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

from warnings import filterwarnings
filterwarnings(action='ignore')
```

In [2]:
```
train = pd.read_csv(r"C:\Users\Admin\Desktop\DataScience(ProdigyInfotech)\Task2_Titanic\train1.csv")
trial = pd.read_csv(r"C:\Users\Admin\Desktop\DataScience(ProdigyInfotech)\Trial.csv")
```

In [3]:
```
print(train.head())
print(trial.head())
```

```
   PassengerId  Survived  Pclass            Name  Gender   Age  SibSp  Parch  \
0            1         0       3         shivani  female  22.0      1      0
1            2         1       1            isha  female  38.0      1      0
2            3         1       3           bablu    male  26.0      0      0
3            4         1       1           pinki    male  35.0      1      0
4            5         0       3  surender singh    male  35.0      0      0

             Ticket     Fare Cabin Embarked
0         A/5 21171   7.2500   C85        S
1          PC 17599  71.2833   NaN        C
2  STON/O2. 3101282   7.9250  C123        S
3            113803  53.1000   NaN        S
4            373450   8.0500   NaN        S
  student name  std div  roll no
0         Aria    2   A       21
1         Veda    3   A       29
2        Beena    7   C       12
3        Seema    1   C       34
4        Arohi    3   B       19
```

In [4]:
```
train.head()
```

Out[4]:

	PassengerId	Survived	Pclass	Name	Gender	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	shivani	female	22.0	1	0	A/5 21171	7.2500	C85	S
1	2	1	1	isha	female	38.0	1	0	PC 17599	71.2833	NaN	C
2	3	1	3	bablu	male	26.0	0	0	STON/O2. 3101282	7.9250	C123	S
3	4	1	1	pinki	male	35.0	1	0	113803	53.1000	NaN	S
4	5	0	3	surender singh	male	35.0	0	0	373450	8.0500	NaN	S

In [5]:
```
train.tail()
```

Out[5]:

	PassengerId	Survived	Pclass	Name	Gender	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
59	60	0	3	mohd. aslam	male	11.0	5	2	113509	46.9000	NaN	S
60	61	0	3	bharat damor	male	22.0	0	0	113509	7.2292	B28	C
61	62	1	1	alok kumar thakur	male	38.0	0	0	113509	80.0000	C83	NaN
62	63	0	1	najim urf babbu	male	45.0	1	0	113509	83.4750	NaN	S
63	64	0	3	amar singh	male	4.0	3	2	113509	27.9000	NaN	S

In [6]:
```
train
```

Out[6]:

	PassengerId	Survived	Pclass	Name	Gender	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	shivani	female	22.0	1	0	A/5 21171	7.2500	C85	S
1	2	1	1	isha	female	38.0	1	0	PC 17599	71.2833	NaN	C
2	3	1	3	bablu	male	26.0	0	0	STON/O2. 3101282	7.9250	C123	S
3	4	1	1	pinki	male	35.0	1	0	113803	53.1000	NaN	S
4	5	0	3	surender singh	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
59	60	0	3	mohd. aslam	male	11.0	5	2	113509	46.9000	NaN	S
60	61	0	3	bharat damor	male	22.0	0	0	113509	7.2292	B28	C
61	62	1	1	alok kumar thakur	male	38.0	0	0	113509	80.0000	C83	NaN
62	63	0	1	najim urf babbu	male	45.0	1	0	113509	83.4750	NaN	S
63	64	0	3	amar singh	male	4.0	3	2	113509	27.9000	NaN	S

64 rows × 12 columns

In [7]:
```
train.shape
```

Out[7]: (64, 12)

In [8]:
```
#Checking for Null values
train.isnull().sum()
```

Out[8]:
```
PassengerId     0
Survived        0
Pclass          0
Name            0
Gender          0
Age            15
SibSp           0
Parch           0
Ticket          1
Fare            0
Cabin          50
Embarked        1
dtype: int64
```

In [9]:
```
#Description of dataset
train.describe(include="all")
```

Out[9]:

	PassengerId	Survived	Pclass	Name	Gender	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
count	64.000000	64.0000	64.000000	64	64	49.000000	64.000000	64.000000	63	64.000000	14	63
unique	NaN	NaN	NaN	62	2	NaN	NaN	NaN	54	NaN	14	3
top	NaN	NaN	NaN	anita	male	NaN	NaN	NaN	113509	NaN	C85	S
freq	NaN	NaN	NaN	2	40	NaN	NaN	NaN	9	NaN	1	40
mean	32.500000	0.4375	2.359375	NaN	NaN	27.234694	0.781250	0.437500	NaN	30.210158	NaN	NaN
std	18.618987	0.5000	0.842656	NaN	NaN	16.528732	1.147375	1.037013	NaN	39.276369	NaN	NaN
min	1.000000	0.0000	1.000000	NaN	NaN	2.000000	0.000000	0.000000	NaN	7.225000	NaN	NaN
25%	16.750000	0.0000	2.000000	NaN	NaN	15.000000	0.000000	0.000000	NaN	8.044800	NaN	NaN
50%	32.500000	0.0000	3.000000	NaN	NaN	27.000000	0.000000	0.000000	NaN	17.900000	NaN	NaN
75%	48.250000	1.0000	3.000000	NaN	NaN	38.000000	1.000000	0.000000	NaN	32.415625	NaN	NaN
max	64.000000	1.0000	3.000000	NaN	NaN	66.000000	5.000000	5.000000	NaN	263.000000	NaN	NaN

In [10]:
```
#Display shape
train.shape
```

Out[10]:
```
(64, 12)
```

In [11]:
```
train.corr
```

Out[11]:
```
<bound method DataFrame.corr of     PassengerId  Survived  Pclass              Name  Gender   Age  SibSp  \
0             1         0       3           shivani  female  22.0      1
1             2         1       1              isha  female  38.0      1
2             3         1       3             bablu    male  26.0      0
3             4         1       1             pinki    male  35.0      1
4             5         0       3    surender singh    male  35.0      0
..          ...       ...     ...               ...     ...   ...    ...
59           60         0       3       mohd. aslam    male  11.0      5
60           61         0       3      bharat damor    male  22.0      0
61           62         1       1  alok kumar thakur    male  38.0      0
62           63         0       1    najim urf babbu    male  45.0      1
63           64         0       3        amar singh    male   4.0      3

    Parch            Ticket     Fare Cabin Embarked
0       0         A/5 21171   7.2500   C85        S
1       0          PC 17599  71.2833   NaN        C
2       0  STON/O2. 3101282   7.9250  C123        S
3       0            113803  53.1000   NaN        S
4       0            373450   8.0500   NaN        S
..    ...               ...      ...   ...      ...
59      2            113509  46.9000   NaN        S
60      0            113509   7.2292   B28        C
61      0            113509  80.0000   C83      NaN
62      0            113509  83.4750   NaN        S
63      2            113509  27.9000   NaN        S

[64 rows x 12 columns]>
```

In [12]:
```
train.groupby('Survived').mean
```

Out[12]:
```
<bound method GroupBy.mean of <pandas.core.groupby.generic.DataFrameGroupBy object at 0x000002A652890800>>
```

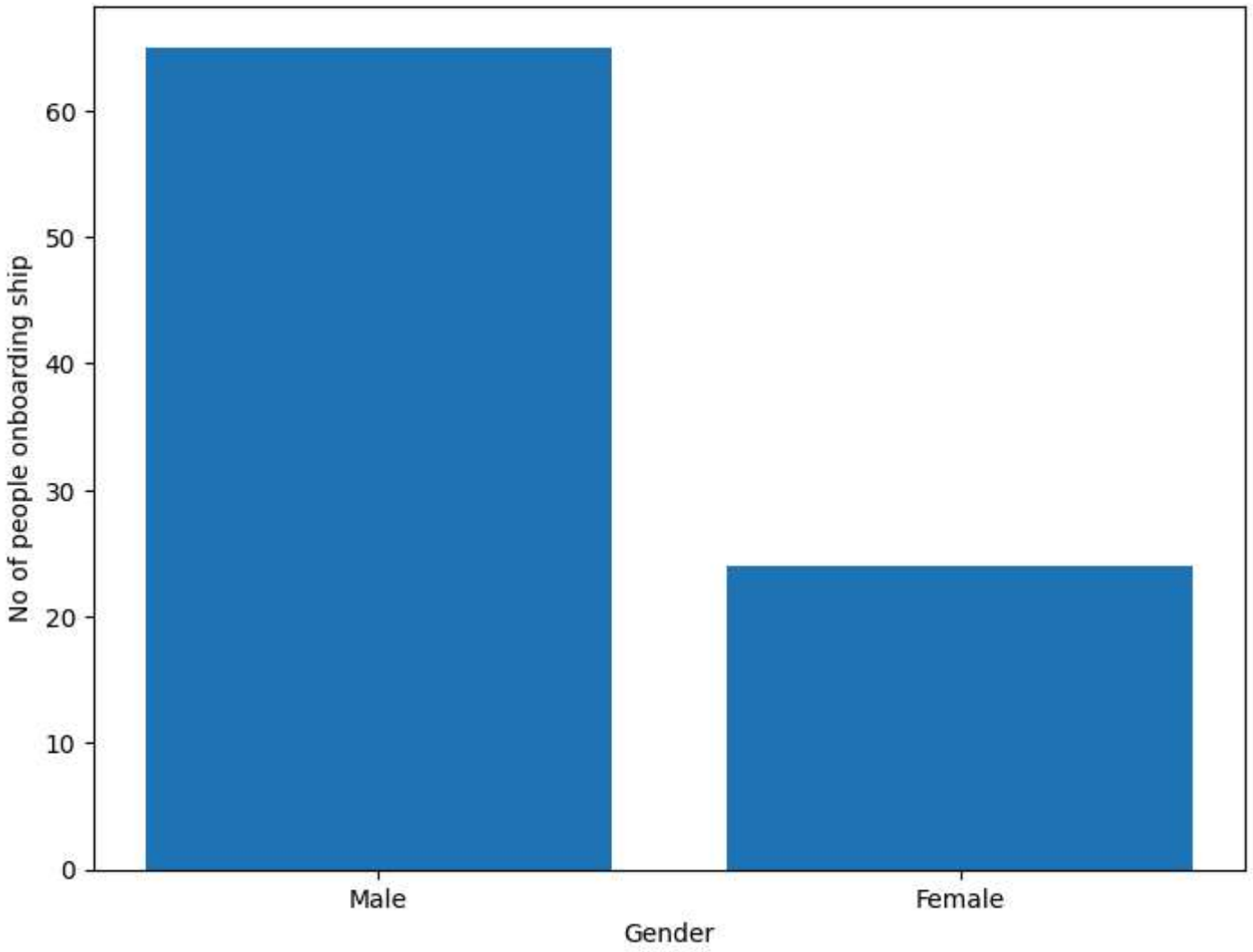
In [13]:
```
male_ind = len(train[train['Gender'] == 'male'])
print("No of Males in Titanic:",male_ind)
```

```
No of Males in Titanic: 40
```

In [14]:
```
female_ind = len(train[train['Gender'] == 'female'])
print("No of Females in Titanic:",female_ind)
```

```
No of Females in Titanic: 24
```

In [15]:
```
#Plotting
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
gender = ['Male','Female']
index = [65,24]
ax.bar(gender,index)
plt.xlabel("Gender")
plt.ylabel("No of people onboarding ship")
plt.show()
```



In [16]:
```
alive = len(train[train['Survived'] == 1])
dead = len(train[train['Survived'] == 0])
```

In [17]:
```
train.groupby('Gender')[['Survived']].mean()
```

Out[17]:

Gender	Survived
female	0.5
male	0.4

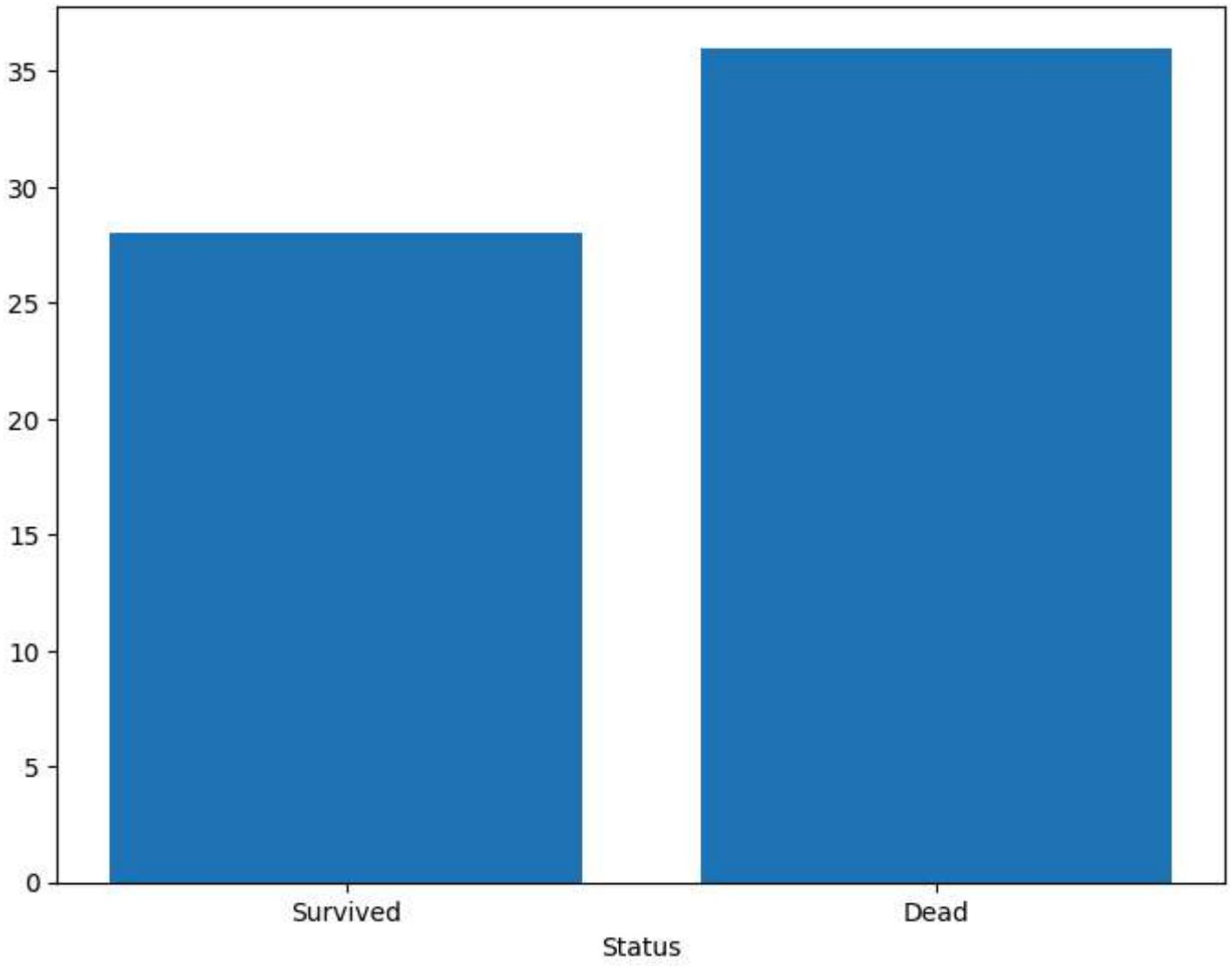
In [18]:
```
train.groupby('Age')[['Survived']].mean()
```

Out[18]:

Age	Survived
2.0	0.000000
3.0	1.000000
4.0	0.500000
5.0	1.000000
7.0	0.000000
8.0	0.000000
11.0	0.000000
14.0	0.666667
15.0	1.000000
18.0	0.000000
19.0	0.500000
20.0	0.000000
21.0	0.333333
22.0	0.000000
26.0	1.000000
27.0	0.500000
28.0	0.500000
28.5	0.000000
29.0	1.000000
31.0	0.000000
34.0	1.000000

Age	Survived
35.0	0.333333
38.0	1.000000
39.0	0.000000
40.0	0.000000
42.0	0.000000
45.0	0.000000
49.0	1.000000
54.0	0.000000
55.0	1.000000
58.0	1.000000
65.0	0.000000
66.0	0.000000

In [19]:
```
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
status = ['Survived','Dead']
ind = [alive,dead]
ax.bar(status,ind)
plt.xlabel("Status")
plt.show()
```

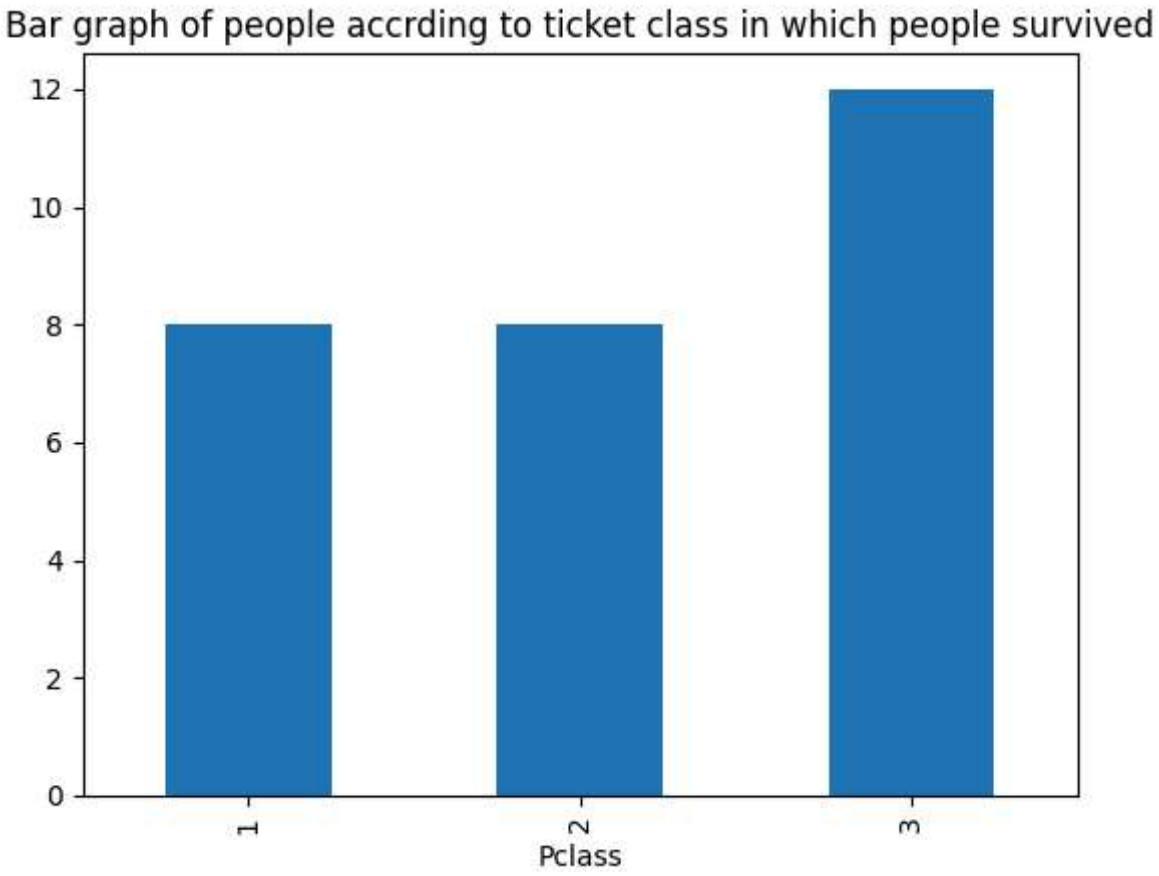


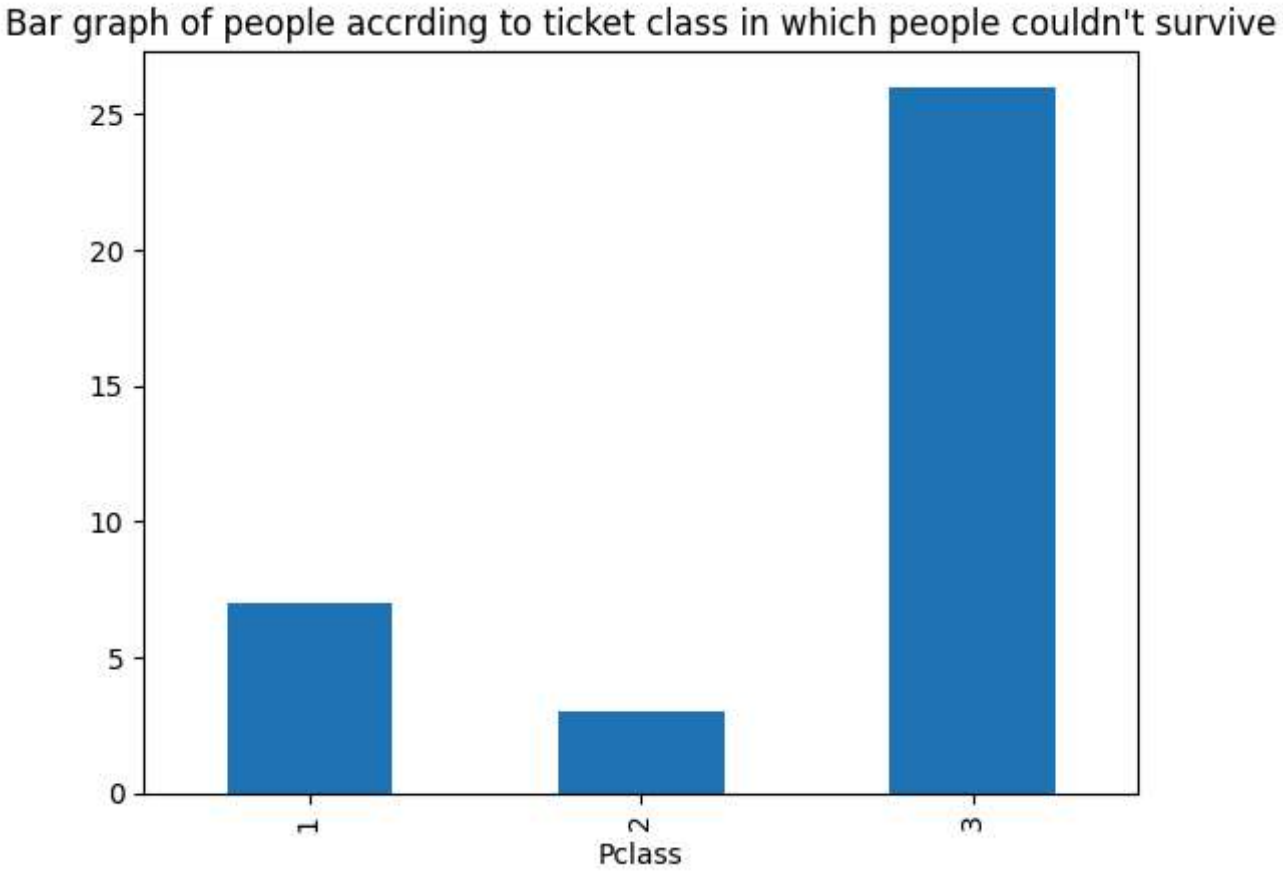
In [20]:
```
plt.figure(1)
train.loc[train['Survived'] == 1, 'Pclass'].value_counts().sort_index().plot.bar()
plt.title('Bar graph of people accrding to ticket class in which people survived')


plt.figure(2)
```

```
train.loc[train['Survived'] == 0, 'Pclass'].value_counts().sort_index().plot.bar()
plt.title('Bar graph of people accrding to ticket class in which people couldn\'t survive')
```

Out[20]: Text(0.5, 1.0, "Bar graph of people accrding to ticket class in which people couldn't survive")

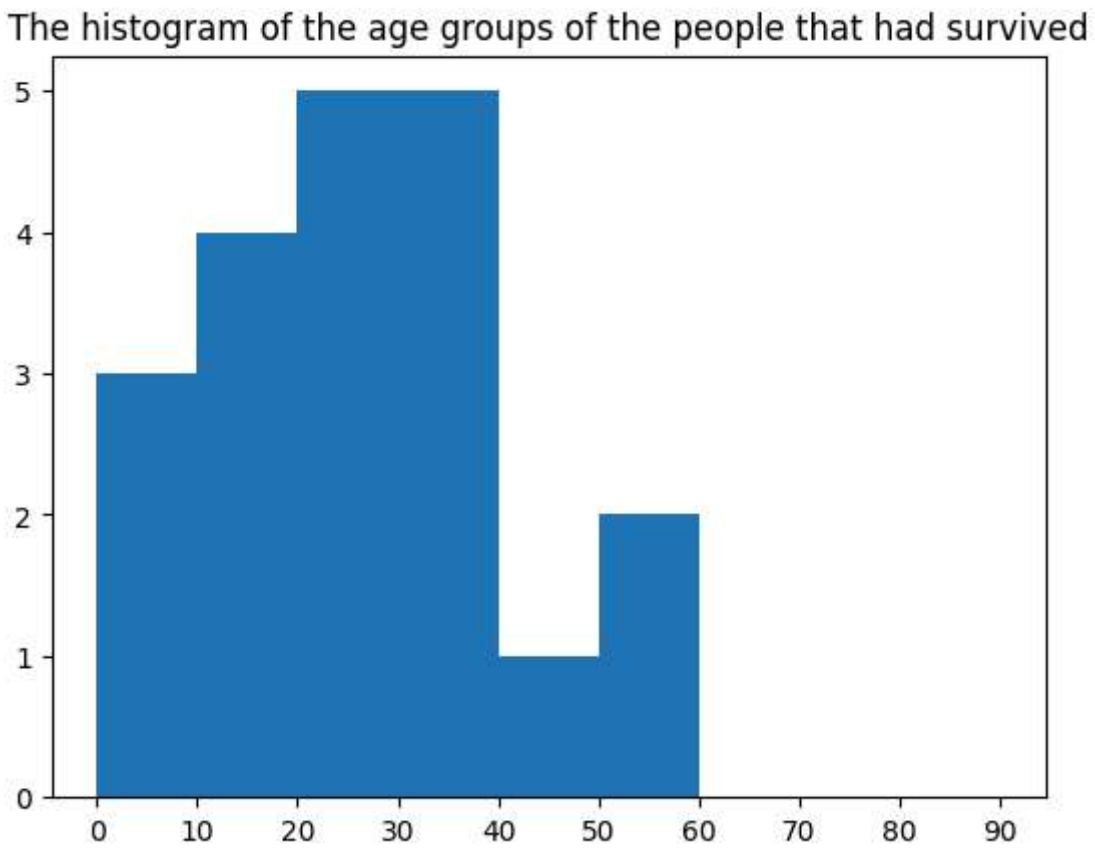


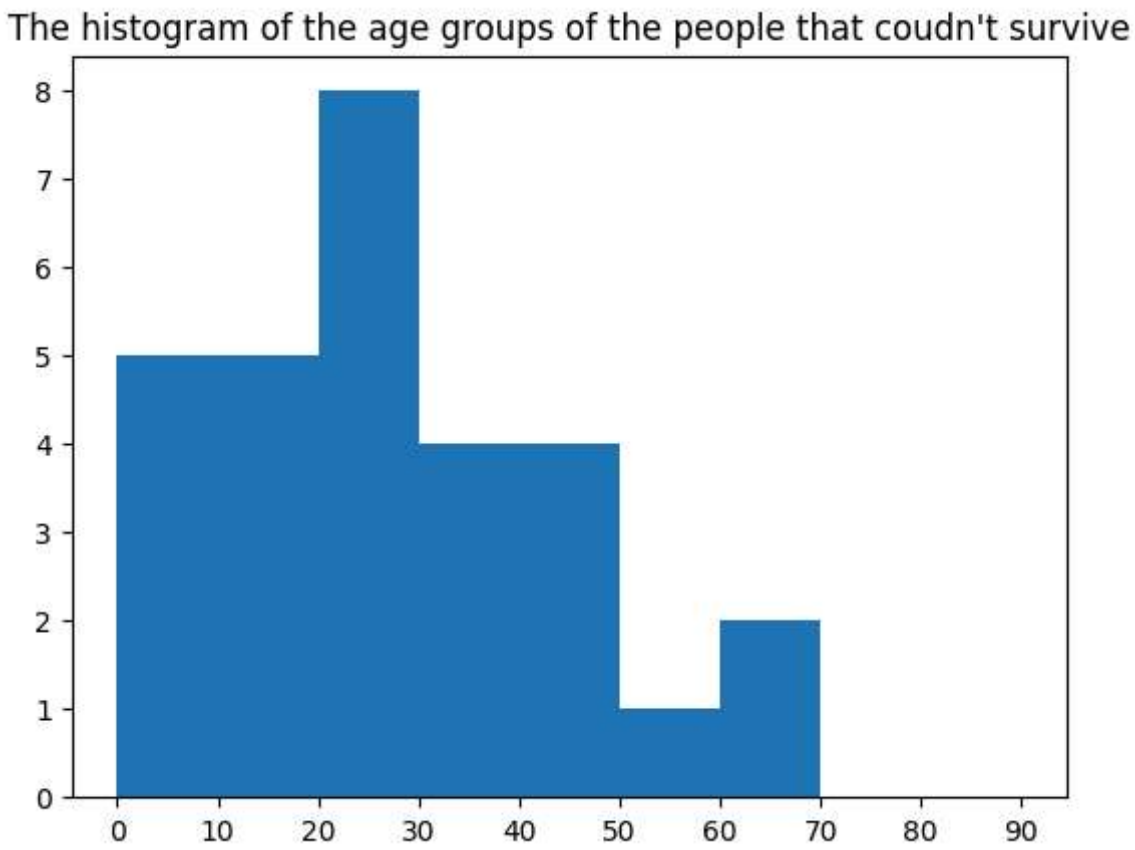


In [21]:
```
plt.figure(1)
age  = train.loc[train.Survived == 1, 'Age']
plt.title('The histogram of the age groups of the people that had survived')
plt.hist(age, np.arange(0,100,10))
plt.xticks(np.arange(0,100,10))


plt.figure(2)
age  = train.loc[train.Survived == 0, 'Age']
plt.title('The histogram of the age groups of the people that coudn\'t survive')
plt.hist(age, np.arange(0,100,10))
plt.xticks(np.arange(0,100,10))
```

Out[21]:
```
([<matplotlib.axis.XTick at 0x2a654cbede0>,
  <matplotlib.axis.XTick at 0x2a654d4e7e0>,
  <matplotlib.axis.XTick at 0x2a654d4c950>,
  <matplotlib.axis.XTick at 0x2a6550bf7d0>,
  <matplotlib.axis.XTick at 0x2a6550bfda0>,
  <matplotlib.axis.XTick at 0x2a6550f0a70>,
  <matplotlib.axis.XTick at 0x2a654d4c410>,
  <matplotlib.axis.XTick at 0x2a6550f0650>,
  <matplotlib.axis.XTick at 0x2a6550f1d60>,
  <matplotlib.axis.XTick at 0x2a6550f2660>],
 [Text(0, 0, '0'),
  Text(10, 0, '10'),
  Text(20, 0, '20'),
  Text(30, 0, '30'),
  Text(40, 0, '40'),
  Text(50, 0, '50'),
  Text(60, 0, '60'),
  Text(70, 0, '70'),
  Text(80, 0, '80'),
  Text(90, 0, '90')])
```


The histogram of the age groups of the people that had survived
0
1
2
3
4
5
0
10
20
30
40
50
60
70
80
90



In [22]:
```
train[["SibSp", "Survived"]].groupby(['SibSp'], as_index=False).mean().sort_values(by='Survived', ascending=False)
```

Out[22]:

	SibSp	Survived
1	1	0.52381
0	0	0.50000
2	2	0.00000
3	3	0.00000
4	4	0.00000
5	5	0.00000

In [23]:
```
train[["Pclass", "Survived"]].groupby(['Pclass'], as_index=False).mean().sort_values(by='Survived', ascending=False)
```

Out[23]:

	Pclass	Survived
1	2	0.727273
0	1	0.533333
2	3	0.315789

In [24]:
```
train[["Age", "Survived"]].groupby(['Age'], as_index=False).mean().sort_values(by='Age', ascending=True)
```

Out[24]:

	Age	Survived
0	2.0	0.000000
1	3.0	1.000000
2	4.0	0.500000
3	5.0	1.000000
4	7.0	0.000000
5	8.0	0.000000
6	11.0	0.000000
7	14.0	0.666667
8	15.0	1.000000
9	18.0	0.000000
10	19.0	0.500000
11	20.0	0.000000
12	21.0	0.333333
13	22.0	0.000000
14	26.0	1.000000
15	27.0	0.500000
16	28.0	0.500000
17	28.5	0.000000
18	29.0	1.000000
19	31.0	0.000000
20	34.0	1.000000
21	35.0	0.333333

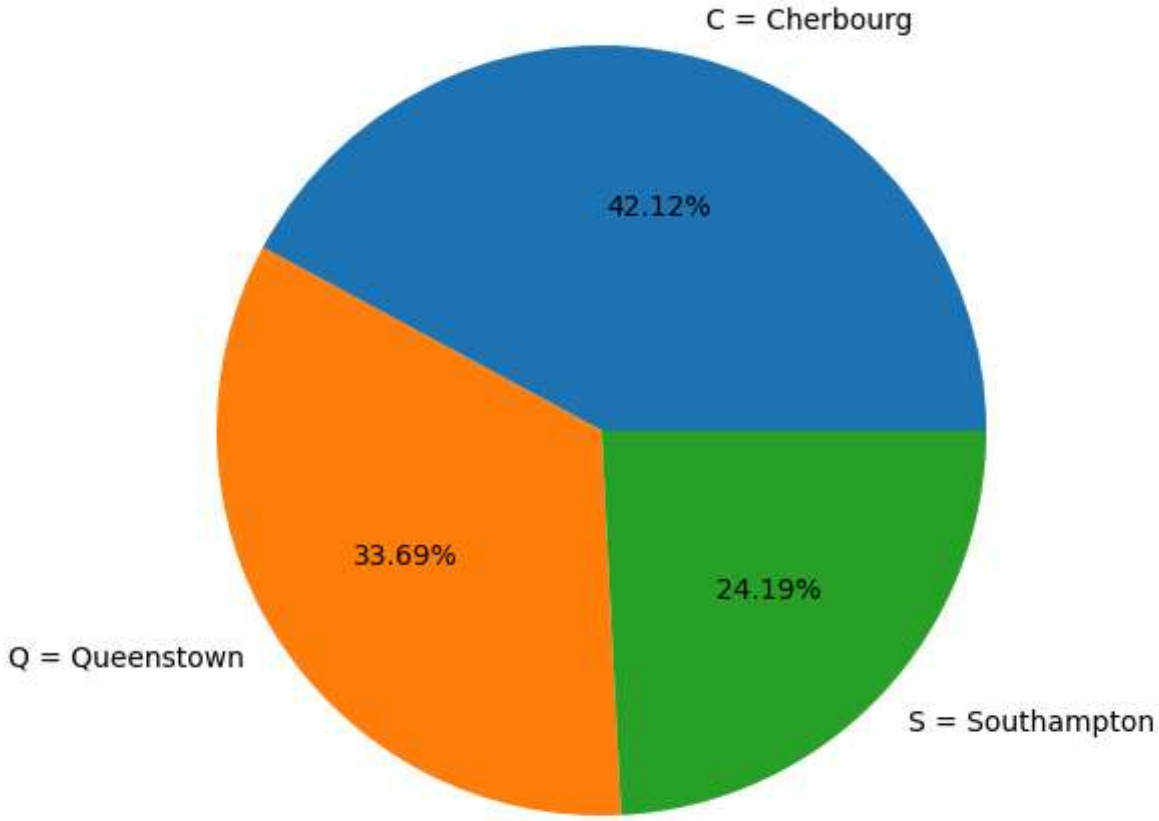
	Age	Survived
22	38.0	1.000000
23	39.0	0.000000
24	40.0	0.000000
25	42.0	0.000000
26	45.0	0.000000
27	49.0	1.000000
28	54.0	0.000000
29	55.0	1.000000
30	58.0	1.000000
31	65.0	0.000000
32	66.0	0.000000

In [25]:
```
train[["Embarked", "Survived"]].groupby(['Embarked'], as_index=False).mean().sort_values(by='Survived', ascending=False)
```

Out[25]:

	Embarked	Survived
1	Q	0.571429
0	C	0.500000
2	S	0.375000

In [26]:
```
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.axis('equal')
l = ['C = Cherbourg', 'Q = Queenstown', 'S = Southampton']
s = [0.625000,0.500000,0.358974]
ax.pie(s, labels = l,autopct='%1.2f%%')
plt.show()
```



In [27]:
```
train.describe(include="all")
```

Out[27]:

	PassengerId	Survived	Pclass	Name	Gender	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
count	64.000000	64.0000	64.000000	64	64	49.000000	64.000000	64.000000	63	64.000000	14	63
unique	NaN	NaN	NaN	62	2	NaN	NaN	NaN	54	NaN	14	3
top	NaN	NaN	NaN	anita	male	NaN	NaN	NaN	113509	NaN	C85	S
freq	NaN	NaN	NaN	2	40	NaN	NaN	NaN	9	NaN	1	40
mean	32.500000	0.4375	2.359375	NaN	NaN	27.234694	0.781250	0.437500	NaN	30.210158	NaN	NaN
std	18.618987	0.5000	0.842656	NaN	NaN	16.528732	1.147375	1.037013	NaN	39.276369	NaN	NaN
min	1.000000	0.0000	1.000000	NaN	NaN	2.000000	0.000000	0.000000	NaN	7.225000	NaN	NaN
25%	16.750000	0.0000	2.000000	NaN	NaN	15.000000	0.000000	0.000000	NaN	8.044800	NaN	NaN
50%	32.500000	0.0000	3.000000	NaN	NaN	27.000000	0.000000	0.000000	NaN	17.900000	NaN	NaN
75%	48.250000	1.0000	3.000000	NaN	NaN	38.000000	1.000000	0.000000	NaN	32.415625	NaN	NaN
max	64.000000	1.0000	3.000000	NaN	NaN	66.000000	5.000000	5.000000	NaN	263.000000	NaN	NaN

In [28]:
```
#Feature Selection
column_train=['Age','Pclass','SibSp','Parch','Fare','Gender','Embarked']
#training values
X=train[column_train]
#target value
Y=train['Survived']
```

In [29]:
```
X['Age'].isnull().sum()
X['Pclass'].isnull().sum()
X['SibSp'].isnull().sum()
X['Parch'].isnull().sum()
X['Fare'].isnull().sum()
X['Gender'].isnull().sum()
X['Embarked'].isnull().sum()
```

Out[29]:
```
np.int64(1)
```

In [30]:
```
X['Age']=X['Age'].fillna(X['Age'].median())
X['Age'].isnull().sum()
```

Out[30]:
```
np.int64(0)
```

In [31]:
```
X['Embarked'] = train['Embarked'].fillna(method ='pad')
X['Embarked'].isnull().sum()
```

Out[31]:
```
np.int64(0)
```

In [32]:
```
d={'male':0, 'female':1}
X['Gender']=X['Gender'].apply(lambda x:d[x])
X['Gender'].head()
```

Out[32]:
```
0    1
1    1
2    0
3    0
4    0
Name: Gender, dtype: int64
```

In [33]:
```
e={'C':0, 'Q':1 ,'S':2}
X['Embarked']=X['Embarked'].apply(lambda x:e[x])
X['Embarked'].head()
```

Out[33]:
```
0    2
1    0
2    2
3    2
4    2
Name: Embarked, dtype: int64
```

In [37]:
```
results = pd.DataFrame({
    'Age': ['Logistic Regression','Survived', 'Dead','No injuries' ,'Deep injuries'],
    'Ticket': [0.75,0.66,0.76,0.66,0.74]})

result_df = results.sort_values(by='Ticket', ascending=False)
result_df = result_df.set_index('Ticket')
result_df.head(9)
```

Out[37]:

Ticket	Age
0.76	Dead
0.75	Logistic Regression
0.74	Deep injuries
0.66	Survived
0.66	No injuries

In [ ]: