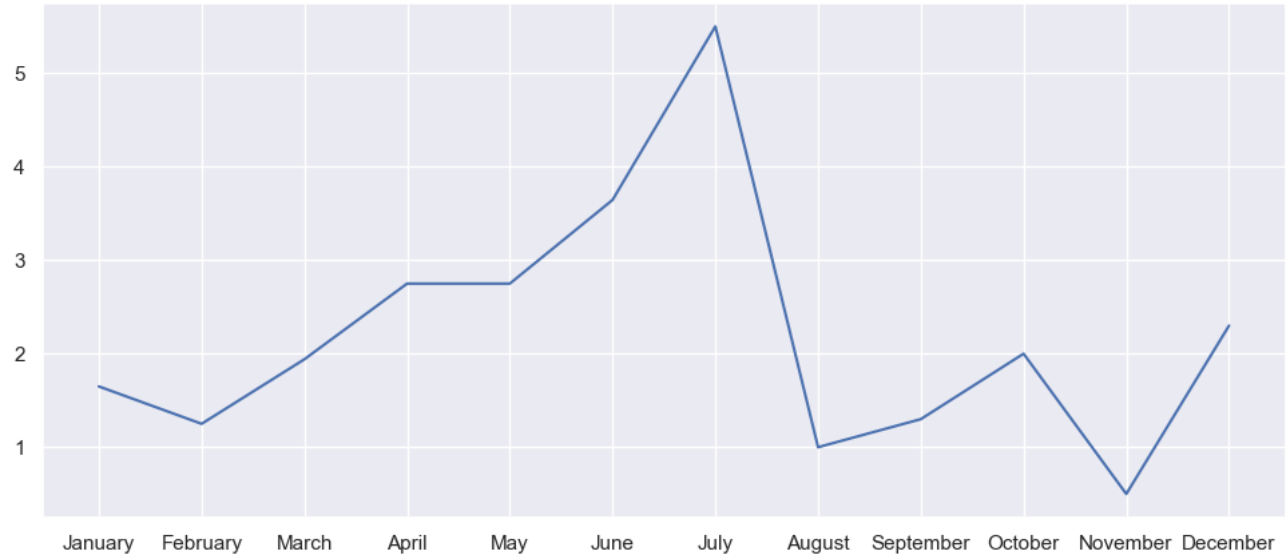


In [ ]:
```
plt.figure(figsize=(12,5))   #first number is width, 2nd one is height
plt.plot( df['Month'], df['Temperature'], label = 'Temperature')

# Display the legend
plt.legend()
```

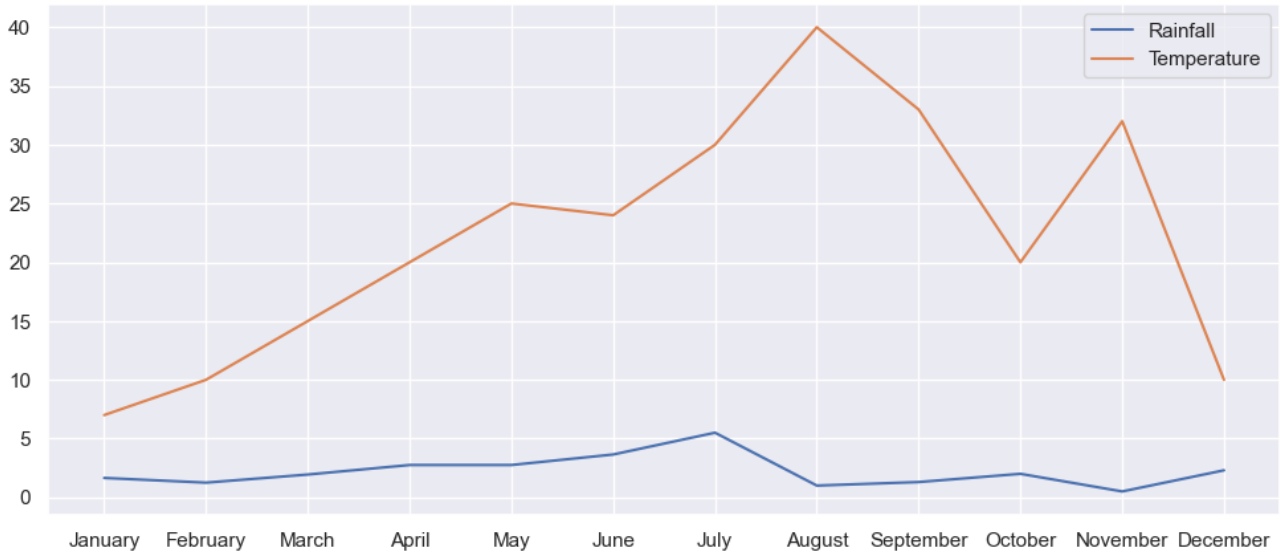
```
plt.show()
```



In [ ]:
```
#Plot Rainfall
plt.figure(figsize=(12,5))
plt.plot( df['Month'], df['Rainfall'], label = 'Rainfall')
plt.show()
```



In [ ]:
```
#Plot Both temperature and Rainfall against Month
plt.figure(figsize=(12,5))
plt.plot( df['Month'], df['Rainfall'], label = 'Rainfall')
plt.plot( df['Month'], df['Temperature'], label = 'Temperature')
plt.legend()   #To show us different colors according to label
plt.show()
```



Task 3: Scatterplot Graph

In [ ]:
```
# Import the needed library
import seaborn as sns
```

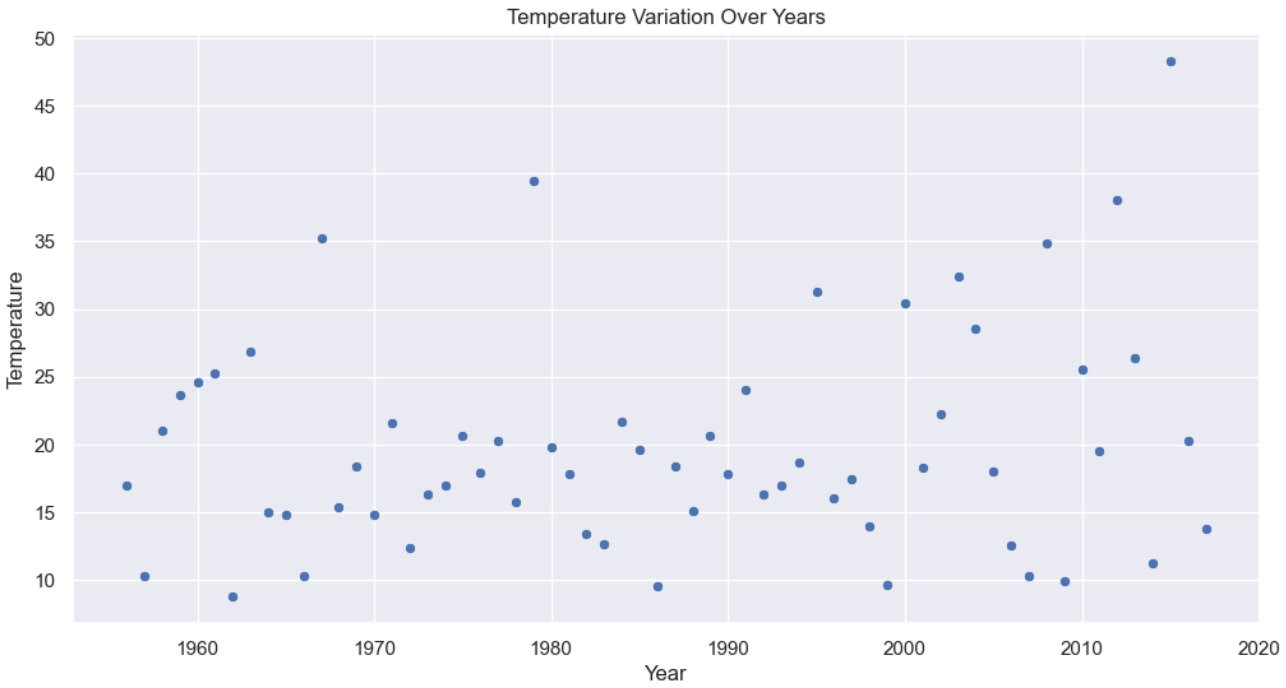
In [ ]:
```
df = pd.read_csv('tempYearly.csv')
```

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

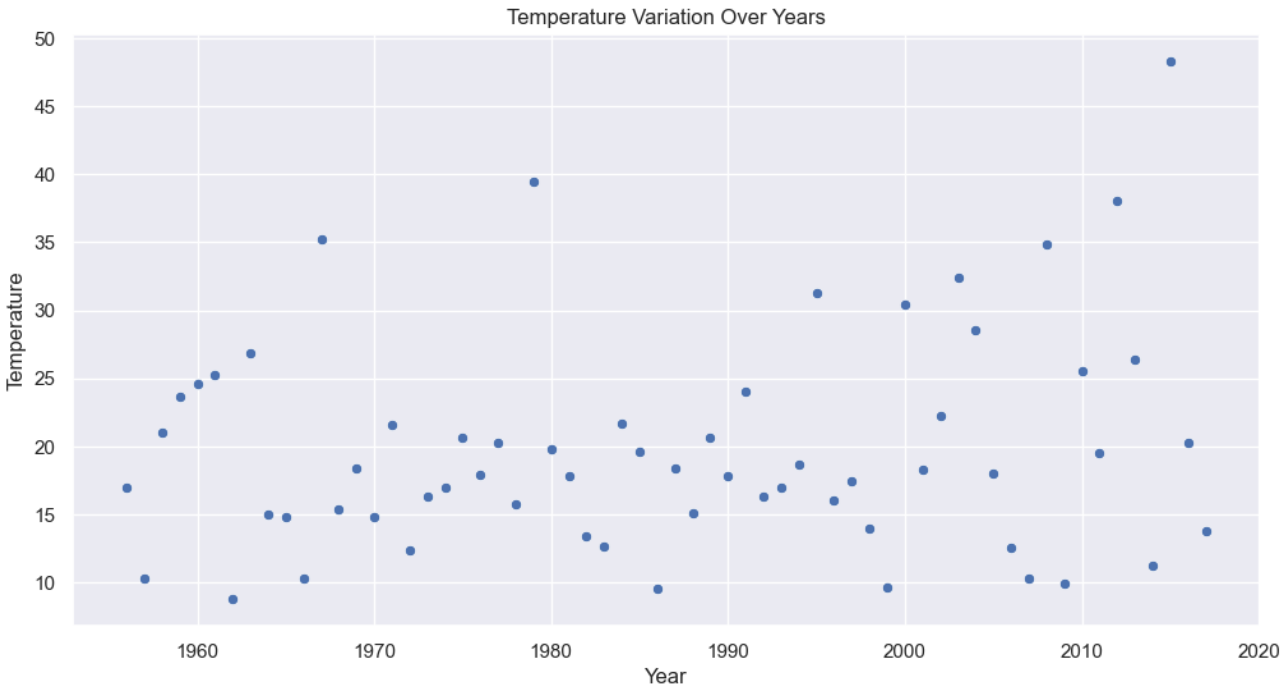
Out[ ]:

	Temperature	Year	Rainfall
0	16.99	1956	1.01
1	10.34	1957	1.66
2	21.01	1958	3.50
3	23.68	1959	3.31
4	24.59	1960	3.61

In [ ]:
```
# plot the scatter figure
sns.scatterplot(x='Year', y='Temperature', data=df)

# Set the title for the scatter plot
plt.title('Temperature Variation Over Years')

plt.show()
```

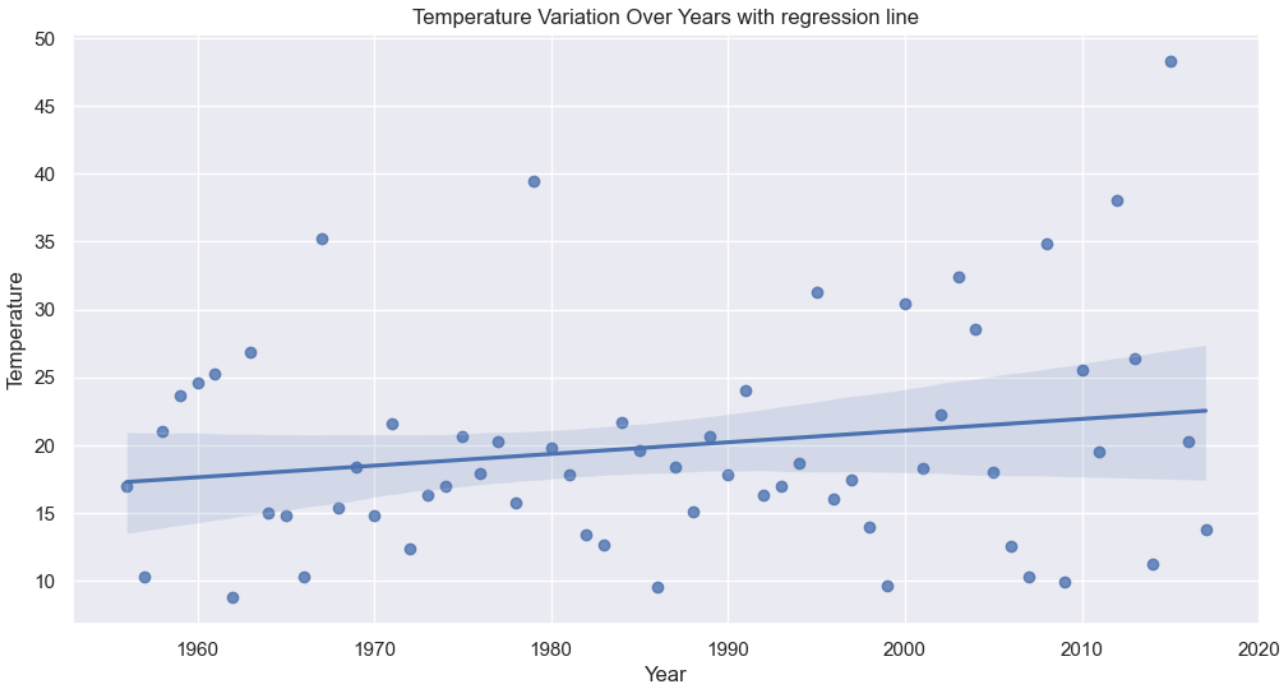


In [ ]:
```
#To increase width of plot, set the figure size of the plot to (12, 6) inches
sns.set(rc={'figure.figsize':(12,6)})

# plot the scatter figure
sns.scatterplot(x='Year', y='Temperature', data=df)

# Set the title for the scatter plot
plt.title('Temperature Variation Over Years')
plt.show()
```

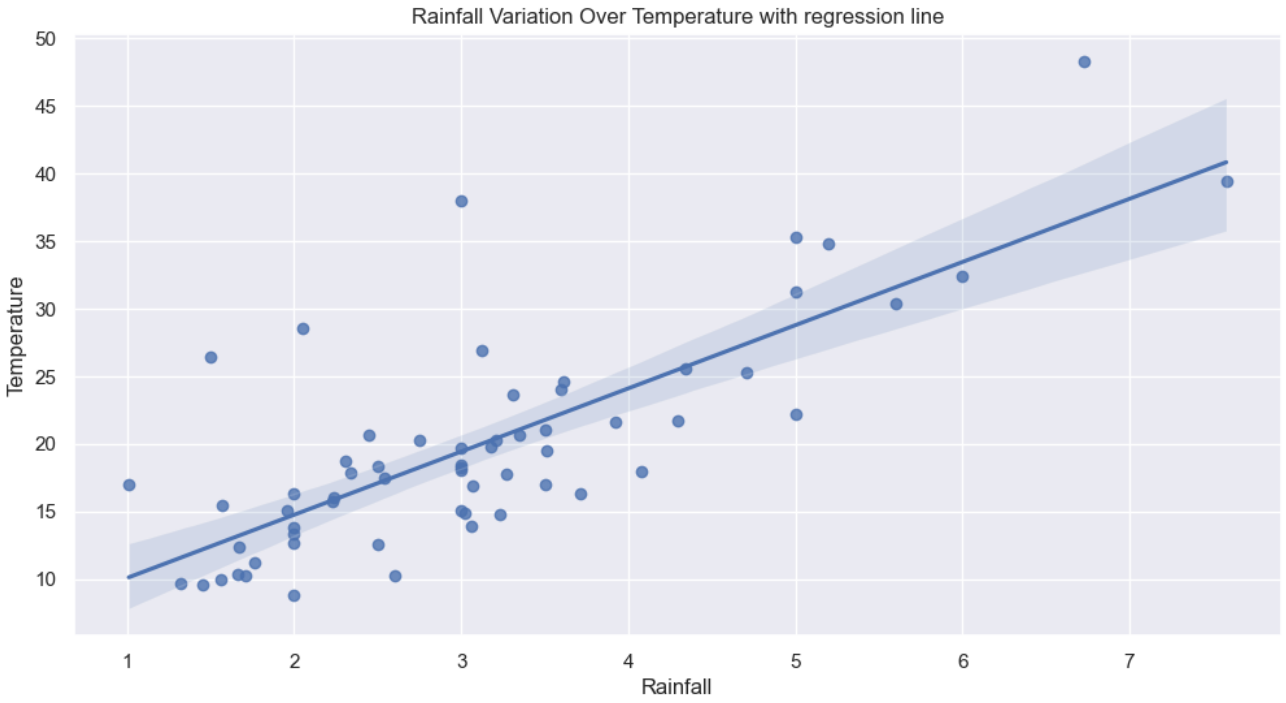


In [ ]:
```
#regplot to see regression line
sns.set(rc={'figure.figsize': (12, 6)})  # Set the figure size of the plot

sns.regplot(x='Year', y='Temperature', data=df)  # Create a scatter plot with a regressi
# Set the title for the scatter plot
plt.title('Temperature Variation Over Years with regression line')
```

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



In [ ]:
```
#regplot to see regression line with rainfall and temp
sns.set(rc={'figure.figsize':(12,6)})
sns.regplot(x='Rainfall', y='Temperature', data=df)  # Create a scatter plot with a regr

# Set the title for the scatter plot
plt.title('Rainfall Variation Over Temperature with regression line')
plt.show()
```



Important analysis: The above shows us the correlation between rainfall and temperature.

Task 4: Seaborn Heatmap

To visualize large and small values quickly.

Has a color key on the side.

In [ ]:
```
data = pd.read_csv(r'birthYearly.csv')
data
```

Out[ ]:

	month	year	births
0	January	2000	101
1	February	2000	168
2	March	2000	271
3	April	2000	229
4	May	2000	287
...	...	...	...
79	August	2006	243
80	September	2006	174
81	October	2006	62
82	November	2006	175
83	December	2006	134

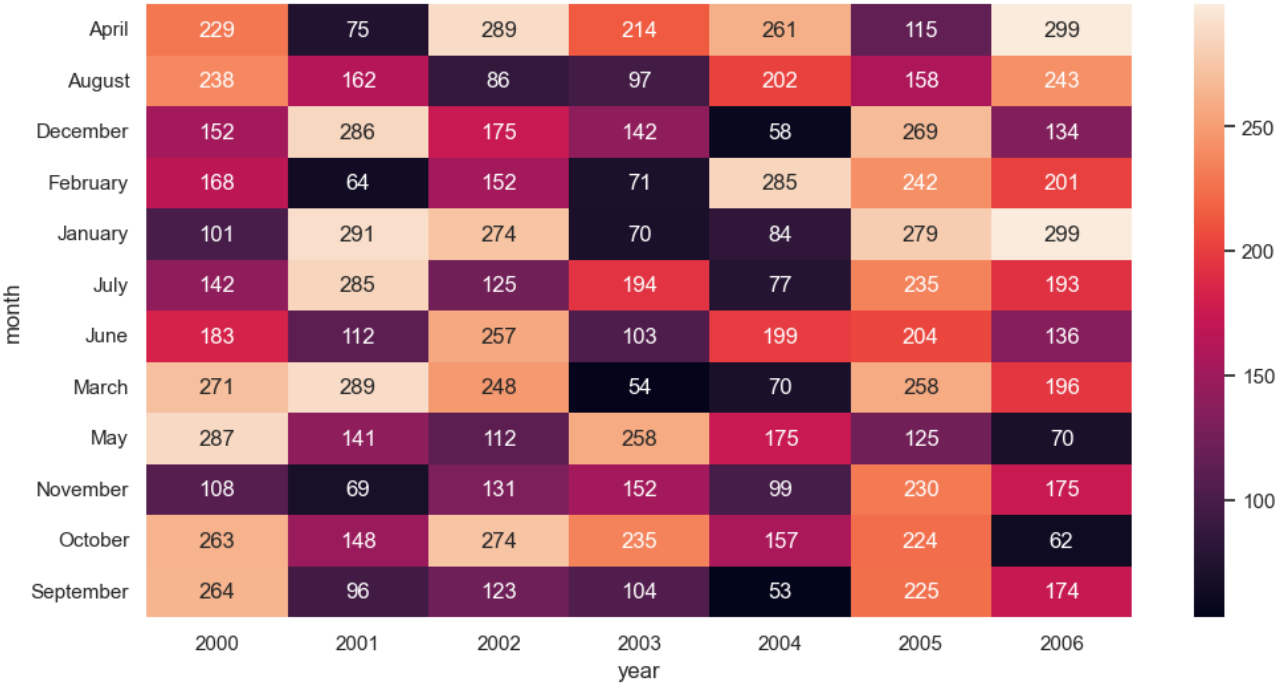
84 rows × 3 columns

In [ ]:
```
# We want to draw a heatmap but we will get an error: ValueError: could not convert stri
# because the data is not in correct format for the heat map
#We need to organize the data using pivots
# sns.heatmap(data,
#             annot = True,    #we want our numbers inside the heatmap
#            fmt = "d"      #decimal format data
```

In [ ]:
```
#Pivoting the table
dataP = data.pivot("month", "year", "births")
print(dataP)
```

```
year       2000  2001  2002  2003  2004  2005  2006
month
April       229    75   289   214   261   115   299
August      238   162    86    97   202   158   243
December    152   286   175   142    58   269   134
February    168    64   152    71   285   242   201
January     101   291   274    70    84   279   299
July        142   285   125   194    77   235   193
June        183   112   257   103   199   204   136
March       271   289   248    54    70   258   196
May         287   141   112   258   175   125    70
November    108    69   131   152    99   230   175
October     263   148   274   235   157   224    62
September   264    96   123   104    53   225   174
```

In [ ]:
```
# plot the heatmap
sns.heatmap(dataP,
            annot=True,  # Display the data values inside the heatmap
            fmt="d"       # Use decimal format for the annotations
           )
plt.show()
```



We can quickly draw some conclusions from this heatmap like:

1) V.small births in September 2004

2) Alot of births in April 2006

Task 5: Seaborn JointPlot

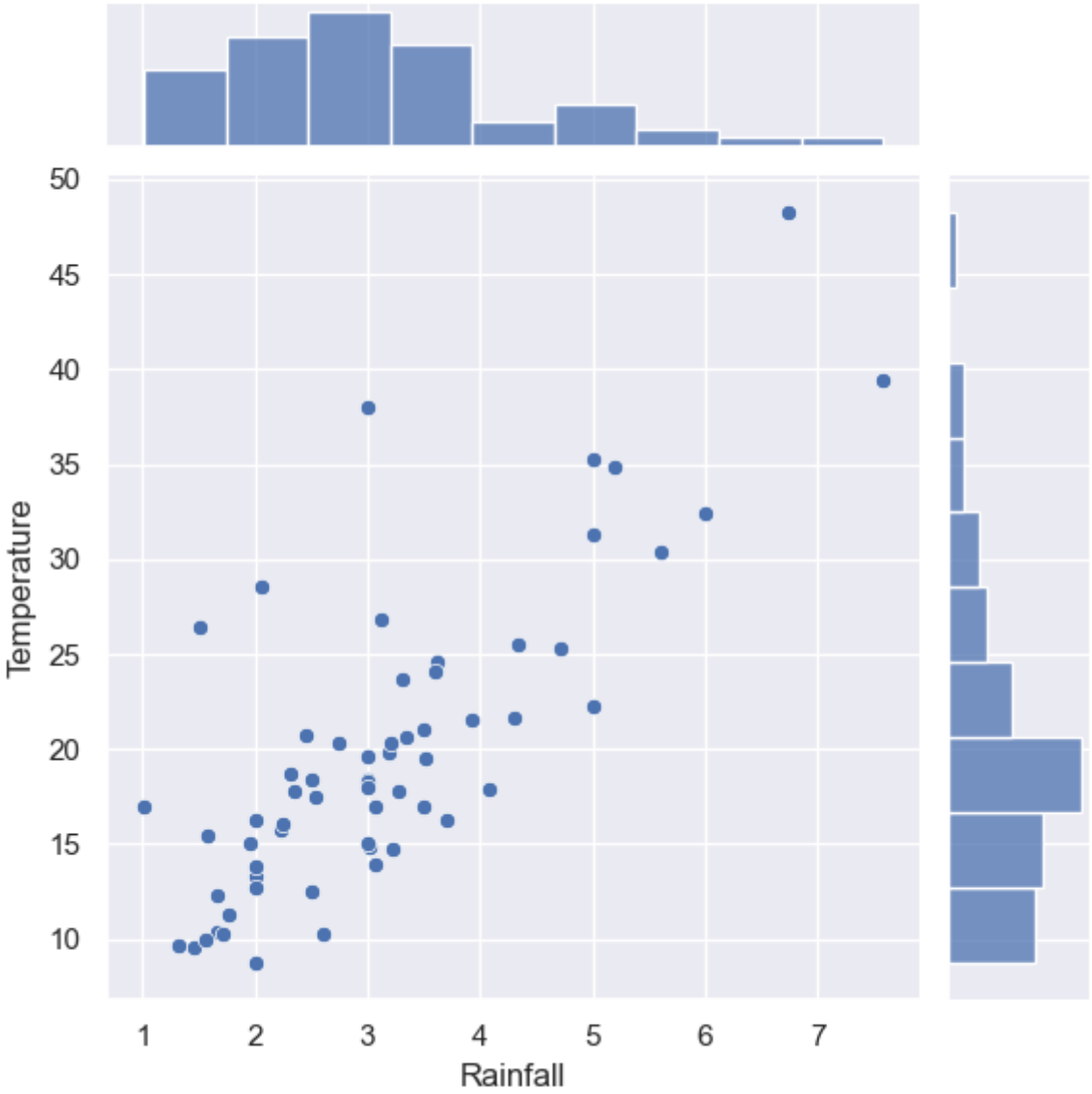
Good for looking at distribution and data points at the same time

In [ ]:
```
data = pd.read_csv(r'tempYearly.csv')
data.head()
```

Out[ ]:

	Temperature	Year	Rainfall
0	16.99	1956	1.01
1	10.34	1957	1.66
2	21.01	1958	3.50
3	23.68	1959	3.31
4	24.59	1960	3.61

In [ ]:
```
#We will draw a joint plot which will get scatterplot in middle and histograms on the si
sns.jointplot(x=data['Rainfall'], y=data['Temperature'])

plt.show()
```

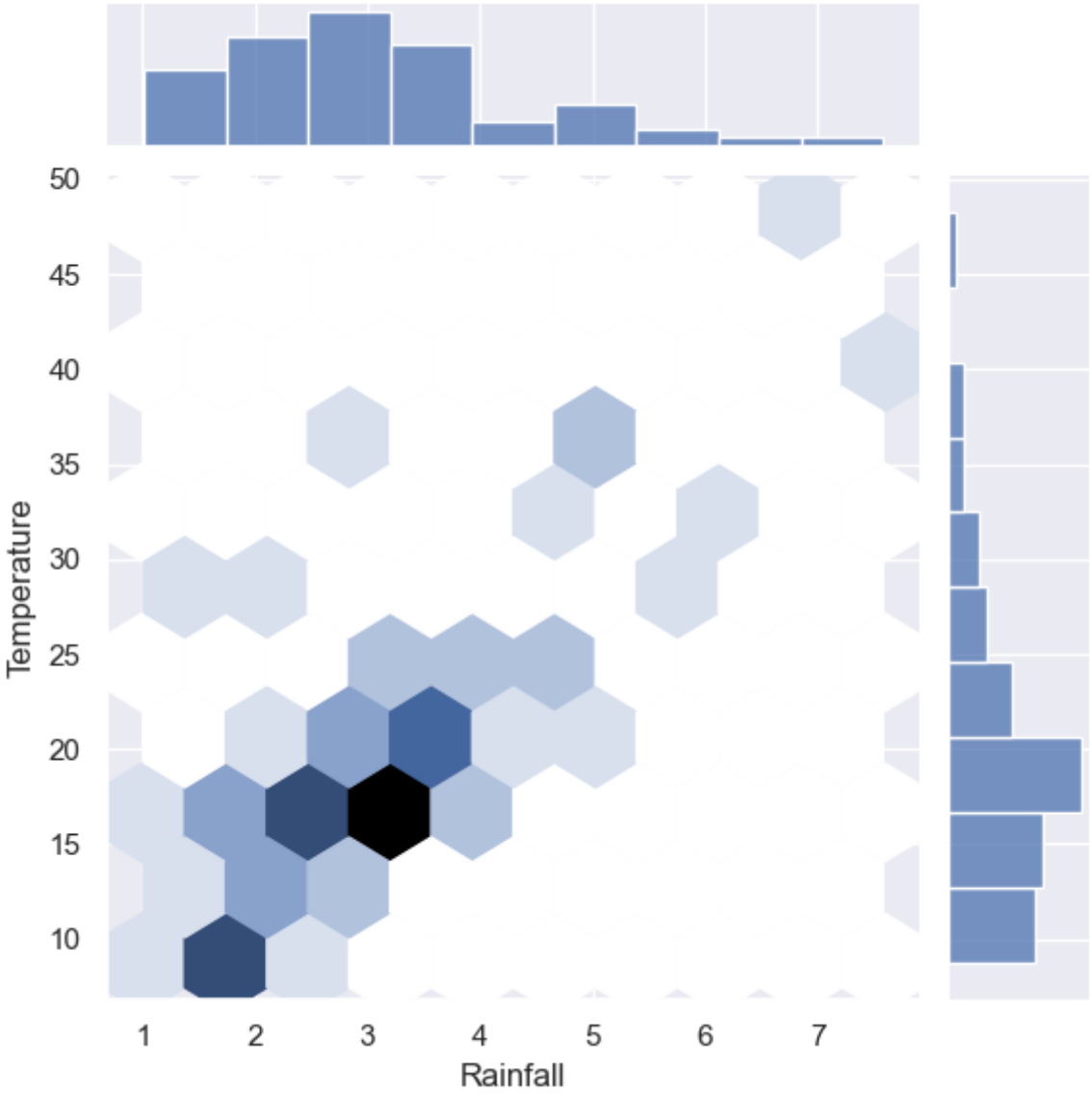


In the above jointplot:

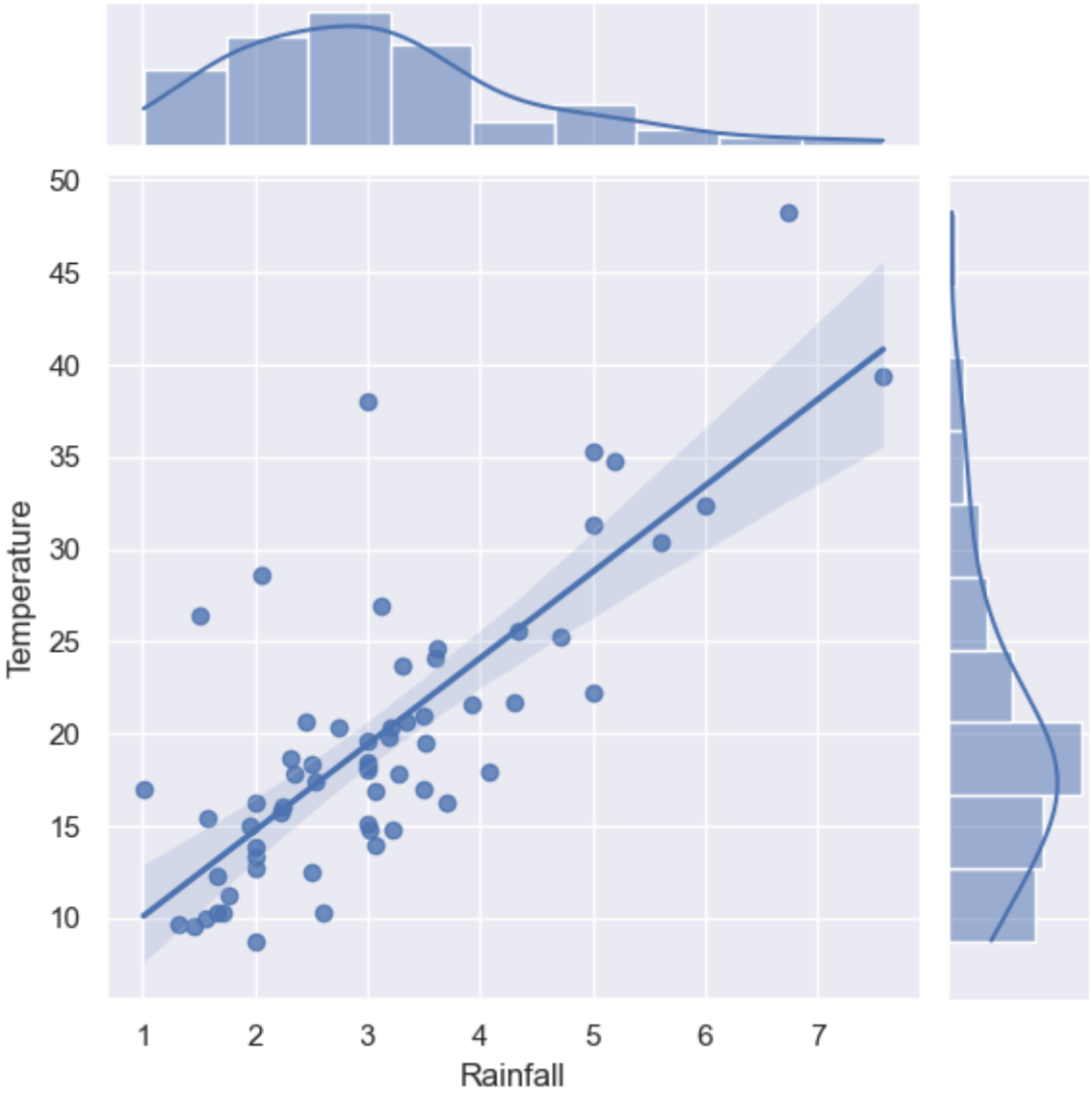
Scatterplot can show us outliers, we can imagine a regression line and see the outliers

Histograms can show us the distibution of the data

In [ ]:
```
sns.jointplot(x=data['Rainfall'], y=data['Temperature'], kind = "hex") #Hexagram to visu
plt.show()
```



In [ ]:
```
#Regression Line, so we can see a possible +ve correlation
sns.jointplot(x=data['Rainfall'], y=data['Temperature'], kind = "reg")
plt.show()
```



Many thanks to MarwaEshra and Instructor (David Dalsveen)