NAME: KELVIN J ANIL
REG NO: 21BCE0002

COLAB LINK:
https://colab.research.google.com/drive/1SweHvjCMbcwP8uwN4NqPOsHUkAKqdEZB?usp=sharing

1. Download the dataset: Dataset

2. Load the dataset into the tool.

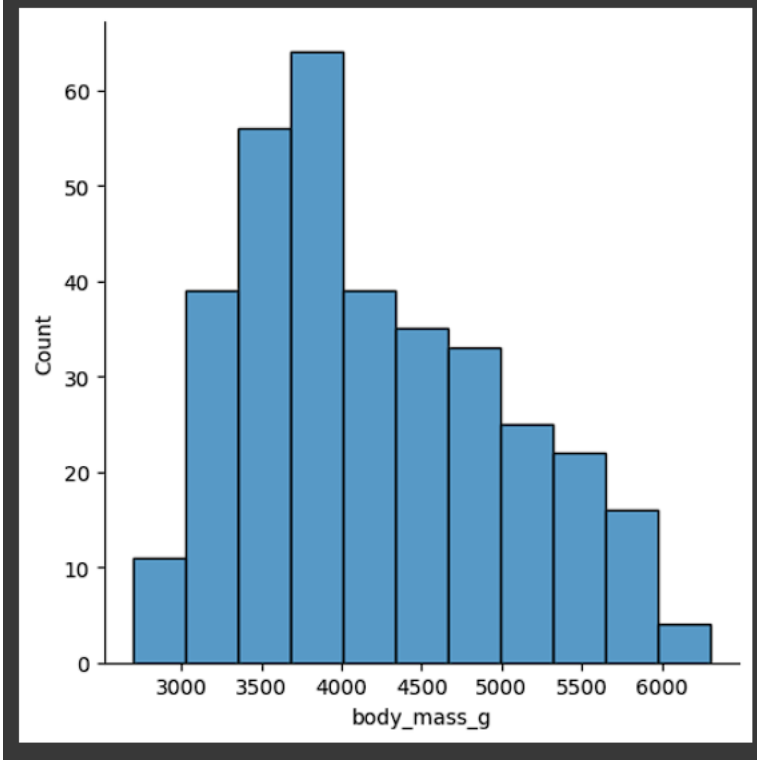
```
# Loading the Dataset
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv("/content/penguins_size.csv")
```

```
[3] df
```

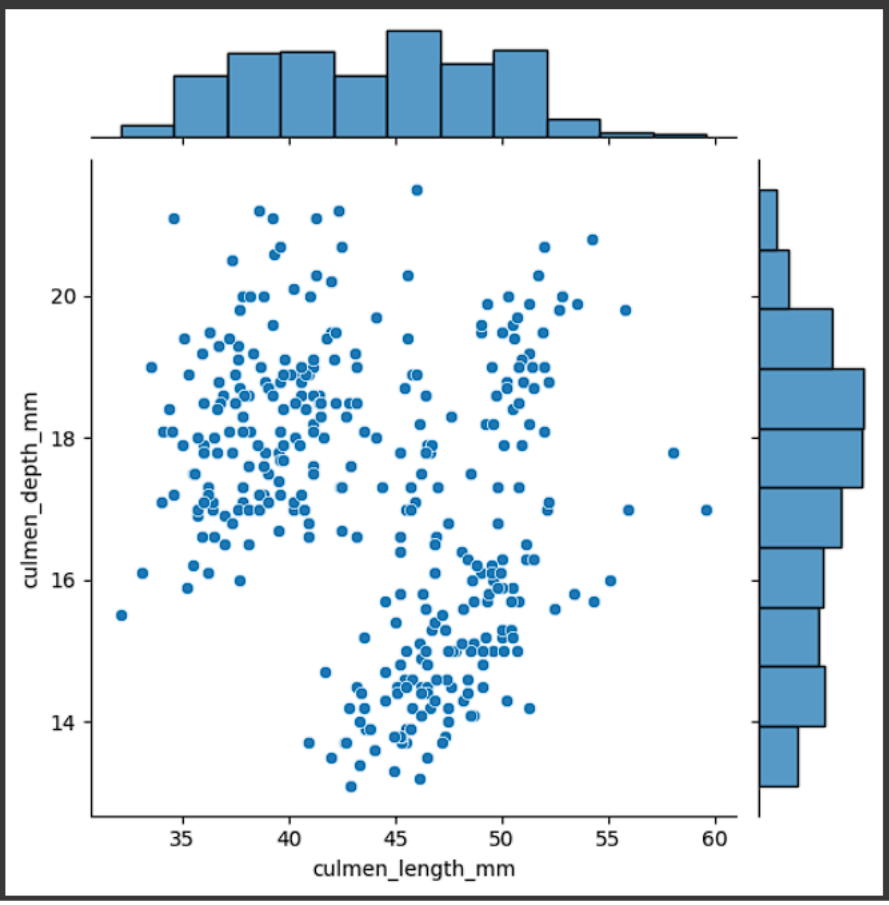
	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	MALE
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	FEMALE
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	FEMALE
3	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0	FEMALE
...	...	...	...	...	...	...	...
339	Gentoo	Biscoe	NaN	NaN	NaN	NaN	NaN

- Univariate Analysis
- Bi- Variate Analysis
- Multi-Variate Analysis

```
#Univariate Analysis
from matplotlib import rcParams
sns.displot(df["body_mass_g"])
plt.show()
```

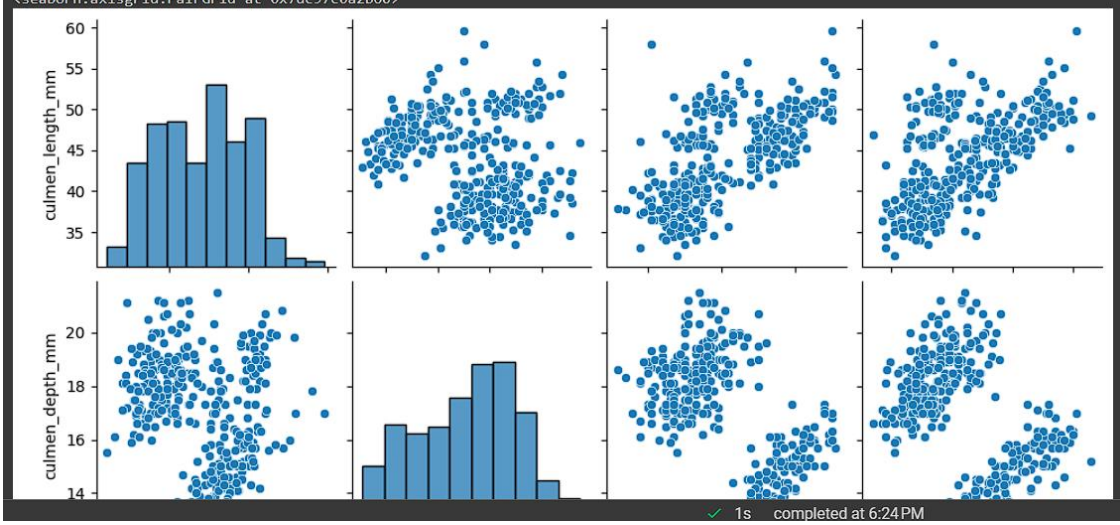


```
# bivariate Analysis
sns.jointplot(data=df,x="culmen_length_mm",y="culmen_depth_mm")
plt.show()
```

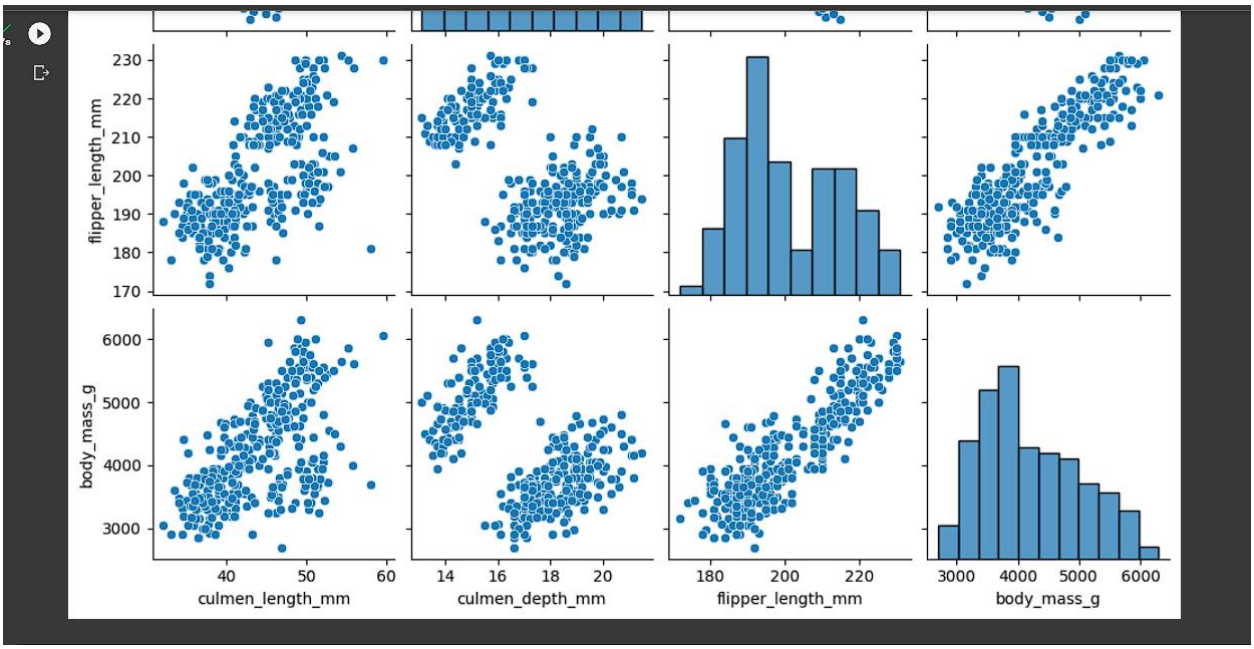



```
# multivariate analysis
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x7dc97c0a2b60>
```



✓ 1s completed at 6:24 PM



4. Perform descriptive statistics on the dataset.

```
[10] # descriptive Statistics
     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

5. Check for Missing values and deal with them.

```
[12] # Checking for missing values
     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
```

```
# Checking how many missing values
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
```

```
[15] # Handling missing values with Median
     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().iloc[0],inplace=True)
```

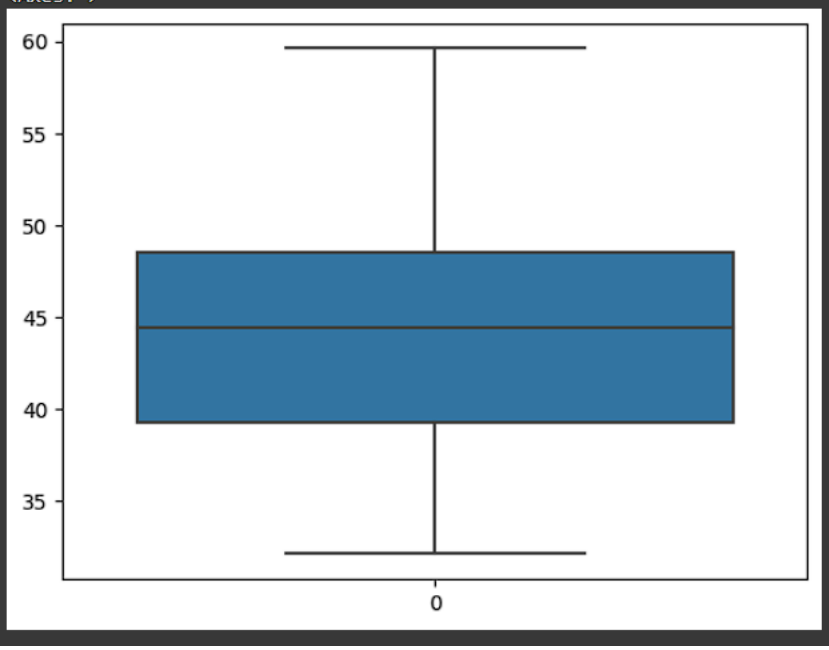
```
# Checking again for missing values
df.isnull().sum()
```

```
species              0
island               0
culmen_length_mm     0
culmen_depth_mm      0
flipper_length_mm    0
body_mass_g          0
sex                  0
dtype: int64
```

6. Find the outliers and replace them outliers

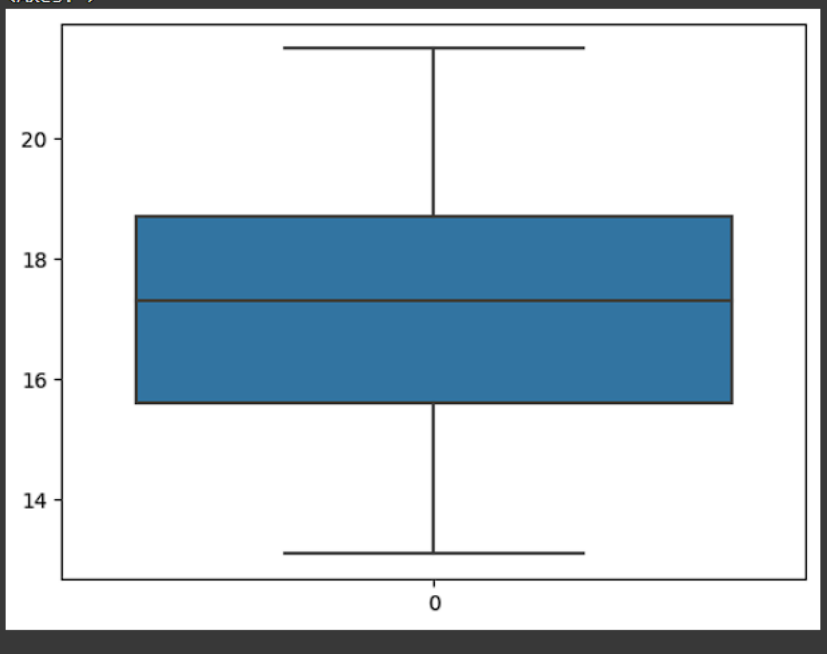
```
# finding outliers and replacing them
sns.boxplot(df["culmen_length_mm"])
```

```
<Axes: >
```

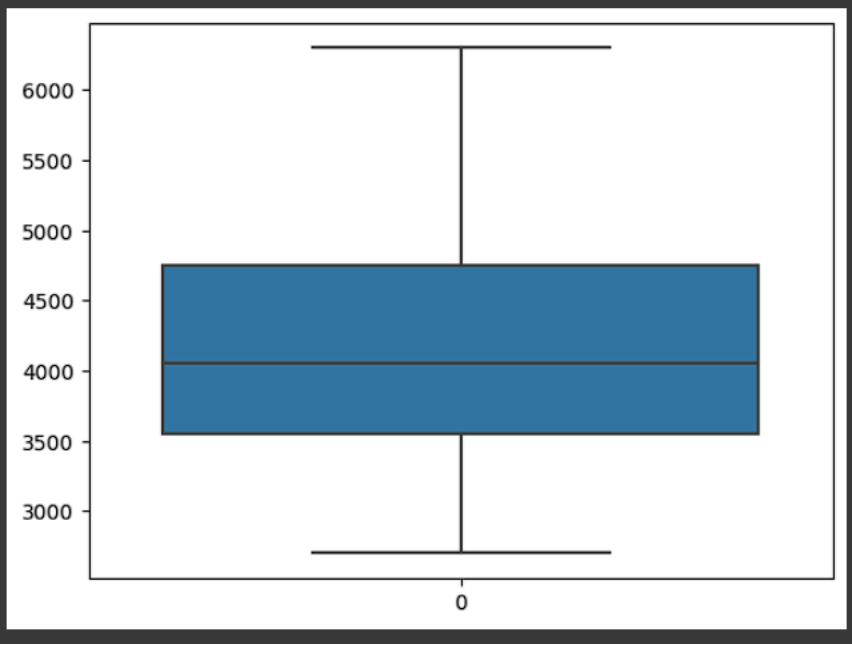



```
sns.boxplot(df["culmen_depth_mm"])
```

```
<Axes: >
```



```
[23] sns.boxplot(df["body_mass_g"])
     plt.show()
```




```
[24] # there are no outliers in this dataset
```

7.Check the correlation of independent variables with the target

```
[32] # correlation
     df.corr().species.sort_values(ascending=False)
```

```
species              1.000000
flipper_length_mm    0.850819
body_mass_g          0.747547
culmen_length_mm     0.728706
sex                 -0.003823
island              -0.635659
culmen_depth_mm     -0.741282
Name: species, dtype: float64
```

8. Check for Categorical columns and perform encoding.

```
[33] # checking for categorical columns and perform encoding
     from sklearn.preprocessing import LabelEncoder
     le_sex = LabelEncoder()
     le_species = LabelEncoder()
     le_island = LabelEncoder()
     df["sex"] = le_sex.fit_transform(df["sex"])
     df["species"] = le_species.fit_transform(df["species"])
     df["island"] = le_island.fit_transform(df["island"])
     df.head()
```

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
0	0	2	39.10	18.7	181.0	3750.0	2
1	0	2	39.50	17.4	186.0	3800.0	1
2	0	2	40.30	18.0	195.0	3250.0	1
3	0	2	44.45	17.3	197.0	4050.0	2
4	0	2	36.70	19.3	193.0	3450.0	1

9. Split the data into dependent and independent variables.

```
[34] # split the data into dependent and independent variables
     X=df.drop(columns=["species"],axis=1)
     X.head()
```

	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
0	2	39.10	18.7	181.0	3750.0	2
1	2	39.50	17.4	186.0	3800.0	1
2	2	40.30	18.0	195.0	3250.0	1
3	2	44.45	17.3	197.0	4050.0	2
4	2	36.70	19.3	193.0	3450.0	1

```
Y = df["species"]
Y.head()
```

```
0    0
1    0
2    0
3    0
4    0
Name: species, dtype: int64
```

10. Scaling the data

```
[38] # Scalling the data
     from sklearn.preprocessing import MinMaxScaler
     scale = MinMaxScaler()
     X_scaled = pd.DataFrame(scale.fit_transform(X),columns=X.columns)
     X_scaled.head()
```

	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
0	1.0	0.254545	0.666667	0.152542	0.291667	1.0
1	1.0	0.269091	0.511905	0.237288	0.305556	0.5
2	1.0	0.298182	0.583333	0.389831	0.152778	0.5
3	1.0	0.449091	0.500000	0.423729	0.375000	1.0
4	1.0	0.167273	0.738095	0.355932	0.208333	0.5

11. Split the data into training and testing

12.check the training and testing data shape.

```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X_scaled, Y, test_size=0.2, random_state=42)

[41] X_train.shape

     (275, 6)

[42] X_test.shape

     (69, 6)

[43] Y_train.shape

     (275,)

[45] Y_test.shape

     (69,)
```