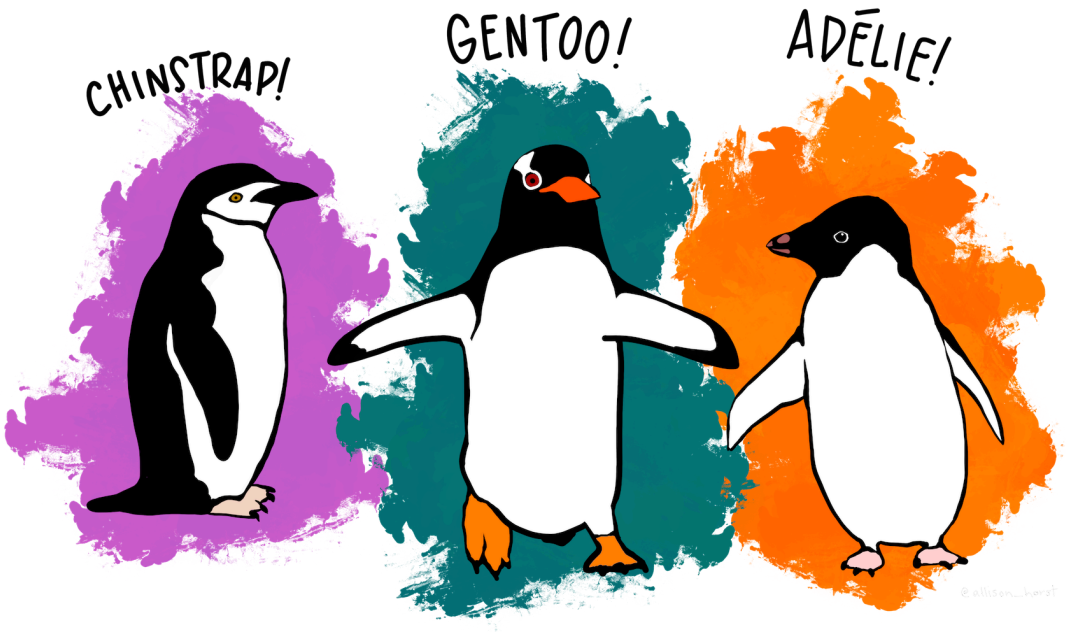
Exploratory Data Analysis — Palmer Penguins 🐧



In this notebook you will practise core EDA skills:

1. Loading & inspecting a dataset
2. Handling missing values
3. Computing summary statistics
4. Visualising distributions and relationships

Dataset: Palmer Penguins (shipped with `seaborn`).
Each row describes a penguin observed on three islands in the Palmer Archipelago, Antarctica.

Fill in every cell marked with `# YOUR CODE HERE` (the surrounding code and comments tell you what is expected).

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_theme(style="whitegrid")
%matplotlib inline
```

1. Load the dataset

Load the penguins dataset from seaborn using `sns.load_dataset("penguins")` and store it in a variable called `df`.
Then display the first 5 rows.

In [2]:
```
df = sns.load_dataset("penguins")
df.head()
```

Out[2]:

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass
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

2. Basic inspection

2a. Print the shape of the dataframe (rows, columns).

In [3]:
```
print(df.shape)
```

```
(344, 7)
```

2b. Display the data types of each column using `.dtypes`

In [4]:
```
df.dtypes
```

Out[4]:
```
species              str
island               str
bill_length_mm     float64
bill_depth_mm      float64
flipper_length_mm  float64
body_mass_g        float64
sex                  str
dtype: object
```

2c. Use `.info()` to get a concise summary including non-null counts.

In [5]:
```
df.info()
```

```
<class 'pandas.DataFrame'>
RangeIndex: 344 entries, 0 to 343
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   species            344 non-null    str
 1   island             344 non-null    str
 2   bill_length_mm     342 non-null    float64
 3   bill_depth_mm      342 non-null    float64
 4   flipper_length_mm  342 non-null    float64
 5   body_mass_g        342 non-null    float64
 6   sex                333 non-null    str
dtypes: float64(4), str(3)
memory usage: 18.9 KB
```

3. Missing values

https://www.geeksforgeeks.org/python/python-pandas-isnull-and-notnull/

3a. Count the number of missing values per column.

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
species              0
island               0
bill_length_mm       2
bill_depth_mm        2
flipper_length_mm    2
body_mass_g          2
sex                 11
dtype: int64
```

3b. Compute the percentage of missing values per column (round to 2 decimals).

In [7]:
```
(df.isnull().sum() / len(df) * 100).round(2)
```

Out[7]:
```
species             0.00
island              0.00
bill_length_mm      0.58
bill_depth_mm       0.58
flipper_length_mm   0.58
body_mass_g         0.58
sex                 3.20
dtype: float64
```

3c. Drop rows with *any* missing value and store the result in `df_clean` with `.drop`. Print its new shape.

In [8]:
```
df_clean = df.dropna()
print(df_clean.shape)
```

```
(333, 7)
```

4. Summary statistics

4a. Show summary statistics for all *numerical* columns using `.describe()`.

In [9]:
```
df_clean.describe()
```

Out[9]:

	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
count	333.000000	333.000000	333.000000	333.000000
mean	43.992793	17.164865	200.966967	4207.057057
std	5.468668	1.969235	14.015765	805.215802
min	32.100000	13.100000	172.000000	2700.000000
25%	39.500000	15.600000	190.000000	3550.000000
50%	44.500000	17.300000	197.000000	4050.000000
75%	48.600000	18.700000	213.000000	4775.000000
max	59.600000	21.500000	231.000000	6300.000000

4b. Compute the **mean** of `bill_length_mm` and `bill_depth_mm` grouped by `species`.

In [10]:
```
df_clean.groupby("species")[["bill_length_mm", "bill_depth_mm"]].mean()
```

Out[10]:

species	bill_length_mm	bill_depth_mm
Adelie	38.823973	18.347260
Chinstrap	48.833824	18.420588
Gentoo	47.568067	14.996639

4c. Count the number of penguins per `island`.

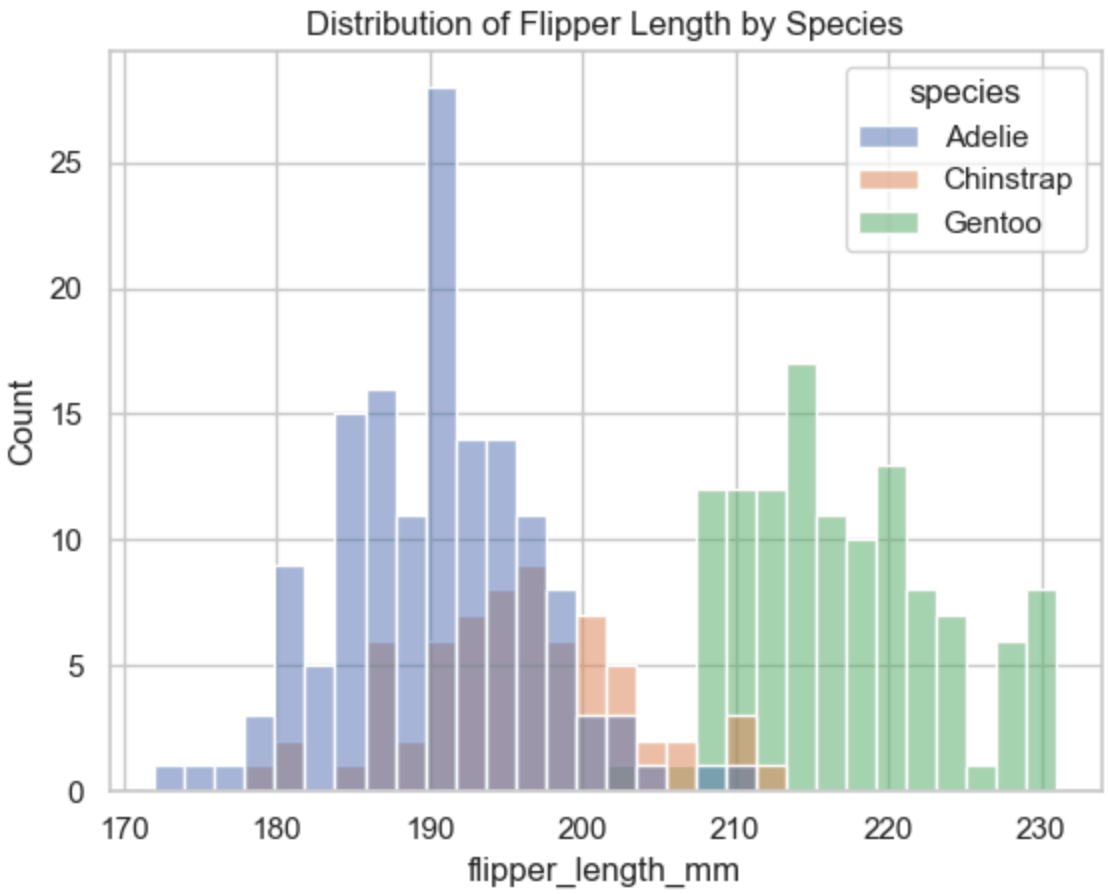
In [11]:
```
df_clean["island"].value_counts()
```

Out[11]:
```
island
Biscoe       163
Dream        123
Torgersen     47
Name: count, dtype: int64
```

5. Visualisations

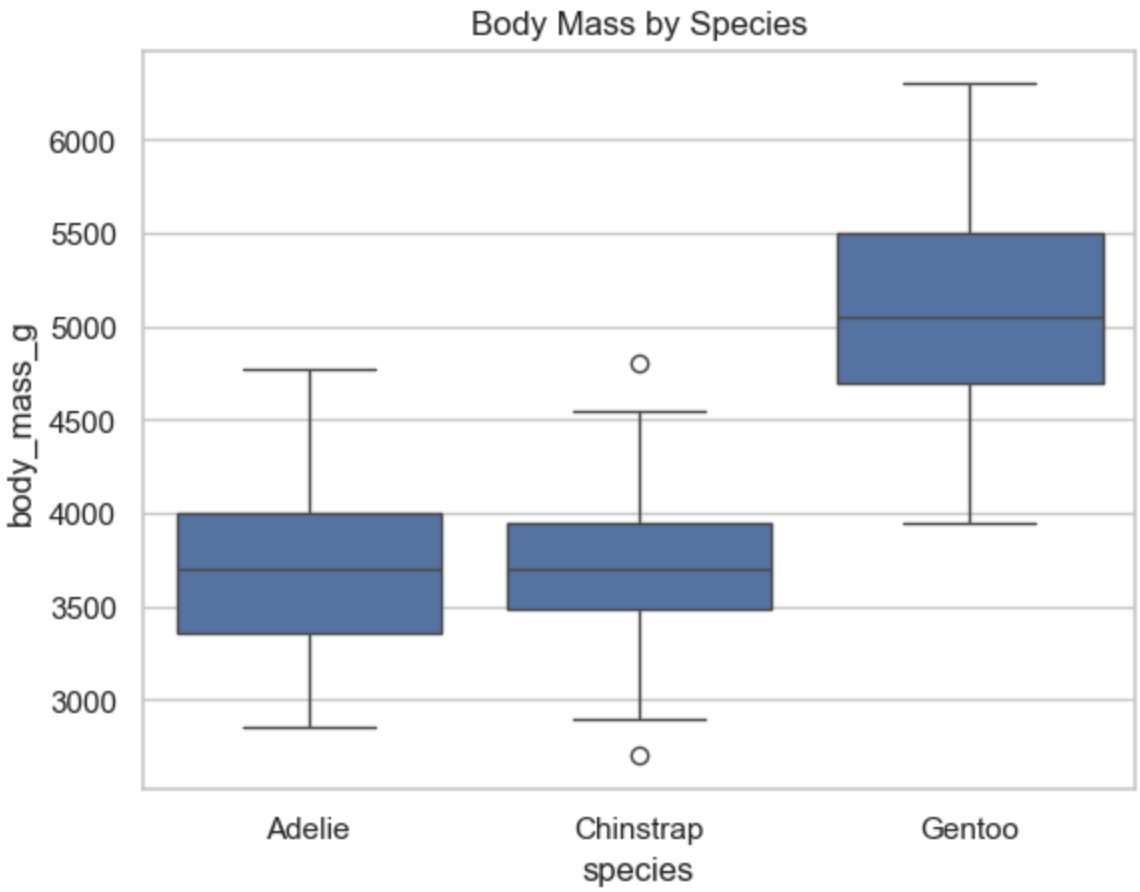
5a. Plot a histogram of `flipper_length_mm`, coloured by `species`, with 30 bins. Add a title.

In [12]:
```
sns.histplot(data=df_clean, x="flipper_length_mm", hue="species", bins=30)
plt.title("Distribution of Flipper Length by Species")
plt.show()
```



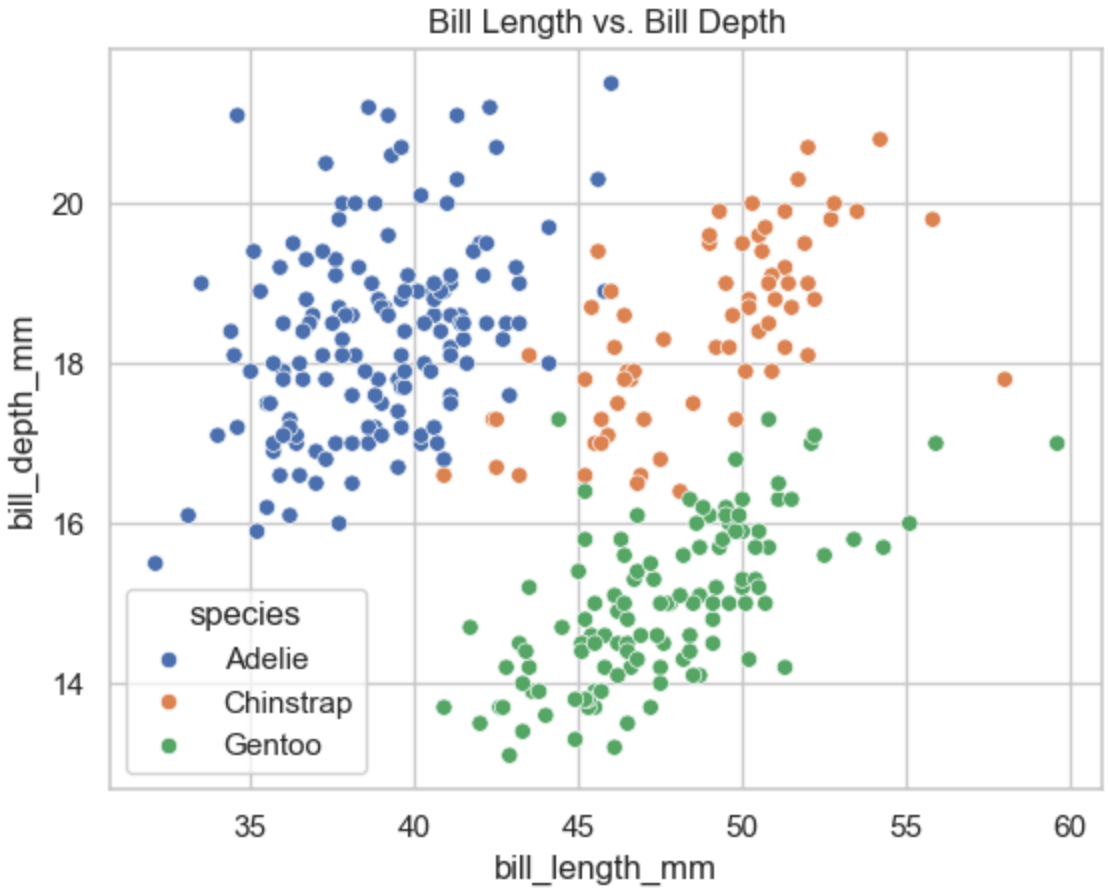
5b. Create a boxplot of `body_mass_g` for each `species`.

In [13]:
```
sns.boxplot(data=df_clean, x="species", y="body_mass_g")
plt.title("Body Mass by Species")
plt.show()
```



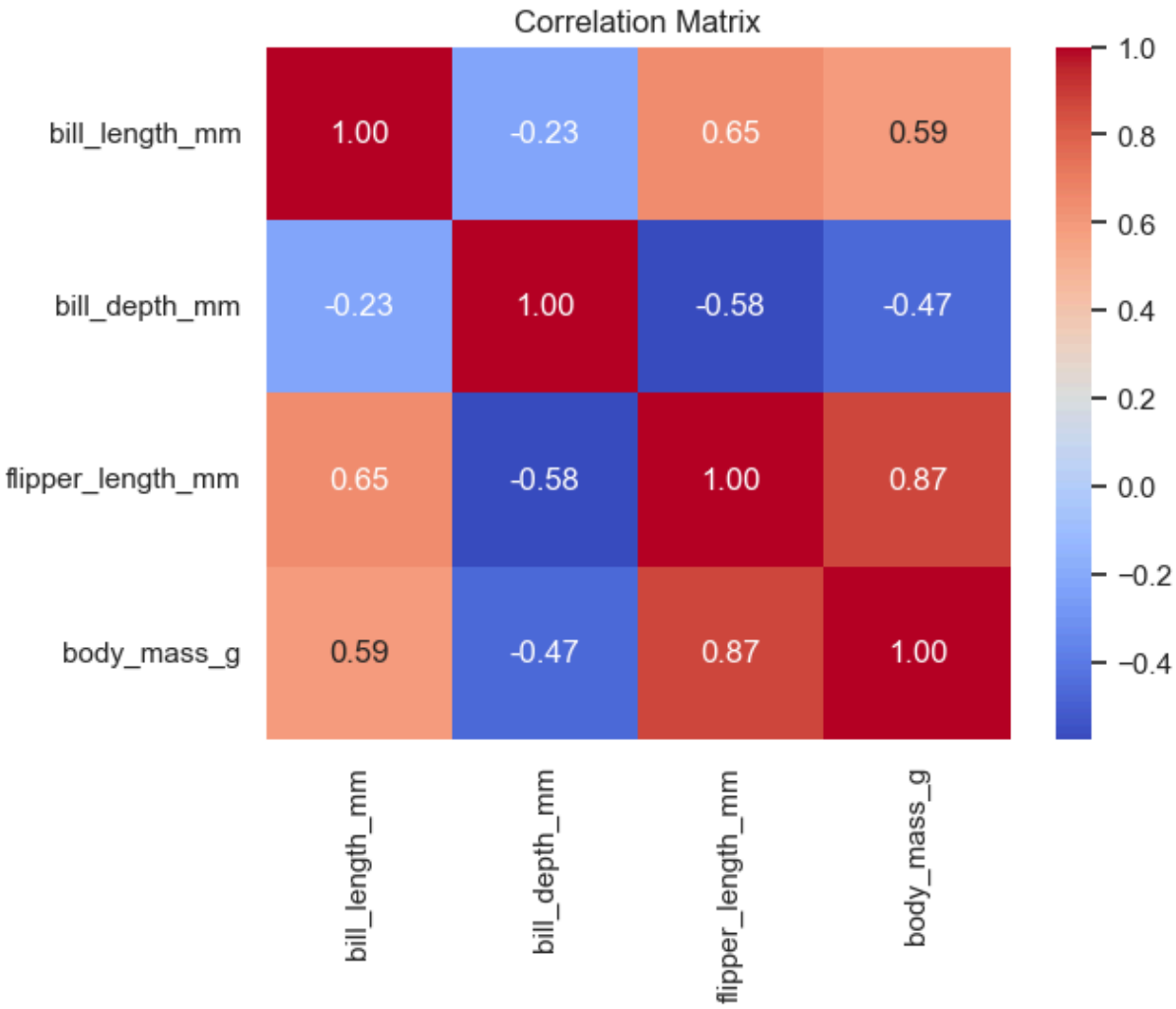
5c. Make a scatter plot of `bill_length_mm` (x) vs. `bill_depth_mm` (y), coloured by `species`.

In [14]:
```
sns.scatterplot(data=df_clean, x="bill_length_mm", y="bill_depth_mm", hue="s
plt.title("Bill Length vs. Bill Depth")
plt.show()
```



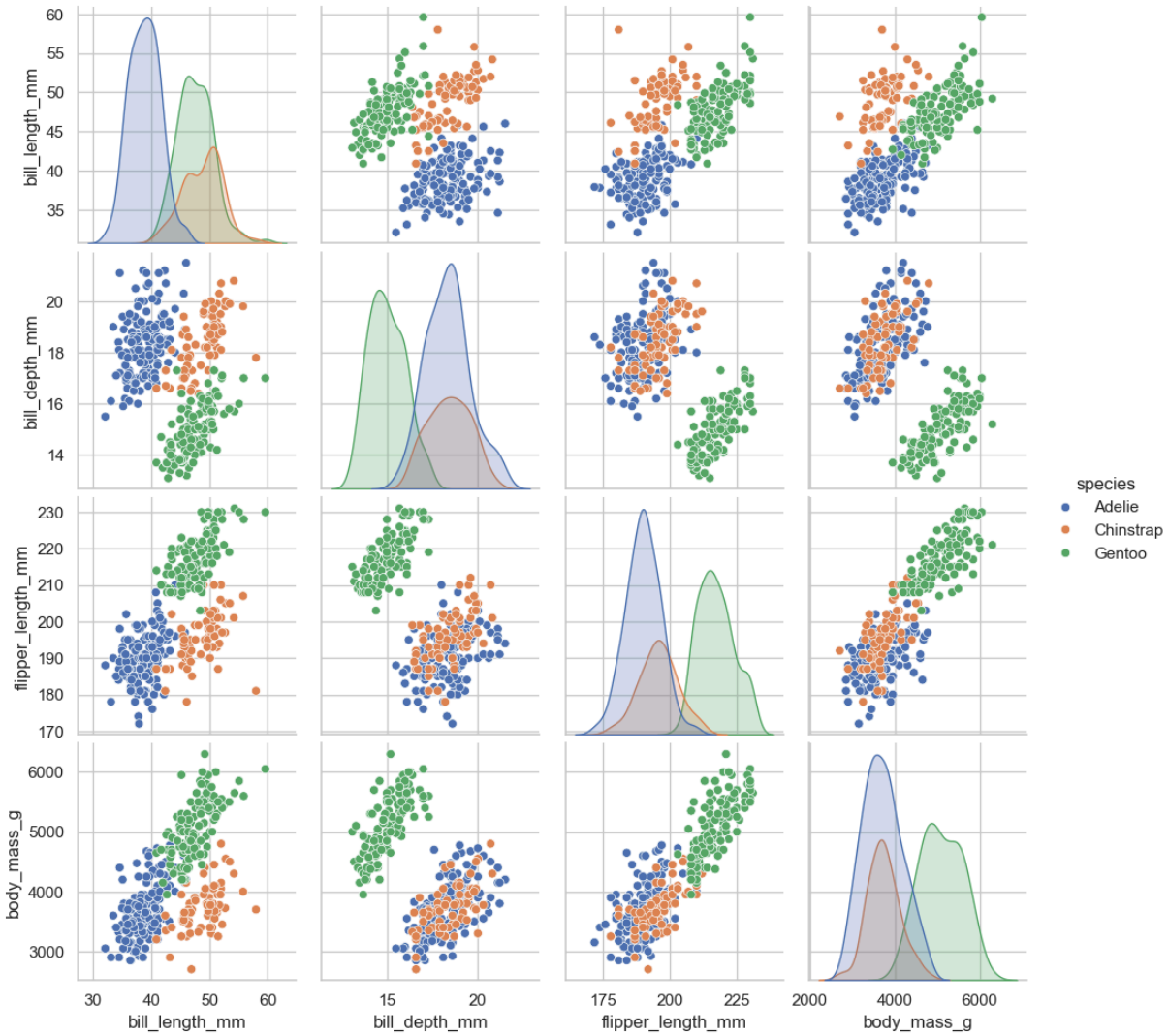
5d. Compute the correlation matrix of the numerical columns in `df_clean` and visualise it as a heatmap with annotations.

In [15]:
```
corr = df_clean.select_dtypes(include="number").corr()
sns.heatmap(corr, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Correlation Matrix")
plt.show()
```



5e. Create a pair plot of the numerical features, coloured by `species`.

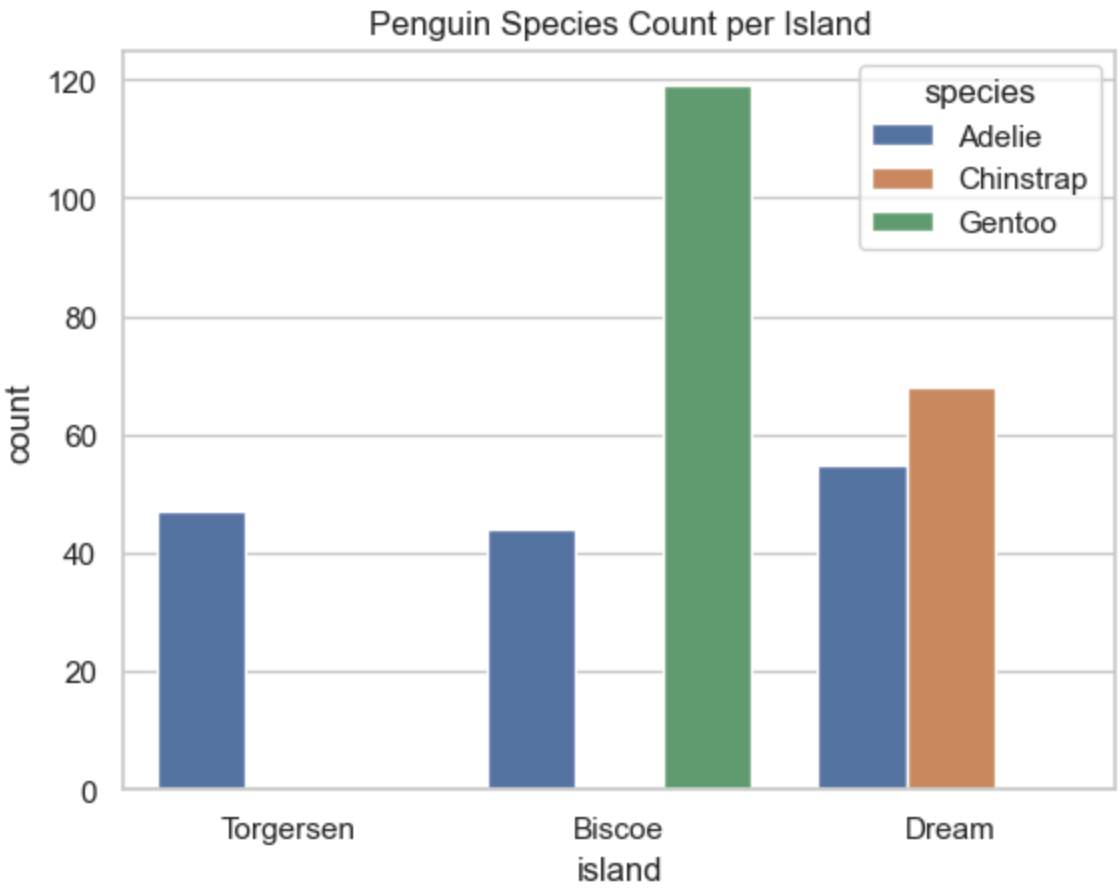
In [16]:
```
sns.pairplot(df_clean, hue="species")
plt.show()
```



6. Bonus

6. Create a `countplot` showing the number of penguins per `species` on each `island`.
Hint: use the `hue` parameter.

In [17]:
```
sns.countplot(data=df_clean, x="island", hue="species")
plt.title("Penguin Species Count per Island")
plt.show()
```



Well done! You have completed a basic EDA of the Palmer Penguins dataset.