Observations and Insights

In [1]:

```
# Dependencies and Setup
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
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)
# print(mouse_metadata)
study_results = pd.read_csv(study_results_path)
# print(study_results)

# Combine the data into a single dataset
combined_data = pd.merge(study_results,mouse_metadata,on="Mouse ID", how="left")
combined_data

# Display the data table for preview
# combined_data

# drop_duplicates()
# combined_data.sort_values("Mouse ID",inplace = True)
# combined_data.drop_duplicates(subset ="Mouse ID",
#                        keep = False, inplace = True)
# combined_data.describe()
```

Out[1]:

	Mouse ID	Timepoint	Tumor Volume (mm3)	Metastatic Sites	Drug Regimen	Sex	Age_months	Weight (g)
0	b128	0	45.000000	0	Capomulin	Female	9	22
1	f932	0	45.000000	0	Ketapril	Male	15	29
2	g107	0	45.000000	0	Ketapril	Female	2	29
3	a457	0	45.000000	0	Ketapril	Female	11	30
4	c819	0	45.000000	0	Ketapril	Male	21	25
...	...	...	...	...	...	...	...	...
1888	r944	45	41.581521	2	Capomulin	Male	12	25
1889	u364	45	31.023923	3	Capomulin	Male	18	17
1890	p438	45	61.433892	1	Ceftamin	Female	11	26
1891	x773	45	58.634971	4	Placebo	Female	21	30
1892	b879	45	72.555239	2	Stelasyn	Female	4	26

1893 rows × 8 columns

In [2]:

```
# Checking the number of mice.
number_of_mouse = len(combined_data['Mouse ID'].unique())
number_of_mouse
```

Out[2]:

```
249
```

In [3]:

```
# Getting the duplicate mice by ID number that shows up for Mouse ID and Timep
oint.

duplicated_ID = combined_data.loc[combined_data.duplicated(subset=["Mouse ID",
"Timepoint"]),"Mouse ID"].unique()
duplicated_ID
```

Out[3]:

```
array(['g989'], dtype=object)
```

In [4]:

```
# Optional: Get all the data for the duplicate mouse ID.
combined_data.loc[combined_data["Mouse ID"]=='g989']
```

Out[4]:

	Mouse ID	Timepoint	Tumor Volume (mm3)	Metastatic Sites	Drug Regimen	Sex	Age_months	Weight (g)
107	g989	0	45.000000	0	Propriva	Female	21	26
137	g989	0	45.000000	0	Propriva	Female	21	26
329	g989	5	48.786801	0	Propriva	Female	21	26
360	g989	5	47.570392	0	Propriva	Female	21	26
620	g989	10	51.745156	0	Propriva	Female	21	26
681	g989	10	49.880528	0	Propriva	Female	21	26
815	g989	15	51.325852	1	Propriva	Female	21	26
869	g989	15	53.442020	0	Propriva	Female	21	26
950	g989	20	55.326122	1	Propriva	Female	21	26
1111	g989	20	54.657650	1	Propriva	Female	21	26
1195	g989	25	56.045564	1	Propriva	Female	21	26
1380	g989	30	59.082294	1	Propriva	Female	21	26
1592	g989	35	62.570880	2	Propriva	Female	21	26

In [5]:

```
# Create a clean DataFrame by dropping the duplicate mouse by its ID.
combined_data = combined_data.loc[combined_data["Mouse ID"]!='g989']
combined_data
```

Out[5]:

	Mouse ID	Timepoint	Tumor Volume (mm3)	Metastatic Sites	Drug Regimen	Sex	Age_months	Weight (g)
0	b128	0	45.000000	0	Capomulin	Female	9	22
1	f932	0	45.000000	0	Ketapril	Male	15	29
2	g107	0	45.000000	0	Ketapril	Female	2	29
3	a457	0	45.000000	0	Ketapril	Female	11	30
4	c819	0	45.000000	0	Ketapril	Male	21	25
...	...	...	...	...	...	...	...	...
1888	r944	45	41.581521	2	Capomulin	Male	12	25
1889	u364	45	31.023923	3	Capomulin	Male	18	17
1890	p438	45	61.433892	1	Ceftamin	Female	11	26
1891	x773	45	58.634971	4	Placebo	Female	21	30
1892	b879	45	72.555239	2	Stelasyn	Female	4	26

1880 rows × 8 columns

In [6]:

```
# Checking the number of mice in the clean DataFrame.
number_of_mice = len(combined_data["Mouse ID"].unique())
number_of_mice
```

Out[6]:

```
248
```

Summary Statistics

In [7]:

```
# Generate a summary statistics table of mean, median, variance, standard deviation, and SEM of the tumor volume for each regimen
mean = combined_data.groupby(["Drug Regimen"]).mean()["Tumor Volume (mm3)"]
mean

median = combined_data.groupby(["Drug Regimen"]).median()["Tumor Volume (mm3)"]
median

variance = combined_data.groupby(["Drug Regimen"]).var()["Tumor Volume (mm3)"]
variance

st_dev = combined_data.groupby(["Drug Regimen"]).std()["Tumor Volume (mm3)"]
st_dev

SEM = combined_data.groupby(["Drug Regimen"]).sem()["Tumor Volume (mm3)"]
SEM
# Use groupby and summary statistical methods to calculate the following properties of each drug regimen:
# mean, median, variance, standard deviation, and SEM of the tumor volume.
# Assemble the resulting series into a single summary dataframe.

statistic = pd.DataFrame({"mean": mean,
                          "median": median,
                          "variance": variance,
                          "standard deviation": st_dev,
                          "SEM": SEM})
statistic
```

Out[7]:

	mean	median	variance	standard deviation	SEM
Drug Regimen					
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 [8]:

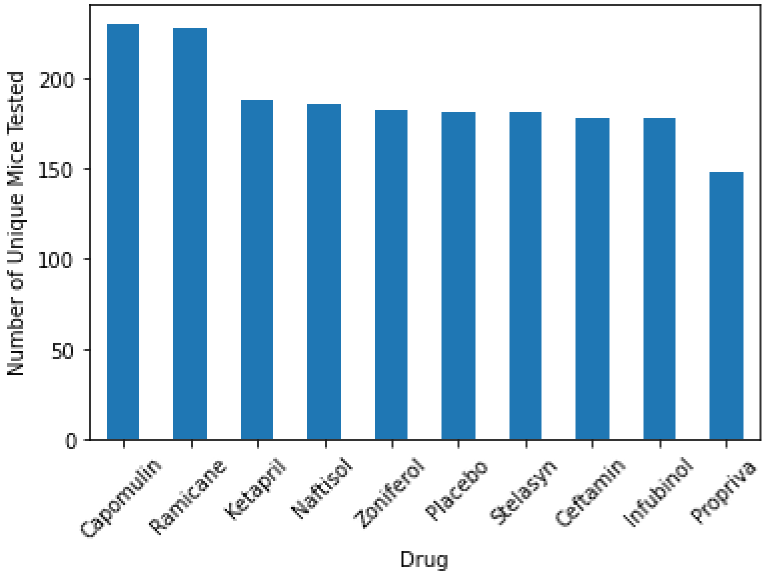
```
# Generate a summary statistics table of mean, median, variance, standard deviation, and SEM of the tumor volume for each regimen
summary_table = combined_data.groupby(["Drug Regimen"]).agg({"Tumor Volume (mm3)":["mean","median","var","std","sem"]})
summary_table
# Using the aggregation method, produce the same summary statistics in a single line
```

Out[8]:

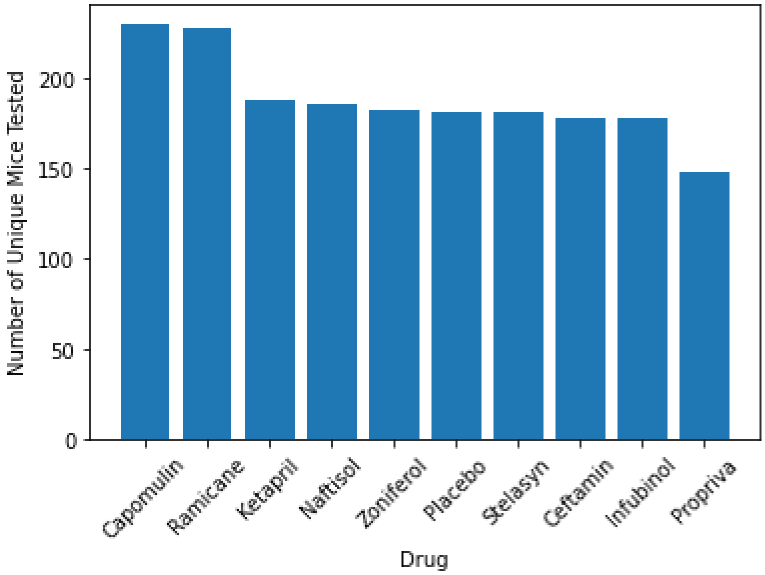
	Tumor Volume (mm3)				
	mean	**median**	**var**	**std**	**sem**
Drug Regimen					
Capomulin	40.675741	41.557809	24.947764	4.994774	0.329346
Ceftamin	52.591172	51.776157	39.290177	6.268188	0.469821
Infubinol	52.884795	51.820584	43.128684	6.567243	0.492236
Ketapril	55.235638	53.698743	68.553577	8.279709	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 [9]:
```
# Generate a bar plot showing the total number of unique mice tested on each d
rug regimen using pandas.
drug_count = combined_data["Drug Regimen"].value_counts()
drug_count
drug_count.plot(kind="bar", rot=45)
plt.xlabel("Drug")
plt.ylabel("Number of Unique Mice Tested")
plt.show()
```

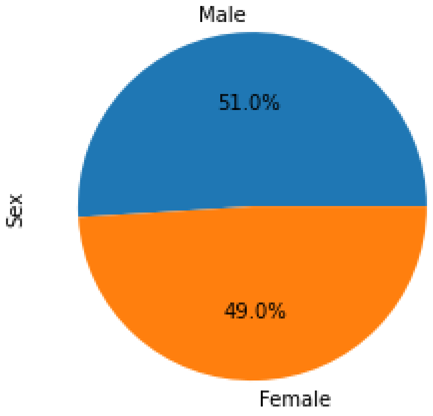


In [10]:
```
# Generate a bar plot showing the total number of unique mice tested on each d
rug regimen using pyplot.
plt.bar(drug_count.index.values,drug_count.values)
plt.xticks(rotation=45)
plt.xlabel("Drug")
plt.ylabel("Number of Unique Mice Tested")
plt.show()
```



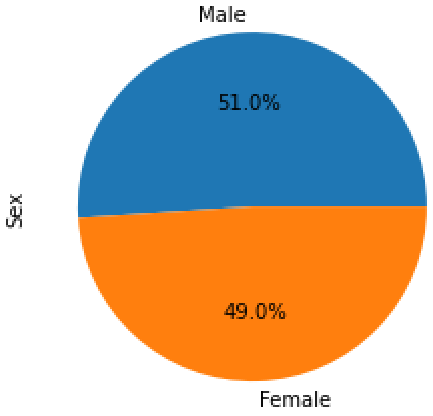
In [11]:

```
# Generate a pie plot showing the distribution of female versus male mice using pandas

gender_count = combined_data["Sex"].value_counts()
gender_count
gender_count.plot(kind="pie",autopct="%1.1f%%")
plt.show()
```



In [12]:

```
# Generate a pie plot showing the distribution of female versus male mice using pyplot
plt.pie(gender_count.values,labels=gender_count.index.values,autopct="%1.1f%%")
plt.ylabel("Sex")
plt.show()
```



Quartiles, Outliers and Boxplots

In [13]:

```
# Calculate the final tumor volume of each mouse across four of the treatment regimens:
# Capomulin, Ramicane, Infubinol, and Ceftamin

# Start by getting the last (greatest) timepoint for each mouse
last_tp_df = combined_data.drop_duplicates(subset=['Mouse ID'],keep='last')
last_tp_df.head()

# Prepare separate df for each drug
Capomulin_df = last_tp_df[last_tp_df['Drug Regimen']=='Capomulin']
Ramicane_df = last_tp_df[last_tp_df['Drug Regimen']=='Ramicane']
Infubinol_df = last_tp_df[last_tp_df['Drug Regimen']=='Infubinol']
Ceftamin_df = last_tp_df[last_tp_df['Drug Regimen']=='Ceftamin']
```

In [14]:

```
# Merge this group df with the original dataframe to get the tumor volume at t
he last timepoint
Capomulin_merge = pd.merge(Capomulin_df,combined_data,on=["Mouse ID","Drug Reg
imen","Sex","Age_months","Weight (g)","Timepoint","Tumor Volume (mm3)","Metast
atic Sites"],how='left')
Capomulin_merge
Ramicane_merge = pd.merge(Capomulin_merge,Ramicane_df,on=["Mouse ID","Drug Reg
imen","Sex","Age_months","Weight (g)","Timepoint","Tumor Volume (mm3)","Metast
atic Sites"],how='outer')
Ramicane_merge
Infubinol_merge = pd.merge(Ramicane_merge,Infubinol_df,on=["Mouse ID","Drug Re
gimen","Sex","Age_months","Weight (g)","Timepoint","Tumor Volume (mm3)","Metas
tatic Sites"],how='outer')
Infubinol_merge
Total_merge = pd.merge(Infubinol_merge,Ceftamin_df,on=["Mouse ID","Drug Regime
n","Sex","Age_months","Weight (g)","Timepoint","Tumor Volume (mm3)","Metastati
c Sites"],how='outer')
Total_merge.head
```

Out[14]:

```
<bound method NDFrame.head of    Mouse ID  Timepoint  Tumor Volume (mm3)  Met
astatic Sites Drug Regimen  \
0      w150         10           39.952347                 0    Capomulin
1      r157         15           46.539206                 0    Capomulin
2      f966         20           30.485985                 0    Capomulin
3      j246         35           38.753265                 1    Capomulin
4      b128         45           38.982878                 2    Capomulin
..      ...        ...                 ...               ...          ...
95     k210         45           68.923185                 3     Ceftamin
96     x822         45           61.386660                 3     Ceftamin
97     l733         45           64.299830                 1     Ceftamin
98     o287         45           59.741901                 4     Ceftamin
99     p438         45           61.433892                 1     Ceftamin

       Sex  Age_months  Weight (g)
0     Male          23          23
1     Male          22          25
2     Male          16          17
3   Female          21          21
4   Female           9          22
..     ...         ...         ...
95    Male          15          28
96    Male           3          29
97  Female           4          30
98    Male           2          28
99  Female          11          26

[100 rows x 8 columns]>
```

In [15]:

```
# Put treatments into a list for for loop (and later for plot labels)
# Create empty list to fill with tumor vol data (for plotting)
# Calculate the IQR and quantitatively determine if there are any potential outliers.

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

    # add subset

    # Determine outliers using upper and lower bounds

Total_tumor_vol_data = Total_merge['Tumor Volume (mm3)']
quartiles = Total_tumor_vol_data.quantile([.25,.5,.75])
lowerq = quartiles[0.25]
upperq = quartiles[0.75]
iqr = upperq-lowerq


lower_bound = lowerq - (1.5*iqr)
upper_bound = upperq + (1.5*iqr)
print(f"The four treatments potential outliers could be values below {lower_bound} and above {upper_bound} could be outliers.")
```

```
The four treatments potential outliers could be values below 3.0739668737499812 and above 94.04403868375002 could be outliers.
```

In [16]:

```
#### CAPOMULIN
# Merge this group df with the original dataframe to get the tumor volume at the last timepoint
Capomulin_merge = pd.merge(Capomulin_df,combined_data,on=["Mouse ID","Drug Regimen","Sex","Age_months","Weight (g)","Timepoint","Tumor Volume (mm3)","Metastatic Sites"],how='left')
Capomulin_merge.head()

# Preparing Tumor volume data for Capomulin
cap_tumor_vol_data = Capomulin_merge['Tumor Volume (mm3)']

quartiles = cap_tumor_vol_data.quantile([.25,.5,.75])
lowerq = quartiles[0.25]
upperq = quartiles[0.75]
iqr = upperq-lowerq


lower_bound = lowerq - (1.5*iqr)
upper_bound = upperq + (1.5*iqr)
print(f"Capomulin potential outliers could be values below {lower_bound} and above {upper_bound} could be outliers.")
```

```
Capomulin potential outliers could be values below 20.70456164999999 and above 51.83201549 could be outliers.
```

In [17]:

```
#### RAMICANE
# Merge this group df with the original dataframe to get the tumor volume at the last timepoint
Ramicane_merge = pd.merge(Ramicane_df,combined_data,on=["Mouse ID","Drug Regimen","Sex","Age_months","Weight (g)","Timepoint","Tumor Volume (mm3)","Metastatic Sites"],how='left')
Ramicane_merge.head()

# Preparing Tumor volume data for Ramicane
ram_tumor_vol_data = Ramicane_merge['Tumor Volume (mm3)']

quartiles2 = ram_tumor_vol_data.quantile([.25,.5,.75])
lowerq2 = quartiles2[0.25]
upperq2 = quartiles2[0.75]
iqr2 = upperq2-lowerq2


lower_bound2 = lowerq2 - (1.5*iqr2)
upper_bound2 = upperq2 + (1.5*iqr2)
print(f"Capomulin potential outliers could be values below {lower_bound2} and above {upper_bound2} could be outliers.")
```

```
Capomulin potential outliers could be values below 17.912664470000003 and above 54.30681135 could be outliers.
```

In [18]:

```
#### INFUBINOL
# Merge this group df with the original dataframe to get the tumor volume at the last timepoint
Infubinol_merge = pd.merge(Infubinol_df,combined_data,on=["Mouse ID","Drug Regimen","Sex","Age_months","Weight (g)","Timepoint","Tumor Volume (mm3)","Metastatic Sites"],how='left')
Infubinol_merge.head()

# Preparing Tumor volume data for infubinol
infu_tumor_vol_data = Infubinol_merge['Tumor Volume (mm3)']

quartiles3 = infu_tumor_vol_data.quantile([.25,.5,.75])
lowerq3 = quartiles3[0.25]
upperq3 = quartiles3[0.75]
iqr3 = upperq3-lowerq3


lower_bound3 = lowerq3 - (1.5*iqr3)
upper_bound3 = upperq3 + (1.5*iqr3)
print(f"Capomulin potential outliers could be values below {lower_bound3} and above {upper_bound3} could be outliers.")
```

```
Capomulin potential outliers could be values below 36.83290494999999 and above 82.74144559000001 could be outliers.
```

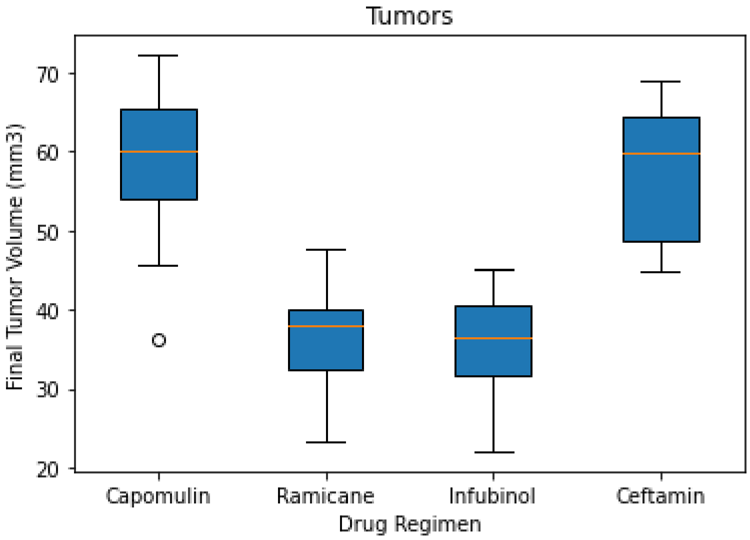
In [19]:

```
#### CEFTAMINE
# Merge this group df with the original dataframe to get the tumor volume at the last timepoint

Ceftamin_merge = pd.merge(Ceftamin_df,combined_data,on=["Mouse ID","Drug Regimen","Sex","Age_months","Weight (g)","Timepoint","Tumor Volume (mm3)","Metastatic Sites"],how='left')
Ceftamin_merge.head()

# Preparing Tumor volume data for Ceftmaine

ceft_tumor_vol_data = Ceftamin_merge['Tumor Volume (mm3)']

quartiles4 = ceft_tumor_vol_data.quantile([.25,.5,.75])
lowerq4 = quartiles4[0.25]
upperq4 = quartiles4[0.75]
iqr4 = upperq4-lowerq4


lower_bound4 = lowerq4 - (1.5*iqr4)
upper_bound4 = upperq4 + (1.5*iqr4)
print(f"Capomulin potential outliers could be values below {lower_bound4} and above {upper_bound4} could be outliers.")
```

```
Capomulin potential outliers could be values below 25.355449580000002 and above 87.66645829999999 could be outliers.
```

In [20]:

```
data_to_plot = [infu_tumor_vol_data,cap_tumor_vol_data,ram_tumor_vol_data,ceft_tumor_vol_data]

fig1, ax1 = plt.subplots()
ax1.set_title('Tumors')
ax1.set_ylabel('Final Tumor Volume (mm3)')
ax1.set_xlabel('Drug Regimen')

ax1.boxplot(data_to_plot,labels=["Capomulin","Ramicane","Infubinol","Ceftamin"], patch_artist=True, vert=True)

plt.savefig('boxplot')
plt.show()
```



Line and Scatter Plots

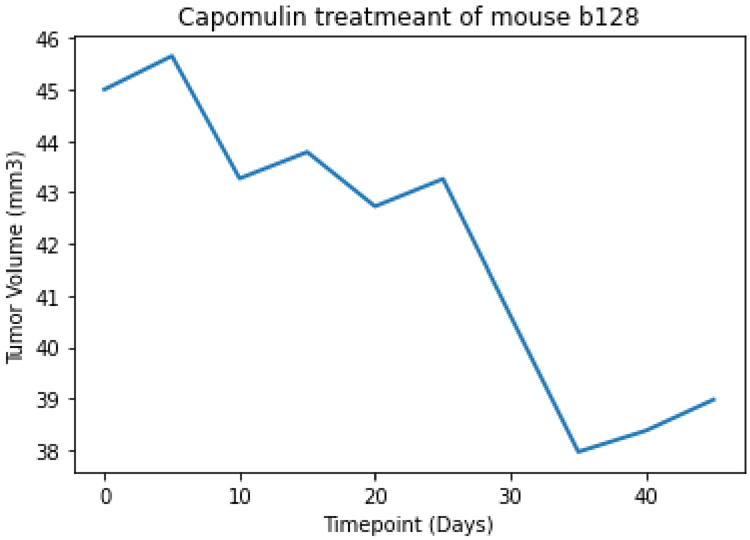
In [21]:

```
# Generate a line plot of tumor volume vs. time point for a mouse treated with Capomulin
Cap_line = combined_data[combined_data["Drug Regimen"] == "Capomulin"]
Cap_line
forline_df = Cap_line.loc[Cap_line["Mouse ID"] == "b128",:]
forline_df.head()
```

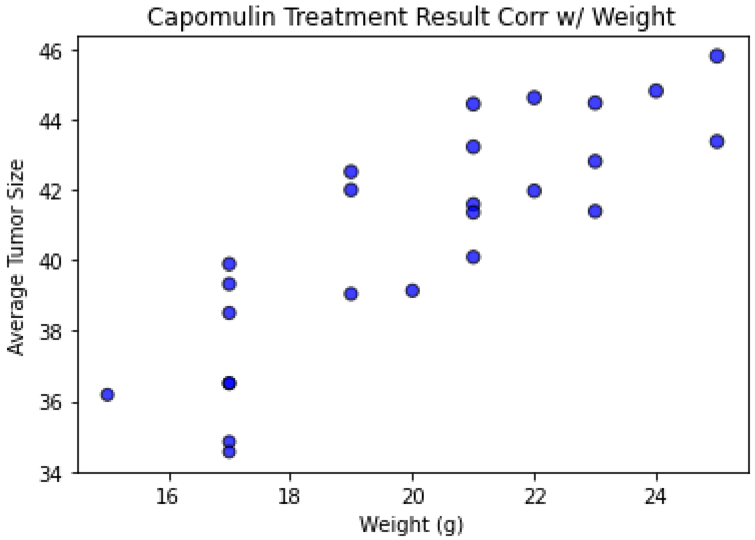
Out[21]:

	Mouse ID	Timepoint	Tumor Volume (mm3)	Metastatic Sites	Drug Regimen	Sex	Age_months	Weight (g)
0	b128	0	45.000000	0	Capomulin	Female	9	22
472	b128	5	45.651331	0	Capomulin	Female	9	22
679	b128	10	43.270852	0	Capomulin	Female	9	22
719	b128	15	43.784893	0	Capomulin	Female	9	22
933	b128	20	42.731552	0	Capomulin	Female	9	22

In [22]:
```
x_axisTP = forline_df["Timepoint"]
tumsiz = forline_df["Tumor Volume (mm3)"]

plt.title('Capomulin treatmeant of mouse b128')
plt.plot(x_axisTP, tumsiz,linewidth=2, markersize=12)
plt.xlabel('Timepoint (Days)')
plt.ylabel('Tumor Volume (mm3)')

plt.savefig('linechart')
plt.show()
```



In [23]:
```
# Generate a scatter plot of average tumor volume vs. mouse weight for the Cap
omulin regimen
cap_regimen =Cap_line.groupby("Mouse ID").mean()
x_axis = cap_regimen["Weight (g)"]
y_axis = cap_regimen["Tumor Volume (mm3)"]

plt.scatter(x_axis, y_axis, marker="o",facecolors="blue", edgecolors="black",s
=y_axis,alpha=0.75)
plt.xlabel("Weight (g)")
plt.ylabel("Average Tumor Size")
plt.title("Capomulin Treatment Result Corr w/ Weight")
plt.show()
```



Correlation and Regression

In [24]:
```
# Calculate the correlation coefficient and linear regression model

corr=round(st.pearsonr(cap_regimen['Weight (g)'],cap_regimen['Tumor Volume (mm
3)'])[0],2)
print(f"The correlation between mouse weight and average tumor volume is {corr
}")
```

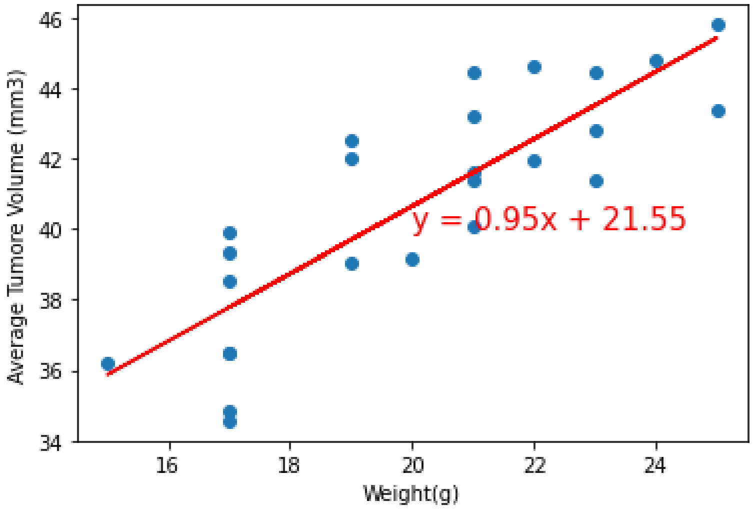
```
The correlation between mouse weight and average tumor volume is 0.84
```

In [25]:
```
model=st.linregress(cap_regimen['Weight (g)'],cap_regimen['Tumor Volume (mm3)'
])
model
```

Out[25]:
```
LinregressResult(slope=0.9544396890241045, intercept=21.552160532685015, rval
ue=0.8419363424694718, pvalue=1.3225722434712642e-07, stderr=0.12754359033201
34)
```

In [26]:
```
#capavg['Weight (g)']
m_slope = 0.9544396890241045
m_intercept = 21.552160532685015
```

In [39]:
```
# # for mouse weight and average tumor volume for the Capomulin regimen

x_values = cap_regimen["Weight (g)"]
y_values = cap_regimen["Tumor Volume (mm3)"]
m_slope, m_intercept, m_rvalue, m_pvalue, m_std_err = st.linregress(x_values,
y_values)
regress_values = x_values * m_slope + m_intercept
line_eq = "y = " + str(round(m_slope,2)) + "x + " + str(round(m_intercept,2))

plt.scatter(x_values,y_values)
plt.plot(x_values, regress_values,"r-")
plt.annotate(line_eq,(20,40),fontsize=15,color="red")
plt.xlabel('Weight(g)')
plt.ylabel('Average Tumore Volume (mm3)')
plt.show()

# # murder_rate = crime_data.iloc[:, 5]
# # m_slope, m_int, m_r, m_p, m_std_err = stats.linregress(year, murder_rate)
# # m_fit = m_slope * year + m_int
# # plt.scatter(year,murder_rate)
# # plt.plot(year,m_fit,"--")
# # plt.xticks(year, rotation=90)
# # plt.xlabel('Year')
# # plt.ylabel('Murder Rate')
# # plt.show()


# y_values = cap_regimen['Weight (g)']*m_slope+m_intercept
# plt.scatter(cap_regimen['Weight (g)'],cap_regimen['Tumor Volume (mm3)'])
# plt.plot(cap_regimen['Weight (g)'],y_values,color="red")
# plt.xlabel('Weight(g)')
# plt.ylabel('Average Tumore Volume (mm3)')


# plt.savefig('linearregression')
# plt.show()
```



In [ ]: