```
# Latihan 1
# import library pandas
import pandas as pd

# Import library numpy
import numpy as np

# Import library matplotlib dan seaborn untuk visualisasi
import matplotlib.pyplot as plt
import seaborn as sns

# me-non aktifkan peringatan pada python
import warnings
warnings.filterwarnings('ignore')

#Panggil file (load file bernama Iris_AfterClean.csv) dan simpan dalam
dataframe
dataset =pd.read_csv("Iris_AfterClean.csv")
iris = pd.DataFrame(dataset)

# tampilkan 5 baris data
iris.head(5)
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	4.6	3.1	1.5	0.2	Iris-setosa
1	5.0	3.6	1.4	0.2	Iris-setosa
2	5.4	3.9	1.7	0.4	Iris-setosa
3	4.9	3.1	1.5	0.1	Iris-setosa
4	5.4	3.7	1.5	0.2	Iris-setosa

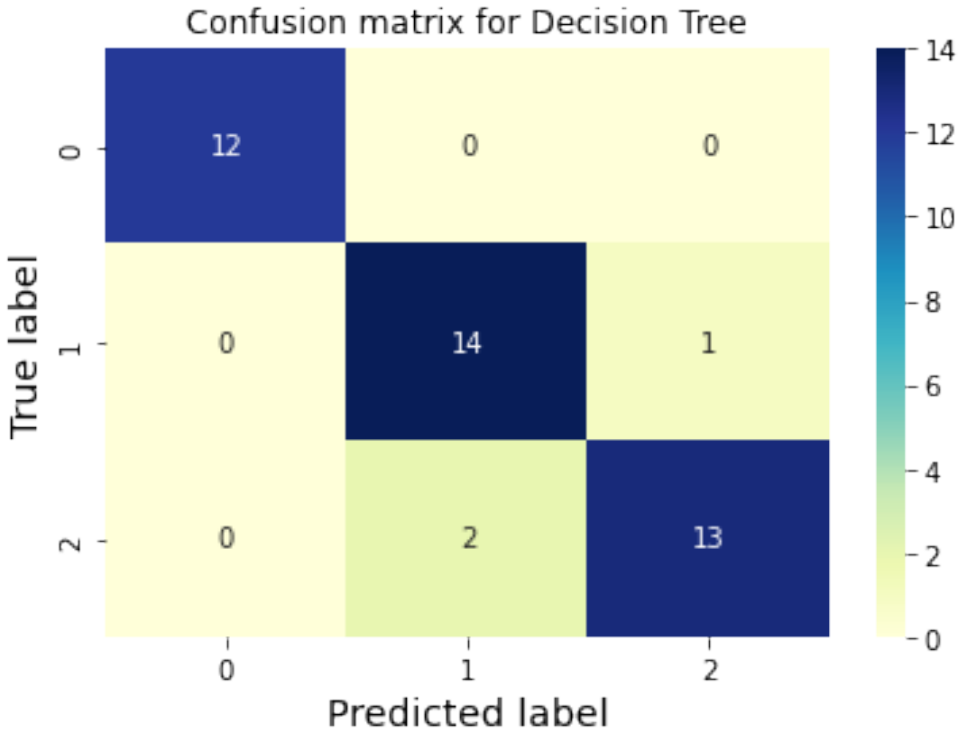
```
# Latihan 2
X=iris.iloc[:,0:4].values
y=iris.iloc[:,4].values

# Latihan 3
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)

#Metrics
from sklearn.metrics import make_scorer,
accuracy_score,precision_score
from sklearn.metrics import classification_report

# Import Library Confussion Matrix
```

```
from sklearn.metrics import confusion_matrix

from sklearn.metrics import
accuracy_score ,precision_score,recall_score,f1_score

#Model Select
from sklearn.model_selection import
KFold,train_test_split,cross_val_score

# Import Library Random Forest Classifier
from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import train_test_split

from sklearn.tree import DecisionTreeClassifier

# from sklearn import linear_model
from sklearn import linear_model

# Latihan 5
#Train and Test split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,rando
m_state=0)

# Latihan 6
random_forest = RandomForestClassifier(n_estimators=100)
random_forest.fit(X_train, y_train)
Y_prediction = random_forest.predict(X_test)
accuracy_rf=round(accuracy_score(y_test,Y_prediction)* 100, 2)
acc_random_forest = round(random_forest.score(X_train, y_train) * 100,
2)

cm = confusion_matrix(y_test, Y_prediction)
accuracy = accuracy_score(y_test,Y_prediction)
precision =precision_score(y_test, Y_prediction,average='micro')
recall =  recall_score(y_test, Y_prediction,average='micro')
f1 = f1_score(y_test,Y_prediction,average='micro')
print('Confusion matrix for Random Forest\n',cm)
print('accuracy_random_Forest : %.3f' %accuracy)
print('precision_random_Forest : %.3f' %precision)
print('recall_random_Forest : %.3f' %recall)
print('f1-score_random_Forest : %.3f' %f1)

Confusion matrix for Random Forest
 [[12  0  0]
 [ 0 14  1]
 [ 0  2 13]]
accuracy_random_Forest : 0.929
precision_random_Forest : 0.929
```

```
recall_random_Forest : 0.929
f1-score_random_Forest : 0.929
```

```
# Latihan 7
from sklearn import metrics
cm = confusion_matrix(y_test, Y_prediction)
p = sns.heatmap(pd.DataFrame(cm), annot=True, cmap="YlGnBu" ,fmt='g')
plt.title('Confusion matrix for Decision Tree')
plt.ylabel('True label', fontsize=14)
plt.xlabel('Predicted label', fontsize=14)
```

```
Text(0.5, 15.0, 'Predicted label')
```



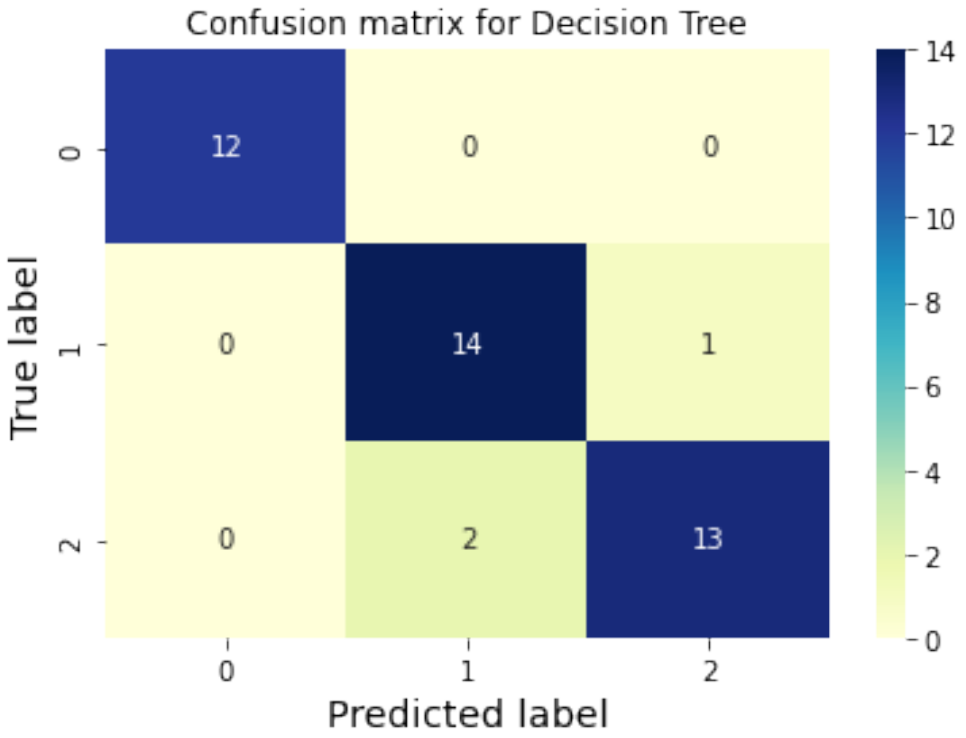

```
# Latihan 8
decision_tree = DecisionTreeClassifier()
decision_tree.fit(X_train, y_train)
Y_pred = decision_tree.predict(X_test)
accuracy_dt=round(accuracy_score(y_test,Y_pred)* 100, 2)
acc_decision_tree = round(decision_tree.score(X_train, y_train) * 100,
2)

cm = confusion_matrix(y_test, Y_pred)
accuracy = accuracy_score(y_test,Y_pred)
precision =precision_score(y_test, Y_pred,average='micro')
recall =  recall_score(y_test, Y_pred,average='micro')
f1 = f1_score(y_test,Y_pred,average='micro')
print('Confusion matrix for DecisionTree\n',cm)
print('accuracy_DecisionTree: %.3f' %accuracy)
```

```
print('precision_DecisionTree: %.3f' %precision)
print('recall_DecisionTree: %.3f' %recall)
print('f1-score_DecisionTree : %.3f' %f1)
```

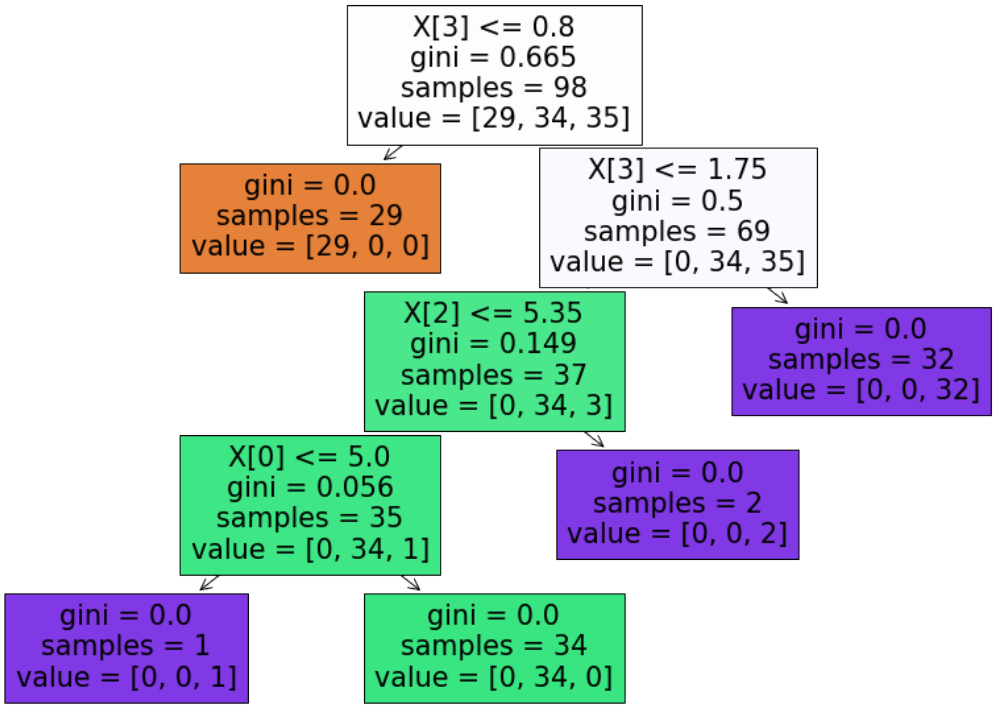
```
Confusion matrix for DecisionTree
 [[12  0  0]
 [ 0 12  3]
 [ 0  2 13]]
accuracy_DecisionTree: 0.881
precision_DecisionTree: 0.881
recall_DecisionTree: 0.881
f1-score_DecisionTree : 0.881
```

```
# Latihan 9
from sklearn import metrics
cdt = confusion_matrix(y_test, Y_prediction)
p = sns.heatmap(pd.DataFrame(cdt), annot=True, cmap="YlGnBu" ,fmt='g')
plt.title('Confusion matrix for Decision Tree')
plt.ylabel('True label', fontsize=14)
plt.xlabel('Predicted label', fontsize=14)
```

```
Text(0.5, 15.0, 'Predicted label')
```




```
# Latihan 10
from sklearn.tree import plot_tree
plt.figure(figsize = (15,10))
plot_tree(decision_tree.fit(X_train, y_train)  ,filled=True)
plt.show()
```


X[3] <= 0.8
gini = 0.665
samples = 98
value = [29, 34, 35]
gini = 0.0
samples = 29
value = [29, 0, 0]
X[3] <= 1.75
gini = 0.5
samples = 69
value = [0, 34, 35]
X[2] <= 5.35
gini = 0.149
samples = 37
value = [0, 34, 3]
gini = 0.0
samples = 32
value = [0, 0, 32]
X[0] <= 5.0
gini = 0.056
samples = 35
value = [0, 34, 1]
gini = 0.0
samples = 2
value = [0, 0, 2]
gini = 0.0
samples = 1
value = [0, 0, 1]
gini = 0.0
samples = 34
value = [0, 34, 0]