Implement Decision Tree using "Iris" dataset and compute accuracy score, confusion matrix.

In [17]:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import LabelEncoder
```

In [18]:

```
dat=pd.read_csv('iris.csv')
```

In [19]:

```
dat['Iris-setosa'].value_counts()
```

Out[19]:

```
Iris-versicolor    50
Iris-virginica     50
Iris-setosa        49
Name: Iris-setosa, dtype: int64
```

In [20]:

```
lr=LabelEncoder()
dat['Iris-setosa']=lr.fit_transform(dat['Iris-setosa'])
dat
```

Out[20]:

	5.1	3.5	1.4	0.2	Iris-setosa
0	4.9	3.0	1.4	0.2	0
1	4.7	3.2	1.3	0.2	0
2	4.6	3.1	1.5	0.2	0
3	5.0	3.6	1.4	0.2	0
4	5.4	3.9	1.7	0.4	0
...	...	...	...	...	...
144	6.7	3.0	5.2	2.3	2
145	6.3	2.5	5.0	1.9	2
146	6.5	3.0	5.2	2.0	2
147	6.2	3.4	5.4	2.3	2
148	5.9	3.0	5.1	1.8	2

149 rows × 5 columns

In [21]:

```
x=dat.drop('Iris-setosa',axis=1)
y=dat.iloc[:,4]
```

In [22]:

```
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.2)
```

In [23]:

```
model=DecisionTreeClassifier()
model.fit(xtrain,ytrain)
```

Out[23]:

```
DecisionTreeClassifier()
```

**In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.**

In [24]:

```
pred=model.predict(xtest)
```

In [25]:

```
import sklearn.metrics as mc
mc.confusion_matrix(ytest,pred)
```

Out[25]:

```
array([[10,  0,  0],
       [ 0, 13,  0],
       [ 0,  1,  6]], dtype=int64)
```

In [26]:

```
mc.accuracy_score(ytest,pred)
```

Out[26]:

```
0.9666666666666667
```

In [ ]:

```

```