In [19]:
```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
```

In [20]:
```
train_data = pd.read_csv("train.csv")
test_data = pd.read_csv("test.csv")
```

In [21]:
```
train_data
```

Out[21]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Emba
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	

891 rows × 12 columns

In [22]:

```
train_data.isnull().sum()
```

Out[22]:

```
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
```

In [24]:

```
train_data.dropna(inplace = True)
```

In [25]:

```
train_data.isnull().sum()
```

Out[25]:

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

In [27]:

```
test_data
```

Out[27]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...
413	1305	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S
414	1306	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C
415	1307	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S
416	1308	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S
417	1309	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C

418 rows × 11 columns

In [29]:

```
test_data.isnull().sum()
```

Out[29]:

```
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

In [30]:

```
test_data.dropna(inplace = True)
```

In [32]:

```
test_data.isnull().sum()
```

Out[32]:

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

In [33]:

```
X_train = train_data[['Pclass', 'Sex', 'Age', 'SibSp', 'Parch']]
y_train = train_data['Survived']
X_test = test_data[['Pclass', 'Sex', 'Age', 'SibSp', 'Parch']]
```

In [34]:

```
X_train = pd.get_dummies(X_train, columns=['Sex'], drop_first=True)
X_test = pd.get_dummies(X_test, columns=['Sex'], drop_first=True)
```

In [35]:

```
model = DecisionTreeClassifier()
```

In [36]:

```
model.fit(X_train, y_train)
```

Out[36]:

```
DecisionTreeClassifier()
```

In [37]:

```
# Define the test cases
test_case_1 = pd.DataFrame({'Pclass': [1], 'Sex': ['female'], 'Age': [42], 'SibSp': [2],
test_case_2 = pd.DataFrame({'Pclass': [3], 'Sex': ['male'], 'Age': [30], 'SibSp': [0], '

# Encode 'Sex' column to match the encoding used during preprocessing
test_case_1['Sex'] = test_case_1['Sex'].apply(lambda x: 0 if x == 'female' else 1)
test_case_2['Sex'] = test_case_2['Sex'].apply(lambda x: 0 if x == 'female' else 1)

# Make predictions for the test cases
prediction_1 = model.predict(test_case_1)
prediction_2 = model.predict(test_case_2)

# Interpret the predictions
if prediction_1[0] == 1:
    result_1 = "Survived"
else:
    result_1 = "Did not survive"

if prediction_2[0] == 1:
    result_2 = "Survived"
else:
    result_2 = "Did not survive"

# Print the results
print("Test Case 1:", result_1)
print("Test Case 2:", result_2)
```

```
Test Case 1: Survived
Test Case 2: Did not survive

C:\Users\Lenovo\anaconda3\lib\site-packages\sklearn\base.py:493: FutureWarning: The fe
ature names should match those that were passed during fit. Starting version 1.2, an e
rror will be raised.
Feature names unseen at fit time:
- Sex
Feature names seen at fit time, yet now missing:
- Sex_male

  warnings.warn(message, FutureWarning)
C:\Users\Lenovo\anaconda3\lib\site-packages\sklearn\base.py:493: FutureWarning: The fe
ature names should match those that were passed during fit. Starting version 1.2, an e
rror will be raised.
Feature names unseen at fit time:
- Sex
Feature names seen at fit time, yet now missing:
- Sex_male

  warnings.warn(message, FutureWarning)
```

In [ ]: