Pymaceuticals Inc.

Analysis

- Add your analysis here.

In [2]:

```
# 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 DataFrame
data_df = pd.merge(study_results, mouse_metadata, on='Mouse ID', how
# Display the data table for preview
data_df.head()
```

Alt+Q

Out[2]:

	Mouse ID	Timepoint	Tumor Volume (mm3)	Metastatic Sites	Drug Regimen	Sex	Age_months	Weight (g)
0	b128	0	45.0	0	Capomulin	Female	9	22
1	f932	0	45.0	0	Ketapril	Male	15	29
2	g107	0	45.0	0	Ketapril	Female	2	29
3	a457	0	45.0	0	Ketapril	Female	11	30
4	c819	0	45.0	0	Ketapril	Male	21	25

In [3]:

```
# Checking the number of mice.
mouse_count = len(data_df["Mouse ID"].unique())
mouse_count
```

Out[3]: 249

In [4]:

```
# Our data should be uniquely identified by Mouse ID and Timepoint
# Get the duplicate mice by ID number that shows up for Mouse ID and
duplicates = data_df.loc[data_df.duplicated(subset = ["Mouse ID", "
duplicates
```

Out[4]:

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

In [5]:

```
# Optional: Get all the data for the duplicate mouse ID.

```

In [6]:

```
# Create a clean DataFrame by dropping the duplicate mouse by its ID
clean_df = data_df[data_df['Mouse ID'].isin(duplicates)==False]
clean_df
```

Out[6]:

	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

In [7]:

```
# Checking the number of mice in the clean DataFrame.
clean_mice = clean_df["Mouse ID"].nunique()

clean_mice
```

Out[7]:

```
248
```

Summary Statistics

In [8]:

```
mean = clean_df.groupby(["Drug Regimen"])["Tumor Volume (mm3)"].mean
median = clean_df.groupby(["Drug Regimen"])["Tumor Volume (mm3)"].me
var = clean_df.groupby(["Drug Regimen"])["Tumor Volume (mm3)"].var()
std = clean_df.groupby(["Drug Regimen"])["Tumor Volume (mm3)"].std()
sem = clean_df.groupby(["Drug Regimen"])["Tumor Volume (mm3)"].sem()

summary_stat = pd.DataFrame({"Mean Tumor Volume":mean,
                             "Median Tumor Volume":median,
                            "Tumor Volume Variance":var,
                            "Tumor Volume Std. Dev.":std,
                            "Tumor Volume Std. Err.":sem})
# Display the Summary statistics table grouped by 'Drug Regimen' col
summary_stat
```

Out[8]:

Drug Regimen	Mean Tumor Volume	Median Tumor Volume	Tumor Volume Variance	Tumor Volume Std. Dev.	Tumor Volume Std. Err.
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 [9]:

```
# Generate a summary statistics table of mean, median, variance, sta

# Group data by Drug Regimen
summary_df = clean_df.groupby('Drug Regimen')

# Calculate the mean, median, standard deviation, and sem for each d
tumor_mean = summary_df['Tumor Volume (mm3)'].mean()
tumor_median = summary_df['Tumor Volume (mm3)'].median()
tumor_stdev = summary_df['Tumor Volume (mm3)'].std()
tumor_sem = summary_df['Tumor Volume (mm3)'].sem()

# Create DataFrame to summarize calculations
summary_grouped_df = pd.DataFrame({'Mean': tumor_mean, 'Median': tur
                                   'Standard Deviation': tumor_stdev

summary_grouped_df.head()
```

Out[9]:

Drug Regimen	Mean	Median	Standard Deviation	SEM
Capomulin	40.675741	41.557809	4.994774	0.329346
Ceftamin	52.591172	51.776157	6.268188	0.469821
Infubinol	52.884795	51.820584	6.567243	0.492236
Ketapril	55.235638	53.698743	8.279709	0.603860

In [10]:

```
mean_mouse = clean_df.groupby(["Drug Regimen"])["Tumor Volume (mm3)"
mean_mouse
median_mouse = clean_df.groupby(["Drug Regimen"])["Tumor Volume (mm3
median_mouse
```

Out[10]:

```
Drug Regimen
Capomulin    41.557809
Ceftamin     51.776157
Infubinol    51.820584
Ketapril     53.698743
Naftisol     52.509285
Placebo      52.288934
Propriva     50.446266
Ramicane     40.673236
Stelasyn     52.431737
Zoniferol    51.818479
Name: Tumor Volume (mm3), dtype: float64
```

In [48]:

```
# Generate a summary statistics table of mean, median, variance, sta

# Use groupby and summary statistical methods to calculate the follo
# mean, median, variance, standard deviation, and SEM of the tumor v
# Assemble the resulting series into a single summary DataFrame.
mean = clean_df.groupby(["Drug Regimen"])["Tumor Volume (mm3)"].mean
median = clean_df.groupby(["Drug Regimen"])["Tumor Volume (mm3)"].me
var = clean_df.groupby(["Drug Regimen"])["Tumor Volume (mm3)"].var()
std = clean_df.groupby(["Drug Regimen"])["Tumor Volume (mm3)"].std()
sem = clean_df.groupby(["Drug Regimen"])["Tumor Volume (mm3)"].sem()

summary_stat = pd.DataFrame({"Mean Tumor Volume":mean,
                             "Median Tumor Volume":median,
                            "Tumor Volume Variance":var,
                            "Tumor Volume Std. Dev.":std,
                            "Tumor Volume Std. Err.":sem})
# Display the Summary statistics table grouped by 'Drug Regimen' col
summary_stat
```

Out[48]:

Drug Regimen	Mean Tumor Volume	Median Tumor Volume	Tumor Volume Variance	Tumor Volume Std. Dev.	Tumor Volume Std. Err.
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

In [12]:

```
summary_agg =  clean_df.groupby(['Drug Regimen'])[['Tumor Volume (mm
summary_agg
```

Out[12]:

	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

In [13]:

```
# A more advanced method to generate a summary statistics table of m
# and SEM of the tumor volume for each regimen (only one method is r

# Using the aggregation method, produce the same summary statistics
summary_agg =  clean_df.groupby(['Drug Regimen'])[['Tumor Volume (mm
summary_agg
```

Out[13]:

	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

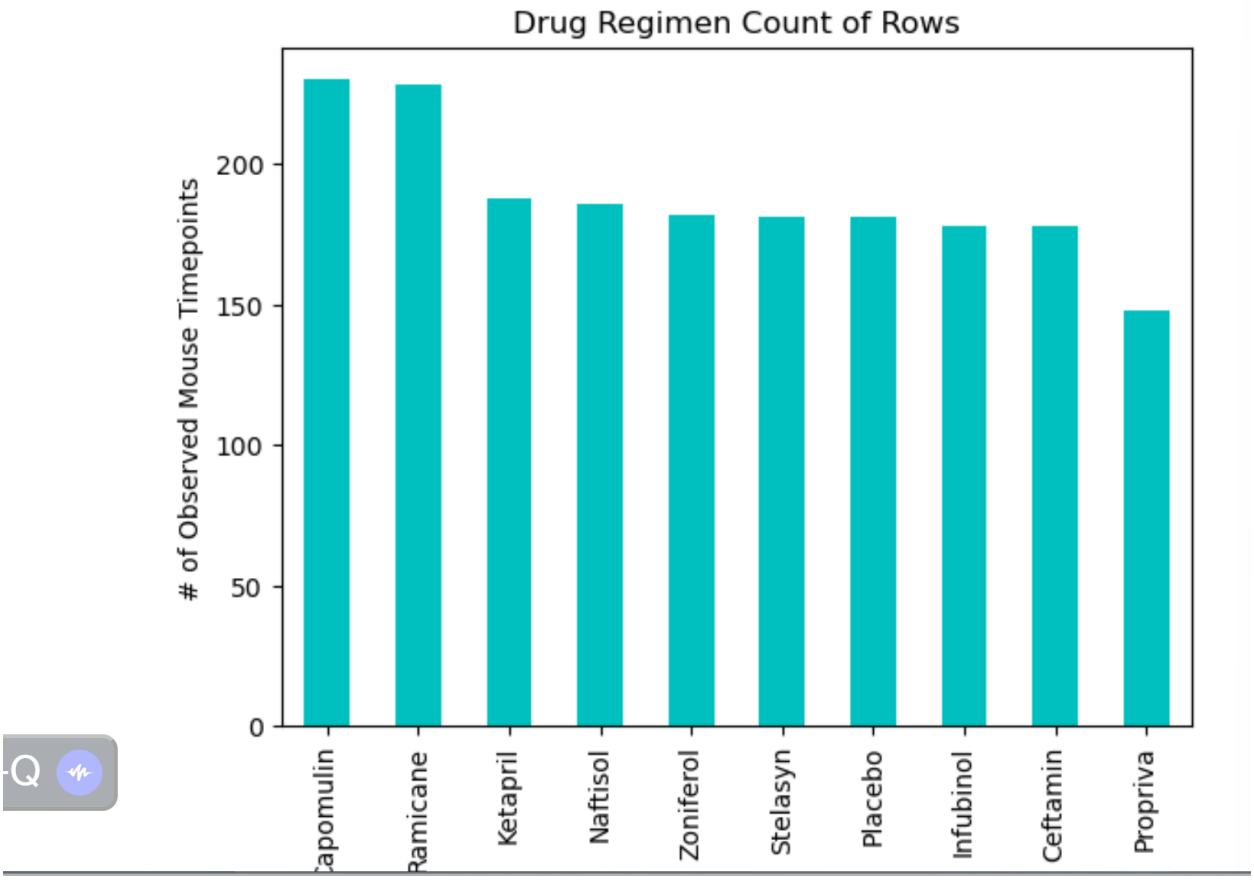
Bar and Pie Charts

In [14]:
```
mice_count = clean_df["Drug Regimen"].value_counts()
mice_count
```

Out[14]:
```
Drug Regimen
Capomulin    230
Ramicane     228
Ketapril     188
Naftisol     186
Zoniferol    182
Stelasyn     181
Placebo      181
Infubinol    178
Ceftamin     178
Propriva     148
Name: count, dtype: int64
```

In [15]:
```
plot_pandas = mice_count.plot.bar(color='b')
# Set the xlabel, ylabel, and title using class methods
plt.xlabel("Drug")
plt.ylabel("Count")
plt.title("Drug Regimen Count of Rows")
```

Out[15]:
```
Text(0.5, 1.0, 'Drug Regimen Count of Rows')
```

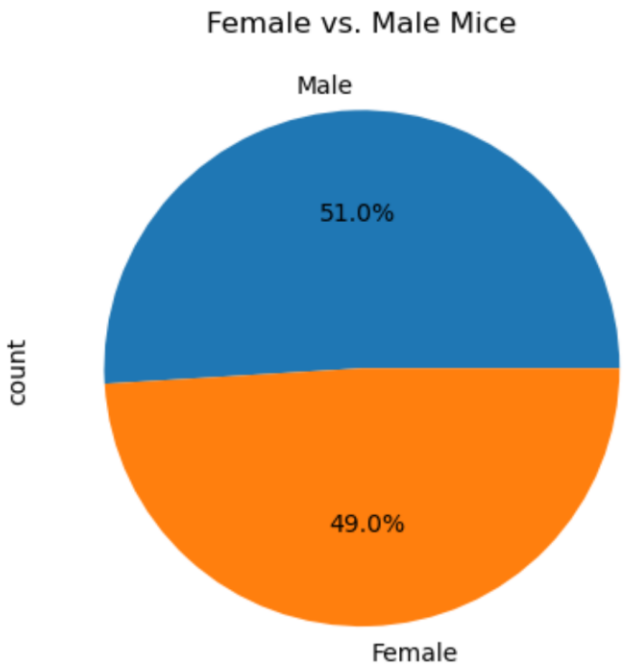


In [19]:

```
# Generate a bar plot showing the total number of rows (Mouse ID/Tir
x_axis = mice_count.index.values
y_axis = mice_count.values

# Create a Pyplot bar plot based off of the group series from before
plt.bar(x_axis, y_axis, color='b', alpha=0.8, align='center')

# Set the xlabel and ylabel
plt.xlabel("Drug Regimen")
plt.ylabel("# of Observed Mouse Timepoints")
plt.xticks(rotation="vertical")

plt.show()
```

In [20]:

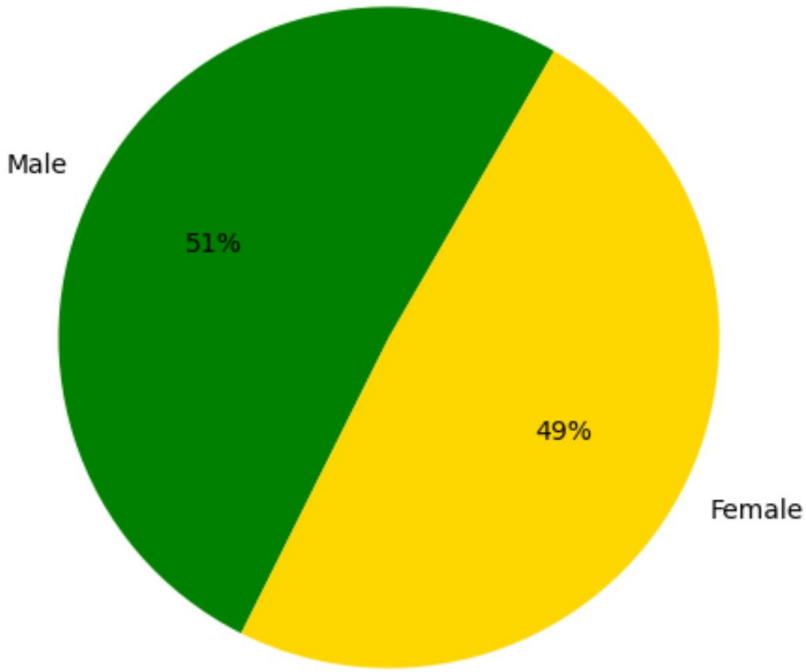
```
gender_data = clean_df["Sex"].value_counts()
plt.title("Female vs. Male Mice")
gender_data.plot.pie(autopct= "%1.1f%%")
plt.show()
```



Alt+Q

In [24]:

```
# Generate a pie plot showing the distribution of female versus male
pies = gender_data.index
pie_votes = gender_data

colors = ["green", "gold"]

plt.figure(figsize = (5,5))
plt.pie(pie_votes, labels = pies, colors = colors, autopct = "%1.0f%
plt.title("Gender Distribution", fontweight = "bold", fontsize = 12)

plt.axis("equal")

plt.show()
```



Quartiles, Outliers and Boxplots

In [25]:
```
cap_df = clean_df.loc[clean_df["Drug Regimen"] == "Capomulin",:]
ram_df = clean_df.loc[clean_df["Drug Regimen"] == "Ramicane", :]
inf_df = clean_df.loc[clean_df["Drug Regimen"] == "Infubinol", :]
cef_df = clean_df.loc[clean_df["Drug Regimen"] == "Ceftamin", :]
```

In [26]:
```
cap_last = cap_df.groupby('Mouse ID').max()['Timepoint']
cap_vol = pd.DataFrame(cap_last)
cap_merge = pd.merge(cap_vol, clean_df, on=("Mouse ID","Timepoint"),
cap_merge.head()
```

Out[26]:

	Mouse ID	Timepoint	Tumor Volume (mm3)	Metastatic Sites	Drug Regimen	Sex	Age_months	Weight (g)
0	b128	45	38.982878	2	Capomulin	Female	9	22
1	b742	45	38.939633	0	Capomulin	Male	7	21
2	f966	20	30.485985	0	Capomulin	Male	16	17
3	g288	45	37.074024	1	Capomulin	Male	3	19
4	g316	45	40.159220	2	Capomulin	Female	22	22

In [28]:
```
Capomulin_tumors = cap_merge["Tumor Volume (mm3)"]

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


print(f"The lower quartile of Capomulin tumors: {lowerq}")
print(f"The upper quartile of Capomulin tumors: {upperq}")
print(f"The interquartile range of Capomulin tumors: {iqr}")
print(f"The median of Capomulin tumors: {quartiles[0.5]} ")

lower_bound = lowerq - (1.5*iqr)
upper_bound = upperq + (1.5*iqr)

print(f"Values below {lower_bound} could be outliers.")
print(f"Values above {upper_bound} could be outliers.")
```

```
The lower quartile of Capomulin tumors: 32.37735684
The upper quartile of Capomulin tumors: 40.1592203
The interquartile range of Capomulin tumors: 7.781863460000004
The median of Capomulin tumors: 38.1251644
Values below 20.70456164999999 could be outliers.
Values above 51.83201549 could be outliers.
```

```
Ramicane_vol = pd.DataFrame(Ramicane_last)
Ramicane_merge = pd.merge(Ramicane_vol, clean_df, on=("Mouse ID","Ti
Ramicane_merge.head()
Ramicane_merge.to_csv("output.csv")
Ramicane_tumors = Ramicane_merge["Tumor Volume (mm3)"]

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


print(f"The lower quartile of Ramicane tumors is: {lowerq}")
print(f"The upper quartile of Ramicane tumors is: {upperq}")
print(f"The interquartile range of Ramicane tumors is: {iqr}")
print(f"The median of Ramicane tumors is: {quartiles[0.5]} ")

lower_bound = lowerq - (1.5*iqr)
upper_bound = upperq + (1.5*iqr)

print(f"Values below {lower_bound} could be outliers.")
print(f"Values above {upper_bound} could be outliers.")
```

```
The lower quartile of Ramicane tumors is: 31.56046955
The upper quartile of Ramicane tumors is: 40.65900627
The interquartile range of Ramicane tumors is: 9.098536719999998
The median of Ramicane tumors is: 36.56165229
Values below 17.912664470000003 could be outliers.
Values above 54.30681135 could be outliers.
```

t+Q

In [30]:

```
# Calculate the final tumor volume of each mouse across four of the
# Capomulin, Ramicane, Infubinol, and Ceftamin
Capomulin_df = clean_df.loc[clean_df["Drug Regimen"] == "Capomulin",
Ramicane_df = clean_df.loc[clean_df["Drug Regimen"] == "Ramicane", 
Infubinol_df = clean_df.loc[clean_df["Drug Regimen"] == "Infubinol",
Ceftamin_df = clean_df.loc[clean_df["Drug Regimen"] == "Ceftamin", 

Capomulin_last = Capomulin_df.groupby('Mouse ID').max()['Timepoint']
Capomulin_vol = pd.DataFrame(Capomulin_last)
Capomulin_merge = pd.merge(Capomulin_vol, clean_df, on=("Mouse ID","
Capomulin_merge.head()
# Start by getting the last (greatest) timepoint for each mouse
Capomulin_tumors = Capomulin_merge["Tumor Volume (mm3)"]

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


print(f"The lower quartile of Capomulin tumors: {lowerq}")
print(f"The upper quartile of Capomulin tumors: {upperq}")
print(f"The interquartile range of Capomulin tumors: {iqr}")
print(f"The median of Capomulin tumors: {quartiles[0.5]} ")

lower_bound = lowerq - (1.5*iqr)
upper_bound = upperq + (1.5*iqr)

print(f"Values below {lower bound} could be outliers.")
```

```
print(f"Values below {lower_bound} could be outliers.")
print(f"Values above {upper_bound} could be outliers.")

# Merge this group df with the original DataFrame to get the tumor v
Ramicane_last = Ramicane_df.groupby('Mouse ID').max()['Timepoint']
Ramicane_vol = pd.DataFrame(Ramicane_last)
Ramicane_merge = pd.merge(Ramicane_vol, clean_df, on=("Mouse ID","Ti
Ramicane_merge.head()
Ramicane_merge.to_csv("output.csv")
Ramicane_tumors = Ramicane_merge["Tumor Volume (mm3)"]

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


print(f"The lower quartile of Ramicane tumors is: {lowerq}")
print(f"The upper quartile of Ramicane tumors is: {upperq}")
print(f"The interquartile range of Ramicane tumors is: {iqr}")
print(f"The median of Ramicane tumors is: {quartiles[0.5]} ")

lower_bound = lowerq - (1.5*iqr)
upper_bound = upperq + (1.5*iqr)

print(f"Values below {lower_bound} could be outliers.")
print(f"Values above {upper_bound} could be outliers.")
```

```
The lower quartile of Capomulin tumors: 32.37735684
The upper quartile of Capomulin tumors: 40.1592203
```

```
The lower quartile of Capomulin tumors: 32.37735684
The upper quartile of Capomulin tumors: 40.1592203
The interquartile range of Capomulin tumors: 7.781863460000004
The median of Capomulin tumors: 38.1251644
Values below 20.70456164999999 could be outliers.
Values above 51.83201549 could be outliers.
The lower quartile of Ramicane tumors is: 31.56046955
The upper quartile of Ramicane tumors is: 40.65900627
The interquartile range of Ramicane tumors is: 9.098536719999998
The median of Ramicane tumors is: 36.56165229
Values below 17.912664470000003 could be outliers.
Values above 54.30681135 could be outliers.
```

```python
Clean_last = clean_df.groupby('Mouse ID').max()['Timepoint']
Clean_vol = pd.DataFrame(Clean_last)
Clean_merge = pd.merge(Clean_vol, clean_df, on=("Mouse ID","Timepoir
Clean_merge.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 248 entries, 0 to 247
Data columns (total 8 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Mouse ID            248 non-null    object 
 1   Timepoint           248 non-null    int64  
 2   Tumor Volume (mm3)  248 non-null    float64
 3   Metastatic Sites    248 non-null    int64  
 4   Drug Regimen        248 non-null    object 
 5   Sex                 248 non-null    object 
 6   Age_months          248 non-null    int64  
```

In [33]:

```
# Put treatments into a list for for loop (and later for plot labels
drugs = ["Capomulin", "Ramicane", "Infubinol", "Ceftamin"]

# Create empty list to fill with tumor vol data (for plotting)
tumor_lists = []

# Calculate the IQR and quantitatively determine if there are any po
for drug in drugs:

    # Locate the rows which contain mice on each drug and get the tu
    sub = Clean_merge.loc[Clean_merge["Drug Regimen"] == drug]

    # add subset
    tumor = sub["Tumor Volume (mm3)"]
    tumor_lists.append(tumor)

    # Determine outliers using upper and lower bounds
    quartiles = tumor.quantile([0.25, 0.75])
    lowerq = quartiles[0.25]
    upperq = quartiles[0.75]
    iqr = upperq-lowerq

    lower_bound = lowerq - (1.5*iqr)
    upper_bound = upperq + (1.5*iqr)

    outliers = tumor.loc[(tumor < lower_bound) | (tumor > upper_boun
    print (drug)
```

Alt+Q

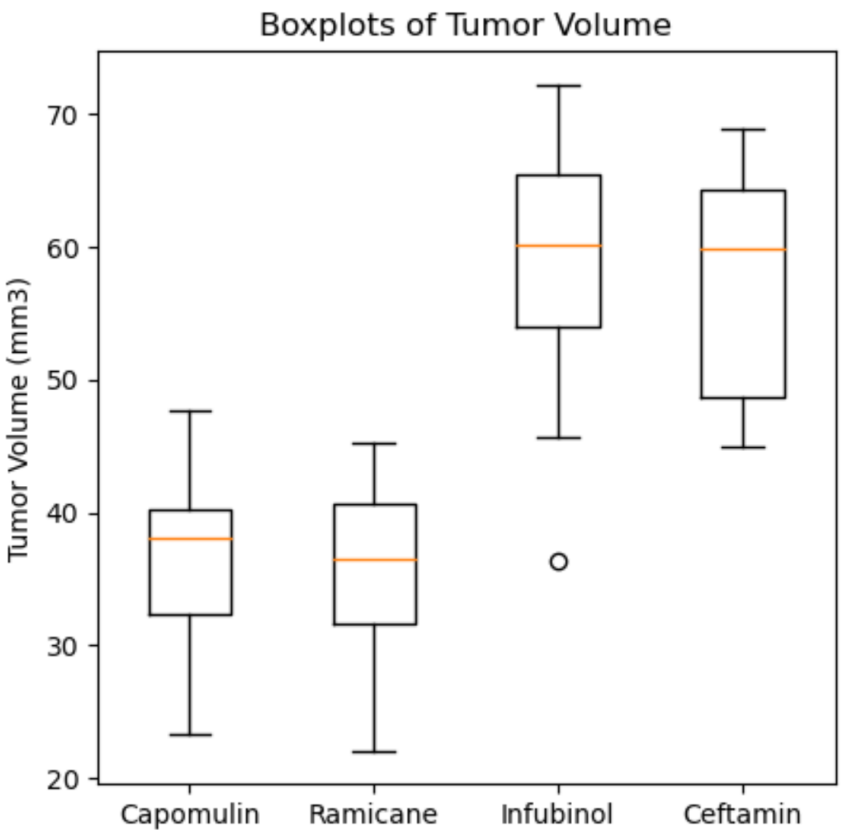
```
    outliers = tumor.loc[(tumor < lower_bound) | (tumor > upper_bour
    print (drug)
    print (outliers)
    print (lowerq)
    print (upperq)
    print (iqr)
```

```
Capomulin
Series([], Name: Tumor Volume (mm3), dtype: float64)
32.37735684
40.1592203
7.781863460000004
Ramicane
Series([], Name: Tumor Volume (mm3), dtype: float64)
31.56046955
40.65900627
9.098536719999998
Infubinol
31    36.321346
Name: Tumor Volume (mm3), dtype: float64
54.04860769
65.52574285
11.477135160000003
Ceftamin
Series([], Name: Tumor Volume (mm3), dtype: float64)
48.72207785
64.29983003
15.577752179999997
```

t+Q

In [34]:

```
# Generate a box plot that shows the distrubution of the tumor volu

plt.figure(figsize = (5, 5))
plt.boxplot(tumor_lists, labels = drugs)
plt.ylabel("Tumor Volume (mm3)")
plt.title("Boxplots of Tumor Volume")
plt.show()
```

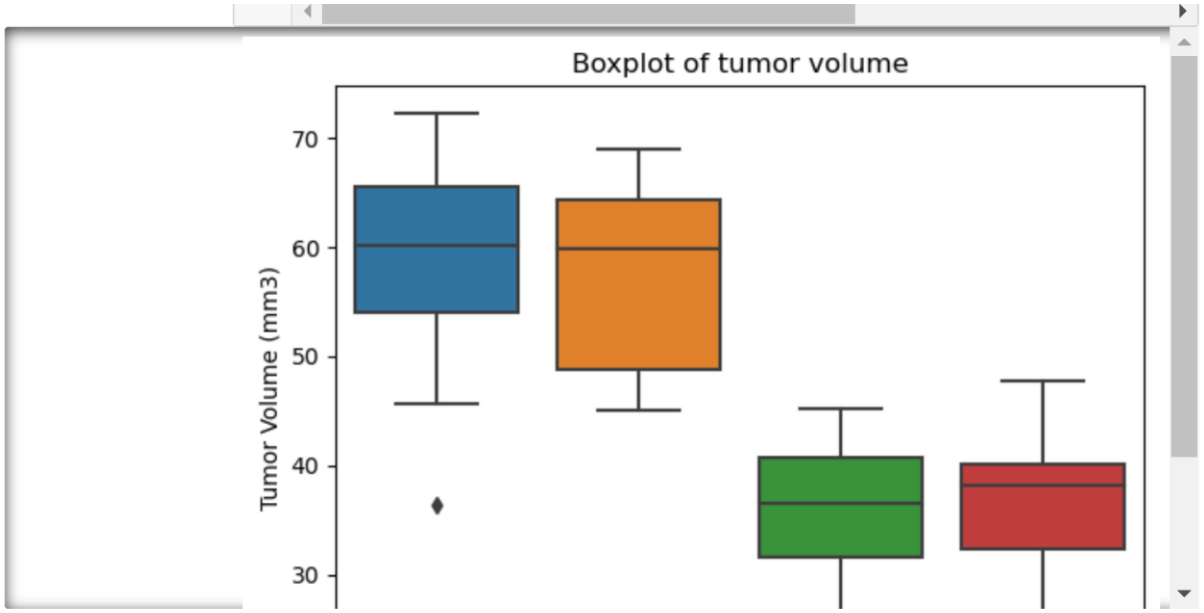


In [35]:

```
1 import seaborn as sns
```

In [36]:

```
1 df4 = Clean_merge.loc[Clean_merge["Drug Regimen"].isin(["Capomulin",
2 sns.boxplot(df4, x = "Drug Regimen", y = "Tumor Volume (mm3)")
3 plt.title("Boxplot of tumor volume")
4 plt.show()
```



Line and Scatter Plots

In [44]:
```
1 df4.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 100 entries, 0 to 245
Data columns (total 8 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Mouse ID            100 non-null    object
 1   Timepoint           100 non-null    int64
 2   Tumor Volume (mm3)  100 non-null    float64
 3   Metastatic Sites    100 non-null    int64
 4   Drug Regimen        100 non-null    object
 5   Sex                 100 non-null    object
 6   Age_months          100 non-null    int64
 7   Weight (g)          100 non-null    int64
dtypes: float64(1), int64(4), object(3)
memory usage: 7.0+ KB
```

In [45]:
```
1 df4
```

Out[45]:

	Mouse ID	Timepoint	Tumor Volume (mm3)	Metastatic Sites	Drug Regimen	Sex	Age_months	Weight (g)
0	a203	45	67.973419	2	Infubinol	Female	20	23

In [42]:

```
# Generate a line plot of tumor volume vs. time point for a single m
#df4.loc[df4["Drug Regimen"] == "Capomulin"]
mouse = "l509"

sub_mouse = df4.loc[df4["Mouse ID"] == mouse]

plt.figure(figsize=(8,6))
plt.plot(sub_mouse.Timepoint, sub_mouse["Tumor Volume (mm3)"])
plt.xlabel("Days")
plt.ylabel("Tumor Size (mm3)")
plt.title(f"Tumor Size over Time l509")
plt. show()
```