In [1]:
```python
# Exploratory Data Analysis(EDA) on Titanic dataset.
```

In [2]:
```python
#Importing Necessary libraries

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

In [3]:
```python
#Importing train dataset

df1 = pd.read_csv('C:/Users/ankit/Desktop/elevate labs/task - 5/titanic/train.csv')
```

In [4]:
```python
#Importing test dataset

df2 = pd.read_csv('C:/Users/ankit/Desktop/elevate labs/task - 5/titanic/test.csv')
```

In [5]:
```python
#Checking first five rows of train dataset

df1.head()
```

Out[5]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [6]:
```python
#Checking first five rows of test dataset

df2.head()
```

Out[6]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

In [7]:
```python
#Appending both the datasets to get one single dataset

df_titanic = pd.concat([df1,df2],ignore_index = True)
```

In [8]:
```python
#Getting overview of full titanic dataset

df_titanic.head()
```

Out[8]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [9]:
```python
#Listing down all the columns
df_titanic.columns.values
```

Out[9]:
```
array(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'], dtype=object)
```

In [10]:
```
#Getting the high level overview of the dataset.

df_titanic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  1309 non-null   int64  
 1   Survived     1309 non-null   int64  
 2   Pclass       1309 non-null   int64  
 3   Name         1309 non-null   object 
 4   Sex          1309 non-null   object 
 5   Age          1046 non-null   float64
 6   SibSp        1309 non-null   int64  
 7   Parch        1309 non-null   int64  
 8   Ticket       1309 non-null   object 
 9   Fare         1308 non-null   float64
 10  Cabin        295 non-null    object 
 11  Embarked     1307 non-null   object 
dtypes: float64(2), int64(5), object(5)
memory usage: 122.8+ KB
```

In [11]:
```
#Checking total null values present in each column of dataset

df_titanic.isnull().sum()
```

Out[11]:
```
PassengerId       0
Survived          0
Pclass            0
Name              0
Sex               0
Age             263
SibSp             0
Parch             0
Ticket            0
Fare              1
Cabin          1014
Embarked          2
dtype: int64
```

In [12]:
```
#Few Conclusions:
# 1) There are missing values in Age, Cabin, Fare and Embarked columns
# 2) More than 75% values are missing from Cabin column, so will be dropping it.
# 3) Few columns have inappropriate datatype
```

In [13]:
```
#Dropping Cabin column
df_titanic.drop(columns = ['Cabin'], inplace=True)
```

In [14]:
```
df_titanic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  1309 non-null   int64  
 1   Survived     1309 non-null   int64  
 2   Pclass       1309 non-null   int64  
 3   Name         1309 non-null   object 
 4   Sex          1309 non-null   object 
 5   Age          1046 non-null   float64
 6   SibSp        1309 non-null   int64  
 7   Parch        1309 non-null   int64  
 8   Ticket       1309 non-null   object 
 9   Fare         1308 non-null   float64
 10  Embarked     1307 non-null   object 
dtypes: float64(2), int64(5), object(4)
memory usage: 112.6+ KB
```

In [15]:
```
#Imputing the missing values in age by using mean of age column

df_titanic['Age'].fillna(df_titanic['Age'].mean(), inplace=True)
```

In [16]:
```
df_titanic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  1309 non-null   int64  
 1   Survived     1309 non-null   int64  
 2   Pclass       1309 non-null   int64  
 3   Name         1309 non-null   object 
 4   Sex          1309 non-null   object 
 5   Age          1309 non-null   float64
 6   SibSp        1309 non-null   int64  
 7   Parch        1309 non-null   int64  
 8   Ticket       1309 non-null   object 
 9   Fare         1308 non-null   float64
 10  Embarked     1307 non-null   object 
dtypes: float64(2), int64(5), object(4)
memory usage: 112.6+ KB
```

In [17]:
```
#Imputing the missing value in fare column by using mean of fare column

df_titanic['Fare'].fillna(df_titanic['Fare'].mean(), inplace=True)
```

In [18]:
```
#Checking the frequency of values in Embarked column

df_titanic['Embarked'].value_counts()
```

Out[18]:
```
Embarked
S    914
C    270
Q    123
Name: count, dtype: int64
```

In [19]:
```
#S is the most occured value, So Replacing the null value from Embarked column to 'S'

df_titanic['Embarked'].fillna('S', inplace = True)
```

In [20]:
```
#Again checking the overview of dataset
df_titanic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  1309 non-null   int64  
 1   Survived     1309 non-null   int64  
 2   Pclass       1309 non-null   int64  
 3   Name         1309 non-null   object 
 4   Sex          1309 non-null   object 
 5   Age          1309 non-null   float64
 6   SibSp        1309 non-null   int64  
 7   Parch        1309 non-null   int64  
 8   Ticket       1309 non-null   object 
 9   Fare         1309 non-null   float64
 10  Embarked     1309 non-null   object 
dtypes: float64(2), int64(5), object(4)
memory usage: 112.6+ KB
```

In [21]:
```
#Checking any null value in dataset
df_titanic.isnull().sum()
```

Out[21]:
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

In [22]:
```
#Describing the dataset

df_titanic.describe()
```

Out[22]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	1309.000000	1309.000000	1309.000000	1309.000000	1309.000000	1309.000000	1309.000000
mean	655.000000	0.377387	2.294882	29.881138	0.498854	0.385027	33.295479
std	378.020061	0.484918	0.837836	12.883193	1.041658	0.865560	51.738879
min	1.000000	0.000000	1.000000	0.170000	0.000000	0.000000	0.000000
25%	328.000000	0.000000	2.000000	22.000000	0.000000	0.000000	7.895800
50%	655.000000	0.000000	3.000000	29.881138	0.000000	0.000000	14.454200
75%	982.000000	1.000000	3.000000	35.000000	1.000000	0.000000	31.275000
max	1309.000000	1.000000	3.000000	80.000000	8.000000	9.000000	512.329200

In [23]:
```
#Checking the distribution of 'Sibsp' and 'Parch' column as they are categorical column
df_titanic['SibSp'].value_counts()
```

Out[23]:
```
SibSp
0    891
1    319
2     42
4     22
3     20
8      9
5      6
Name: count, dtype: int64
```

In [24]:
```
df_titanic['Parch'].value_counts()
```

Out[24]:
```
Parch
0    1002
1     170
2     113
3       8
5       6
4       6
6       2
9       2
Name: count, dtype: int64
```

In [25]:
```
#Changing the datatype of columns
#Choosing the category datatype as these columns have fixed number of possible values
df_titanic['Survived'] = df_titanic['Survived'].astype('category')
df_titanic['Pclass'] = df_titanic['Pclass'].astype('category')
df_titanic['Sex'] = df_titanic['Sex'].astype('category')
df_titanic['Age'] = df_titanic['Age'].astype('int')
df_titanic['Embarked'] = df_titanic['Embarked'].astype('category')
```

In [26]:
```
#Checking the datatype of all the columns
df_titanic.dtypes
```

Out[26]:
```
PassengerId       int64
Survived       category
Pclass         category
Name             object
Sex            category
Age               int32
SibSp             int64
Parch             int64
Ticket           object
Fare            float64
Embarked       category
dtype: object
```

In [27]:
```
#Final checking of dataset
df_titanic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  1309 non-null   int64
 1   Survived     1309 non-null   category
 2   Pclass       1309 non-null   category
 3   Name         1309 non-null   object
 4   Sex          1309 non-null   category
 5   Age          1309 non-null   int32
 6   SibSp        1309 non-null   int64
 7   Parch        1309 non-null   int64
 8   Ticket       1309 non-null   object
 9   Fare         1309 non-null   float64
 10  Embarked     1309 non-null   category
dtypes: category(4), float64(1), int32(1), int64(3), object(2)
memory usage: 72.2+ KB
```

In [28]:
```
#So we have no missing values, all columns have proper datatype.
```

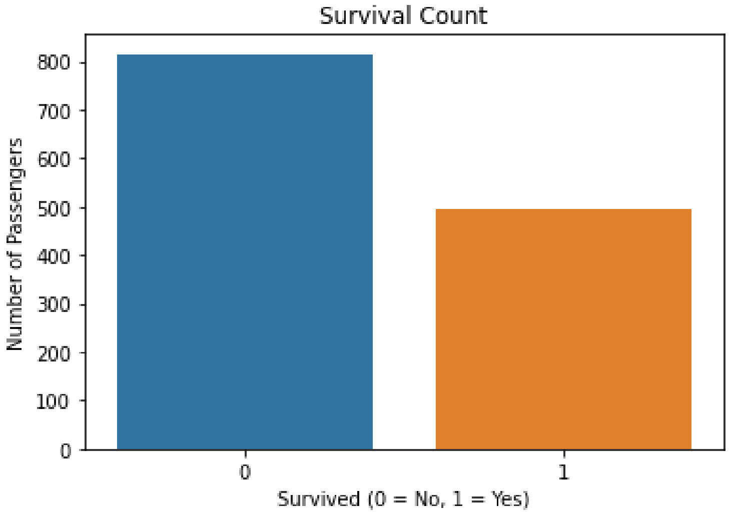
In [29]:
```
df_titanic.describe()
```

Out[29]:

	PassengerId	Age	SibSp	Parch	Fare
count	1309.000000	1309.000000	1309.000000	1309.000000	1309.000000
mean	655.000000	29.685256	0.498854	0.385027	33.295479
std	378.020061	12.899824	1.041658	0.865560	51.738879
min	1.000000	0.000000	0.000000	0.000000	0.000000
25%	328.000000	22.000000	0.000000	0.000000	7.895800
50%	655.000000	29.000000	0.000000	0.000000	14.454200
75%	982.000000	35.000000	1.000000	0.000000	31.275000
max	1309.000000	80.000000	8.000000	9.000000	512.329200

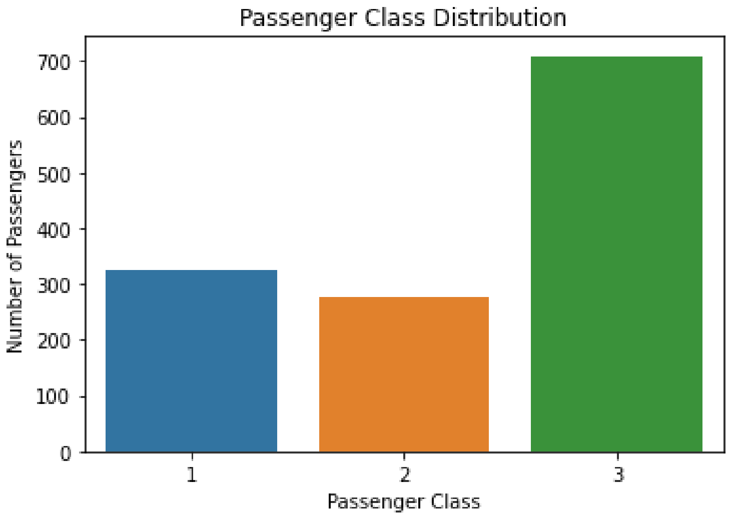
In [30]:
```
#We can get following insights from this:
# 1) 25% of the population were below 22 years of age, while 75% of the population were below 35 years of age. So, mostly
#    young people were travelling in the titanic.
# 2) Average fare value was 33.29
```

In [31]:
```
#Univariate Analysis
# Checking the survived column

sns.countplot(data = df_titanic, x='Survived')
plt.title("Survival Count")
plt.xlabel('Survived (0 = No, 1 = Yes)')
plt.ylabel('Number of Passengers')
plt.show()

death_percent = round((df_titanic['Survived'].value_counts().values[0]/1309)*100)
total_passengers = len(df_titanic)
print(f'Out of {total_passengers} people, {death_percent}% people died in accident.')
```



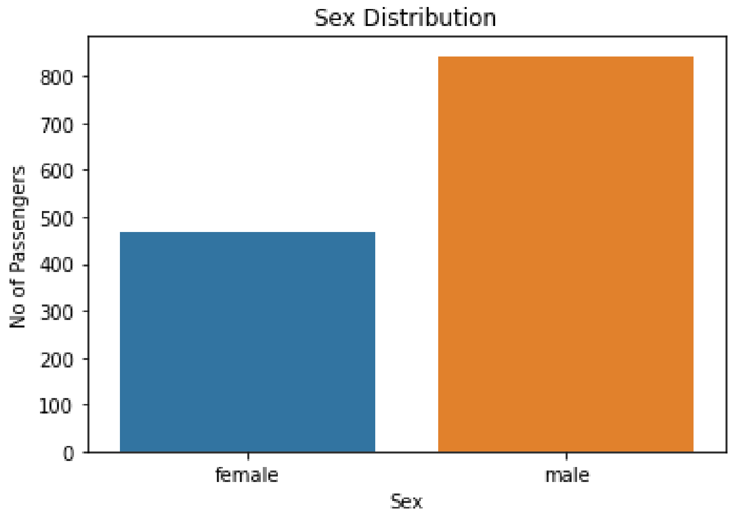

```
Out of 1309 people, 62% people died in accident.
```

In [32]:
```
#Checking Pclass column

sns.countplot(x='Pclass', data=df_titanic)
plt.title('Passenger Class Distribution')
plt.xlabel('Passenger Class')
plt.ylabel('Number of Passengers')
plt.show()

#Conclusion: Pclass 3 was the most crowded class.
```



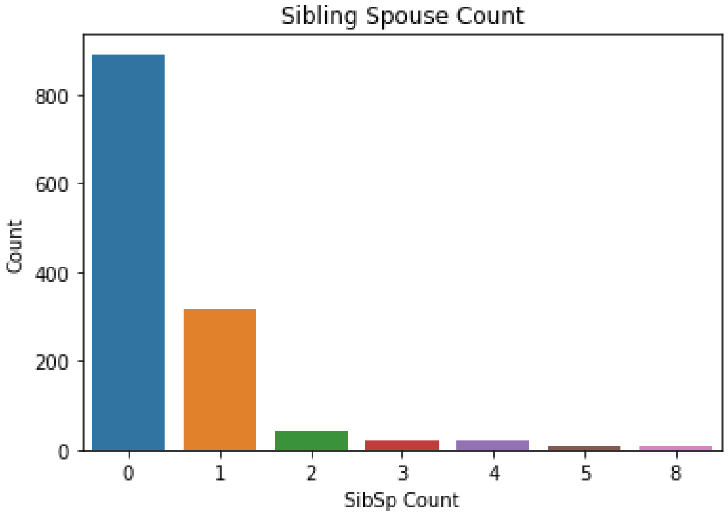
In [33]:
```
#Checking sex column
print((df_titanic['Sex'].value_counts()/1309)*100)

sns.countplot(x = 'Sex', data = df_titanic)
plt.title("Sex Distribution")
plt.xlabel('Sex')
plt.ylabel('No of Passengers')
plt.show()

#Conclusion: Male passengers are approximately 64% of total passengers.
```

```
Sex
male      64.400306
female    35.599694
Name: count, dtype: float64
```



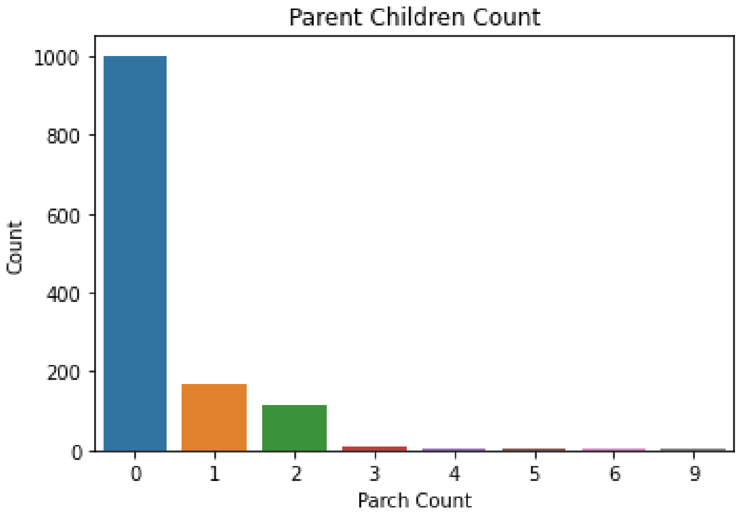
In [34]:
```
#Checking Sibsp column

print(df_titanic['SibSp'].value_counts())

sns.countplot(x='SibSp', data=df_titanic)
plt.title("Sibling Spouse Count")
plt.xlabel("SibSp Count")
plt.ylabel("Count")
plt.show()
```

```
SibSp
0    891
1    319
2     42
4     22
3     20
8      9
5      6
Name: count, dtype: int64
```



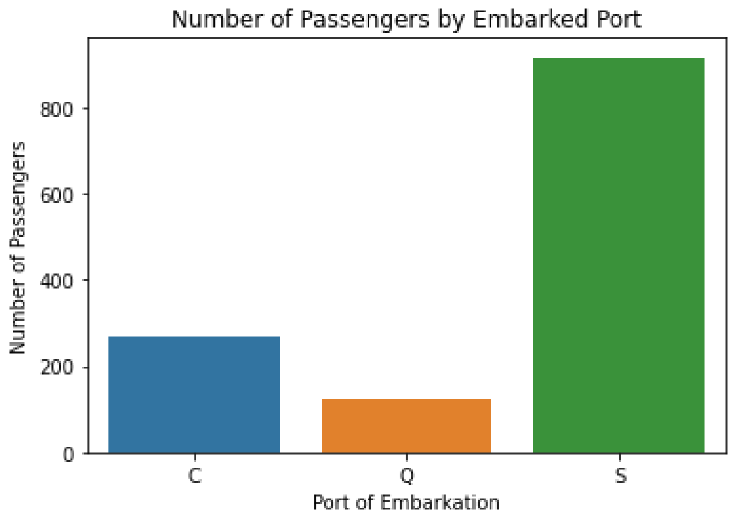
In [35]:
```python
#Checking Parch column

print(df_titanic['Parch'].value_counts()/1309*100)

sns.countplot(x='Parch', data=df_titanic)
plt.title("Parent Children Count")
plt.xlabel("Parch Count")
plt.ylabel("Count")
plt.show()
```

```
Parch
0    76.546982
1    12.987013
2     8.632544
3     0.611154
5     0.458365
4     0.458365
6     0.152788
9     0.152788
Name: count, dtype: float64
```

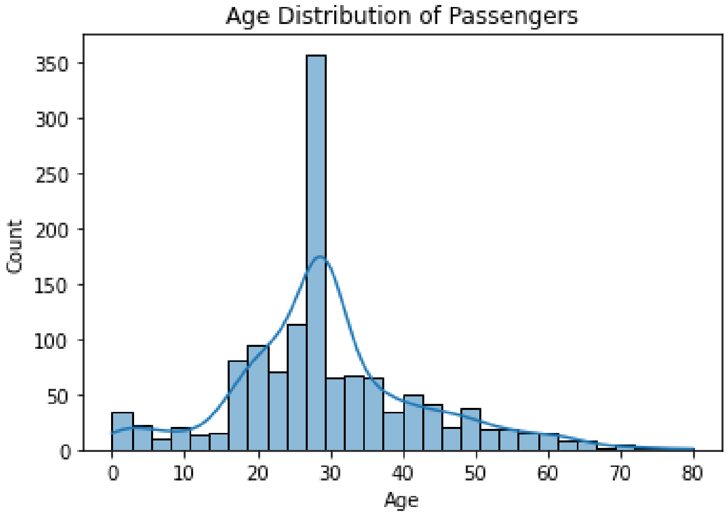


In [36]:
```
#Checking Embarked column

print(df_titanic['Embarked'].value_counts()/1309*100)

sns.countplot(x='Embarked', data=df_titanic)
plt.title('Number of Passengers by Embarked Port')
plt.xlabel('Port of Embarkation')
plt.ylabel('Number of Passengers')
plt.show()

#Conclusion: Almost 70% passengers were travelling to Southampton port.
```

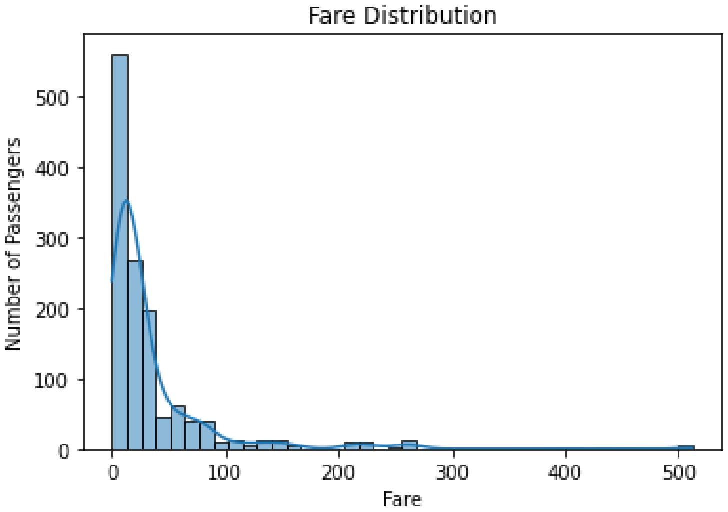
```
Embarked
S    69.977082
C    20.626432
Q     9.396486
Name: count, dtype: float64
```



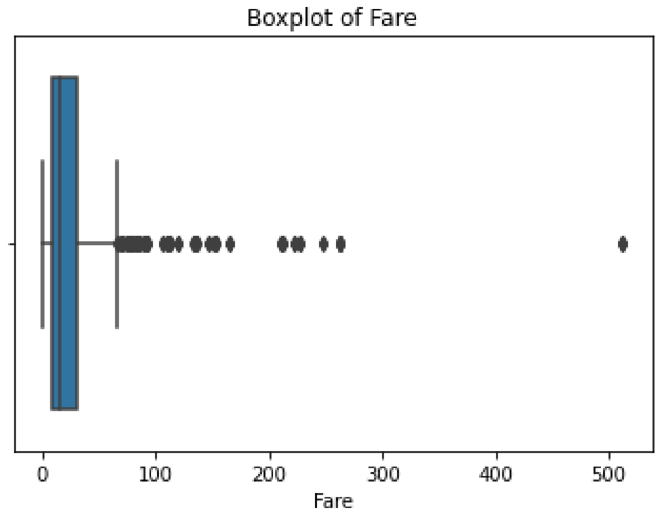
In [37]:
```
#Checking Age column

sns.histplot(df_titanic['Age'], bins=30, kde=True)
plt.title('Age Distribution of Passengers')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()

#Conclusion: Most of the passengers were between 20 years to 40 years of age.
```



In [38]:
```
#Checking Fare column

sns.histplot(df_titanic['Fare'], bins=40, kde=True)
plt.title('Fare Distribution')
plt.xlabel('Fare')
plt.ylabel('Number of Passengers')
plt.show()

#Conclusion: fare distribution is highly right skewed - most of the passengers paid low fares and only few paid very high fares
```



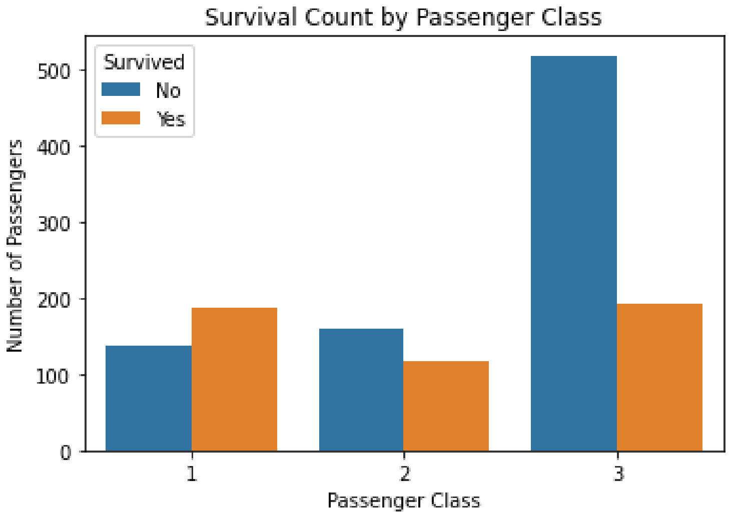
In [39]:
```
sns.boxplot( x='Fare', data=df_titanic)
plt.title('Boxplot of Fare')
plt.xlabel('Fare')
plt.show()
```



In [40]:
```
#Checking how many passengers are between 200 to 300 of fare and how many are there for above 300

print('No of passengers ranging between fare of $200 to $300:', df_titanic[(df_titanic['Fare']>200)&(df_titanic['Fare']<300)].s
print('No of passengers having greater than $300:', df_titanic[(df_titanic['Fare']>300)].shape[0])
```

```
No of passengers ranging between fare of $200 to $300: 34
No of passengers having greater than $300: 4
```

In [41]:
```
#Checking the survival count by Passenger Class

sns.countplot(x='Pclass', hue='Survived', data=df_titanic)
plt.title('Survival Count by Passenger Class')
plt.xlabel('Passenger Class')
plt.ylabel('Number of Passengers')
plt.legend(title='Survived', labels=['No', 'Yes'])
plt.show()

#Conclusion: Mortality rate for Pclass 3 passengers is highest
```

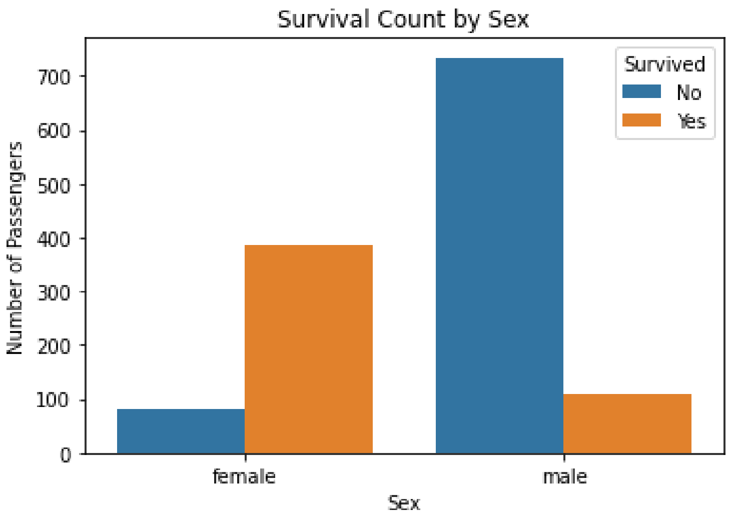


In [42]:
```
pd.crosstab(df_titanic['Pclass'],df_titanic['Survived']).apply(lambda x:round((x/x.sum())*100,1),axis=1)

#Conclusion:
#     1)In Pclass 1: 57.6% survived, 42.4% died

#     2)In Pclass 3: only 26.9% survived, 73.1% died
```

Out[42]:

Survived	0	1
Pclass		
1	42.4	57.6
2	57.8	42.2
3	73.1	26.9

In [43]:
```
# Suvival count by Sex

sns.countplot(x='Sex', hue='Survived', data=df_titanic)
plt.title('Survival Count by Sex')
plt.xlabel('Sex')
plt.ylabel('Number of Passengers')
plt.legend(title='Survived', labels=['No', 'Yes'])
plt.show()

# Conclusion: Males had higher mortality rate.
```

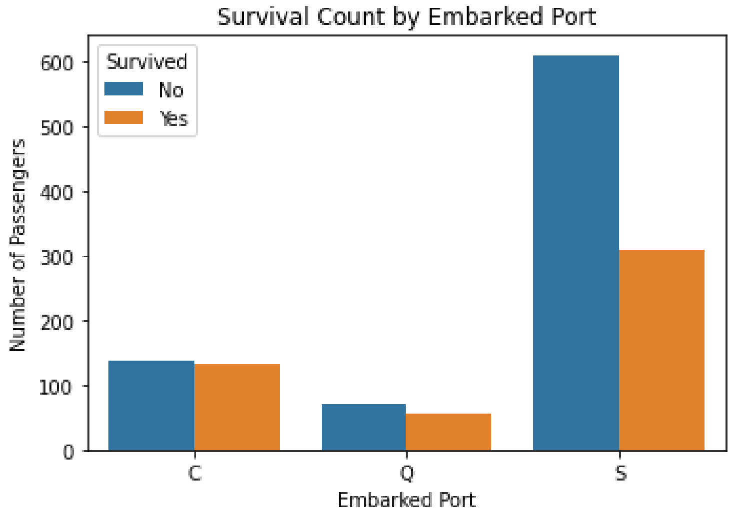


In [44]:
```
pd.crosstab(df_titanic['Sex'],df_titanic['Survived']).apply(lambda x:round((x/x.sum())*100,1),axis=1)

#Conclusion:
#     1)Male Passengers: only 12.9% survived, 87.1% died

#     2)Female Passengers: 82.6% survived, 17.4% died
```

Out[44]:

Survived	0	1
Sex		
female	17.4	82.6
male	87.1	12.9

In [45]:
```
# Suvival count by Embarked Port

sns.countplot(x='Embarked', hue='Survived', data=df_titanic)
plt.title('Survival Count by Embarked Port')
plt.xlabel('Embarked Port')
plt.ylabel('Number of Passengers')
plt.legend(title='Survived', labels=['No', 'Yes'])
plt.show()
```

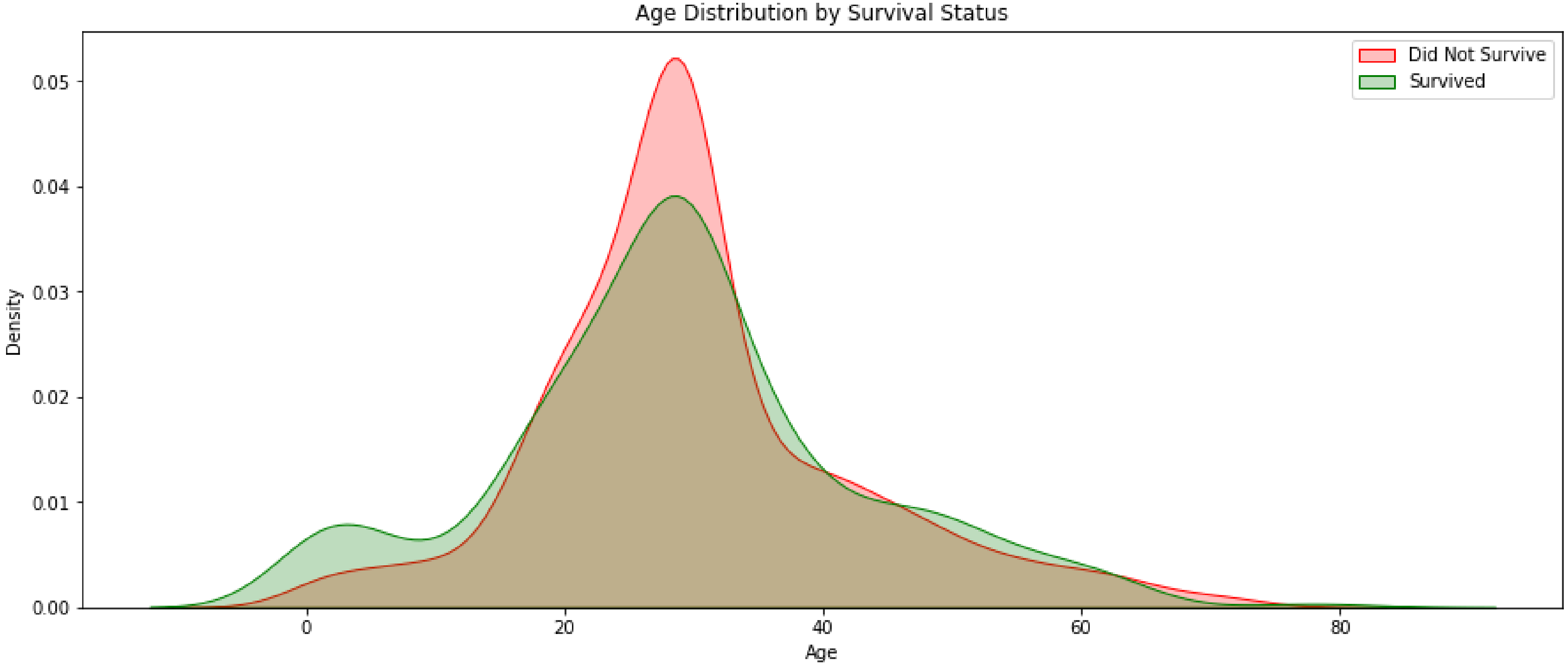


In [46]:
```
pd.crosstab(df_titanic['Embarked'],df_titanic['Survived']).apply(lambda x:round((x/x.sum())*100,1),axis=1)

#Conclusion:
    # Passengers going to S port have higher mortality rate as compared to C and Q.
```

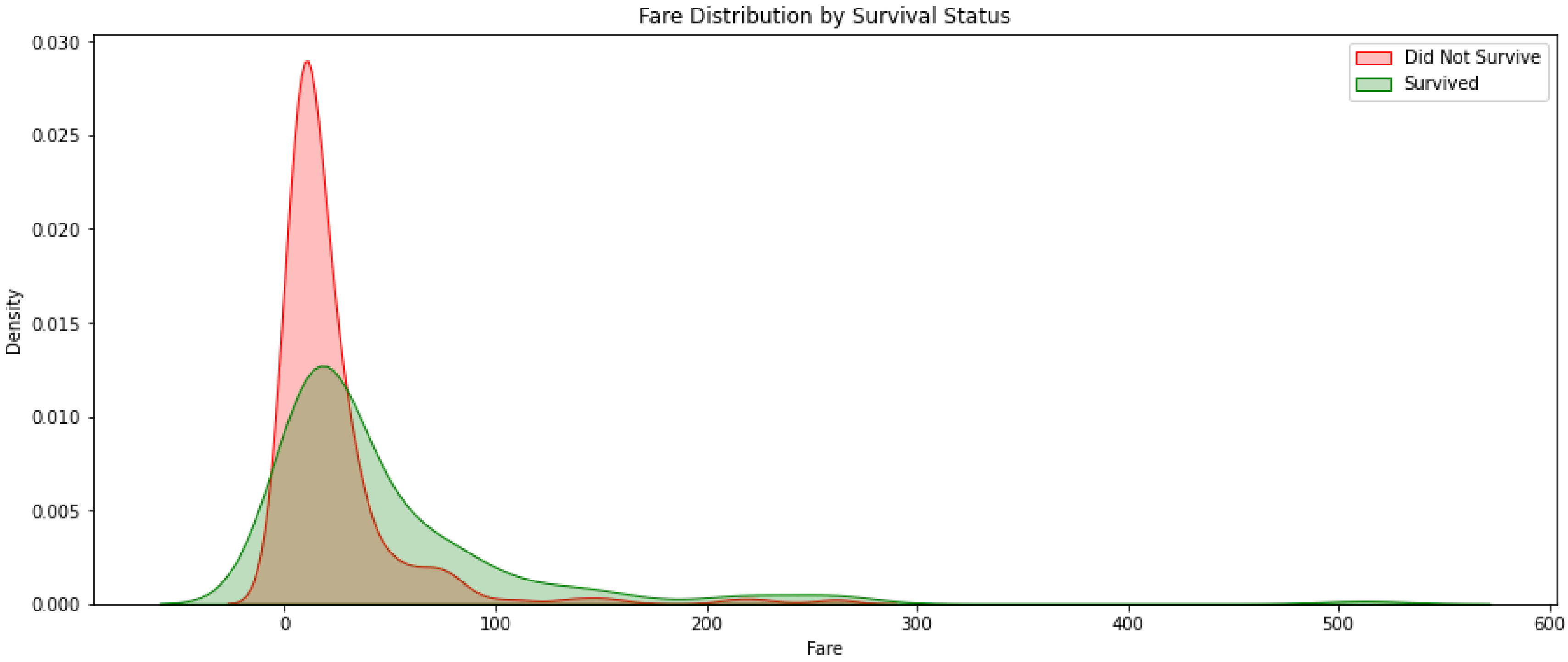
Out[46]:

Survived	0	1
Embarked		
C	50.7	49.3
Q	56.1	43.9
S	66.5	33.5

In [47]:
```python
#Age Distribution by Survival Status

plt.figure(figsize=(15,6))
sns.kdeplot(df_titanic[df_titanic['Survived'] == 0]['Age'], label='Did Not Survive', fill=True, color='red')
sns.kdeplot(df_titanic[df_titanic['Survived'] == 1]['Age'], label='Survived', fill=True, color='green')

plt.title('Age Distribution by Survival Status')
plt.xlabel('Age')
plt.ylabel('Density')
plt.legend()
plt.show()
```

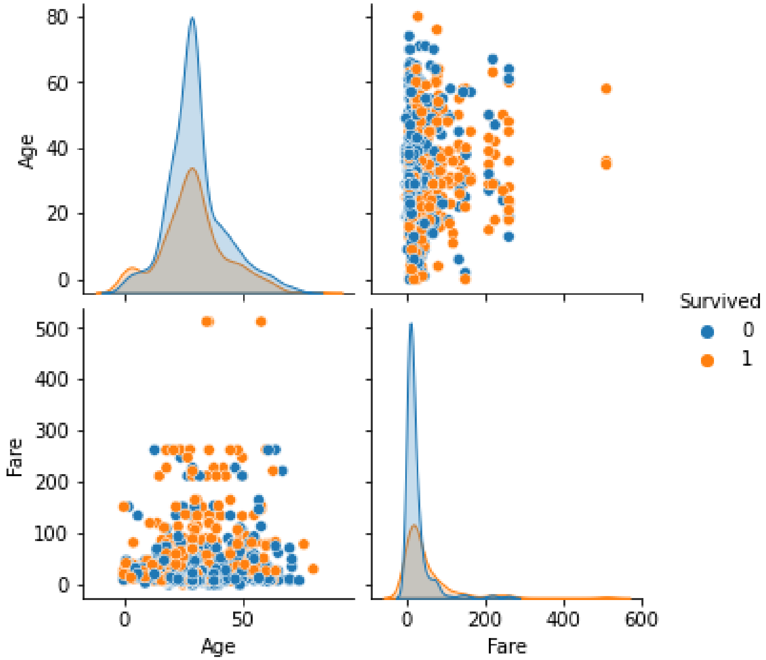


In [48]:
```
#Fare Distribution by Survival Status

plt.figure(figsize=(15,6))
sns.kdeplot(df_titanic[df_titanic['Survived'] == 0]['Fare'], label='Did Not Survive', fill=True, color='red')
sns.kdeplot(df_titanic[df_titanic['Survived'] == 1]['Fare'], label='Survived', fill=True, color='green')

plt.title('Fare Distribution by Survival Status')
plt.xlabel('Fare')
plt.ylabel('Density')
plt.legend()
plt.show()
```

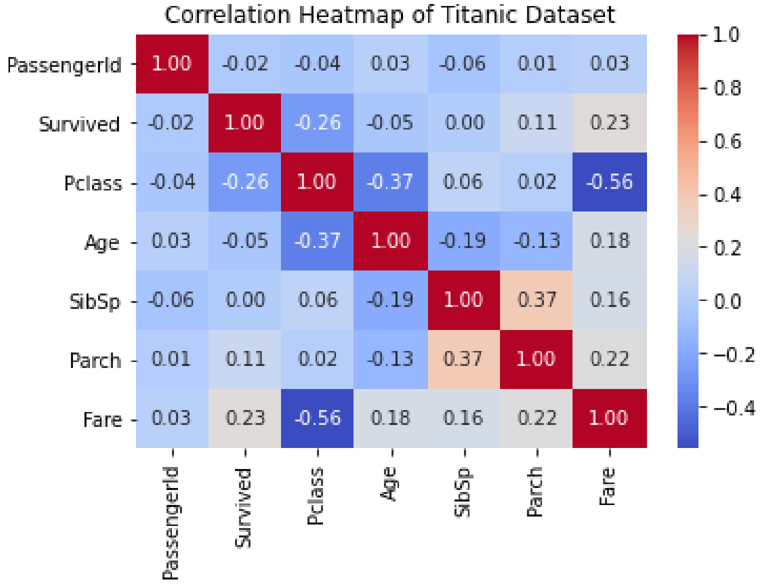


In [49]:
```
sns.pairplot(df_titanic[['Age', 'Fare', 'Pclass', 'Survived']], hue='Survived')
```

Out[49]: <seaborn.axisgrid.PairGrid at 0x1aaeb780af0>



In [50]:
```
df_corr = df_titanic.drop(columns=['Name', 'Ticket','Sex','Embarked'])  # or any other string columns
sns.heatmap(df_corr.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap of Titanic Dataset')
plt.show()

#Conclusion:
    #1) Pclass and fare has negative correlation, so a higher class number having lower fare.
```



In [51]:
```
#Creating new column by the name of family which will be sum of SibSp and Parch
df_titanic['FamilySize'] = df_titanic['SibSp']+df_titanic['Parch']+1
```

In [52]:
```
df_titanic.head()
```

Out[52]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	FamilySize
0	1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.2500	S	2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38	1	0	PC 17599	71.2833	C	2
2	3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.9250	S	1
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1000	S	2
4	5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.0500	S	1

In [53]:
```
#We will group the family size over the family type(Alone, Medium, Large)
def family_group(size):
    if size == 1:
        return 'Alone'
    elif size <= 4:
        return 'Small'
    else:
        return 'Large'

df_titanic['FamilyGroup'] = df_titanic['FamilySize'].apply(family_group)
```

In [54]:
```
df_titanic.head()
```

Out[54]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	FamilySize	FamilyGroup
0	1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.2500	S	2	Small
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38	1	0	PC 17599	71.2833	C	2	Small
2	3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.9250	S	1	Alone
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1000	S	2	Small
4	5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.0500	S	1	Alone

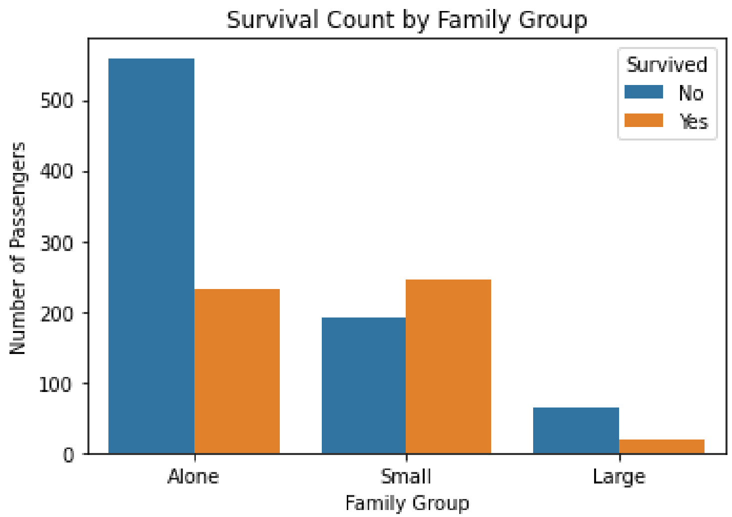
In [56]:
```
#Dropping irrelevant columns now

df_titanic.drop(columns=['SibSp','Parch','FamilySize'], inplace=True)
```

In [57]:
```
df_titanic.sample(5)
```

Out[57]:

	PassengerId	Survived	Pclass	Name	Sex	Age	Ticket	Fare	Embarked	FamilyGroup
505	506	0	1	Penasco y Castellana, Mr. Victor de Satode	male	18	PC 17758	108.9000	C	Small
914	915	0	1	Williams, Mr. Richard Norris II	male	21	PC 17597	61.3792	C	Small
125	126	1	3	Nicola-Yarred, Master. Elias	male	12	2651	11.2417	C	Small
516	517	1	2	Lemore, Mrs. (Amelia Milley)	female	34	C.A. 34260	10.5000	S	Alone
1073	1074	1	1	Marvin, Mrs. Daniel Warner (Mary Graham Carmic...	female	18	113773	53.1000	S	Small

In [58]:
```
sns.countplot(x='FamilyGroup', hue='Survived', data=df_titanic, order=['Alone', 'Small', 'Large'])
plt.title('Survival Count by Family Group')
plt.xlabel('Family Group')
plt.ylabel('Number of Passengers')
plt.legend(title='Survived', labels=['No', 'Yes'])
plt.show()
```



In [59]:
```
pd.crosstab(df_titanic['FamilyGroup'],df_titanic['Survived']).apply(lambda x: round((x/x.sum())*100,1),axis=1)

#Conclusion:
    # 1) 70% of alone passengers died.
    # 2) 78% of large family group died.
    # 3) 44% of medium family group died.
```

Out[59]:

Survived	0	1
FamilyGroup		
Alone	70.8	29.2
Large	78.0	22.0
Small	43.9	56.1

Final Conclusions

1) Out of total, 62% of the total passengers died.

2) Passenger class 3 was the most crowded class, hence it became the most deadliest class.

3) People going to C destination survived more.

4) Almost 70% of the passengers were travelling to Southampton port.

5) Male passengers comprised approximately 64% of total passengers.

6) Chances of Female survival is higher than male survival.

7) Most passengers were between 20-40 years of age.

8) People ranging from 20 years to 35 years of age had a higher chance of not surviving.

9) People travelling with smaller families had higher chance of surviving the accident in comparison to people with large families and travelling alone.