Pymaceuticals Inc.

Analysis

- Add your analysis here.

In [60]:
```
# 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
study_data_complete = pd.merge(study_results,mouse_metadata, how="left", on=["Mouse ID
# Display the data table for preview
study_data_complete.head()
```

Out[60]:

	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 [61]:
```
mouse_metadata.head()
```

Out[61]:

	Mouse ID	Drug Regimen	Sex	Age_months	Weight (g)
0	k403	Ramicane	Male	21	16
1	s185	Capomulin	Female	3	17
2	x401	Capomulin	Female	16	15
3	m601	Capomulin	Male	22	17
4	g791	Ramicane	Male	11	16

In [62]:
```
study_results.head()
```

Out[62]:

	Mouse ID	Timepoint	Tumor Volume (mm3)	Metastatic Sites
0	b128	0	45.0	0
1	f932	0	45.0	0
2	g107	0	45.0	0
3	a457	0	45.0	0
4	c819	0	45.0	0

In [63]:
```
# Checking the number of mice.
# Checking the number of mice.
mice_count = len(study_data_complete["Mouse ID"].unique())
mice_count
```

Out[63]:
```
249
```

In [64]:
```
# 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 Timepoint.
duplicate_miceID = study_data_complete.loc[study_data_complete.duplicated(subset=["Mou
print(f"The Duplicate Mice ID is: {duplicate_miceID}")
```

```
The Duplicate Mice ID is: ['g989']
```

In [65]:
```
# Optional: Get all the data for the duplicate mouse ID.
duplicate_mice_df = study_data_complete.loc[study_data_complete["Mouse ID"]=="g989", :
duplicate_mice_df
```

Out[65]:

	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 [66]:
```
# Create a clean DataFrame by dropping the duplicate mouse by its ID.
clean_df=study_data_complete[study_data_complete["Mouse ID"].isin(duplicate_miceID)==F
clean_df.head()
```

Out[66]:

	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 [67]:
```
# Checking the number of mice in the clean DataFrame.
clean_df_mice_count = len(clean_df["Mouse ID"].unique())
clean_df_mice_count
```

Out[67]:
```
248
```

Summary Statistics

In [68]:
```
# Generate a summary statistics table of mean, median, variance, standard deviation, a

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

mean = clean_df['Tumor Volume (mm3)'].groupby(clean_df['Drug Regimen']).mean()
median = clean_df['Tumor Volume (mm3)'].groupby(clean_df['Drug Regimen']).median()
variance = clean_df['Tumor Volume (mm3)'].groupby(clean_df['Drug Regimen']).var()
standard_deviation = clean_df['Tumor Volume (mm3)'].groupby(clean_df['Drug Regimen']).
sem = clean_df['Tumor Volume (mm3)'].groupby(clean_df['Drug Regimen']).sem()

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

Out[68]:

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

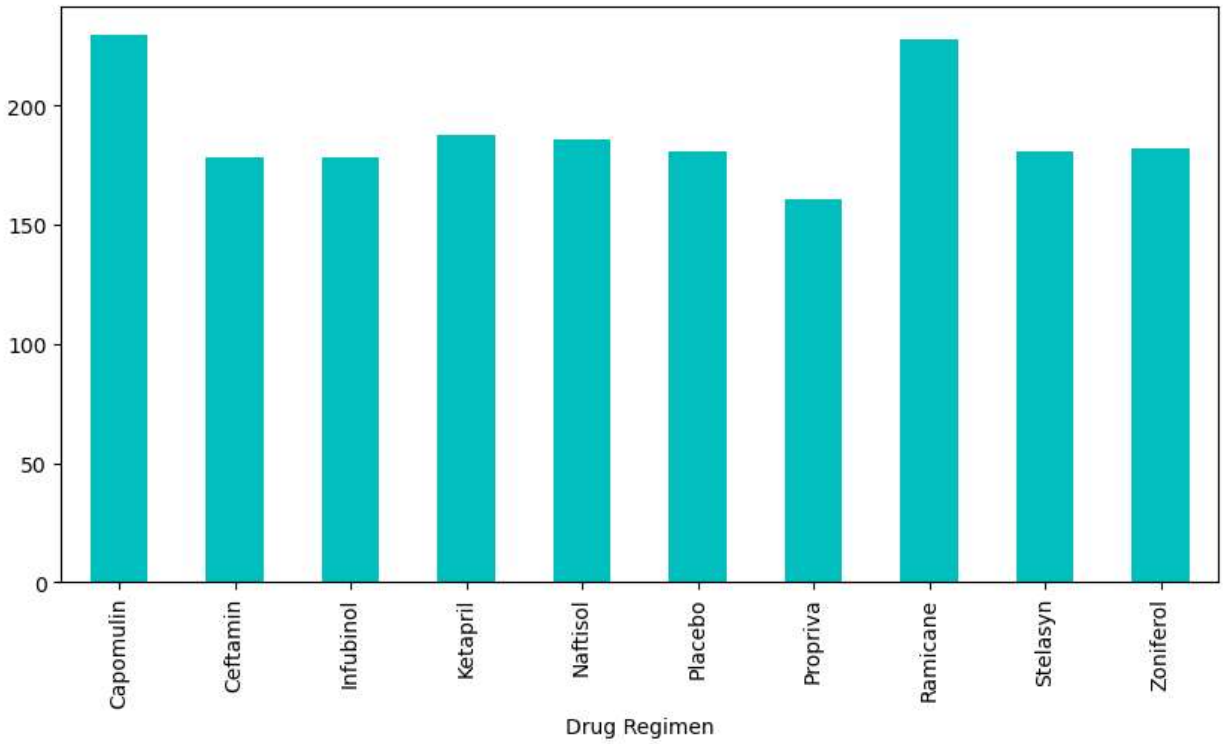
In [69]:
```
# A more advanced method to generate a summary statistics table of mean, median, varia
# and SEM of the tumor volume for each regimen (only one method is required in the sol

# Using the aggregation method, produce the same summary statistics in a single line
summary_table_agg =  clean_df.groupby(['Drug Regimen'])[['Tumor Volume (mm3)']].agg(['
summary_table_agg
```

Out[69]:

	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 [72]:
```
# Generate a bar plot showing the total number of rows (Mouse ID/Timepoints) for each
count_mice_per_drug=study_data_complete.groupby(["Drug Regimen"]).count()["Timepoint"]

plot_pandas=count_mice_per_drug.plot.bar(figsize=(10,5),color='c')
count_mice_per_drug

#Set X-label, Y-Label and Title
plt.xlabel("Drug Regimen")
#plt.ylabel("# of observed Mouse Timepoints")
#plt.title("Total count of Mice per Drug Treatment")

#show plot
plt.show()
```



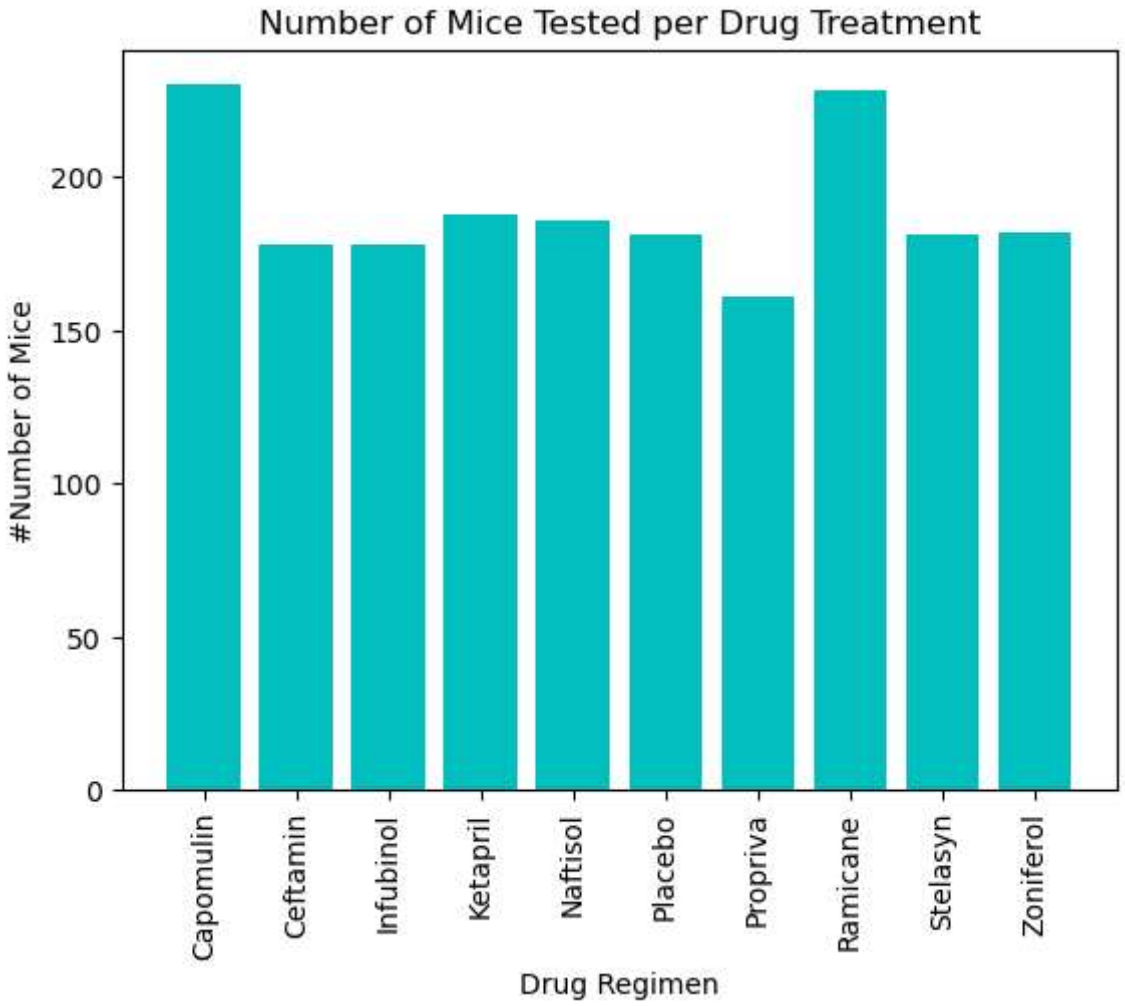
In [73]:
```
count_mice_per_drug.head
```

Out[73]:
```
<bound method NDFrame.head of Drug Regimen
Capomulin    230
Ceftamin     178
Infubinol    178
Ketapril     188
Naftisol     186
Placebo      181
Propriva     161
Ramicane     228
Stelasyn     181
Zoniferol    182
Name: Timepoint, dtype: int64>
```

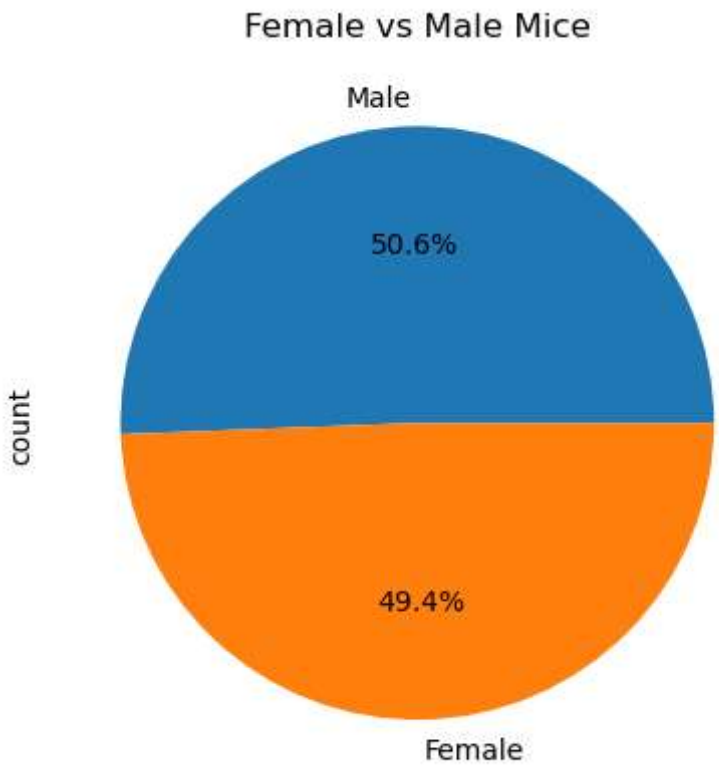
In [13]:
```
# Generate a bar plot showing the total number of rows (Mouse ID/Timepoints) for each

x=count_mice_per_drug.index.values
y=count_mice_per_drug.values

plt.bar(x,y, color="c")
```

```
#Set X-Label, Y-Label and Title
plt.title("Number of Mice Tested per Drug Treatment")
plt.xlabel("Drug Regimen")
plt.ylabel("#Number of Mice")
plt.xticks(rotation="vertical")
#show plot
plt.show
```

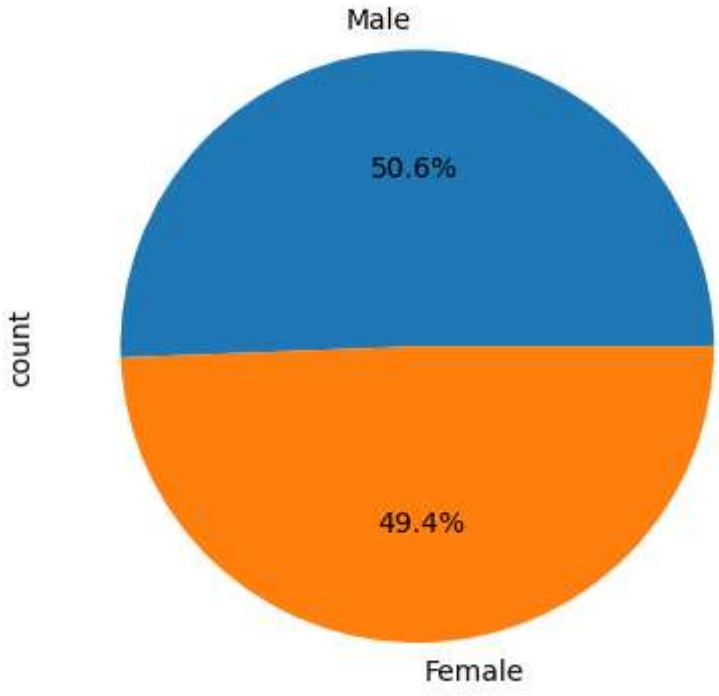
Out[13]: <function matplotlib.pyplot.show(close=None, block=None)>



In [16]:
```
# Generate a pie plot showing the distribution of female versus male mice using Pandas
gender_vals=study_data_complete["Sex"].value_counts()

gender_vals.plot(kind="pie",autopct="%1.1f%%")

plt.title("Female vs Male Mice")

plt.show()
```



In [18]:
```
# Generate a pie plot showing the distribution of female versus male mice using pyplot
labels=["Female","Male"]
sizes=[922,958]
plot=gender_vals.plot.pie(y='Total Count', autopct="%1.1f%%")
plt.title="Female vs Male Mice"
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


# Merge this group df with the original DataFrame to get the tumor volume at the last
```

In [21]:
```
study_data_complete.head()
```

Out[21]:

	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 [22]:
```
study_data_complete.groupby("Mouse ID").Timepoint.max()
```

Out[22]:
```
Mouse ID
a203    45
a251    45
a262    45
a275    45
a366    30
        ..
z435    10
z578    45
z581    45
z795    45
z969    45
Name: Timepoint, Length: 249, dtype: int64
```

In [24]:
```
max_time=study_data_complete.groupby("Mouse ID",as_index=False).Timepoint.max()
max_time.head()
```

Out[24]:

	Mouse ID	Timepoint
0	a203	45
1	a251	45
2	a262	45
3	a275	45
4	a366	30

In [26]:
```
updated_complete_df=pd.merge(max_time,study_data_complete,on=["Mouse ID","Timepoint"],
updated_complete_df.head()
```

Out[26]:

	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
1	a251	45	65.525743	1	Infubinol	Female	21	25
2	a262	45	70.717621	4	Placebo	Female	17	29
3	a275	45	62.999356	3	Ceftamin	Female	20	28
4	a366	30	63.440686	1	Stelasyn	Female	16	29

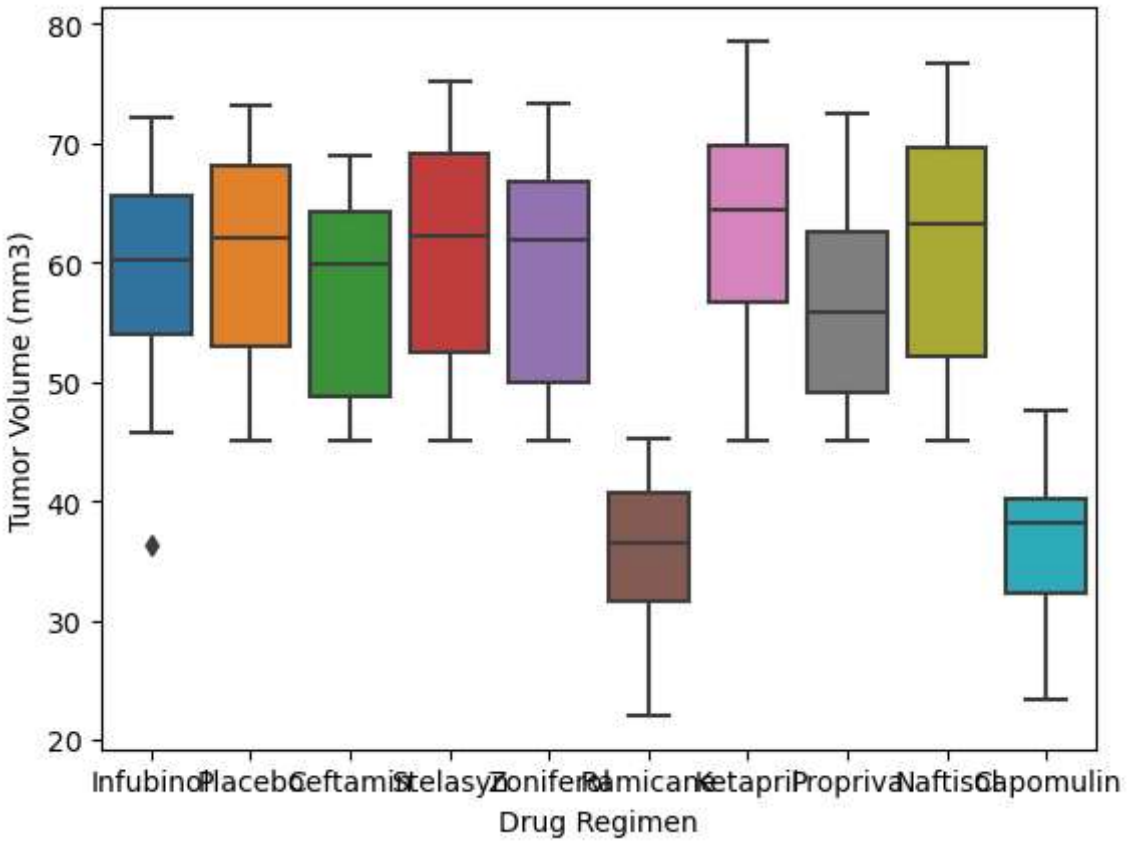
In [27]:
```
updated_complete_df["Mouse ID"].value_counts()
```

Out[27]:
```
Mouse ID
a203    1
s121    1
q597    1
q610    1
q633    1
       ..
j246    1
j296    1
j365    1
j755    1
z969    1
Name: count, Length: 249, dtype: int64
```

In [28]:
```
#use seaborn for box plot

import seaborn as sns
```

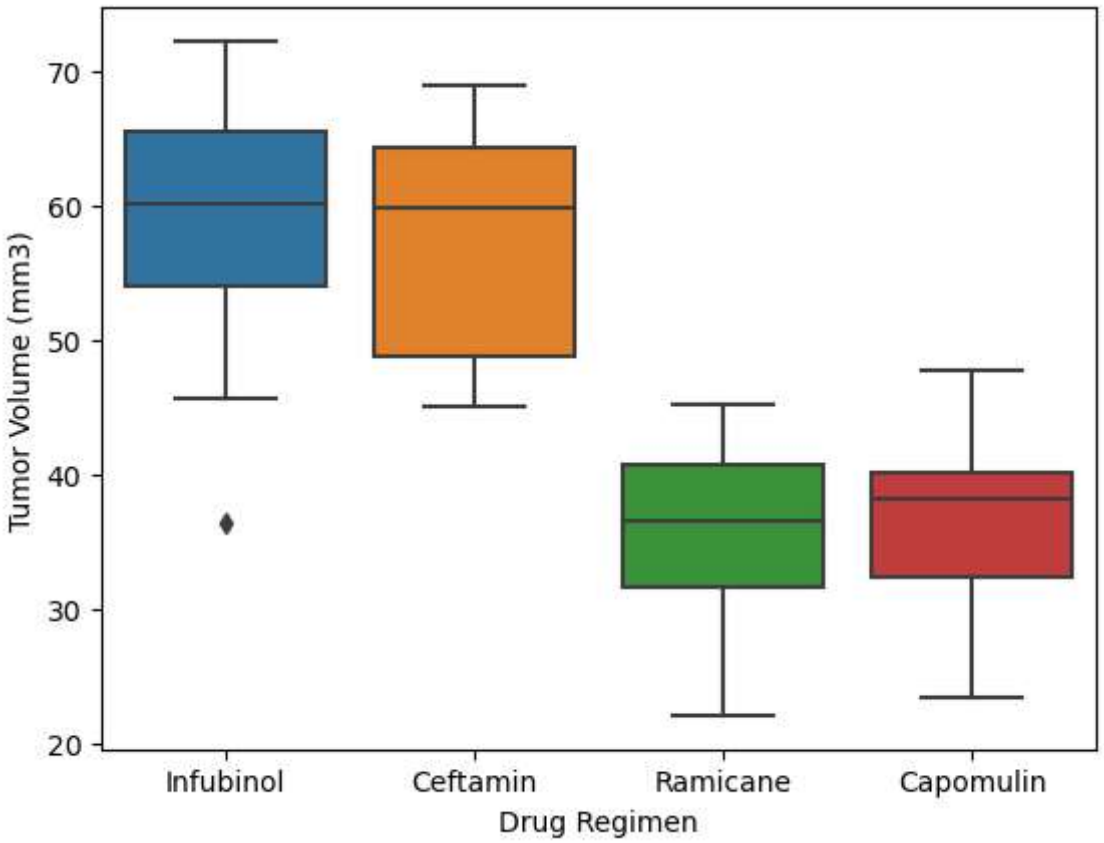
In [30]:
```
sns.boxplot(updated_complete_df,x="Drug Regimen",y="Tumor Volume (mm3)")
```

Out[30]:
```
<Axes: xlabel='Drug Regimen', ylabel='Tumor Volume (mm3)'>
```



In [32]:
```
filtered_updated_complete_df=updated_complete_df.loc[updated_complete_df["Drug Regimer

sns.boxplot(filtered_updated_complete_df,x="Drug Regimen",y="Tumor Volume (mm3)")
```

Out[32]:
```
<Axes: xlabel='Drug Regimen', ylabel='Tumor Volume (mm3)'>
```



In [ ]:
```
#generate data for each box plot and stitch them together
```

In [50]:
```
# 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_list=[]

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

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

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

    # Determine outliers using upper and lower bounds
    quartiles=tumor.quantile([0.25,0.75])
    q1=quartiles[0.25]
    q3=quartiles[0.75]
    iqr=q3-q1

    lower_bound=q1-1.5*iqr
    upper_bound=q3+1.5*iqr

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

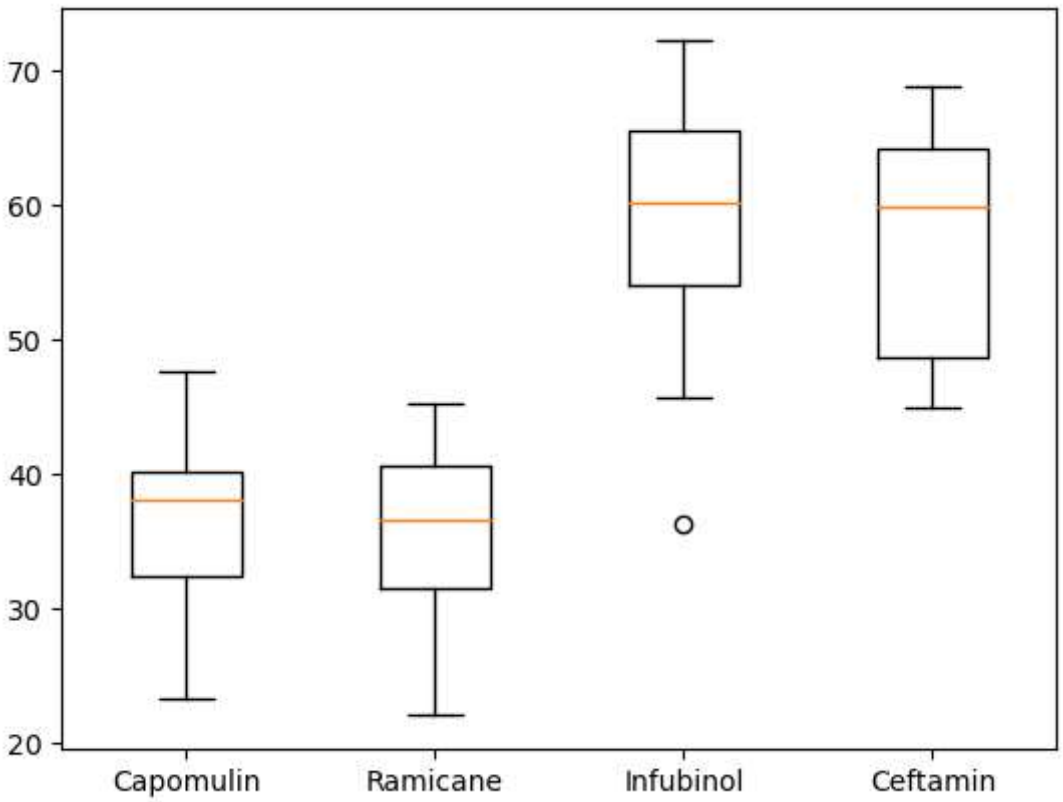
```
Capomulin
Series([], Name: Tumor Volume (mm3), dtype: float64)
Ramicane
Series([], Name: Tumor Volume (mm3), dtype: float64)
Infubinol
31    36.321346
Name: Tumor Volume (mm3), dtype: float64
Ceftamin
Series([], Name: Tumor Volume (mm3), dtype: float64)
```

In [48]:
```
 # Determine outliers using upper and lower bounds
quartiles=tumor.quantile([0.25,0.75])
q1=quartiles[0.25]
q3=quartiles[0.75]
iqr=q3-q1

lower_bound=q1-1.5*iqr
upper_bound=q3+1.5*iqr

outliers=tumor.loc[(tumor<lower_bound) | (tumor>upper_bound)]
outliers
```

Out[48]:
```
Series([], Name: Tumor Volume (mm3), dtype: float64)
```

In [41]:
```
# Generate a box plot that shows the distrubution of the tumor volume for each treatme
#plt.figure(figsize(12,9))
plt.boxplot(tumor_list,labels=drugs)
#plt.ylabel("Tumor Volume (mm3)")
plt.show
```

Out[41]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



Line and Scatter Plots

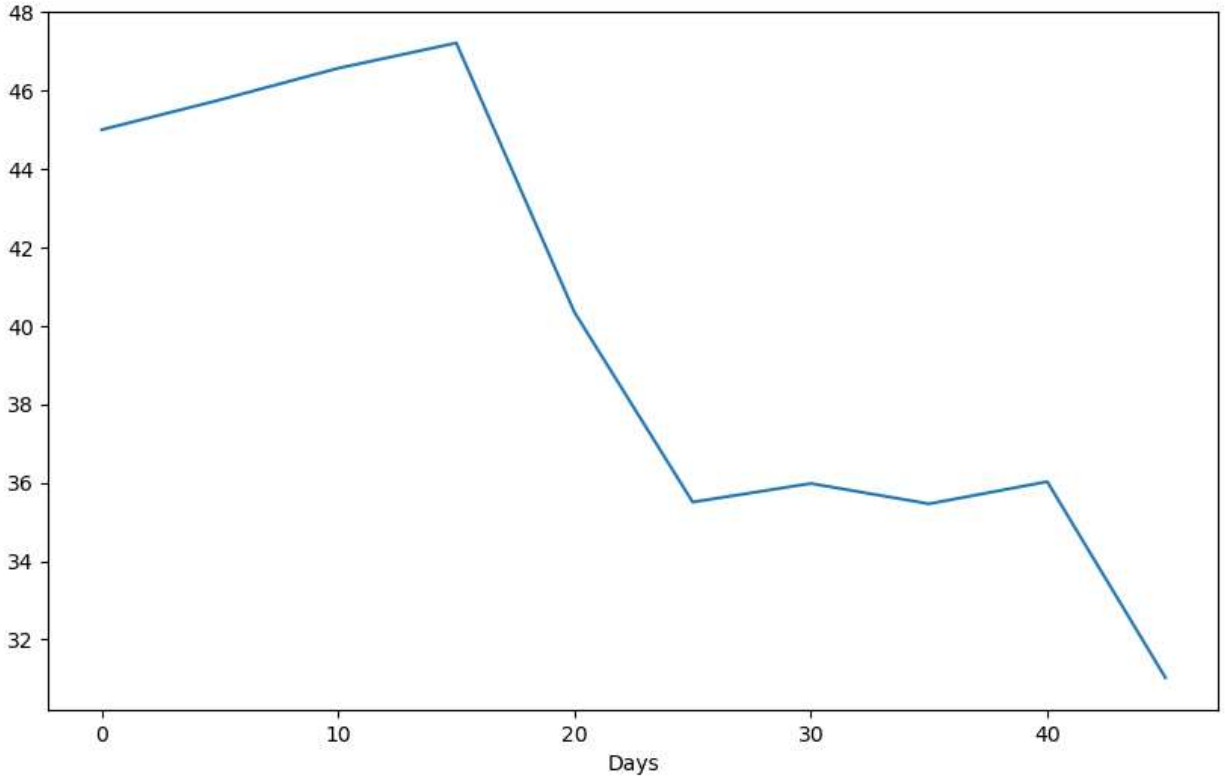
In [52]:
```
# Generate a line plot of tumor volume vs. time point for a single mouse treated with
study_data_complete.loc[study_data_complete["Drug Regimen"]=="Capomulin"]
```

Out[52]:

	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
226	j246	0	45.000000	0	Capomulin	Female	21	21
227	r554	0	45.000000	0	Capomulin	Female	8	17
228	s185	0	45.000000	0	Capomulin	Female	3	17
229	b742	0	45.000000	0	Capomulin	Male	7	21
...	...	...	...	...	...	...	...	...
1854	s710	45	40.728578	1	Capomulin	Female	1	23
1859	j119	45	38.125164	1	Capomulin	Female	7	23
1878	i557	45	47.685963	1	Capomulin	Female	1	24
1888	r944	45	41.581521	2	Capomulin	Male	12	25
1889	u364	45	31.023923	3	Capomulin	Male	18	17

230 rows × 8 columns

In [82]:
```
# Generate a line plot of tumor volume vs. time point for a single mouse treated with

mouse="u364"
sub_mouse=study_data_complete.loc[study_data_complete["Mouse ID"]==mouse]

plt.figure(figsize=(10,6))
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 for Mouse {mouse}")
plt.show()
```

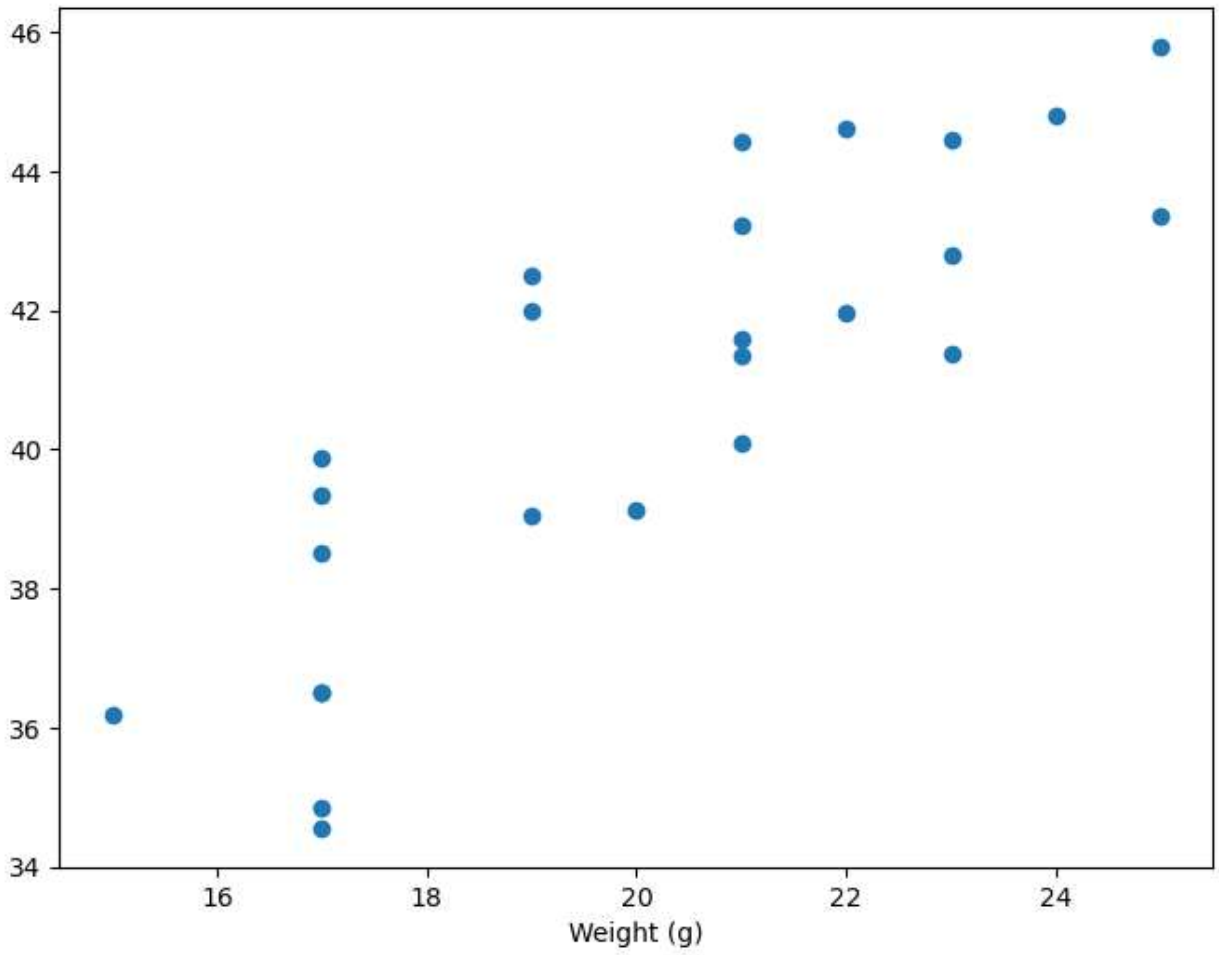


In [86]:
```
# Generate a scatter plot of mouse weight vs. the average observed tumor volume for th
scatter=study_data_complete.loc[study_data_complete["Drug Regimen"]=="Capomulin"]
scatter.groupby("Mouse ID",as_index=False).agg({"Weight (g)":"mean","Tumor Volume (mm3
```

Out[86]:

	Mouse ID	Weight (g)	Tumor Volume (mm3)
0	b128	22.0	41.963636
1	b742	21.0	40.083699
2	f966	17.0	36.505973
3	g288	19.0	41.990097
4	g316	22.0	44.613344
5	i557	24.0	44.805810
6	i738	20.0	39.141053
7	j119	23.0	44.465236
8	j246	21.0	43.216925
9	l509	21.0	44.434474
10	l897	19.0	42.507261
11	m601	17.0	34.847470
12	m957	19.0	39.049816
13	r157	25.0	45.798970
14	r554	17.0	36.509212
15	r944	25.0	43.367364
16	s185	17.0	34.559143
17	s710	23.0	42.803733
18	t565	17.0	39.328725
19	u364	17.0	39.887495
20	v923	21.0	41.581595
21	w150	23.0	41.384825
22	w914	21.0	41.352452
23	x401	15.0	36.182040
24	y793	17.0	38.506829

In [93]:
```
plt.figure(figsize=(8,6))
plt.scatter(scatter_data["Weight (g)"], scatter_data["Tumor Volume (mm3)"])
plt.xlabel("Weight (g)")
#plt.ylabel("Tumor Volume (mm3)")
#plt.title("weight g vs tumor size")
plt.show()
```

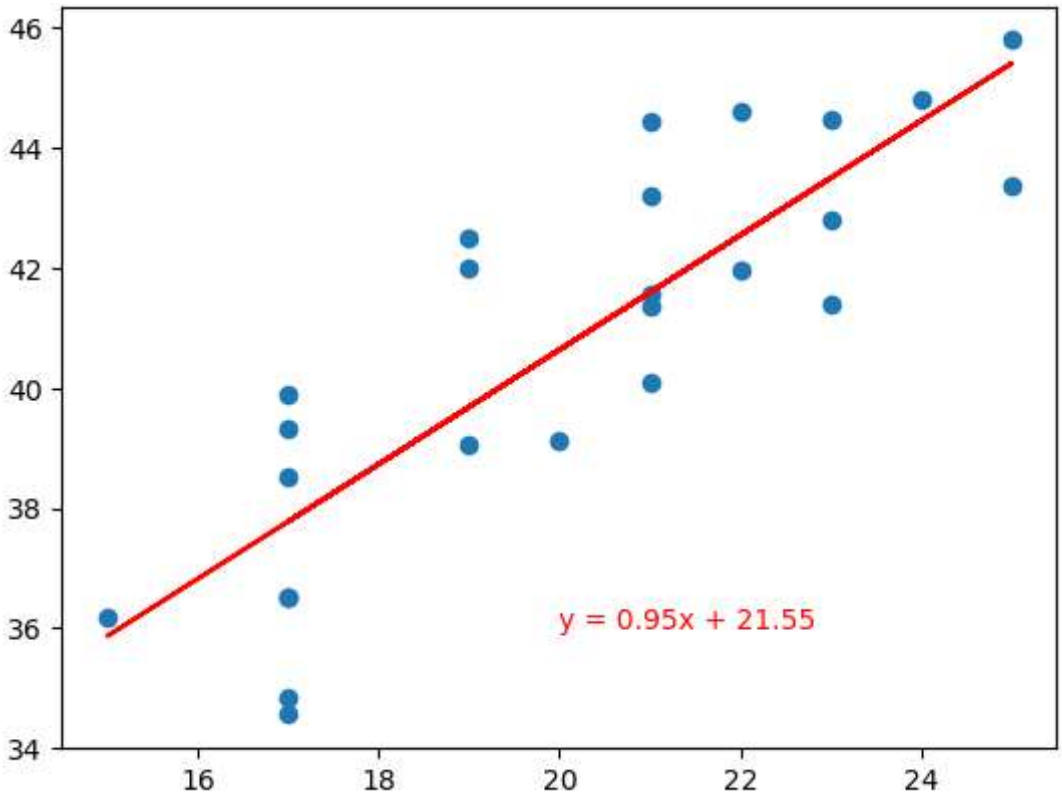


Correlation and Regression

In [96]:
```
from scipy.stats import linregress
```

In [105…
```
# Calculate the correlation coefficient and a linear regression model
# for mouse weight and average observed tumor volume for the entire Capomulin regimen


x_values=scatter_data["Weight (g)"]
y_values=scatter_data["Tumor Volume (mm3)"]

(slope,intercept,rvalue,pvalue,stderr)=linregress(x_values, y_values)
regress_values=x_values*slope+intercept
line_eq="y = " + str(round(slope,2)) + "x + " + str(round(intercept,2))

#Make Plot
plt.scatter(x_values,y_values)
plt.plot(x_values,regress_values,"r-")
plt.annotate(line_eq,(20,36),fontsize=10,color="red")

print(f"The r-squared is: {rvalue**2}")

plt.show()
```

```
The r-squared is: 0.7088568047708723
```



In [ ]: