In [3]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [4]:
```
file_path = "abalone1.xlsx"
df = pd.read_excel(file_path)
```

In [3]:
```
print("Dataset Overview:\n")
print(df.info())
print("\nFirst 5 rows:\n", df.head())
```

```
Dataset Overview:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1999 entries, 0 to 1998
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Sex             1999 non-null   object 
 1   Length          1999 non-null   float64
 2   Diameter        1999 non-null   float64
 3   Height          1999 non-null   float64
 4   Whole_weight    1999 non-null   float64
 5   Shucked_weight  1999 non-null   float64
 6   Viscera_weight  1999 non-null   float64
 7   Shell_weight    1999 non-null   float64
 8   Rings           1999 non-null   int64  
dtypes: float64(7), int64(1), object(1)
memory usage: 140.7+ KB
None

First 5 rows:
   Sex  Length  Diameter  Height  Whole_weight  Shucked_weight  Viscera_weight  \
0   M   0.455     0.365   0.095        0.5140          0.2245          0.1010   
1   M   0.350     0.265   0.090        0.2255          0.0995          0.0485   
2   F   0.530     0.420   0.135        0.6770          0.2565          0.1415   
3   M   0.440     0.365   0.125        0.5160          0.2155          0.1140   
4   I   0.330     0.255   0.080        0.2050          0.0895          0.0395   

   Shell_weight  Rings  
0         0.150     15  
1         0.070      7  
2         0.210      9  
3         0.155     10  
4         0.055      7  
```

In [4]:
```
print("\nMissing Values:\n", df.isnull().sum())
```

```
Missing Values:
 Sex               0
Length             0
Diameter           0
Height             0
Whole_weight       0
Shucked_weight     0
Viscera_weight     0
Shell_weight       0
Rings              0
dtype: int64
```

In [8]:
```
range_df = df.select_dtypes(include=[np.number]).apply(lambda x: x.max() - x.min
print("Range (Max - Min) of Numeric Features:")
print(range_df)
```

```
Range (Max - Min) of Numeric Features:
Length             0.7400
Diameter           0.5950
Height             0.5150
Whole_weight       2.8235
Shucked_weight     1.4870
Viscera_weight     0.7595
Shell_weight       1.0035
Rings             28.0000
dtype: float64
```

In [5]:
```
print("\nSummary Statistics:\n", df.describe())
```

```
Summary Statistics:
              Length     Diameter       Height  Whole_weight  Shucked_weight  \
count  1999.000000  1999.000000  1999.000000   1999.000000      1999.00000
mean      0.521913     0.406523     0.138999      0.824458         0.35721
std       0.122185     0.101312     0.039680      0.496452         0.22598
min       0.075000     0.055000     0.000000      0.002000         0.00100
25%       0.450000     0.345000     0.115000      0.434000         0.18100
50%       0.545000     0.425000     0.140000      0.800000         0.33600
75%       0.610000     0.480000     0.165000      1.137250         0.49825
max       0.815000     0.650000     0.515000      2.825500         1.48800

       Viscera_weight  Shell_weight        Rings
count     1999.000000   1999.000000  1999.000000
mean         0.179646      0.237531     9.941471
std          0.110340      0.140981     3.320558
min          0.000500      0.001500     1.000000
25%          0.092250      0.125000     8.000000
50%          0.170000      0.234000     9.000000
75%          0.249500      0.325000    11.000000
max          0.760000      1.005000    29.000000
```

What are the column names in the dataset?

In [15]:
```
print("Column Names:", df.columns)
```

```
Column Names: Index(['Sex', 'Length', 'Diameter', 'Height', 'Whole_weight', 'Shuc
ked_weight',
       'Viscera_weight', 'Shell_weight', 'Rings'],
      dtype='object')
```

How many unique values are there in the 'Sex' column

In [16]:
```
print("Unique values in 'Sex':", df["Sex"].unique())
```

```
Unique values in 'Sex': ['M' 'F' 'I']
```

What is the average number of rings for male and female abalones?

In [17]:
```
print("Average Rings for Each Sex:")
print(df.groupby("Sex")["Rings"].mean())
```

```
Average Rings for Each Sex:
Sex
F    11.264706
I     7.832565
M    10.706522
Name: Rings, dtype: float64
```

In [11]:
```
print("Standard Deviation of Numeric Features:")
print(df.select_dtypes(include=[np.number]).std())
```

```
Standard Deviation of Numeric Features:
Length            0.122185
Diameter          0.101312
Height            0.039680
Whole_weight      0.496452
Shucked_weight    0.225980
Viscera_weight    0.110340
Shell_weight      0.140981
Rings             3.320558
dtype: float64
```

In [10]:
```
print("Correlation of Features with Rings:")
print(df.corr(numeric_only=True)["Rings"].sort_values(ascending=False))
```

```
Correlation of Features with Rings:
Rings             1.000000
Shell_weight      0.621530
Height            0.589210
Diameter          0.580848
Length            0.559143
Whole_weight      0.540908
Viscera_weight    0.503765
Shucked_weight    0.422398
Name: Rings, dtype: float64
```

In [9]:
```
print("Mean, Min, Max Rings by Sex:")
print(df.groupby("Sex")["Rings"].agg(["mean", "min", "max"]))
```

```
Mean, Min, Max Rings by Sex:
          mean  min  max
Sex
F    11.264706    5   29
I     7.832565    1   21
M    10.706522    3   26
```

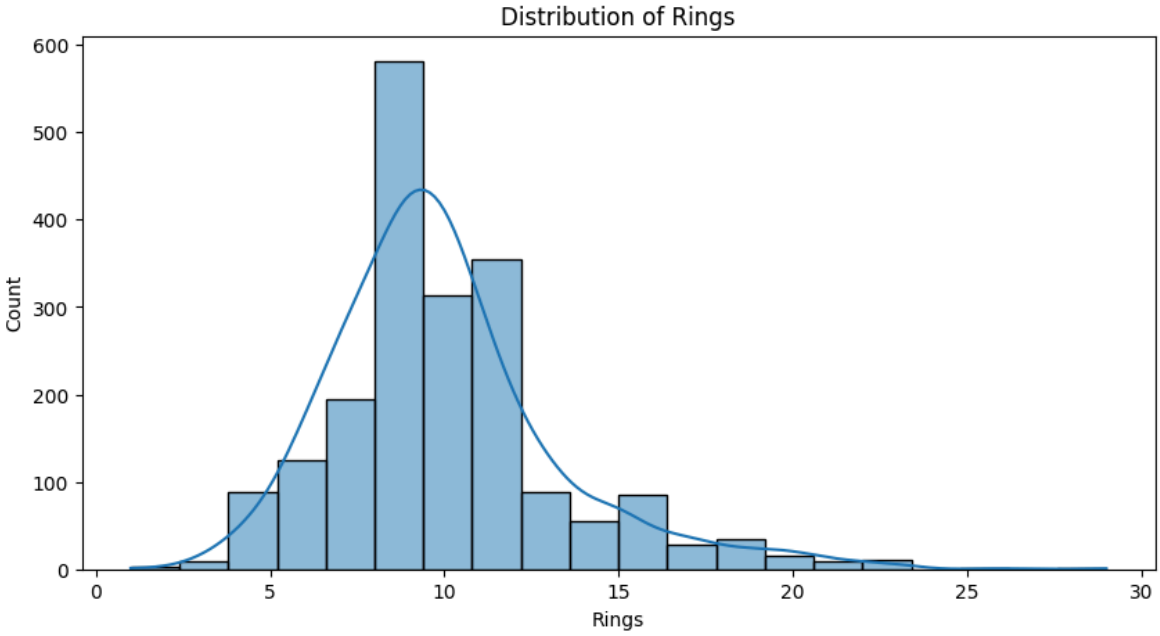
Which feature has the highest correlation with the number of rings?

In [18]:
```
corr_matrix = df_numeric.corr()
most_correlated_feature = corr_matrix["Rings"].drop("Rings").idxmax()
print("Feature most correlated with Rings:", most_correlated_feature)
```

```
Feature most correlated with Rings: Shell_weight
```

What is the distribution of the 'Rings' feature?

In [19]:
```
plt.figure(figsize=(10, 5))
sns.histplot(df["Rings"], bins=20, kde=True)
plt.title("Distribution of Rings")
plt.show()
```



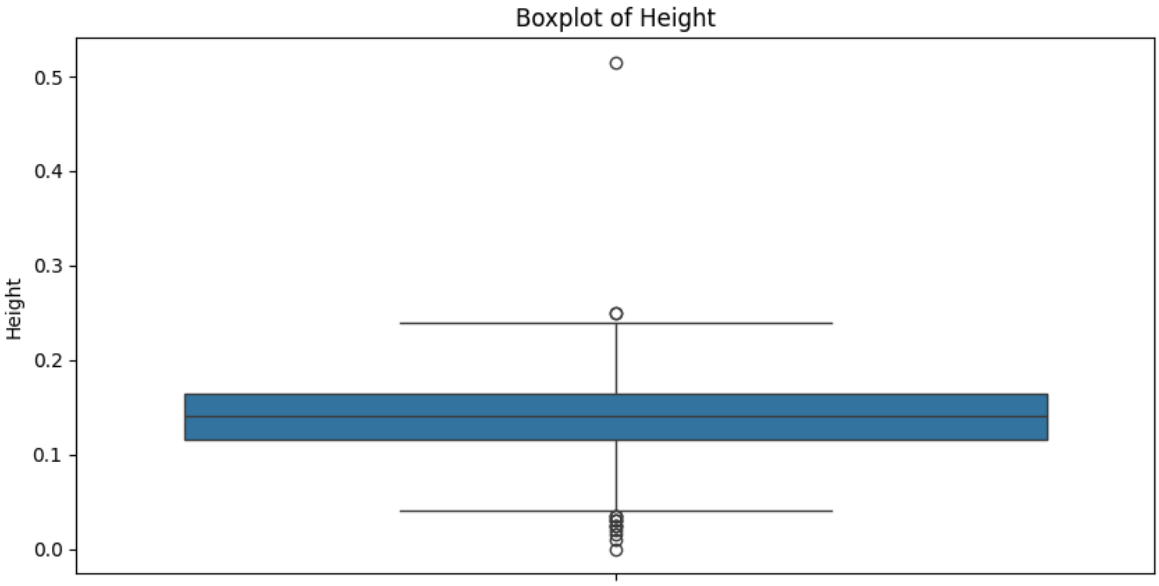
How does the mean length of abalones differ across sexes?

In [20]:
```
print("Mean Length by Sex:")
print(df.groupby("Sex")["Length"].mean())
```

```
Mean Length by Sex:
Sex
F    0.578717
I    0.426075
M    0.559450
Name: Length, dtype: float64
```

Are there any outliers in the 'Height' column?

In [21]:
```
plt.figure(figsize=(10, 5))
sns.boxplot(y=df["Height"])
plt.title("Boxplot of Height")
plt.show()
```



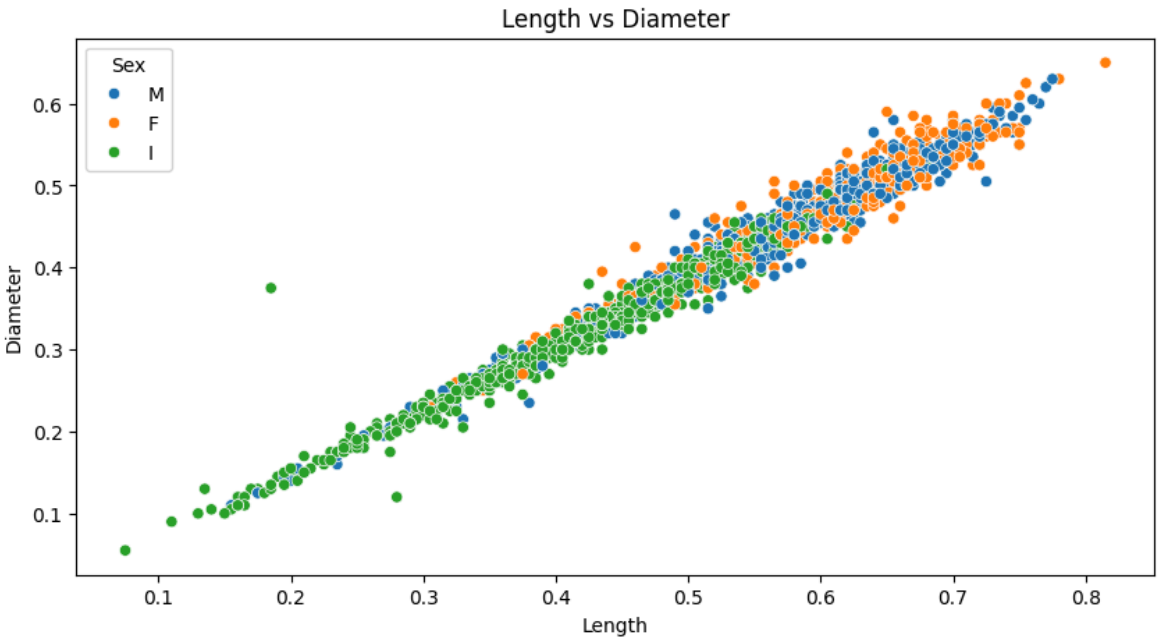
How many abalones are present in each sex category?

In [22]:
```
print("Count of Abalones by Sex:")
print(df["Sex"].value_counts())
```

```
Count of Abalones by Sex:
Sex
M    736
I    651
F    612
Name: count, dtype: int64
```

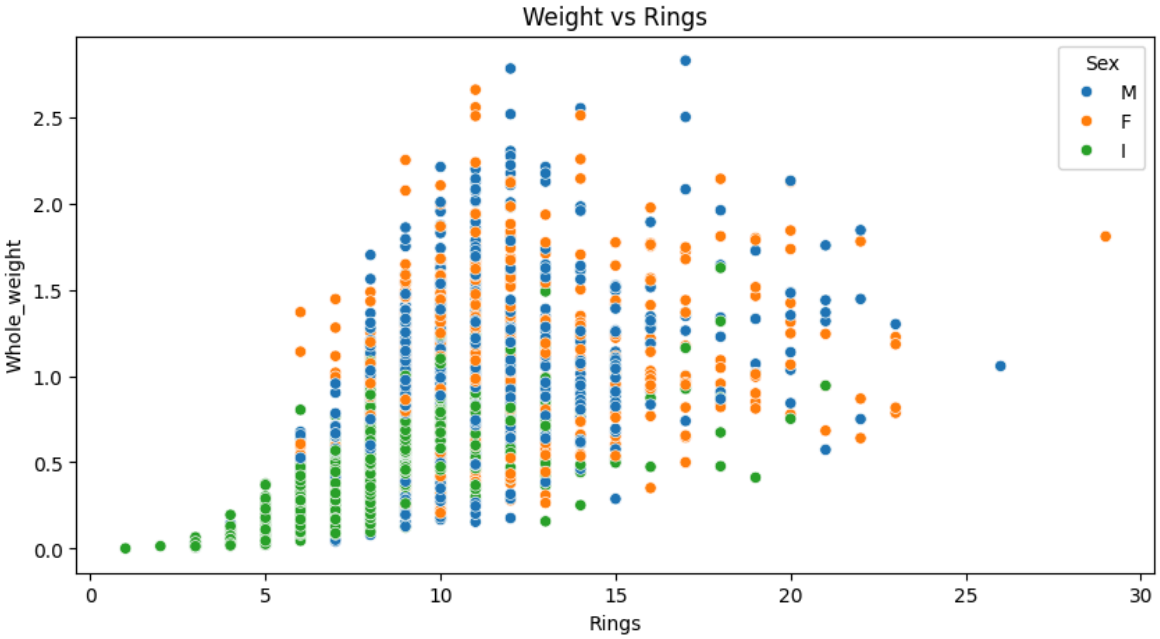
What is the trend of length vs diameter?

In [23]:
```
plt.figure(figsize=(10, 5))
sns.scatterplot(x=df["Length"], y=df["Diameter"], hue=df["Sex"])
plt.title("Length vs Diameter")
plt.show()
```

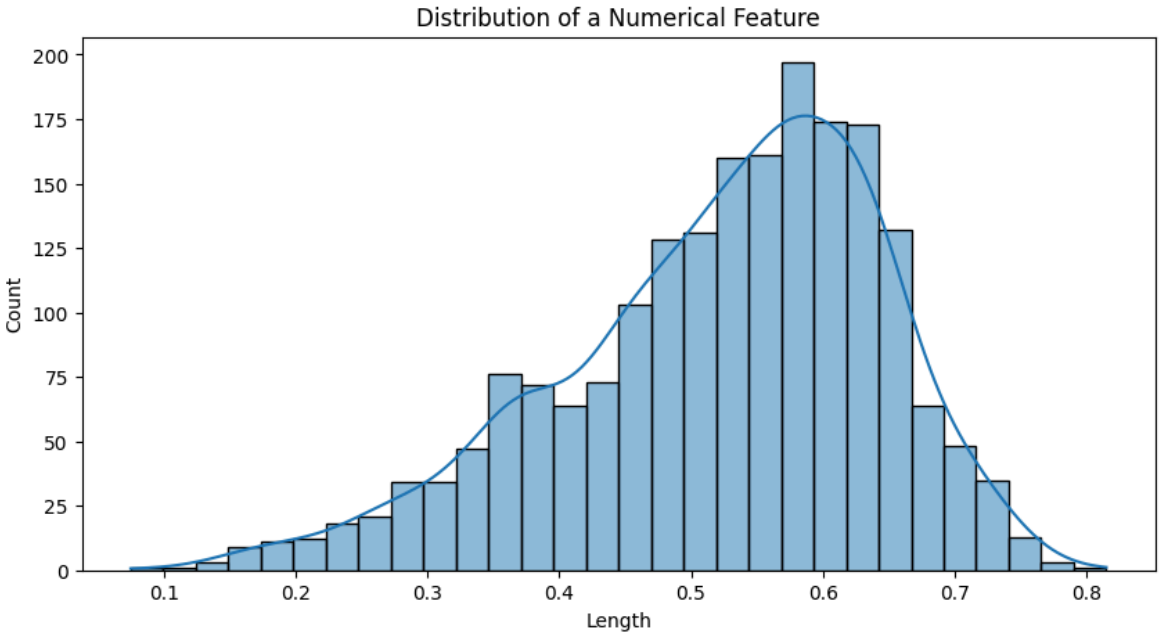


How does weight vary with the number of rings?

In [25]:
```
plt.figure(figsize=(10, 5))
sns.scatterplot(x=df["Rings"], y=df["Whole_weight"], hue=df["Sex"])
plt.title("Weight vs Rings")
plt.show()
```



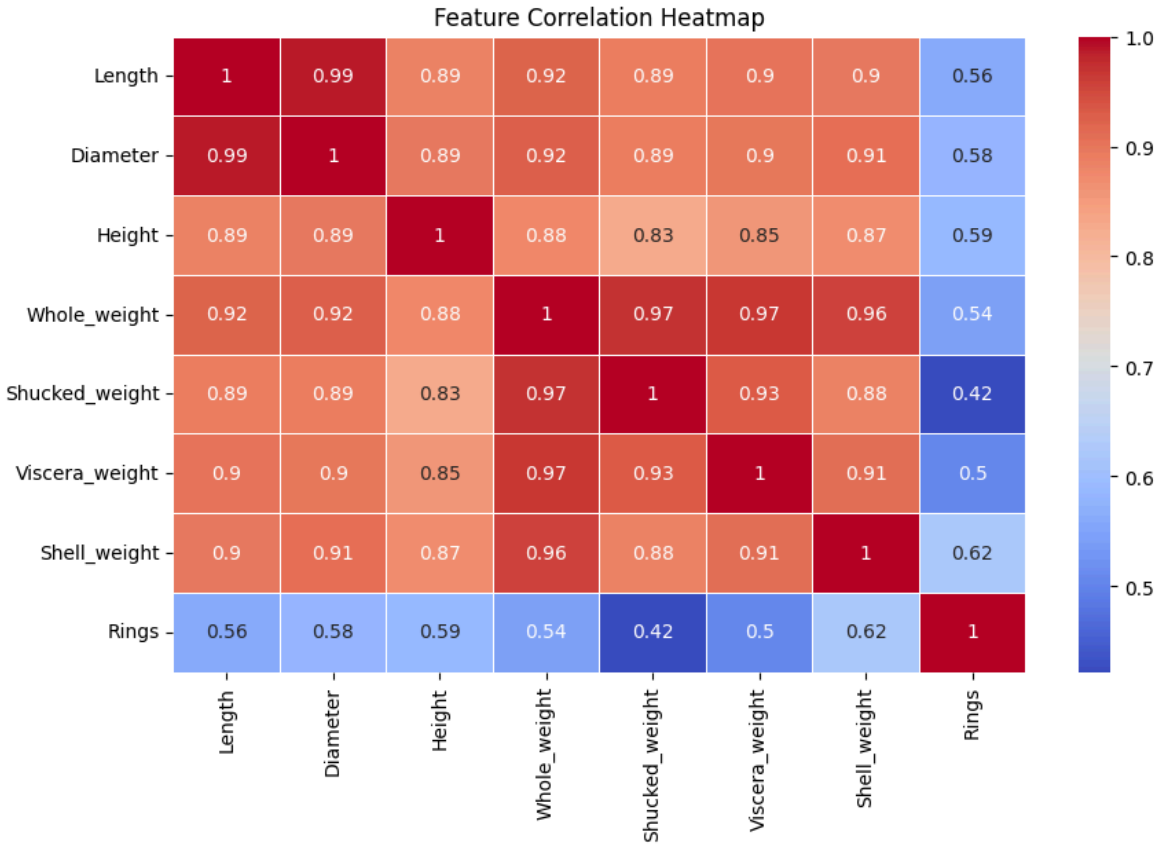
In [6]:
```
plt.figure(figsize=(10, 5))
sns.histplot(df.iloc[:, 1], bins=30, kde=True)
plt.title("Distribution of a Numerical Feature")
plt.show()
```



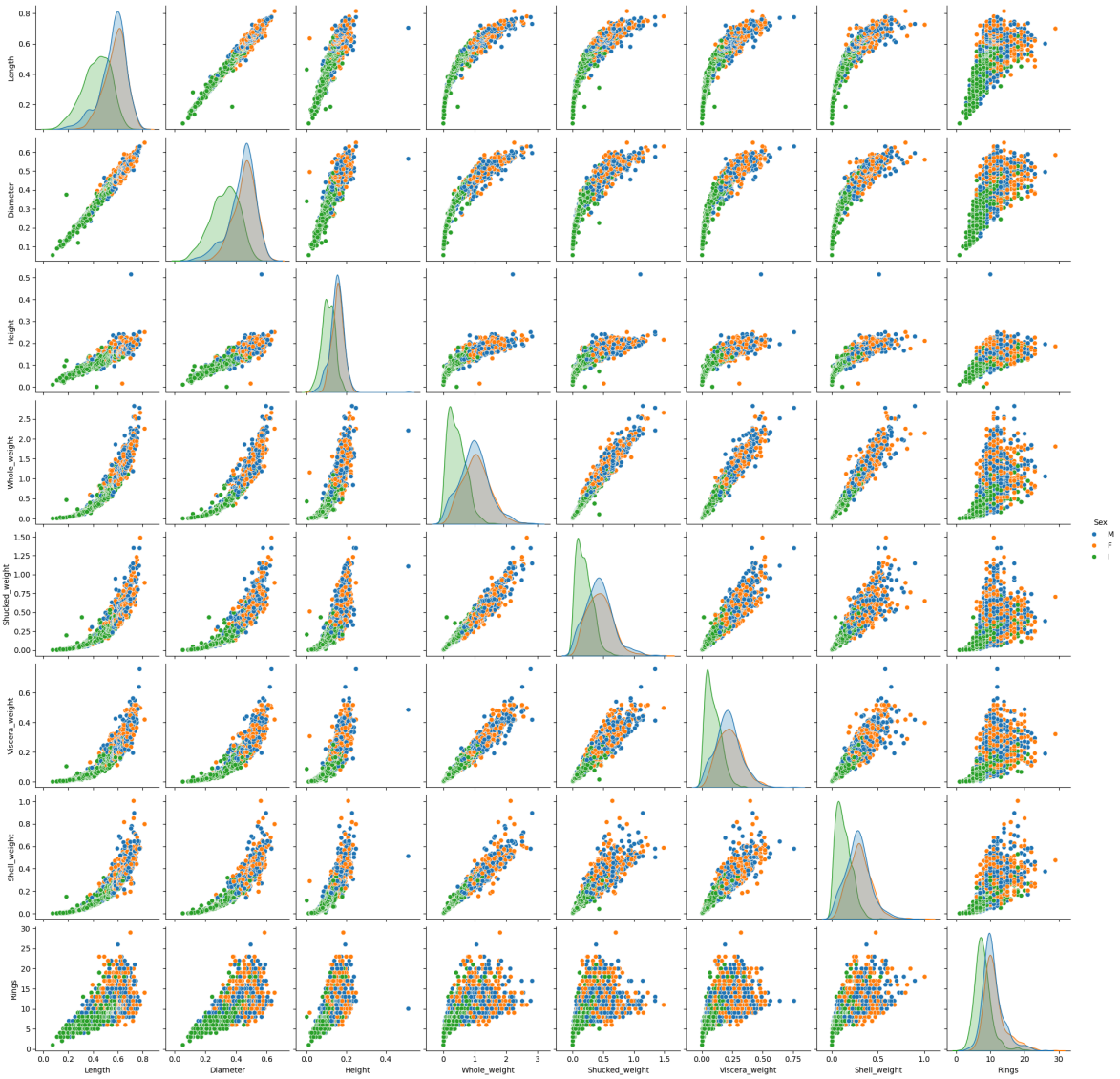
In [8]:
```
# Convert categorical columns to numeric (if necessary)
df_numeric = df.select_dtypes(include=[np.number])  # Select only numeric column

# Correlation heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(df_numeric.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
```

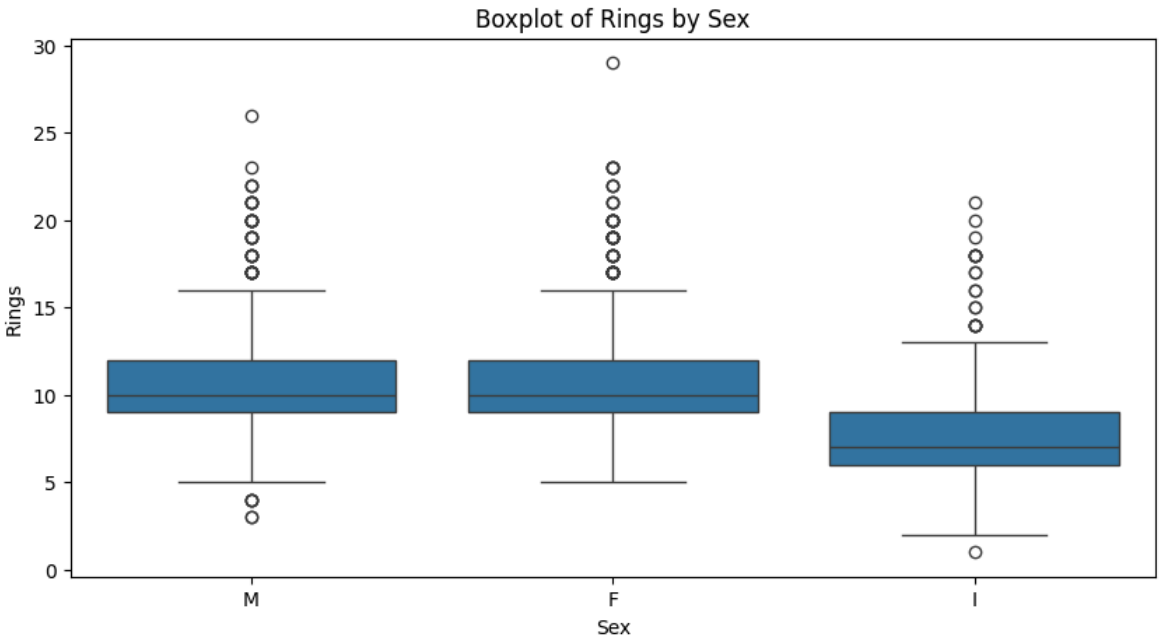
```
plt.title("Feature Correlation Heatmap")
plt.show()
```



In [9]:
```
sns.pairplot(df, hue="Sex")  # Assuming 'Sex' is a column in the dataset
plt.show()
```

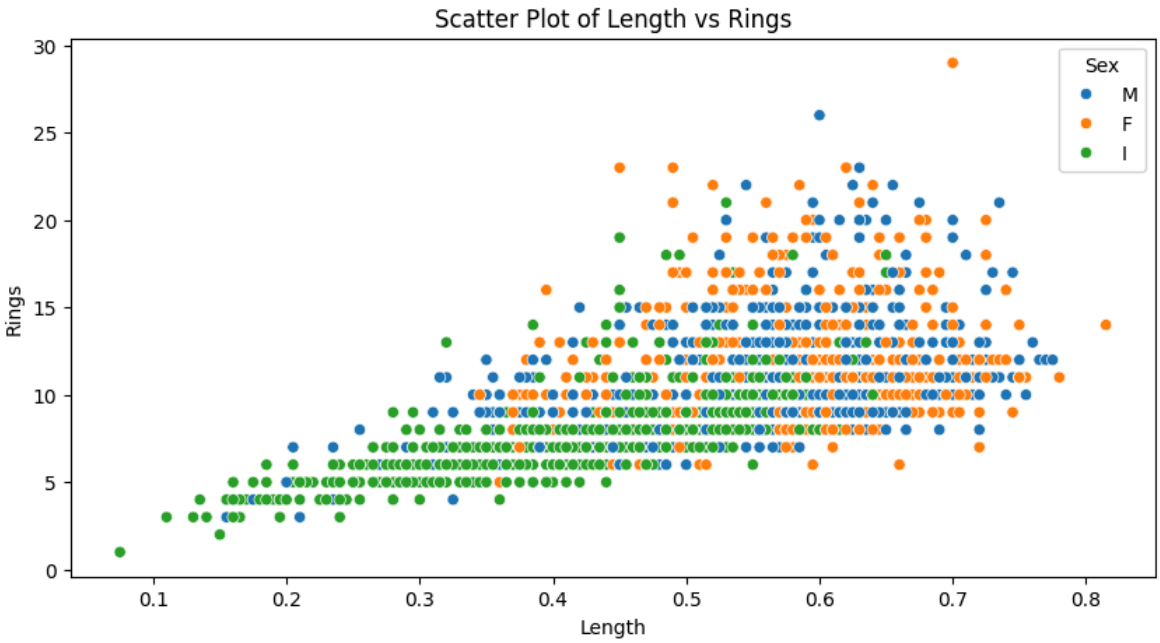


In [10]:
```
plt.figure(figsize=(10, 5))
sns.boxplot(x=df["Sex"], y=df["Rings"])  # Assuming 'Rings' is the target variab
plt.title("Boxplot of Rings by Sex")
plt.show()
```

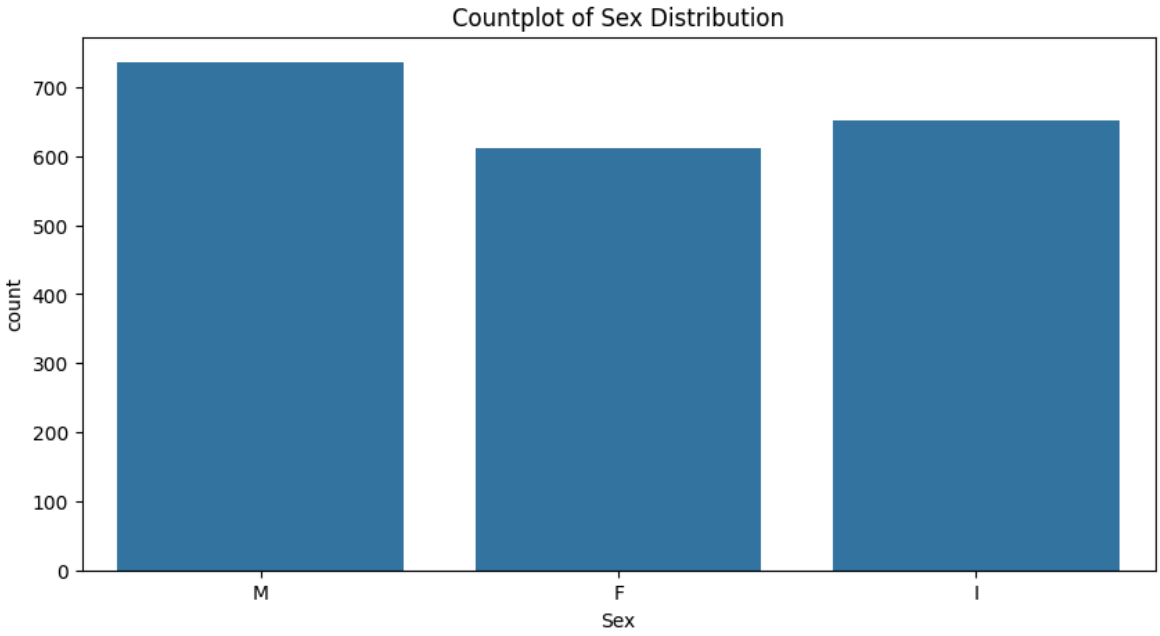


In [11]:
```
plt.figure(figsize=(10, 5))
sns.violinplot(x=df["Sex"], y=df["Rings"])
plt.title("Violin Plot of Rings by Sex")
plt.show()
```

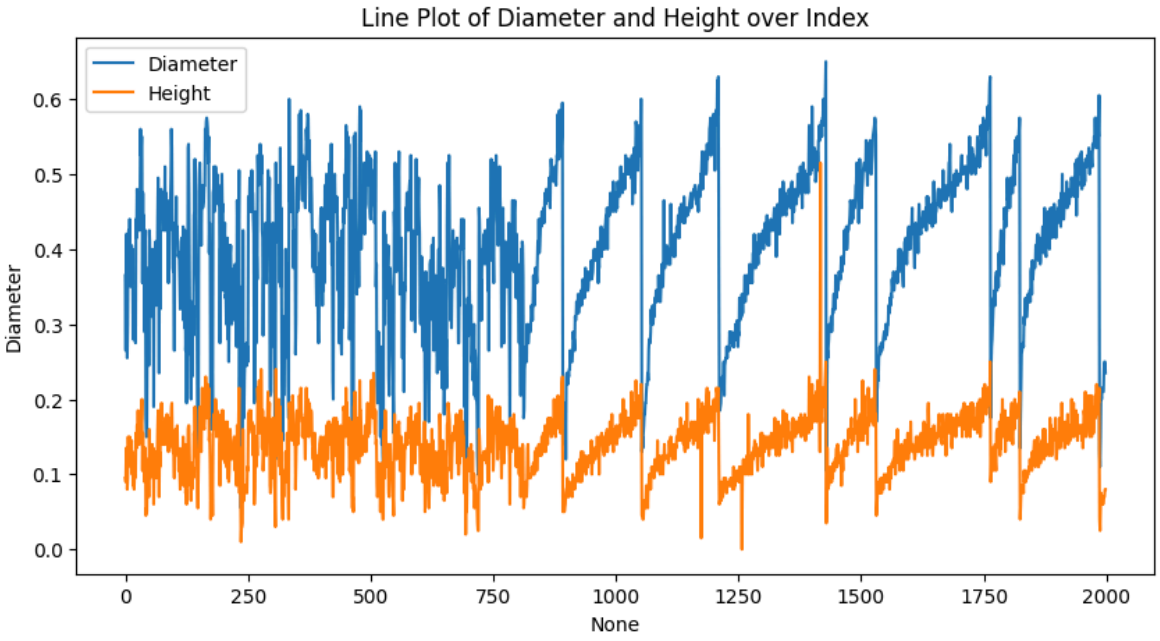
In [12]:
```
plt.figure(figsize=(10, 5))
sns.scatterplot(x=df["Length"], y=df["Rings"], hue=df["Sex"])
plt.title("Scatter Plot of Length vs Rings")
plt.show()
```



In [13]:
```
plt.figure(figsize=(10, 5))
sns.countplot(x=df["Sex"])
plt.title("Countplot of Sex Distribution")
plt.show()
```

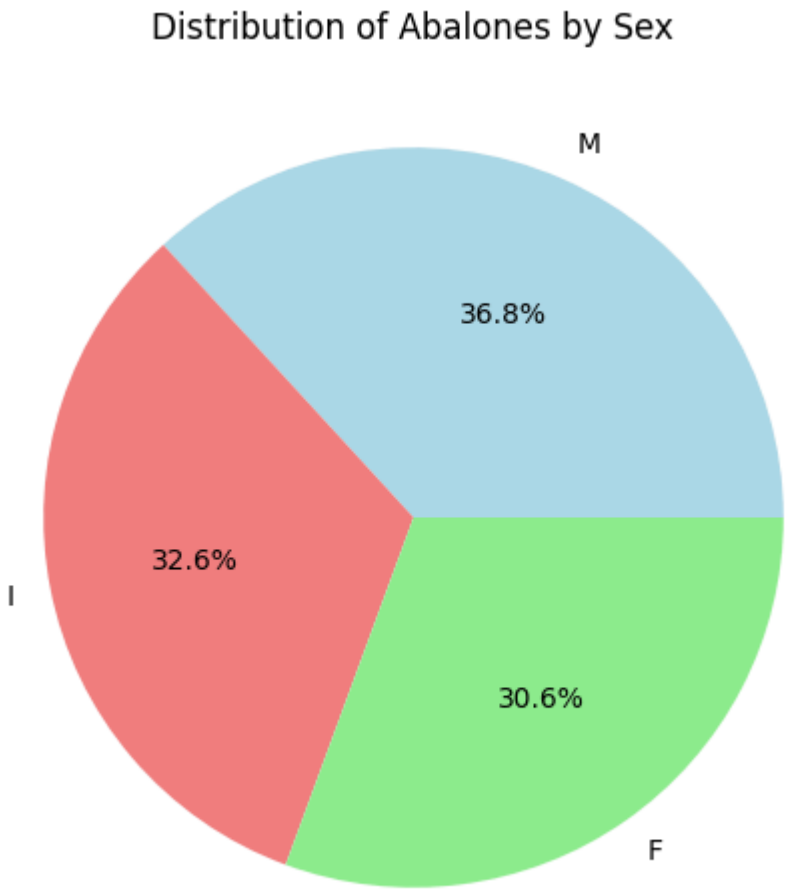


In [29]:
```
plt.figure(figsize=(10, 5))
sns.lineplot(x=df.index, y=df["Diameter"], label="Diameter")
sns.lineplot(x=df.index, y=df["Height"], label="Height")
plt.title("Line Plot of Diameter and Height over Index")
plt.legend()
plt.show()
```



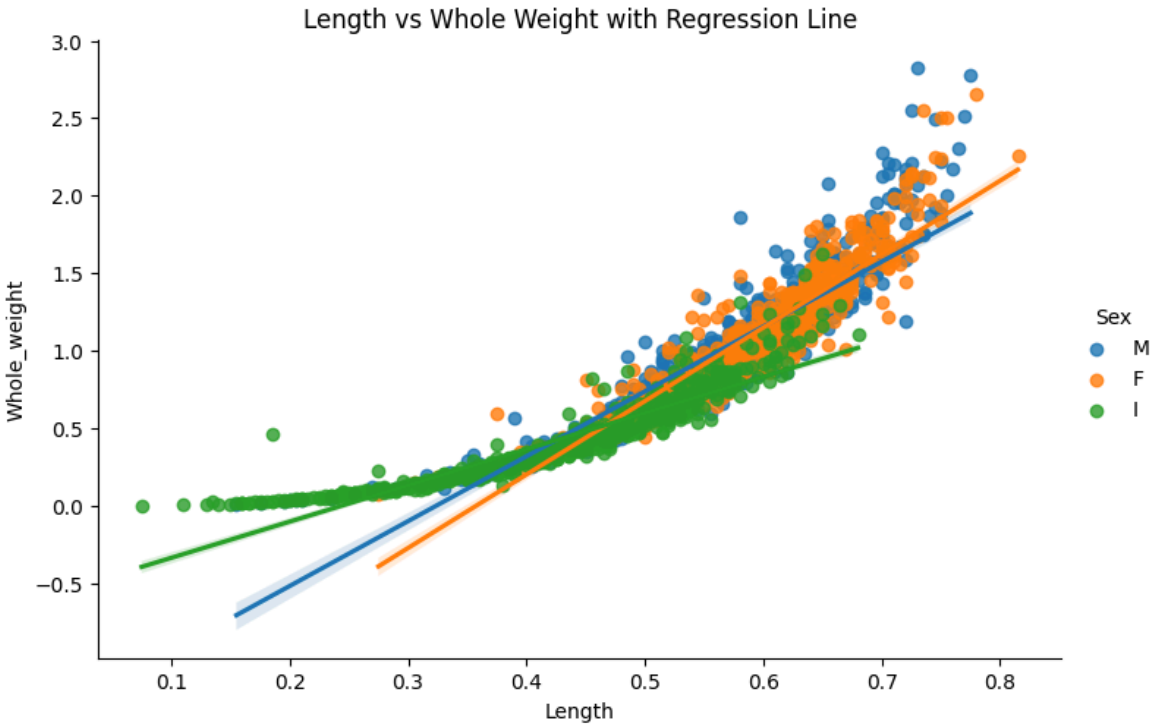
How is the distribution of abalones across different sexes represented in a pie chart?

In [26]:
```
plt.figure(figsize=(6, 6))
df["Sex"].value_counts().plot.pie(autopct="%1.1f%%", colors=["lightblue", "light
plt.title("Distribution of Abalones by Sex")
plt.ylabel("")
plt.show()
```



How does the length of abalones correlate with their whole weight?

In [31]:
```
sns.lmplot(x="Length", y="Whole_weight", hue="Sex", data=df, aspect=1.5)
plt.title("Length vs Whole Weight with Regression Line")
plt.show()
```



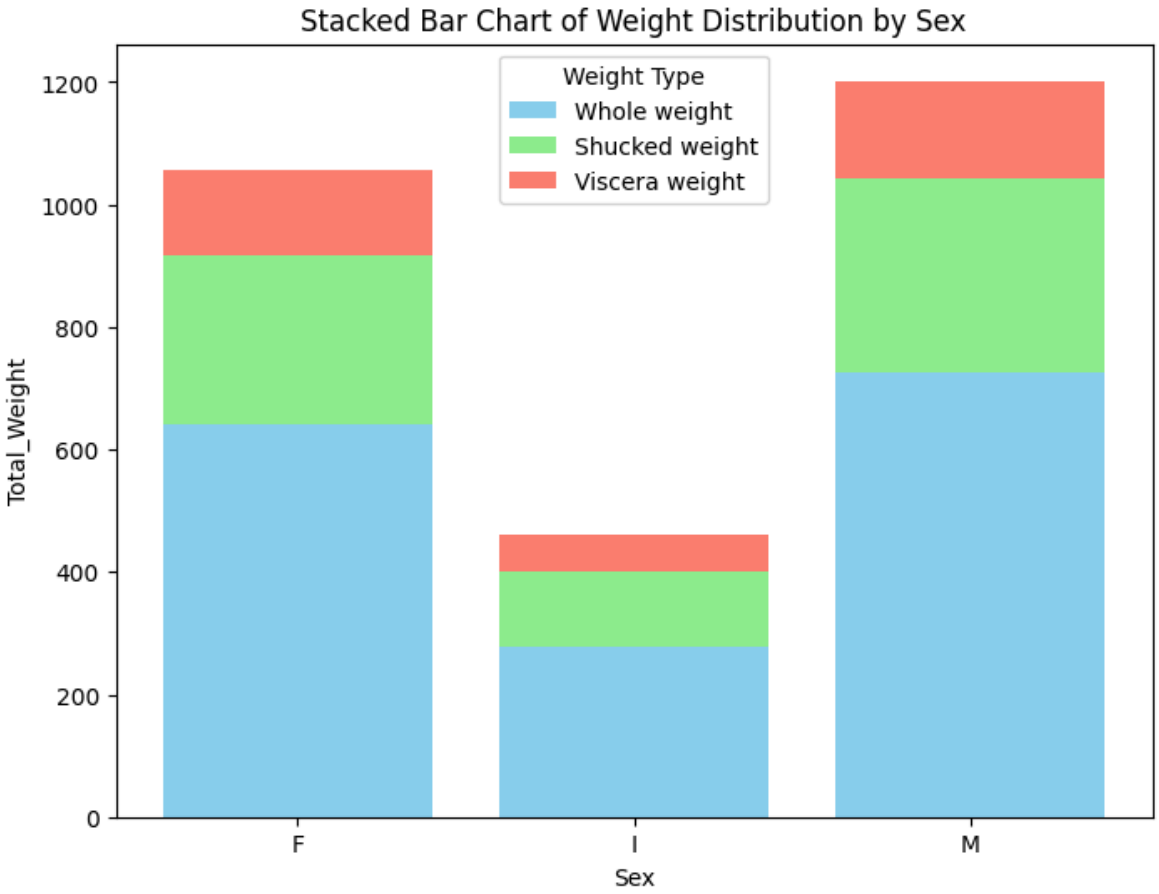
What insights can you draw from the weight distribution among different sexes?

In [37]:
```
grouped_data = df.groupby("Sex")[["Whole_weight", "Shucked_weight", "Viscera_wei

# Extract values for stacked bar chart
sex_categories = grouped_data.index
whole_weight = grouped_data["Whole_weight"]
shucked_weight = grouped_data["Shucked_weight"]
viscera_weight = grouped_data["Viscera_weight"]

# Plot Stacked Bar Chart
plt.figure(figsize=(8, 6))
plt.bar(sex_categories, whole_weight, label='Whole weight', color='skyblue')
plt.bar(sex_categories, shucked_weight, bottom=whole_weight, label='Shucked weig
plt.bar(sex_categories, viscera_weight, bottom=np.add(whole_weight, shucked_weig

# Labels and Title
plt.xlabel("Sex")
plt.ylabel("Total_Weight")
plt.title("Stacked Bar Chart of Weight Distribution by Sex")
plt.legend(title="Weight Type")
plt.show()
```

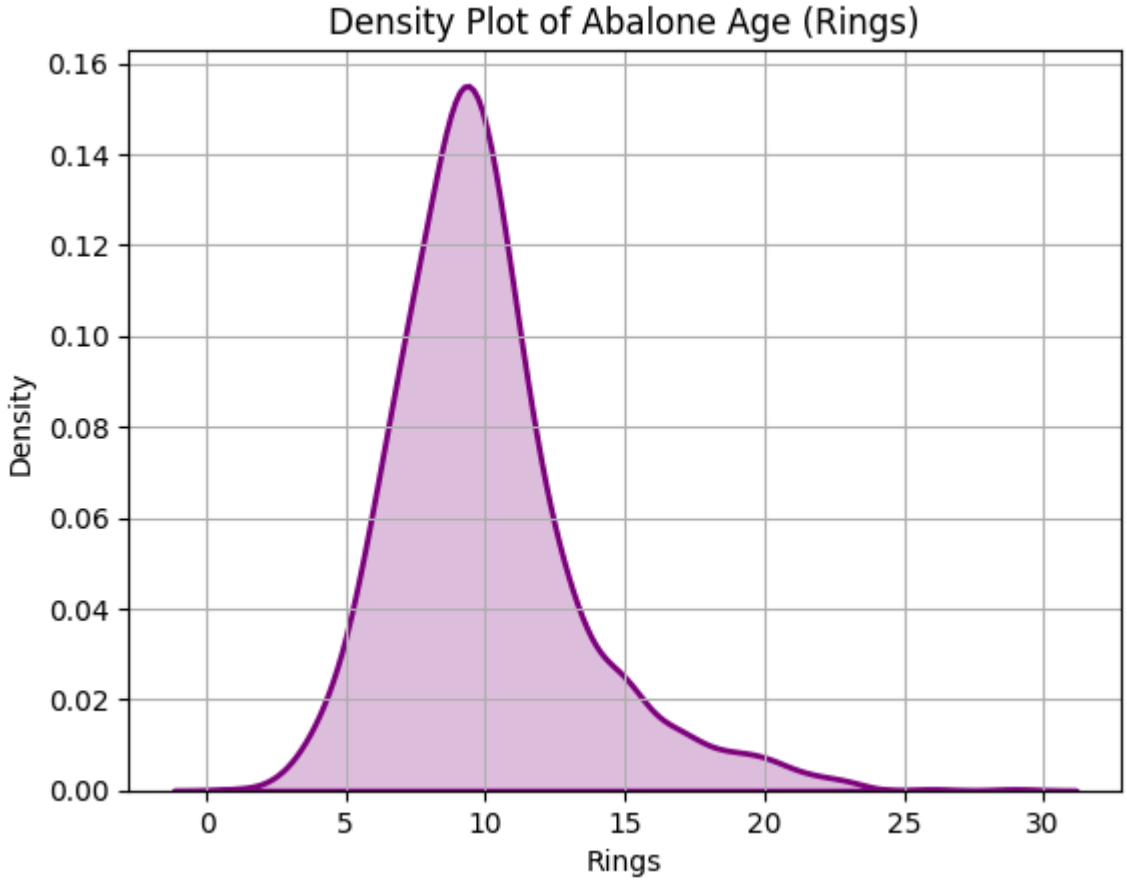


In [7]:
```
sns.kdeplot(data=df, x="Rings", shade=True, color="purple", linewidth=2)
plt.title("Density Plot of Abalone Age (Rings)")
plt.xlabel("Rings")
plt.ylabel("Density")
plt.grid(True)
plt.show()
```

```
C:\Users\SABITHA\AppData\Local\Temp\ipykernel_13924\1480727932.py:1: FutureWarnin
g:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(data=df, x="Rings", shade=True, color="purple", linewidth=2)
```


Density Plot of Abalone Age (Rings)
Density
Rings
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
5
10
15
20
25
30

In [ ]: