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

```
df=pd.read_csv("Titanic-Dataset.csv")
```

```
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	F
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2

```
df.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

```
df.corr()
```

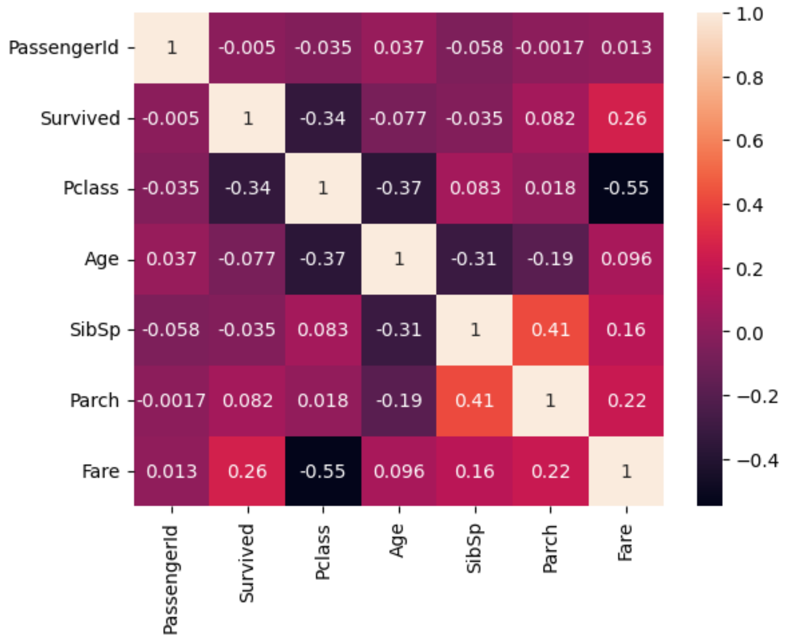
```
    <ipython-input-6-2f6f6606aa2c>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a fu
      df.corr()
                    PassengerId  Survived    Pclass       Age     SibSp     Parch      Fare
```

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

```
    <ipython-input-7-dcd2878cae59>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future ver
      df.corr().Parch.sort_values(ascending=False)
    Parch          1.000000
    SibSp          0.414838
    Fare           0.216225
    Survived       0.081629
    Pclass         0.018443
    PassengerId   -0.001652
    Age           -0.189119
    Name: Parch, dtype: float64
```

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

```
    <ipython-input-8-8df7bcac526d>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a fu
      sns.heatmap(df.corr(),annot=True)
    <Axes: >
```



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

```
    PassengerId      0
    Survived         0
    Pclass           0
    Name             0
    Sex              0
    Age            177
    SibSp            0
    Parch            0
    Ticket           0
    Fare             0
    Cabin          687
    Embarked         2
    dtype: int64
```

```
df.drop(["Cabin"],axis=1,inplace=True)
```

```
mean_age = df['Age'].mean()
mean_age
```

```
    29.69911764705882
```

```
df['Age'].fillna(mean_age, inplace=True)
```

```
mode_embarked = df["Embarked"].mode()[0]
mode_embarked
```

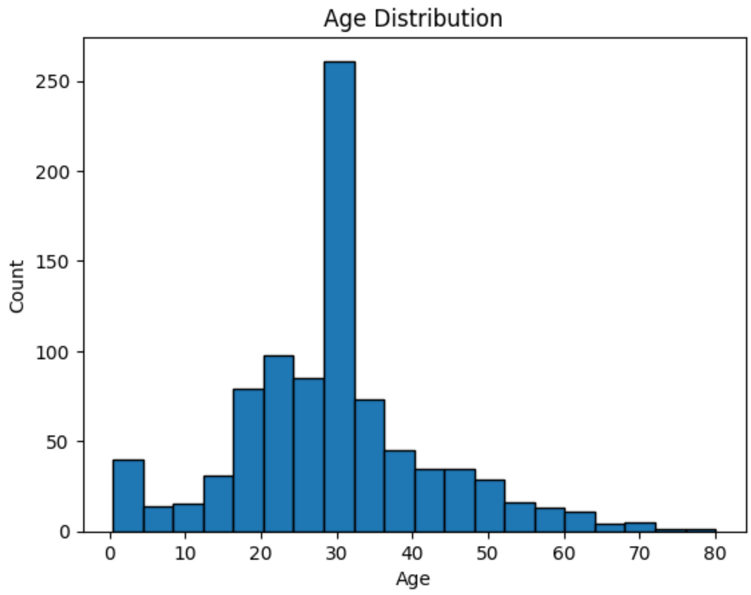
```
    'S'
```

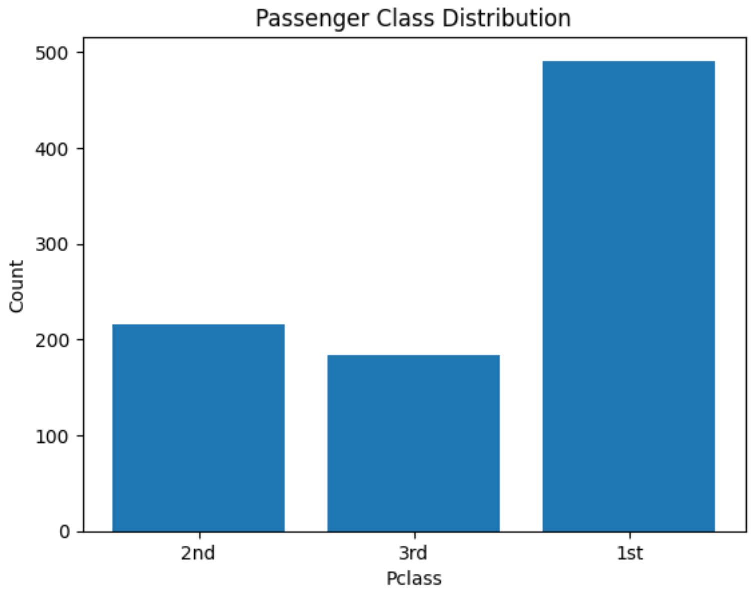
```
df["Embarked"].fillna(mode_embarked,inplace=True)
```

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

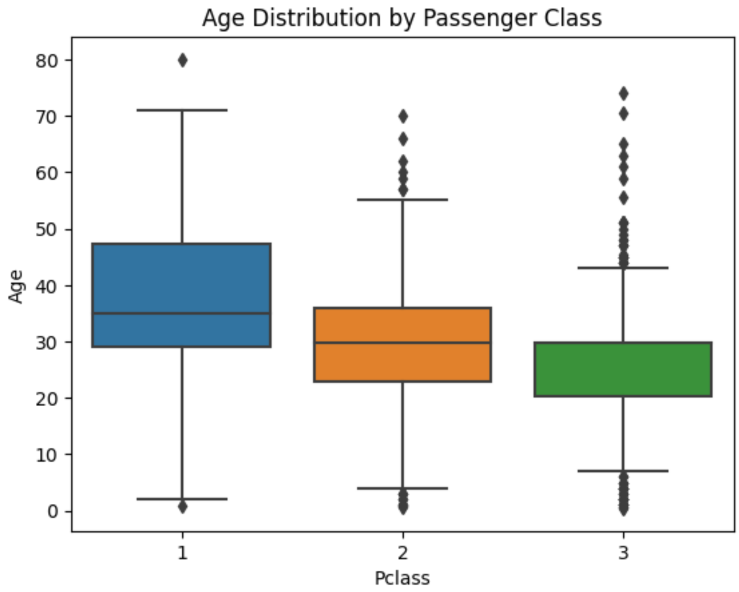
```
    PassengerId    0
    Survived       0
    Pclass         0
    Name           0
    Sex            0
    Age            0
    SibSp          0
    Parch          0
    Ticket         0
    Fare           0
    Embarked       0
    dtype: int64
```

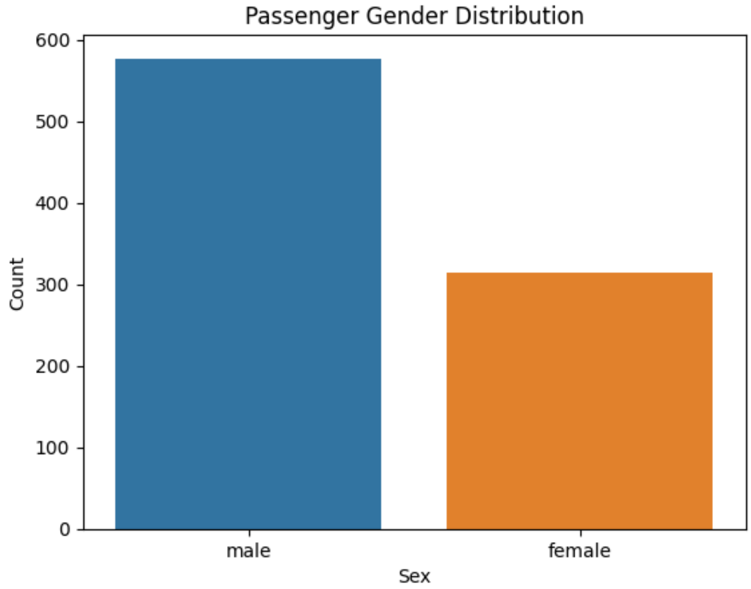
```
# Example 1: Create a histogram of the 'Age' column
plt.hist(df['Age'], bins=20, edgecolor='k')
plt.xlabel('Age')
plt.ylabel('Count')
plt.title('Age Distribution')
plt.show()

# Example 2: Create a bar chart for the 'Pclass' column
pclass_counts = df['Pclass'].value_counts()
plt.bar(pclass_counts.index, pclass_counts.values)
plt.xlabel('Pclass')
plt.ylabel('Count')
plt.title('Passenger Class Distribution')
plt.xticks(pclass_counts.index, labels=['1st', '2nd', '3rd'])
plt.show()
```





```
# Example 1: Create a box plot of 'Age' by 'Pclass'
sns.boxplot(x='Pclass', y='Age', data=df)
plt.xlabel('Pclass')
plt.ylabel('Age')
plt.title('Age Distribution by Passenger Class')
plt.show()

# Example 2: Create a countplot of 'Sex'
sns.countplot(x='Sex', data=df)
plt.xlabel('Sex')
plt.ylabel('Count')
plt.title('Passenger Gender Distribution')
plt.show()
```





```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  891 non-null    int64  
 1   Survived     891 non-null    int64  
 2   Pclass       891 non-null    int64  
 3   Name         891 non-null    object 
 4   Sex          891 non-null    object 
 5   Age          891 non-null    float64
 6   SibSp        891 non-null    int64  
 7   Parch        891 non-null    int64  
 8   Ticket       891 non-null    object 
 9   Fare         891 non-null    float64
 10  Embarked     891 non-null    object 
```

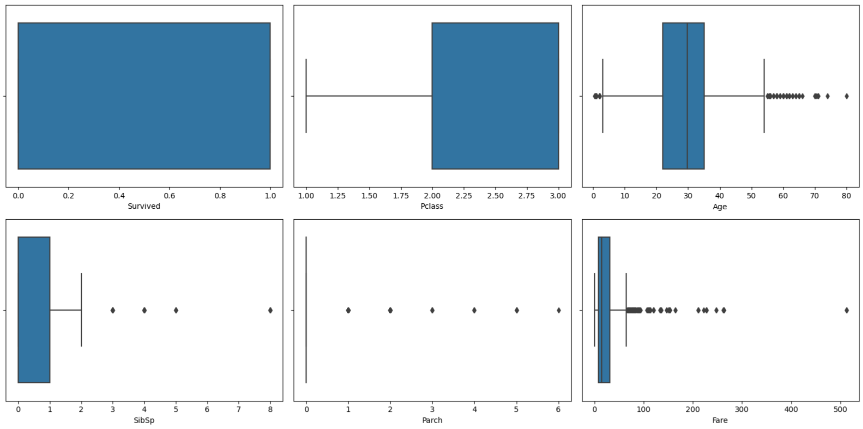
```
dtypes: float64(2), int64(5), object(4)
memory usage: 76.7+ KB
```

```
df.drop(["PassengerId","Name","Ticket"],axis=1,inplace=True)
```

```
df.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.0	1	0	7.2500	S
1	1	1	female	38.0	1	0	71.2833	C
2	1	3	female	26.0	0	0	7.9250	S
3	1	1	female	35.0	1	0	53.1000	S
4	0	3	male	35.0	0	0	8.0500	S

```
# Select numerical columns from the dataset
numerical_attributes = df.select_dtypes(include=['int64', 'float64'])

# Create box plots for each numerical attribute
plt.figure(figsize=(16, 8))  # Adjust the figure size for better visualization

for i, column in enumerate(numerical_attributes.columns):
    plt.subplot(2, 3, i+1)  # Create subplots in a 2x3 grid
    sns.boxplot(x=column, data=numerical_attributes, whis=1.5)  # Adjust whis as needed

plt.tight_layout()
plt.show()
```

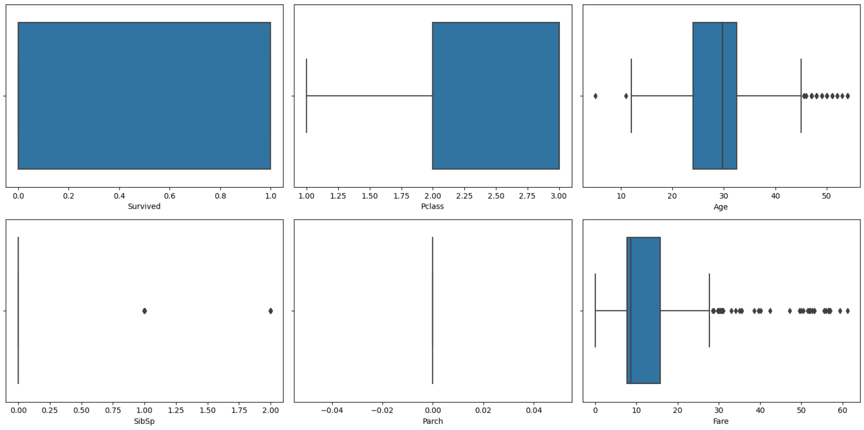


```
# Load your dataset into a pandas DataFrame (assuming your dataset is loaded as 'df')

# Define the numerical attributes
numerical_attributes = ['Age', 'SibSp', 'Parch', 'Fare']

# Define a function to detect and potentially remove outliers
def detect_and_remove_outliers(df, column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    outliers = df[(df[column] < lower_bound) | (df[column] > upper_bound)]
    return outliers

# Detect and potentially remove outliers for each numerical attribute
outliers_dict = {}
```

```
for attribute in numerical_attributes:
    outliers = detect_and_remove_outliers(df, attribute)
    outliers_dict[attribute] = outliers

# Print the detected outliers for each numerical attribute
for attribute, outliers in outliers_dict.items():
    print(f"Outliers in {attribute}:")
    print(outliers)

# Optionally, remove the outliers from the DataFrame
for attribute, outliers in outliers_dict.items():
    df = df[~df.index.isin(outliers.index)]
```

```
Outliers in Age:
     Survived  Pclass     Sex    Age  SibSp  Parch     Fare Embarked
7           0       3    male   2.00      3      1  21.0750        S
11          1       1  female  58.00      0      0  26.5500        S
15          1       2  female  55.00      0      0  16.0000        S
16          0       3    male   2.00      4      1  29.1250        Q
33          0       2    male  66.00      0      0  10.5000        S
..        ...     ...     ...    ...    ...    ...      ...      ...
827         1       2    male   1.00      0      2  37.0042        C
829         1       1  female  62.00      0      0  80.0000        S
831         1       2    male   0.83      1      1  18.7500        S
851         0       3    male  74.00      0      0   7.7750        S
879         1       1  female  56.00      0      1  83.1583        C

[66 rows x 8 columns]
Outliers in SibSp:
     Survived  Pclass     Sex        Age  SibSp  Parch      Fare Embarked
7           0       3    male   2.000000      3      1   21.0750        S
16          0       3    male   2.000000      4      1   29.1250        Q
24          0       3  female   8.000000      3      1   21.0750        S
27          0       1    male  19.000000      3      2  263.0000        S
50          0       3    male   7.000000      4      1   39.6875        S
59          0       3    male  11.000000      5      2   46.9000        S
63          0       3    male   4.000000      3      2   27.9000        S
68          1       3  female  17.000000      4      2    7.9250        S
71          0       3  female  16.000000      5      2   46.9000        S
85          1       3  female  33.000000      3      0   15.8500        S
88          1       1  female  23.000000      3      2  263.0000        S
119         0       3  female   2.000000      4      2   31.2750        S
159         0       3    male  29.699118      8      2   69.5500        S
164         0       3    male   1.000000      4      1   39.6875        S
171         0       3    male   4.000000      4      1   29.1250        Q
176         0       3    male  29.699118      3      1   25.4667        S
180         0       3  female  29.699118      8      2   69.5500        S
182         0       3    male   9.000000      4      2   31.3875        S
201         0       3    male  29.699118      8      2   69.5500        S
229         0       3  female  29.699118      3      1   25.4667        S
233         1       3  female   5.000000      4      2   31.3875        S
261         1       3    male   3.000000      4      2   31.3875        S
266         0       3    male  16.000000      4      1   39.6875        S
278         0       3    male   7.000000      4      1   29.1250        Q
324         0       3    male  29.699118      8      2   69.5500        S
341         1       1  female  24.000000      3      2  263.0000        S
374         0       3  female   3.000000      3      1   21.0750        S
386         0       3    male   1.000000      5      2   46.9000        S
409         0       3  female  29.699118      3      1   25.4667        S
480         0       3    male   9.000000      5      2   46.9000        S
485         0       3  female  29.699118      3      1   25.4667        S
541         0       3  female   9.000000      4      2   31.2750        S
542         0       3  female  11.000000      4      2   31.2750        S
634         0       3  female   9.000000      3      2   27.9000        S
642         0       3  female   2.000000      3      2   27.9000        S
683         0       3    male  14.000000      5      2   46.9000        S
686         0       3    male  14.000000      4      1   39.6875        S
726         1       2  female  30.000000      3      0   21.0000        S
787         0       3    male   8.000000      4      1   29.1250        Q
792         0       3  female  29.699118      8      2   69.5500        S
813         0       3  female   6.000000      4      2   31.2750        S
```

```
import matplotlib.pyplot as plt
import seaborn as sns

# Select numerical columns from the dataset
numerical_attributes = df.select_dtypes(include=['int64', 'float64'])

# Create box plots for each numerical attribute
plt.figure(figsize=(16, 8))  # Adjust the figure size for better visualization

for i, column in enumerate(numerical_attributes.columns):
    plt.subplot(2, 3, i+1)  # Create subplots in a 2x3 grid
    sns.boxplot(x=column, data=numerical_attributes, whis=1.5)  # Adjust whis as needed

plt.tight_layout()
plt.show()
```



```
df.head()
```

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.000000	1	0	7.2500	S
2	1	3	female	26.000000	0	0	7.9250	S
3	1	1	female	35.000000	1	0	53.1000	S
4	0	3	male	35.000000	0	0	8.0500	S
5	0	3	male	29.699118	0	0	8.4583	Q

```
# Assuming 'df' is your DataFrame containing the dataset
X = df.drop('Survived', axis=1)  # Independent variables
y = df['Survived']  # Dependent variable
```

```
X.head()
```

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	male	22.000000	1	0	7.2500	S
2	3	female	26.000000	0	0	7.9250	S
3	1	female	35.000000	1	0	53.1000	S
4	3	male	35.000000	0	0	8.0500	S
5	3	male	29.699118	0	0	8.4583	Q

```
y.head()
```

```
0    0
2    1
3    1
4    0
5    0
Name: Survived, dtype: int64
```

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
X["Sex"]=le.fit_transform(X["Sex"])
Embarked=pd.get_dummies(X["Embarked"],drop_first=True)
Embarked
```

	Q	S
0	0	1
2	0	1
3	0	1
4	0	1
5	1	0
...	...	...
884	0	1
886	0	1
887	0	1
889	0	0
890	1	0

577 rows × 2 columns

```
X=pd.concat([X,Embarked],axis=1)
X.drop(["Embarked"],axis=1,inplace=True)
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=2)
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

```
((461, 8), (116, 8), (461,), (116,))
```

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
x_train=sc.fit_transform(x_train)
x_test=sc.fit_transform(x_test)
x_train
```

```
array([[ 0.6931394 , -1.76523977,  0.0155626 , ..., -0.04945919,
        -0.34879005, -1.74466606],
       [ 0.6931394 ,  0.56649528, -1.4470116 , ..., -0.5034821 ,
        -0.34879005,  0.57317559],
       [ 0.6931394 ,  0.56649528,  0.16538695, ..., -0.5612962 ,
        -0.34879005,  0.57317559],
       ...,
       [ 0.6931394 ,  0.56649528,  0.0155626 , ...,  0.7106139 ,
         2.86705424, -1.74466606],
       [-2.03794831,  0.56649528,  0.0155626 , ..., -1.18255274,
        -0.34879005,  0.57317559],
       [ 0.6931394 , -1.76523977,  0.0155626 , ..., -0.57501479,
         2.86705424, -1.74466606]])
```

```
x_test
```

```
        -0.23462747, -0.37047929,  0.59062442],
       [ 0.62104163,  0.60390884, -0.05807902, -0.39103094,  0.        ,
        -0.59344272, -0.37047929, -1.69312335],
       [ 0.62104163,  0.60390884, -1.09122365, -0.39103094,  0.        ,
        -0.60183606, -0.37047929,  0.59062442],
       [-2.09748023,  0.60390884, -0.09693589, -0.39103094,  0.        ,
         1.40207365, -0.37047929,  0.59062442],
       [ 0.62104163,  0.60390884, -0.09693589, -0.39103094,  0.        ,
        -0.5371402 , -0.37047929,  0.59062442],
       [ 0.62104163,  0.60390884, -1.73693904, -0.39103094,  0.        ,
        -0.60183606, -0.37047929,  0.59062442],
       [ 0.62104163,  0.60390884,  1.10420868, -0.39103094,  0.        ,
         0.82712992, -0.37047929,  0.59062442],
       [ 0.62104163, -1.65587907, -0.09693589, -0.39103094,  0.        ,
        -0.53853349,  2.69920623, -1.69312335],
       [ 0.62104163,  0.60390884,  0.20020713, -0.39103094,  0.        ,
        -0.49796849, -0.37047929,  0.59062442],
       [ 0.62104163, -1.65587907, -1.3495098 , -0.39103094,  0.        ,
        -0.47278847, -0.37047929,  0.59062442],
       [ 0.62104163, -1.65587907, -0.09693589, -0.39103094,  0.        ,
        -0.54937769,  2.69920623, -1.69312335],
       [ 0.62104163,  0.60390884,  2.26649639, -0.39103094,  0.        ,
        -0.54063183, -0.37047929,  0.59062442],
       [-2.09748023,  0.60390884, -0.09693589, -0.39103094,  0.        ,
         2.12390081, -0.37047929, -1.69312335],
       [-0.7382193 , -1.65587907,  1.49163792,  1.87694853,  0.        ,
         0.98240669, -0.37047929,  0.59062442],
       [ 0.62104163, -1.65587907, -1.09122365, -0.39103094,  0.        ,
        -0.37416673, -0.37047929,  0.59062442],
       [-2.09748023,  0.60390884,  1.23335176, -0.39103094,  0.        ,
        -1.19986147, -0.37047929,  0.59062442],
       [ 0.62104163,  0.60390884, -1.3495098 , -0.39103094,  0.        ,
        -0.52419767, -0.37047929,  0.59062442],
       [-0.7382193 ,  0.60390884, -1.47865288, -0.39103094,  0.        ,
        -0.31856086, -0.37047929,  0.59062442],
       [ 0.62104163,  0.60390884, -1.22036673, -0.39103094,  0.        ,
        -0.54063183, -0.37047929,  0.59062442],
       [ 0.62104163, -1.65587907,  0.00649252, -0.39103094,  0.        ,
        -0.54937769,  2.69920623, -1.69312335],
       [-2.09748023,  0.60390884,  1.49163792,  1.87694853,  0.        ,
         3.21119074, -0.37047929,  0.59062442],
       [-0.7382193 ,  0.60390884,  3.04135486, -0.39103094,  0.        ,
         0.98240669, -0.37047929,  0.59062442]])
```

✓ 0s completed at 10:03 PM