Assignment 3 Taniya Hussain 21BKT0083

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Task 1 & 2: Download the Dataset and Load the dataset

```
df=pd.read_csv("penguins_size.csv")
df.head()
```

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mas
0	Adelie	Torgersen	39.1	18.7	181.0	375
1	Adelie	Torgersen	39.5	17.4	186.0	380
2	Adelie	Torgersen	40.3	18.0	195.0	325
3	Adelie	Torgersen	NaN	NaN	NaN	N
4	Adelie	Torgersen	36.7	19.3	193.0	345

```
df.shape
```

```
(344, 7)
```

Task 3: Perform the Below Visualizations

> ### 1. Univariate Analysis

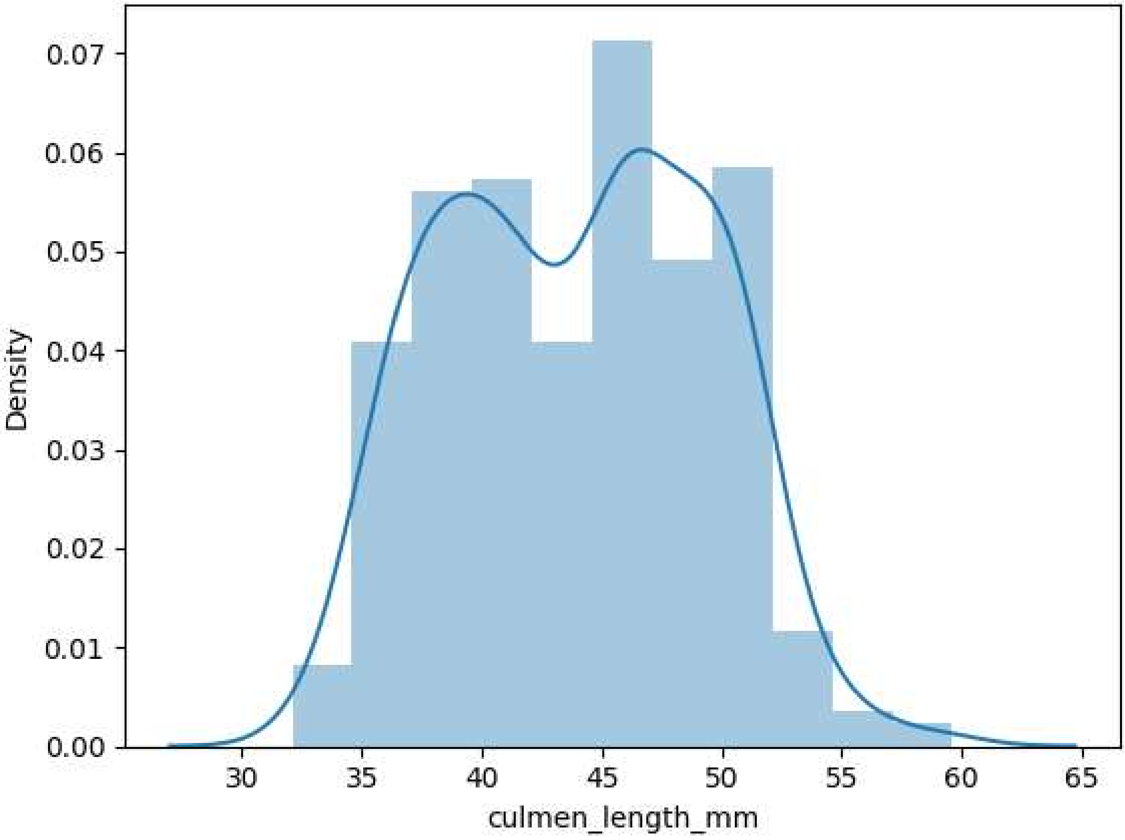
```
df.corr()
```

```
<ipython-input-5-2f6f6606aa2c>:1: FutureWarning: The default value of numeric_only in
  df.corr()
```

	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g
culmen_length_mm	1.000000	-0.235053	0.656181	0.595110
culmen_depth_mm	-0.235053	1.000000	-0.583851	-0.471916
flipper_length_mm	0.656181	-0.583851	1.000000	0.871202
body_mass_g	0.595110	-0.471916	0.871202	1.000000

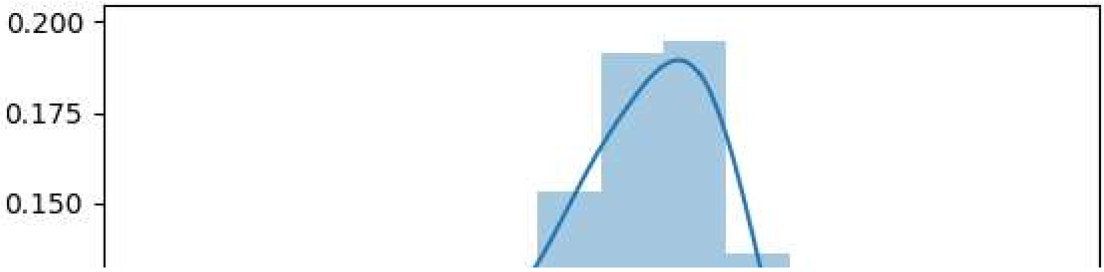
```
sns.distplot(df.culmen_length_mm)
sns.distplot(df.culmen_length_mm)
```

```
<ipython-input-6-24e9b5890c61>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.culmen_length_mm)
<Axes: xlabel='culmen_length_mm', ylabel='Density'>
```



```
sns.distplot(df.culmen_depth_mm)
```

```
<ipython-input-7-4b07ffb4fe44>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df.culmen_depth_mm)
<Axes: xlabel='culmen_depth_mm', ylabel='Density'>
```

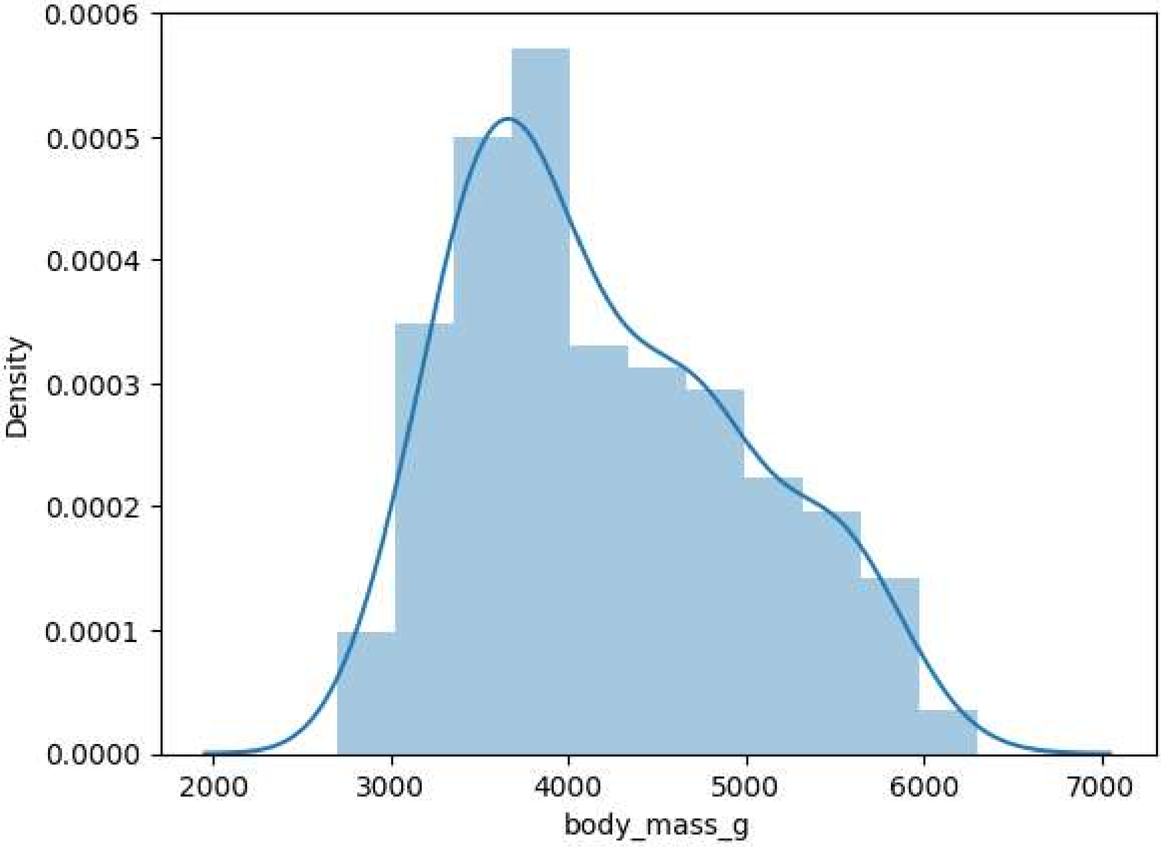


```
sns.distplot(df.flipper_length_mm)
```

```
    <ipython-input-10-4c42e92ff055>:1: UserWarning:
```

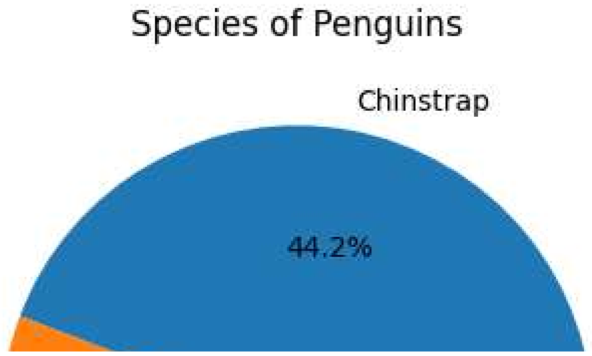
```
sns.distplot(df.body_mass_g)
```

```
    <ipython-input-11-176964dae727>:1: UserWarning:

    `distplot` is a deprecated function and will be removed in seaborn v0.14.0.

    Please adapt your code to use either `displot` (a figure-level function with
    similar flexibility) or `histplot` (an axes-level function for histograms).

    For a guide to updating your code to use the new functions, please see
    https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

      sns.distplot(df.body_mass_g)
    <Axes: xlabel='body_mass_g', ylabel='Density'>
```



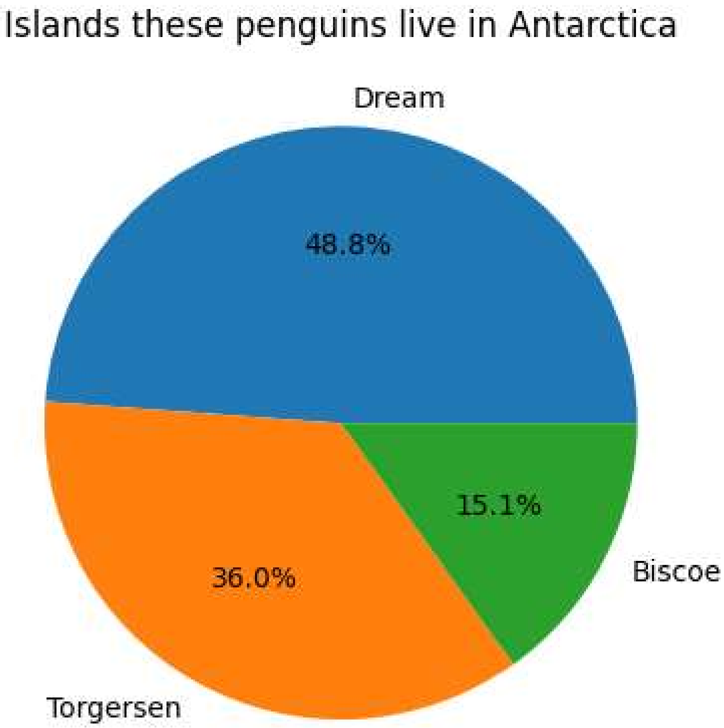
```
plt.pie(df.species.value_counts(),labels=['Chinstrap','Adelie','Gentoo'],autopct='%1.1f%%'
plt.title("Species of Penguins")
```

```
Text(0.5, 1.0, 'Species of Penguins')
```



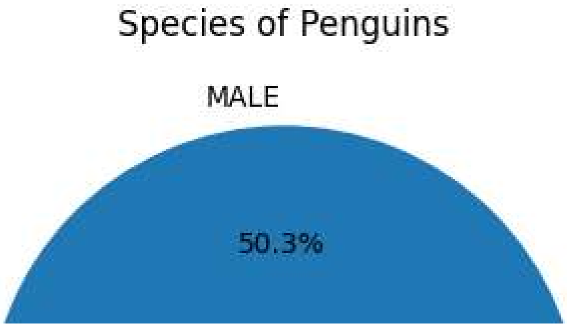
```
plt.pie(df.island.value_counts(),labels=['Dream','Torgersen','Biscoe'],autopct='%1.1f%%')
plt.title("Islands these penguins live in Antarctica")
```

```
Text(0.5, 1.0, 'Islands these penguins live in Antarctica')
```



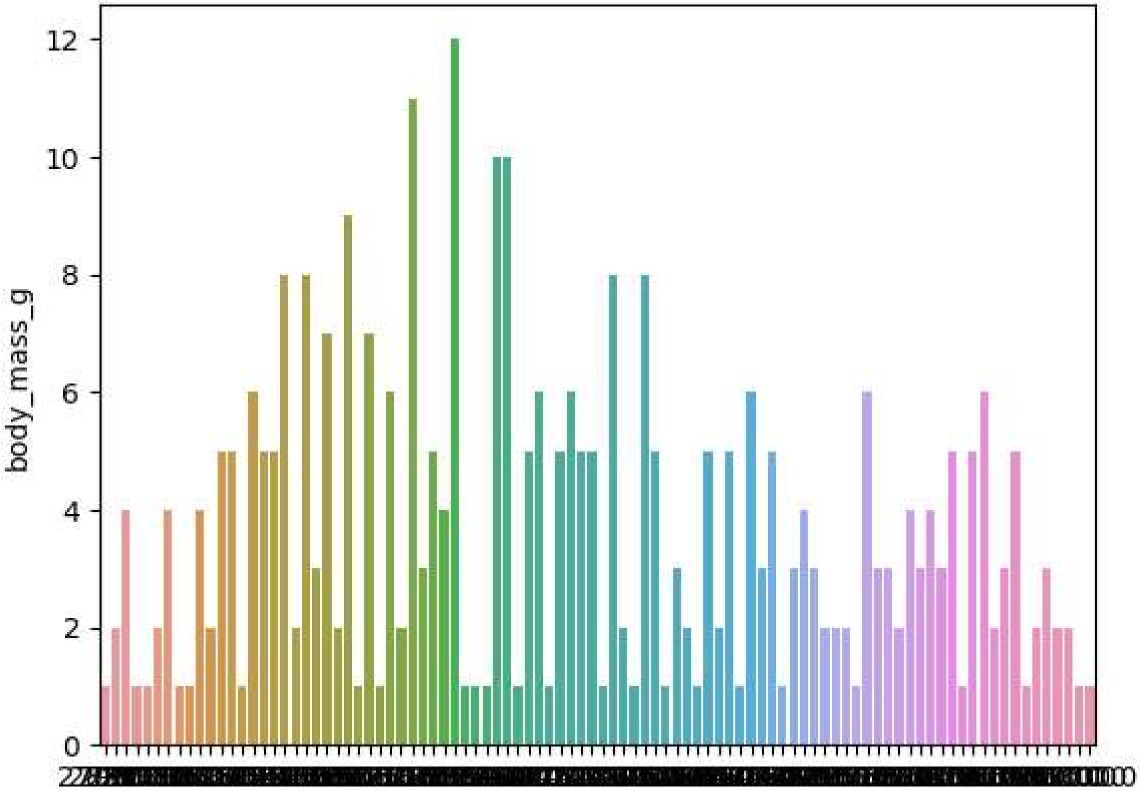
```
plt.pie(df.sex.value_counts(),labels=['MALE','FEMALE','NAN'],autopct='%1.1f%%')
plt.title("Species of Penguins")
```

```
Text(0.5, 1.0, 'Species of Penguins')
```



```
sns.barplot(x=df.body_mass_g.value_counts().index,y=df.body_mass_g.value_counts())
```

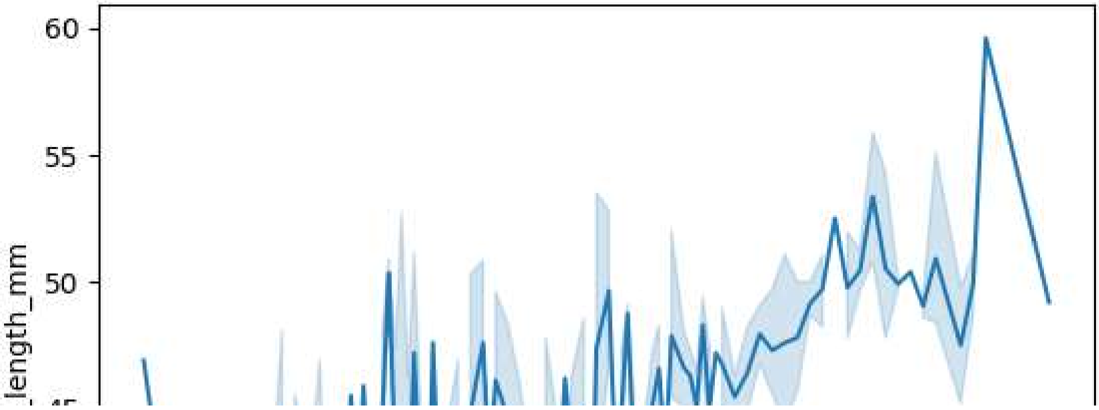
```
<Axes: ylabel='body_mass_g'>
```



2. Bivariate Analysis

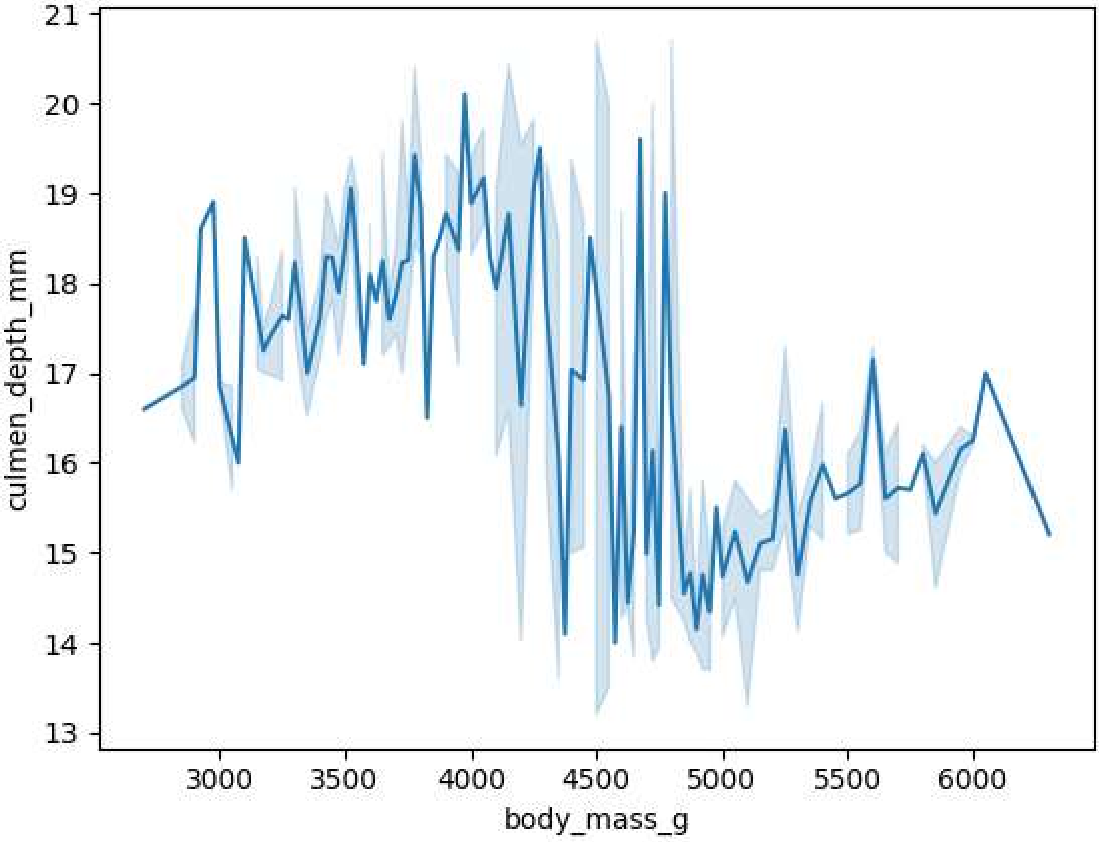
```
sns.lineplot(x=df.body_mass_g,y=df.culmen_length_mm)
```

```
<Axes: xlabel='body_mass_g', ylabel='culmen_length_mm'>
```



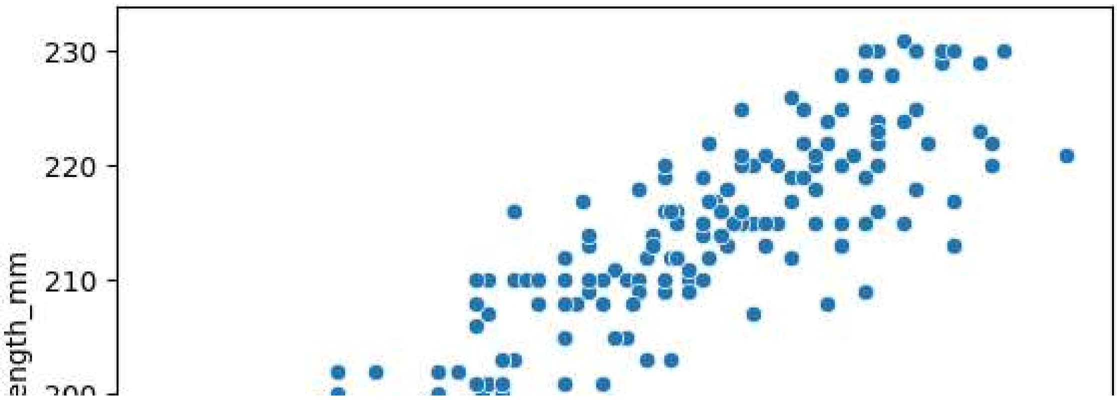
```
sns.lineplot(x=df.body_mass_g,y=df.culmen_depth_mm)
```

```
<Axes: xlabel='body_mass_g', ylabel='culmen_depth_mm'>
```



```
sns.scatterplot(x=df.body_mass_g,y=df.flipper_length_mm)
```

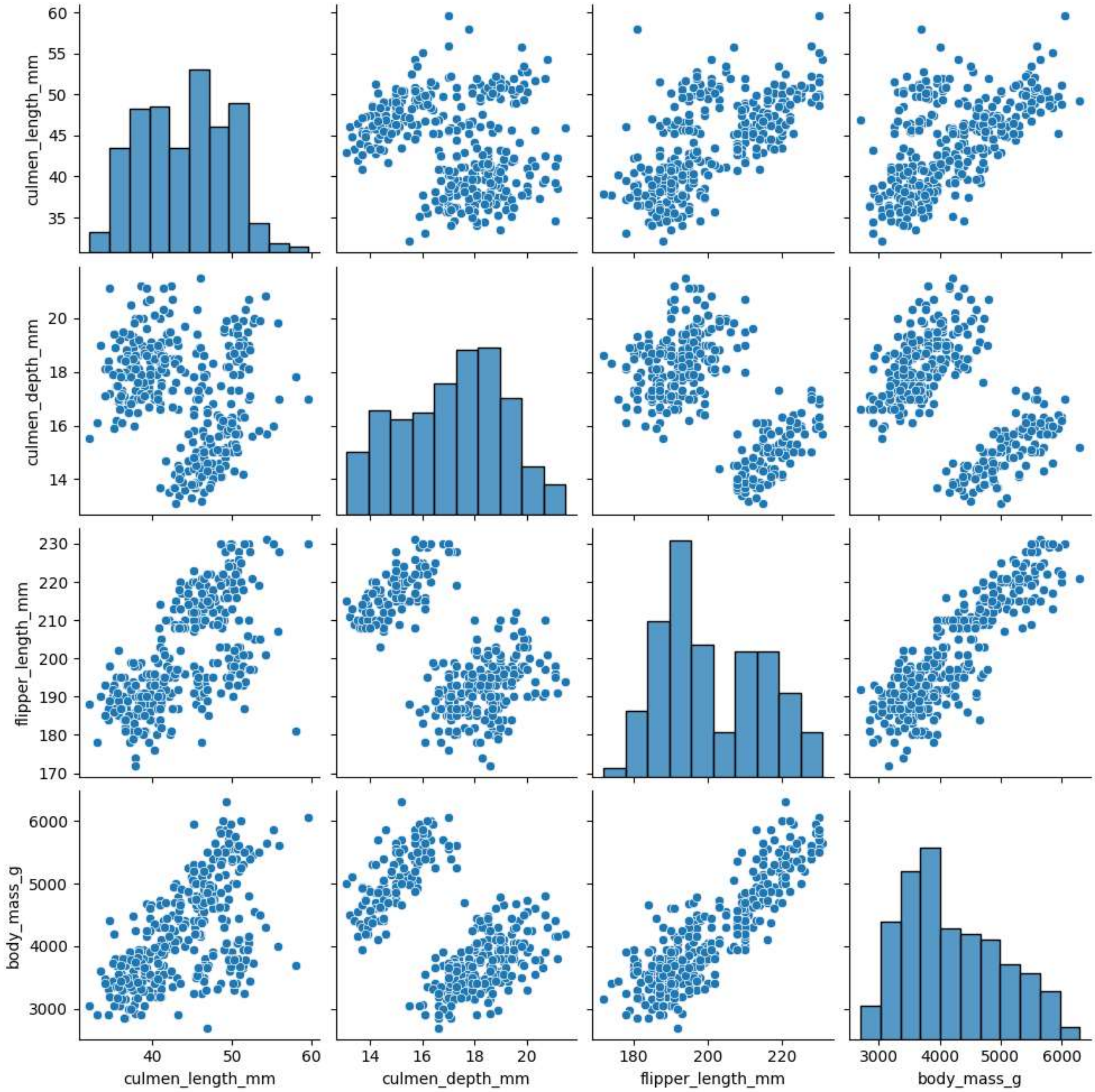
```
<Axes: xlabel='body_mass_g', ylabel='flipper_length_mm'>
```



3. Multivariate Analysis

```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x79ffbc5d6080>
```



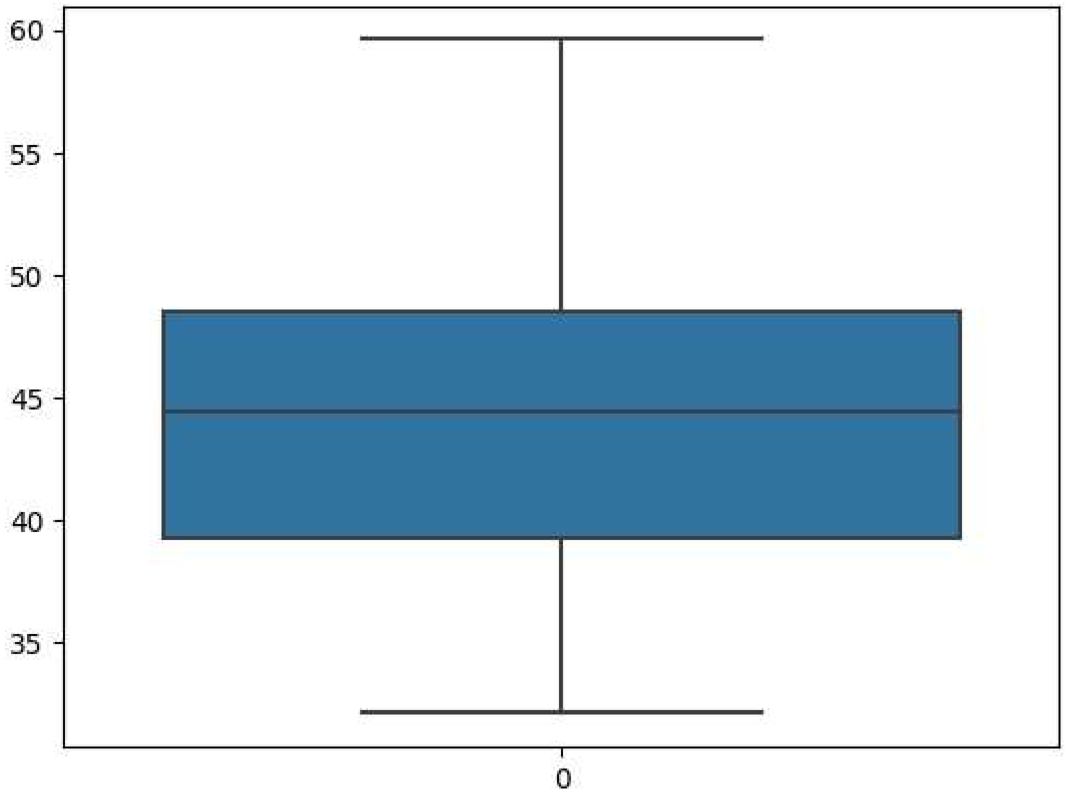
Task 4: Perform descriptive statistics on the dataset.

```
df.describe()
```

	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g
count	342.000000	342.000000	342.000000	342.000000
mean	43.921930	17.151170	200.915205	4201.754386
std	5.459584	1.974793	14.061714	801.954536
min	32.100000	13.100000	172.000000	2700.000000
25%	39.225000	15.600000	190.000000	3550.000000
50%	44.450000	17.300000	197.000000	4050.000000
75%	48.500000	18.700000	213.000000	4750.000000
max	59.600000	21.500000	231.000000	6300.000000

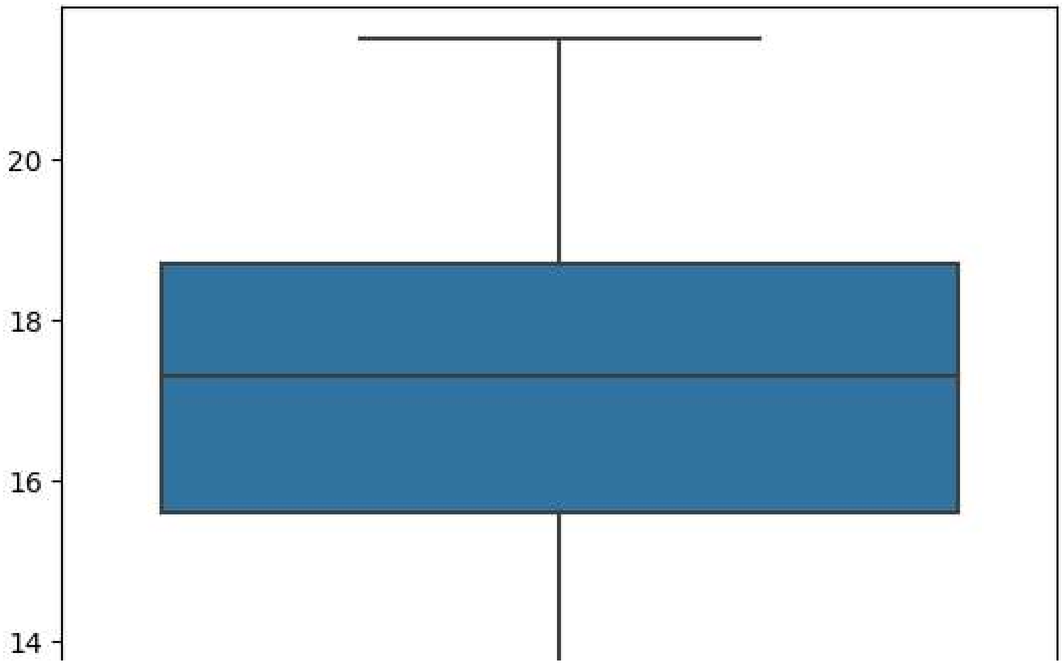
```
sns.boxplot(df.culmen_length_mm)
```

```
<Axes: >
```



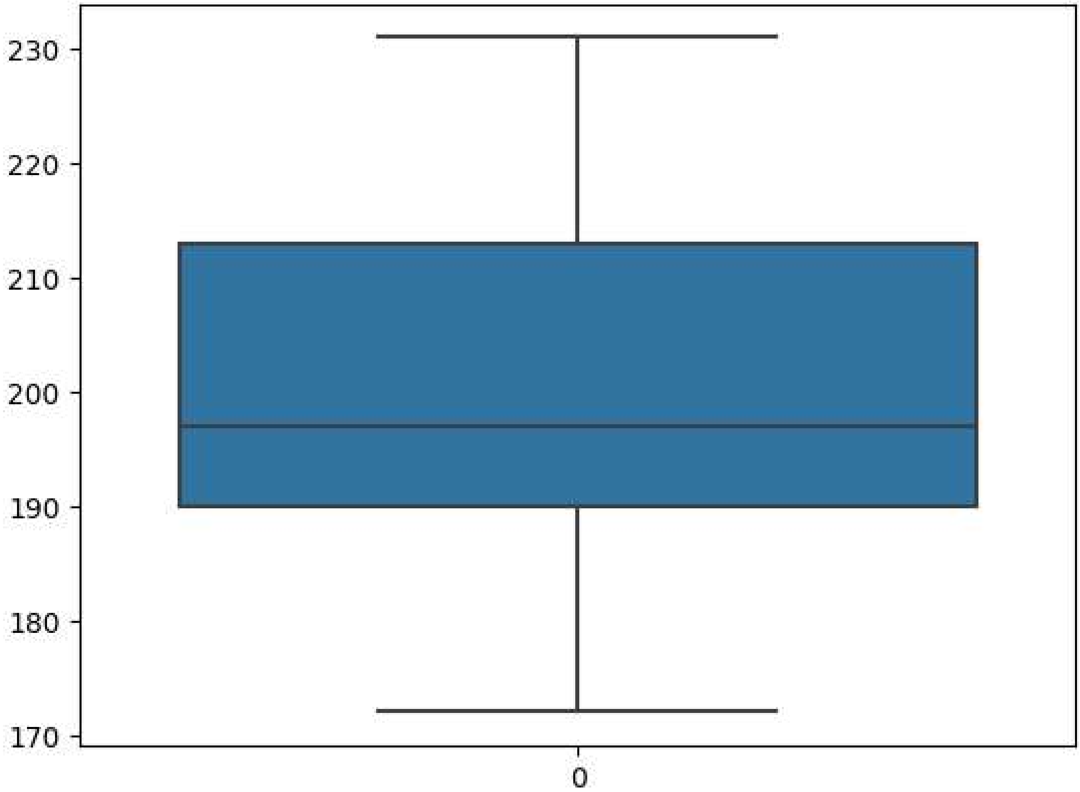
```
sns.boxplot(df.culmen_depth_mm)
```

```
<Axes: >
```



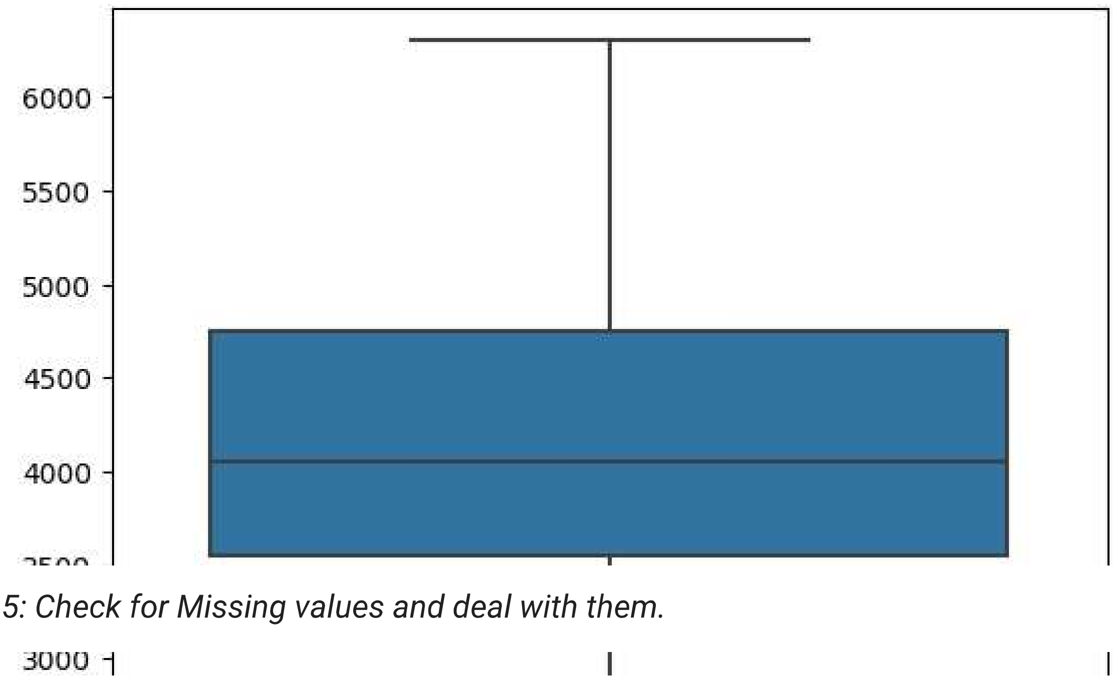
```
sns.boxplot(df.flipper_length_mm)
```

```
<Axes: >
```



```
sns.boxplot(df.body_mass_g)
```

```
<Axes: >
```



Task 5: Check for Missing values and deal with them.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 344 entries, 0 to 343
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   species            344 non-null    object 
 1   island             344 non-null    object 
 2   culmen_length_mm   342 non-null    float64
 3   culmen_depth_mm    342 non-null    float64
 4   flipper_length_mm  342 non-null    float64
 5   body_mass_g        342 non-null    float64
 6   sex                334 non-null    object 
dtypes: float64(4), object(3)
memory usage: 18.9+ KB
```

```
df.isnull().any()
```

```
species              False
island               False
culmen_length_mm      True
culmen_depth_mm       True
flipper_length_mm     True
body_mass_g           True
sex                   True
dtype: bool
```

```
df.isnull().sum()
```

```
species               0
island                0
culmen_length_mm      2
culmen_depth_mm       2
flipper_length_mm     2
body_mass_g           2
```

```
sex                    10
dtype: int64
```

```
df['culmen_length_mm'].fillna(df['culmen_length_mm'].median(),inplace=True)
```

```
df['culmen_depth_mm'].fillna(df['culmen_depth_mm'].median(),inplace=True)
```

```
df['flipper_length_mm'].fillna(df['flipper_length_mm'].median(),inplace=True)
```

```
df['body_mass_g'].fillna(df['body_mass_g'].median(),inplace=True)
```

```
df['sex'].fillna(df['sex'].mode(),inplace=True)
```

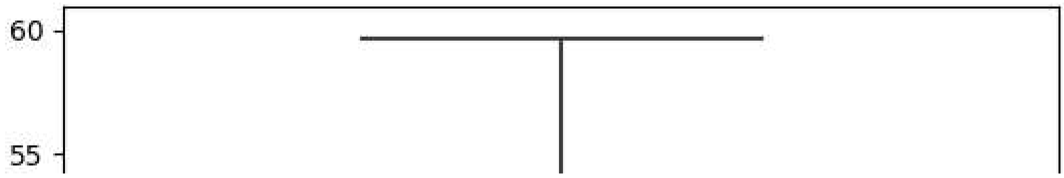
Task 6: Find the outliers and replace them outliers

```
df.head()
```

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass
0	Adelie	Torgersen	39.10	18.7	181.0	375
1	Adelie	Torgersen	39.50	17.4	186.0	380
2	Adelie	Torgersen	40.30	18.0	195.0	325
3	Adelie	Torgersen	44.45	17.3	197.0	405
4	Adelie	Torgersen	36.70	19.3	193.0	345

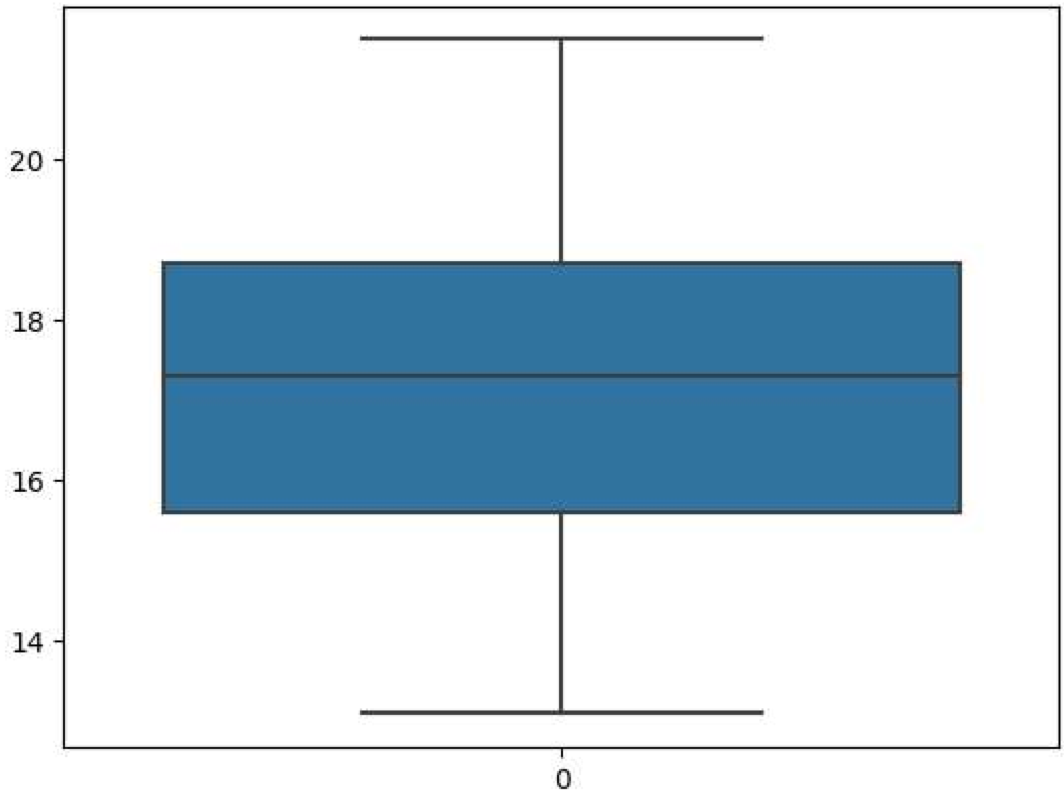
```
sns.boxplot(df.culmen_length_mm)
```

```
<Axes: >
```



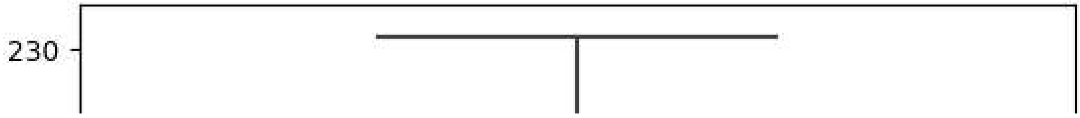
```
sns.boxplot(df.culmen_depth_mm)
```

```
<Axes: >
```



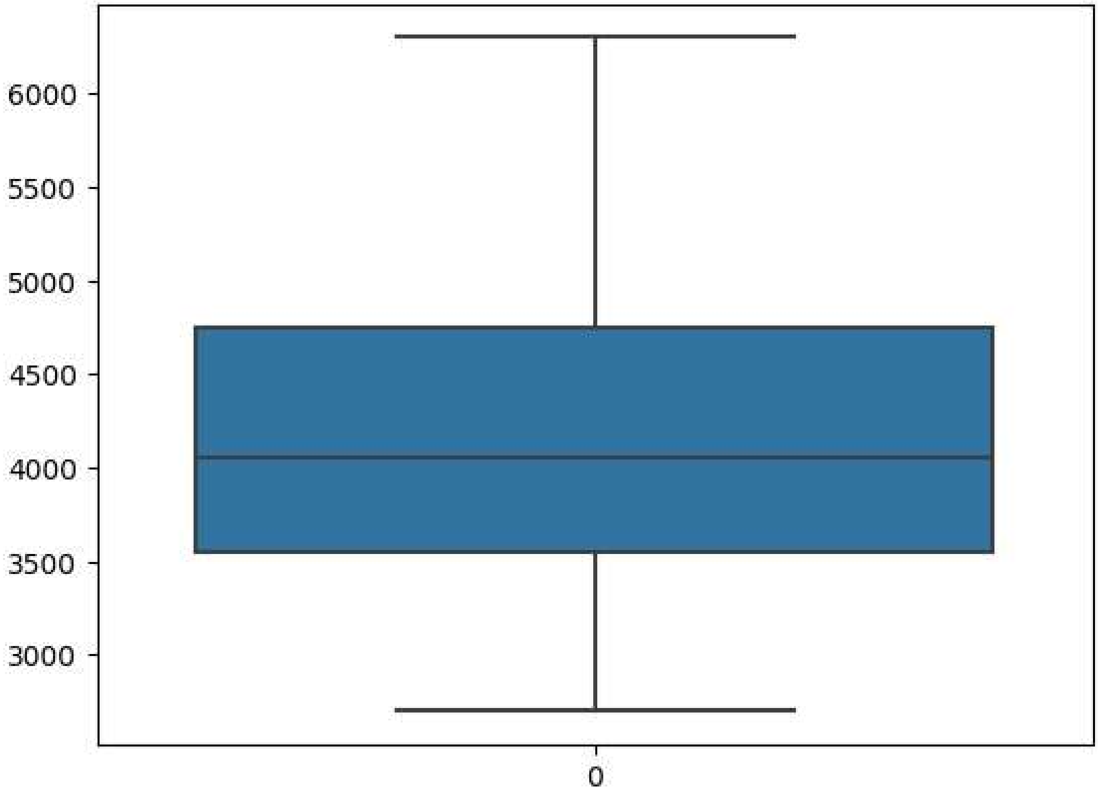
```
sns.boxplot(df.flipper_length_mm)
```

```
<Axes: >
```



```
sns.boxplot(df.body_mass_g)
```

```
<Axes: >
```



There are no outliners as we can see from the boxplot, hence we dont have to replace

Task 7 : Check the correlation of independent variables with the target

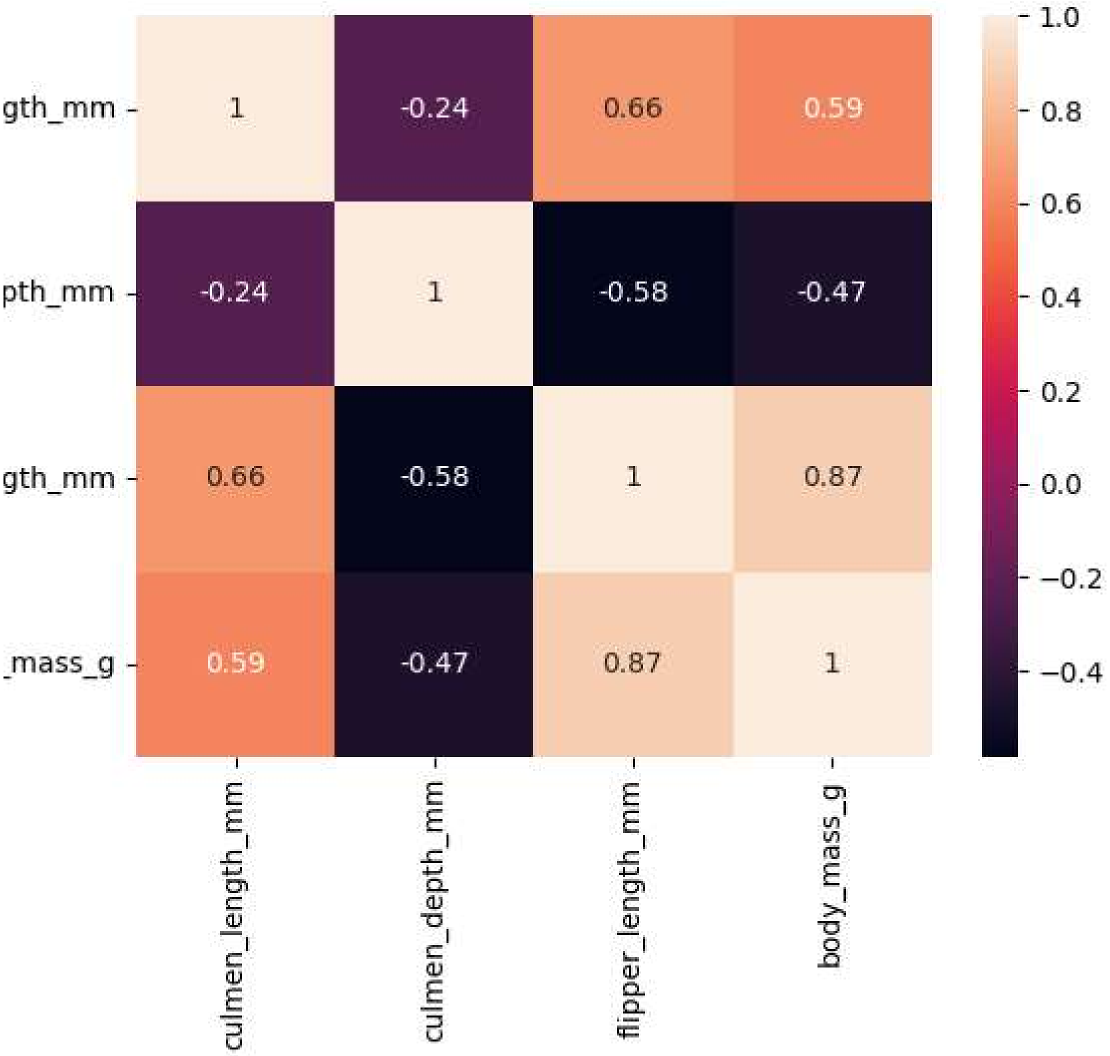
```
df.corr()
```

```
<ipython-input-59-2f6f6606aa2c>:1: FutureWarning: The default value of numeric_only i
  df.corr()
```

	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g
culmen_length_mm	1.000000	-0.235000	0.655858	0.594925
culmen_depth_mm	-0.235000	1.000000	-0.583832	-0.471942
flipper_length_mm	0.655858	-0.583832	1.000000	0.871221
body_mass_g	0.594925	-0.471942	0.871221	1.000000

```
sns.heatmap(df.corr(),annot=True)
```

```
ut-60-8df7bcac526d>:1: FutureWarning: The default value of numeric_only in DataFrame.
p(df.corr(),annot=True)
```



Task 8 : Check for Categorical columns and perform encoding.

```
from sklearn.preprocessing import LabelEncoder
```

```
le = LabelEncoder()#label encoding
```

```
df.sex = le.fit_transform(df.sex)
df.island = le.fit_transform(df.island)
df.species = le.fit_transform(df.species)
```

```
df.head()
```

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g
0	0	2	39.10	18.7	181.0	3750.0
1	0	2	39.50	17.4	186.0	3800.0

```
df.corr().body_mass_g.sort_values(ascending=False)
```

```
body_mass_g          1.000000
flipper_length_mm    0.871221
species              0.747547
culmen_length_mm     0.594925
sex                  0.337485
culmen_depth_mm     -0.471942
island              -0.558500
Name: body_mass_g, dtype: float64
```

Task 9 : Split the data into dependent and independent variables

```
ddf=df
```

```
y=df['body_mass_g']
y
```

```
0      3750.0
1      3800.0
2      3250.0
3      4050.0
4      3450.0
        ...
339    4050.0
340    4850.0
341    5750.0
342    5200.0
343    5400.0
Name: body_mass_g, Length: 344, dtype: float64
```

```
x=ddf.drop(columns=['body_mass_g'],axis=1)
x.head
```

```
<bound method NDFrame.head of      species  island  culmen_length_mm
culmen_depth_mm  flipper_length_mm  \
0          0       2             39.10            18.7              181.0
1          0       2             39.50            17.4              186.0
2          0       2             40.30            18.0              195.0
3          0       2             44.45            17.3              197.0
4          0       2             36.70            19.3              193.0
..       ...     ...               ...             ...                ...
339        2       0             44.45            17.3              197.0
340        2       0             46.80            14.3              215.0
341        2       0             50.40            15.7              222.0
342        2       0             45.20            14.8              212.0
343        2       0             49.90            16.1              213.0

     sex
0      2
```

```
1      1
2      1
3      3
4      1
..   ...
339    3
340    1
341    2
342    1
343    2

[344 rows x 6 columns]>
```

Task 10 : Scaling the data

```
from sklearn.preprocessing import MinMaxScaler
scale =MinMaxScaler()
```

```
x_scaled= pd.DataFrame(scale.fit_transform(x),columns =x.columns)
x_scaled.head()
```

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	sex
0	0.0	1.0	0.254545	0.666667	0.152542	0.666667
1	0.0	1.0	0.269091	0.511905	0.237288	0.333333
2	0.0	1.0	0.298182	0.583333	0.389831	0.333333
3	0.0	1.0	0.449091	0.500000	0.423729	1.000000
4	0.0	1.0	0.167273	0.738095	0.355932	0.333333

Task 11 : Split the data into training and testing

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x_scaled,y,test_size=0.3,random_state=10)
```

Task 12 : check the training and testing data shape.

```
x_train.shape
```

```
(240, 6)
```

```
x_train.head()
```

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	sex
258	1.0	0.0	0.432727	0.059524	0.610169	0.333333
332	1.0	0.0	0.414545	0.250000	0.694915	0.333333
121	0.0	1.0	0.203636	0.797619	0.440678	0.666667
61	0.0	0.0	0.334545	0.952381	0.389831	0.666667

```
y_train.shape
```

```
(240,)
```

```
x_test.shape
```

```
(104, 6)
```

✓ 0s completed at 8:55 PM