Pymaceuticals Inc.

Analysis

In [7]:
```
# Observations.

### 1. The Capomulin and Ramicane drug regimens result in lower tumor volum
### 2. The correlation between mouse weight and tumor volume for the Capomu
### 3. Ketapril is the least effective drug regimen as it has both the high
### 4. The relationship between timepoint and tumor volume varies, starting
```

In [8]:
```
# Dependencies and Setup
import matplotlib.pyplot as plt
import pandas as pd
import scipy.stats as st

# Study data files
mouse_metadata_path = "data/Mouse_metadata.csv"
study_results_path = "data/Study_results.csv"

# Read the mouse data and the study results
mouse_metadata = pd.read_csv(mouse_metadata_path)
study_results = pd.read_csv(study_results_path)

# Combine the data into a single dataset

merge_df = pd.merge(mouse_metadata, study_results, on = "Mouse ID")
# Display the data table for preview
merge_df
```

Out[8]:

	Mouse ID	Drug Regimen	Sex	Age_months	Weight (g)	Timepoint	Tumor Volume (mm3)	Metastatic Sites
0	k403	Ramicane	Male	21	16	0	45.000000	0
1	k403	Ramicane	Male	21	16	5	38.825898	0
2	k403	Ramicane	Male	21	16	10	35.014271	1
3	k403	Ramicane	Male	21	16	15	34.223992	1
4	k403	Ramicane	Male	21	16	20	32.997729	1
...	...	...	...	...	...	...	...	...
1888	z969	Naftisol	Male	9	30	25	63.145652	2
1889	z969	Naftisol	Male	9	30	30	65.841013	3
1890	z969	Naftisol	Male	9	30	35	69.176246	4
1891	z969	Naftisol	Male	9	30	40	70.314904	4

In [9]:
```
# Checking the number of mice.
unique = merge_df["Mouse ID"].value_counts()
len(unique)
```

Out[9]:
```
249
```

In [10]:
```
# Getting the duplicate mice by ID number that shows up for Mouse ID and Ti
truths = merge_df["Mouse ID"].value_counts() > 10
truths

#g989 has more than 10 timepoints so it is the duplicate Mouse ID - Tom.
```

Out[10]:
```
g989     True
z581    False
i901    False
c402    False
k862    False
        ...
u153    False
l872    False
v199    False
x336    False
f932    False
Name: Mouse ID, Length: 249, dtype: bool
```

In [11]:
```
# Optional: Get all the data for the duplicate mouse ID.
merge_df.loc[merge_df["Mouse ID"] == "g989",:]
```

Out[11]:

	Mouse ID	Drug Regimen	Sex	Age_months	Weight (g)	Timepoint	Tumor Volume (mm3)	Metastatic Sites
908	g989	Propriva	Female	21	26	0	45.000000	0
909	g989	Propriva	Female	21	26	0	45.000000	0
910	g989	Propriva	Female	21	26	5	48.786801	0
911	g989	Propriva	Female	21	26	5	47.570392	0
912	g989	Propriva	Female	21	26	10	51.745156	0
913	g989	Propriva	Female	21	26	10	49.880528	0
914	g989	Propriva	Female	21	26	15	51.325852	1
915	g989	Propriva	Female	21	26	15	53.442020	0
916	g989	Propriva	Female	21	26	20	55.326122	1
917	g989	Propriva	Female	21	26	20	54.657650	1
918	g989	Propriva	Female	21	26	25	56.045564	1
919	g989	Propriva	Female	21	26	30	59.082294	1
920	g989	Propriva	Female	21	26	35	62.570880	2

In [12]:
```
# Create a clean DataFrame by dropping the duplicate mouse by its ID.

merge_df = merge_df[merge_df["Mouse ID"] != "g989"]
merge_df
```

Out[12]:

	Mouse ID	Drug Regimen	Sex	Age_months	Weight (g)	Timepoint	Tumor Volume (mm3)	Metastatic Sites
0	k403	Ramicane	Male	21	16	0	45.000000	0
1	k403	Ramicane	Male	21	16	5	38.825898	0
2	k403	Ramicane	Male	21	16	10	35.014271	1
3	k403	Ramicane	Male	21	16	15	34.223992	1
4	k403	Ramicane	Male	21	16	20	32.997729	1
...	...	...	...	...	...	...	...	...
1888	z969	Naftisol	Male	9	30	25	63.145652	2
1889	z969	Naftisol	Male	9	30	30	65.841013	3
1890	z969	Naftisol	Male	9	30	35	69.176246	4
1891	z969	Naftisol	Male	9	30	40	70.314904	4
1892	z969	Naftisol	Male	9	30	45	73.867845	4

1880 rows × 8 columns

In [13]:
```
# Checking the number of mice in the clean DataFrame.
unique = merge_df["Mouse ID"].value_counts()
len(unique)
```

Out[13]: 248

Summary Statistics

In [14]:
```
# Generate a summary statistics table of mean, median, variance, standard d

# Use groupby and summary statistical methods to calculate the following pr
# mean, median, variance, standard deviation, and SEM of the tumor volume.
# Assemble the resulting series into a single summary dataframe.

drug = merge_df.groupby(["Drug Regimen"])
drug_mean = drug.mean()
drug_df = pd.DataFrame(drug_mean["Tumor Volume (mm3)"])
drug_df = drug_df.rename(columns = {"Tumor Volume (mm3)":"Mean Tumor Volume
drug_median = drug.median()
drug_variance = drug.var()
drug_std = drug.std()
drug_sem = drug.sem()
drug_df["Median Tumor Volume"] = drug_median["Tumor Volume (mm3)"]
drug_df["Tumor Volume Variance"] = drug_variance["Tumor Volume (mm3)"]
drug_df["Tumor Volume Standard Deviation"] = drug_std["Tumor Volume (mm3)"]
drug_df["Tumor Volume Std. Error"] = drug_sem["Tumor Volume (mm3)"]
drug_df
```

Out[14]:

Drug Regimen	Mean Tumor Volume	Median Tumor Volume	Tumor Volume Variance	Tumor Volume Standard Deviation	Tumor Volume Std. Error
Capomulin	40.675741	41.557809	24.947764	4.994774	0.329346
Ceftamin	52.591172	51.776157	39.290177	6.268188	0.469821
Infubinol	52.884795	51.820584	43.128684	6.567243	0.492236
Ketapril	55.235638	53.698743	68.553577	8.279709	0.603860
Naftisol	54.331565	52.509285	66.173479	8.134708	0.596466
Placebo	54.033581	52.288934	61.168083	7.821003	0.581331
Propriva	52.320930	50.446266	43.852013	6.622085	0.544332
Ramicane	40.216745	40.673236	23.486704	4.846308	0.320955
Stelasyn	54.233149	52.431737	59.450562	7.710419	0.573111
Zoniferol	53.236507	51.818479	48.533355	6.966589	0.516398

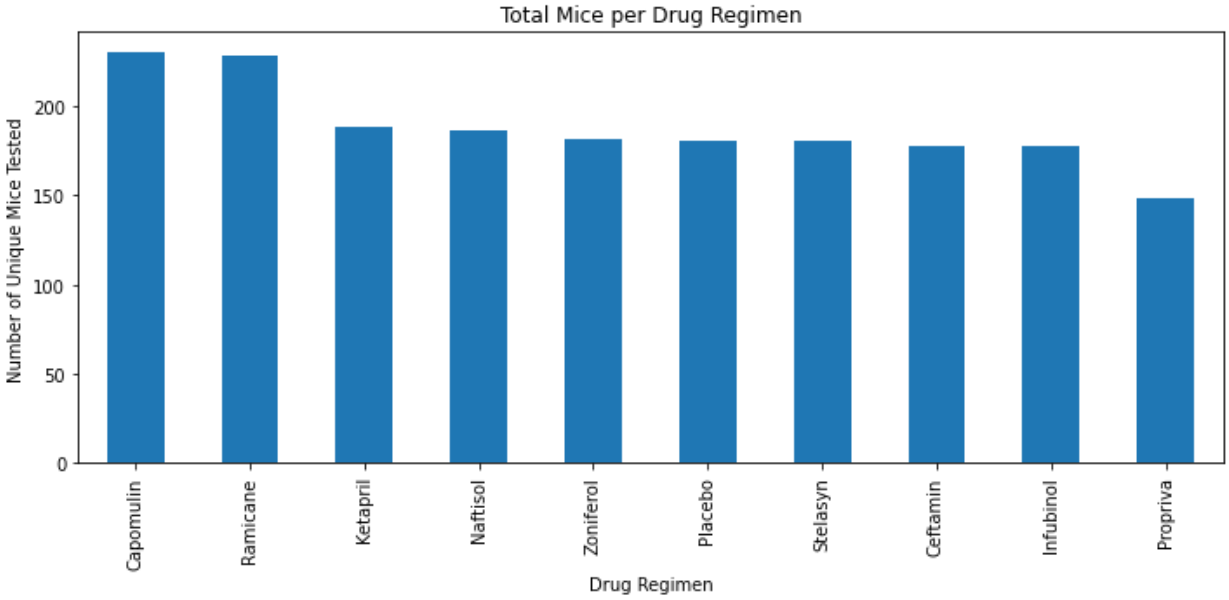
In [15]:
```
# Generate a summary statistics table of mean, median, variance, standard d

# Using the aggregation method, produce the same summary statistics in a si

drug["Tumor Volume (mm3)"].agg(['mean','median','var','std','sem'])
```

Out[15]:

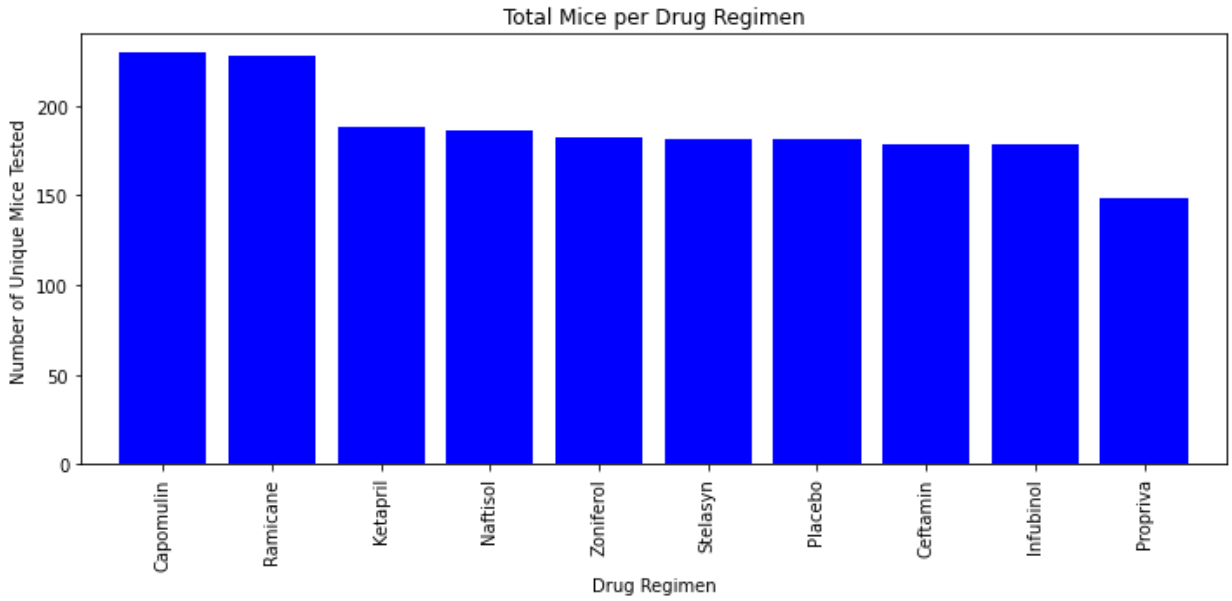
Drug Regimen	mean	median	var	std	sem
Capomulin	40.675741	41.557809	24.947764	4.994774	0.329346
Ceftamin	52.591172	51.776157	39.290177	6.268188	0.469821
Infubinol	52.884795	51.820584	43.128684	6.567243	0.492236
Ketapril	55.235638	53.698743	68.553577	8.279709	0.603860
Naftisol	54.331565	52.509285	66.173479	8.134708	0.596466
Placebo	54.033581	52.288934	61.168083	7.821003	0.581331
Propriva	52.320930	50.446266	43.852013	6.622085	0.544332
Ramicane	40.216745	40.673236	23.486704	4.846308	0.320955
Stelasyn	54.233149	52.431737	59.450562	7.710419	0.573111
Zoniferol	53.236507	51.818479	48.533355	6.966589	0.516398

Bar and Pie Charts

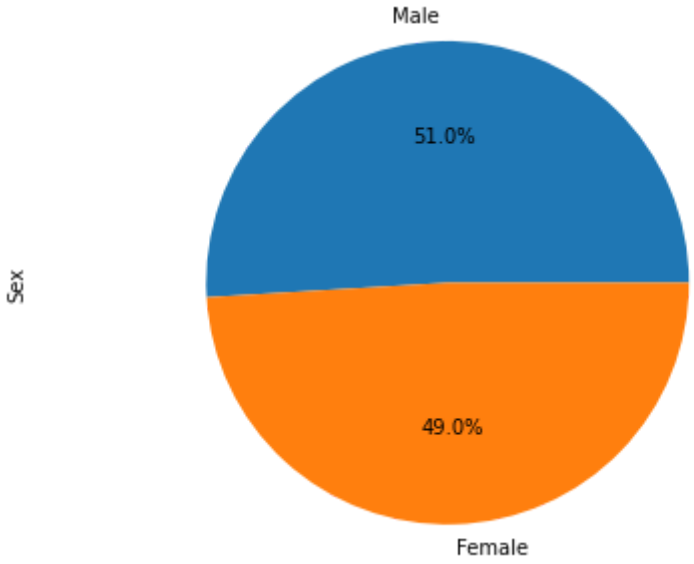
In [16]:
```
# Generate a bar plot showing the total number of unique mice tested on eac
drug_count = merge_df["Drug Regimen"].value_counts()

drug_count.plot(kind="bar", figsize=(10,5), title="Total Mice per Drug Regi
plt.xlabel("Drug Regimen")
plt.ylabel("Number of Unique Mice Tested")
plt.tight_layout()
plt.show()
```



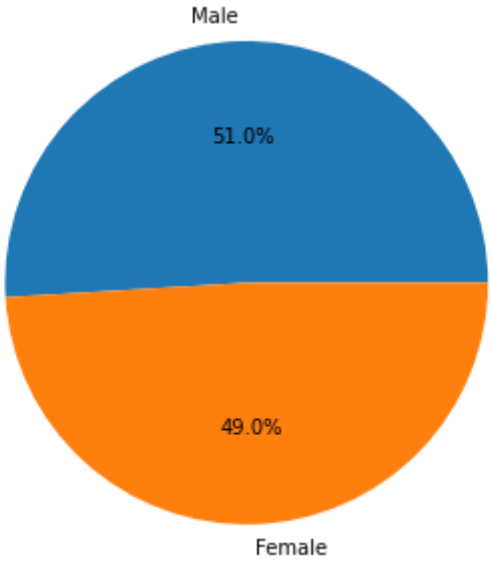
In [17]:
```
# Generate a bar plot showing the total number of unqiue mice tested on eac
import numpy as np
x_axis = np.arange(0, len(drug_count))
count = drug_count
drugs = ["Capomulin","Ramicane","Ketapril","Naftisol","Zoniferol","Stelasyn
tick_locations = []
for x in x_axis:
    tick_locations.append(x)

plt.figure(figsize=(10,5))
plt.title("Total Mice per Drug Regimen")
plt.xlabel("Drug Regimen")
plt.ylabel("Number of Unique Mice Tested")

plt.xlim(-0.75, len(drugs)-.25)
plt.ylim(0, max(count) + 10)

plt.bar(x_axis, count, facecolor="blue", alpha=1, align="center")
plt.xticks(tick_locations, drugs, rotation="vertical")
plt.tight_layout()
plt.show()
```



In [18]:
```
# Generate a pie plot showing the distribution of female versus male mice u

sex_count = merge_df["Sex"].value_counts()
sex_count

sex_pie = sex_count.plot(kind="pie", autopct="%1.1f%%", subplots=True)
plt.tight_layout()
plt.axis("equal")
plt.show()
```



In [19]:
```
# Generate a pie plot showing the distribution of female versus male mice u

sex = ["Male","Female"]
plt.pie(sex_count, labels=sex,
        autopct="%1.1f%%")
plt.tight_layout()
plt.axis("equal")
plt.show()
```



Quartiles, Outliers and Boxplots

In [20]:
```
# Calculate the final tumor volume of each mouse across four of the treatme
# Capomulin, Ramicane, Infubinol, and Ceftamin

# Start by getting the last (greatest) timepoint for each mouse
final_vol = merge_df.groupby(["Mouse ID"])
final_vol_df = pd.DataFrame(final_vol["Timepoint"].max())
final_vol_df




# Merge this group df with the original dataframe to get the tumor volume a

merge5_df = pd.merge(final_vol_df, merge_df, on = "Mouse ID", how = "inner"
merge5_df = merge5_df.loc[(merge5_df["Timepoint_x"] == merge5_df["Timepoint
merge5_df = merge5_df.loc[(merge5_df["Drug Regimen"] == "Capomulin") | (mer
merge5_df = merge5_df[["Drug Regimen","Tumor Volume (mm3)"]]
merge5_df
```

Out[20]:

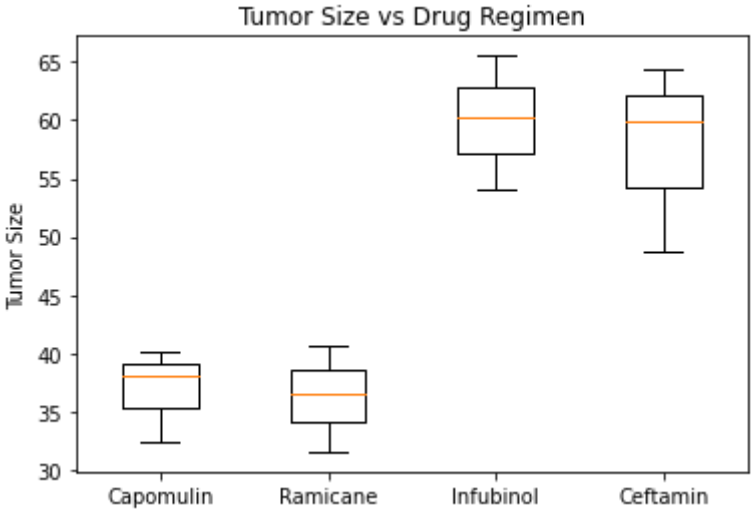
	Drug Regimen	Tumor Volume (mm3)
9	Infubinol	67.973419
19	Infubinol	65.525743
39	Ceftamin	62.999356
66	Ramicane	38.407618
76	Ramicane	43.047543
...	...	...
1812	Ceftamin	68.594745
1822	Capomulin	31.896238
1832	Ceftamin	64.729837
1849	Ramicane	30.638696
1859	Infubinol	62.754451

100 rows × 2 columns

In [21]:
```
# Put treatments into a list for for loop (and later for plot labels)

treatments = ["Capomulin", "Ramicane", "Infubinol", "Ceftamin"]

# Create empty list to fill with tumor vol data (for plotting)

tumor_vol = []


# Calculate the IQR and quantitatively determine if there are any potential

for drug in treatments:

    # Locate the rows which contain mice on each drug and get the tumor vol

    abc = merge5_df.loc[(merge5_df["Drug Regimen"] == drug), :]
    efg = abc["Tumor Volume (mm3)"].quantile([.25,.5,.75])
    tumor_vol.append(efg)

    outliers = []
    lowerq = []
    upperq = []
    iqr = []
    lower_bound = []
    upper_bound = []

    # add subset


    # Determine outliers using upper and lower bounds
    for tumor in tumor_vol:
        lowerq = (efg[0.25])
        upperq = (efg[0.75])
        iqr = upperq-lowerq
        lower_bound = lowerq - (1.5*iqr)
        upper_bound = upperq + (1.5*iqr)
        outliers = abc.loc[(abc["Tumor Volume (mm3)"] < lower_bound) | (abc

    print(f"{drug}'s potential outliers: {outliers}")
```

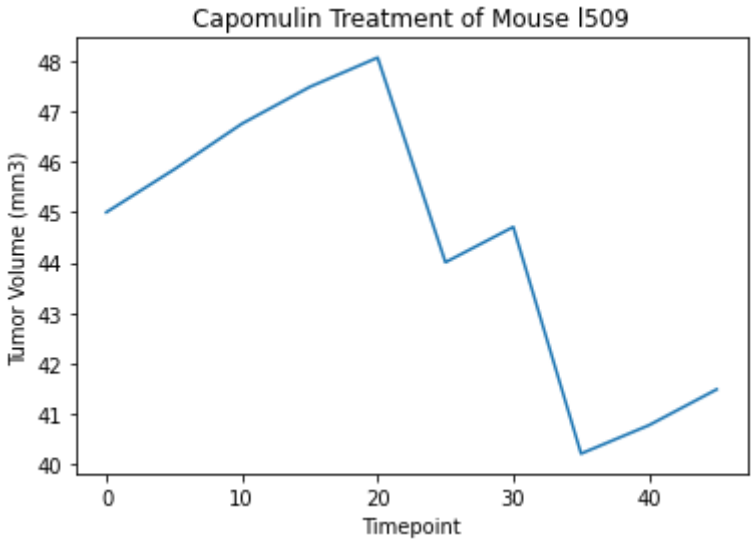
```
Capomulin's potential outliers: Empty DataFrame
Columns: [Drug Regimen, Tumor Volume (mm3)]
Index: []
Ramicane's potential outliers: Empty DataFrame
Columns: [Drug Regimen, Tumor Volume (mm3)]
Index: []
Infubinol's potential outliers:     Drug Regimen  Tumor Volume (mm3)
275    Infubinol           36.321346
Ceftamin's potential outliers: Empty DataFrame
Columns: [Drug Regimen, Tumor Volume (mm3)]
Index: []
```

In [22]:
```
# Generate a box plot of the final tumor volume of each mouse across four r
plots = tumor_vol
fig1, ax1 = plt.subplots()
ax1.set_title('Tumor Size vs Drug Regimen')
ax1.set_ylabel('Tumor Size')
ax1.boxplot(plots)
x_axis = [1,2,3,4]
tick_locations = [value for value in x_axis]
plt.xticks(tick_locations, treatments)
plt.show()
```

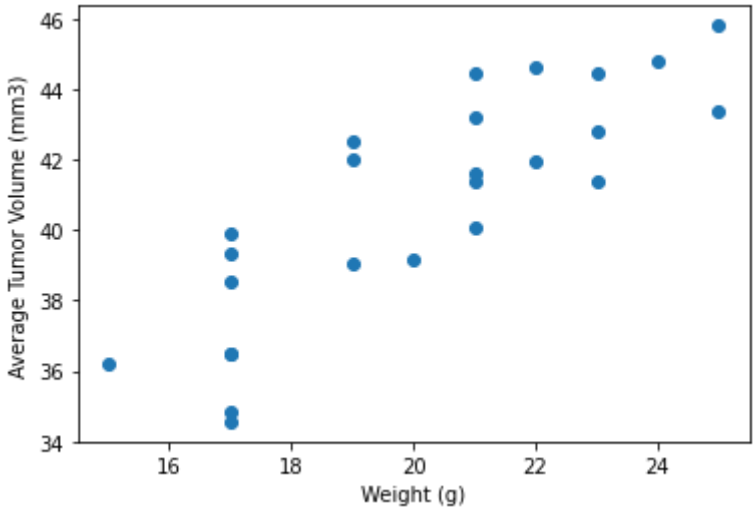


Line and Scatter Plots

In [23]:
```
# Generate a line plot of tumor volume vs. time point for a mouse treated w

merge_df6 = merge_df[["Mouse ID","Timepoint","Tumor Volume (mm3)","Drug Reg
merge_df6 = merge_df6.loc[(merge_df6["Drug Regimen"] == "Capomulin"),:]
merge_df6 = merge_df6.loc[(merge_df6["Mouse ID"] == "l509"),:]
merge_df6
x = merge_df6["Timepoint"]
y = merge_df6["Tumor Volume (mm3)"]
plt.plot(x,y)
plt.title("Capomulin Treatment of Mouse l509")
plt.xlabel("Timepoint")
plt.ylabel("Tumor Volume (mm3)")
plt.show()
```

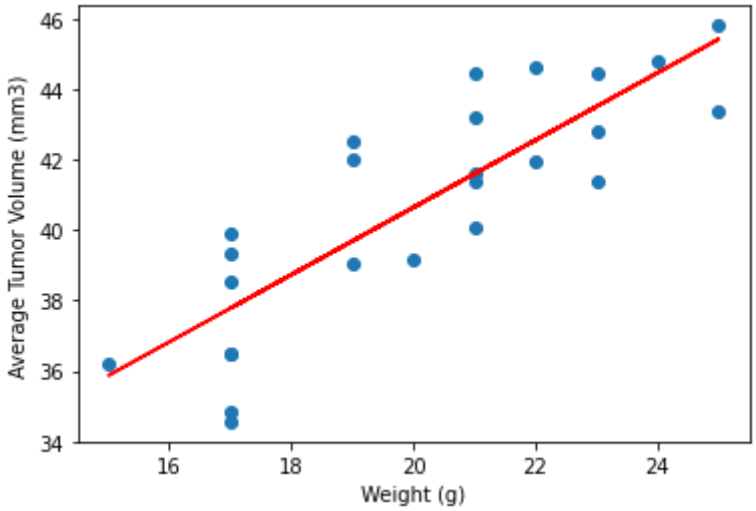


In [24]:
```
# Generate a scatter plot of average tumor volume vs. mouse weight for the
cap = merge_df.loc[(merge_df["Drug Regimen"] == "Capomulin"),:]
avg_vol = cap.groupby("Mouse ID").mean()
avg_vol

x2 = avg_vol["Weight (g)"]
y2 = avg_vol["Tumor Volume (mm3)"]
plt.scatter(x2,y2)
plt.xlabel("Weight (g)")
plt.ylabel("Average Tumor Volume (mm3)")
plt.show()
```



Correlation and Regression

In [24]:
```
# Calculate the correlation coefficient and linear regression model
# for mouse weight and average tumor volume for the Capomulin regimen

from scipy.stats import linregress

(slope, intercept, rvalue, pvalue, stderr) = linregress(x2, y2)
regress_values = x2 * slope + intercept
plt.scatter(x2,y2)
plt.xlabel("Weight (g)")
plt.ylabel("Average Tumor Volume (mm3)")
plt.plot(x2,regress_values,"r-")
plt.show()

print(f"The correlation between mouse weight and the average tumor volume i
```




```
The correlation between mouse weight and the average tumor volume is 0.8
4.
```

In [ ]: