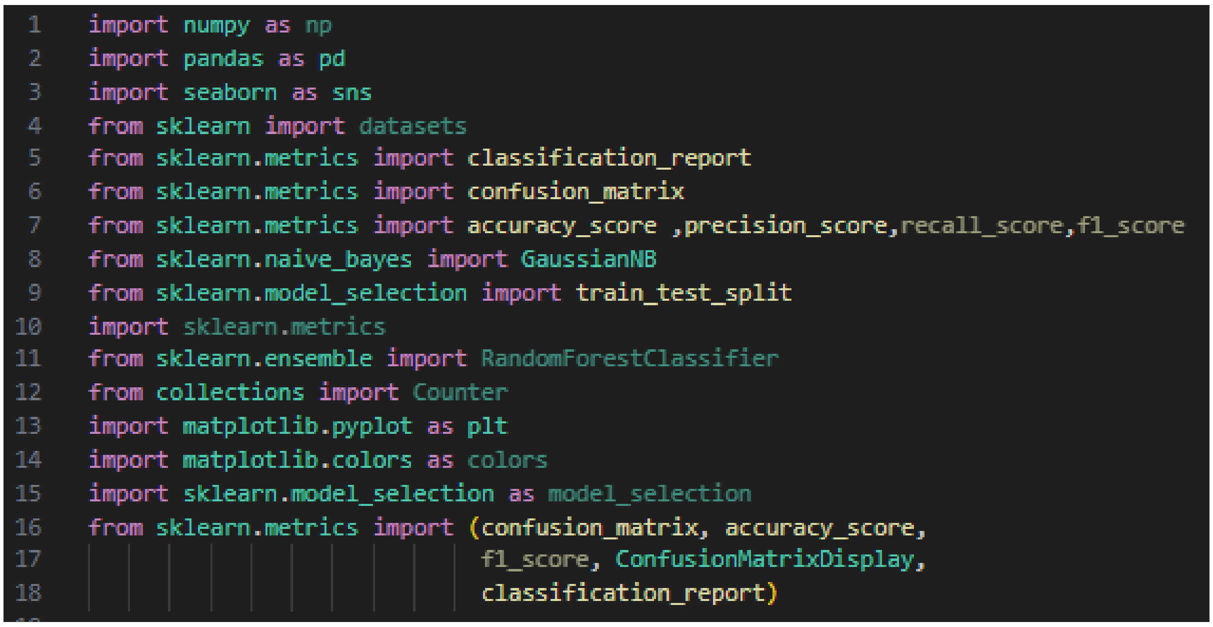
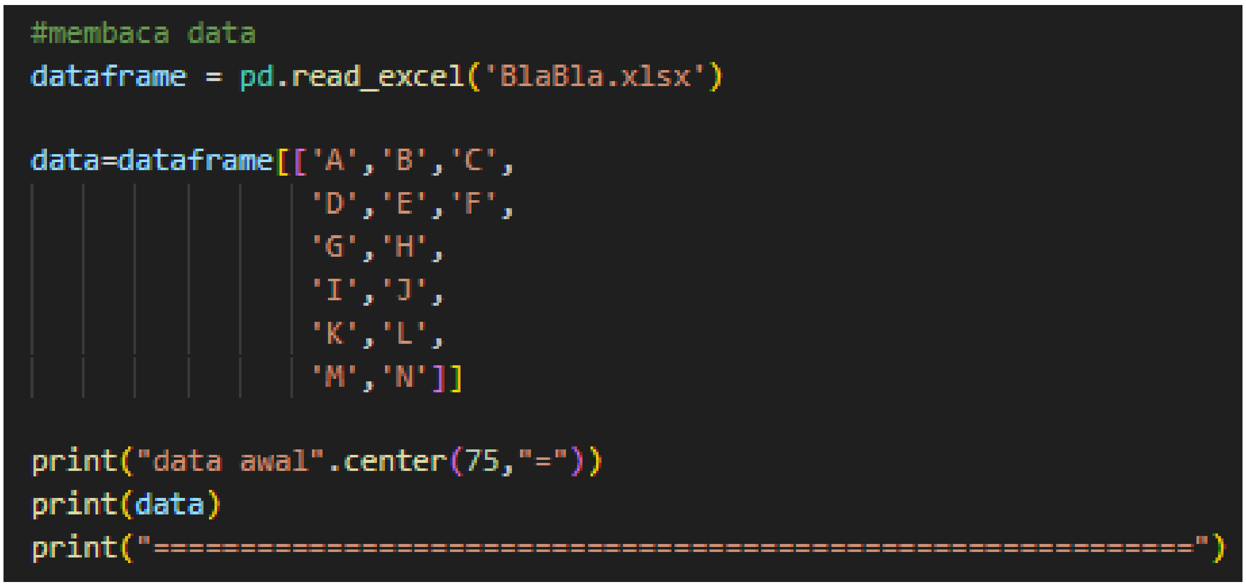
DECISION TREE

1. **Import Library**



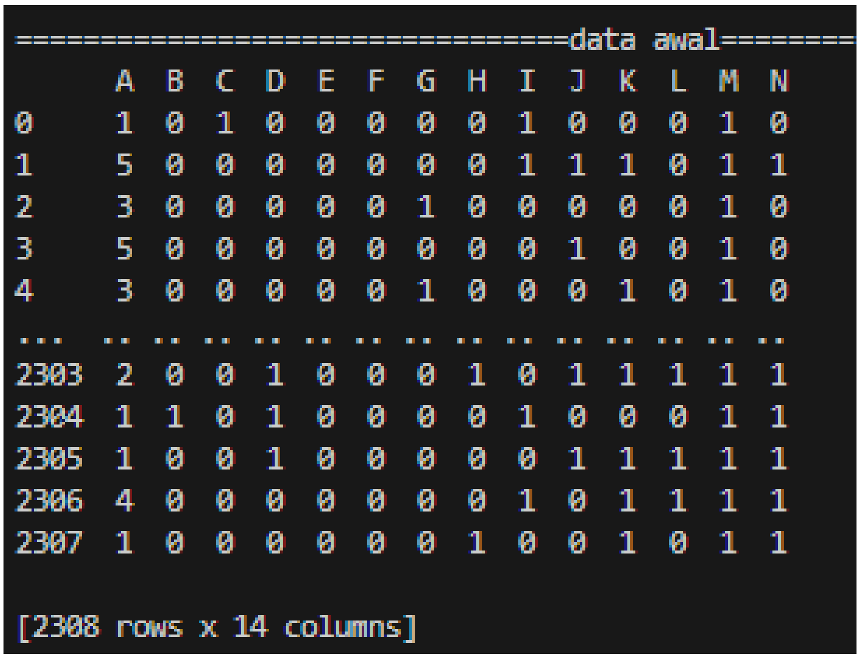
```python
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn import datasets
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score ,precision_score,recall_score,f1_score
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split
import sklearn.metrics
from sklearn.ensemble import RandomForestClassifier
from collections import Counter
import matplotlib.pyplot as plt
import matplotlib.colors as colors
import sklearn.model_selection as model_selection
from sklearn.metrics import (confusion_matrix, accuracy_score,
                             f1_score, ConfusionMatrixDisplay,
                             classification_report)
```

2. **Menampilkan semua data**



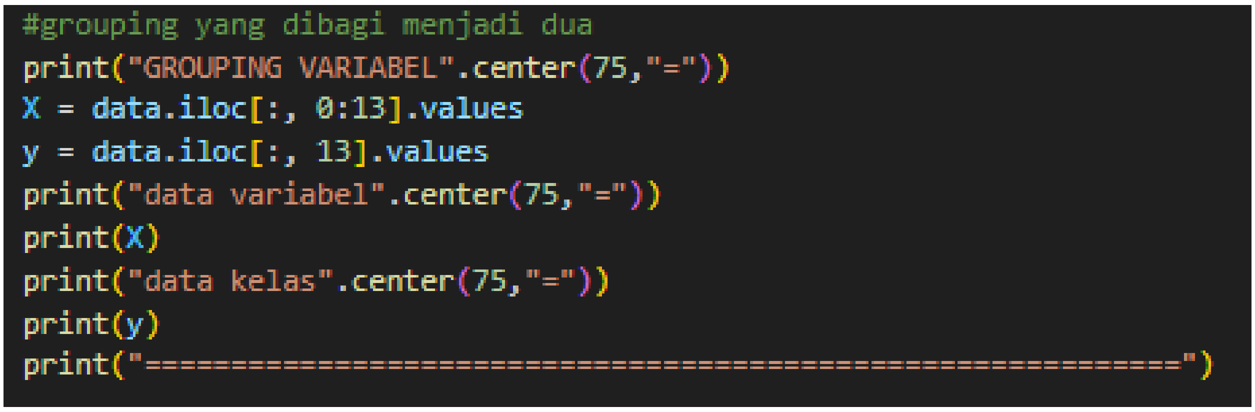
```python
#membaca data
dataframe = pd.read_excel('BlaBla.xlsx')

data=dataframe[['A','B','C',
                'D','E','F',
                'G','H',
                'I','J',
                'K','L',
                'M','N']]

print("data awal".center(75,"="))
print(data)
print("===========================================================================")
```

Hasil :



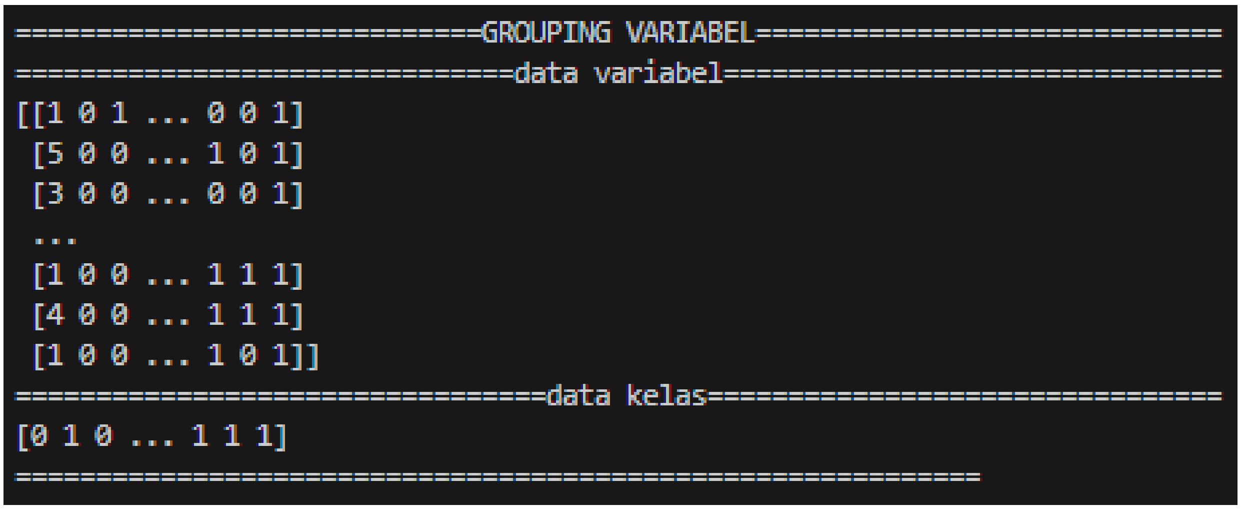
```
=================================data awal========
      A  B  C  D  E  F  G  H  I  J  K  L  M  N
0     1  0  1  0  0  0  0  0  1  0  0  0  1  0
1     5  0  0  0  0  0  0  0  1  1  1  0  1  1
2     3  0  0  0  0  0  1  0  0  0  0  0  1  0
3     5  0  0  0  0  0  0  0  0  1  0  0  1  0
4     3  0  0  0  0  0  1  0  0  0  1  0  1  0
...   .. .. .. .. .. .. .. .. .. .. .. .. .. ..
2303  2  0  0  1  0  0  0  1  0  1  1  1  1  1
2304  1  1  0  1  0  0  0  0  1  0  0  0  1  1
2305  1  0  0  1  0  0  0  0  0  1  1  1  1  1
2306  4  0  0  0  0  0  0  0  1  0  1  1  1  1
2307  1  0  0  0  0  0  0  1  0  0  1  0  1  1

[2308 rows x 14 columns]
```

3. Grouping



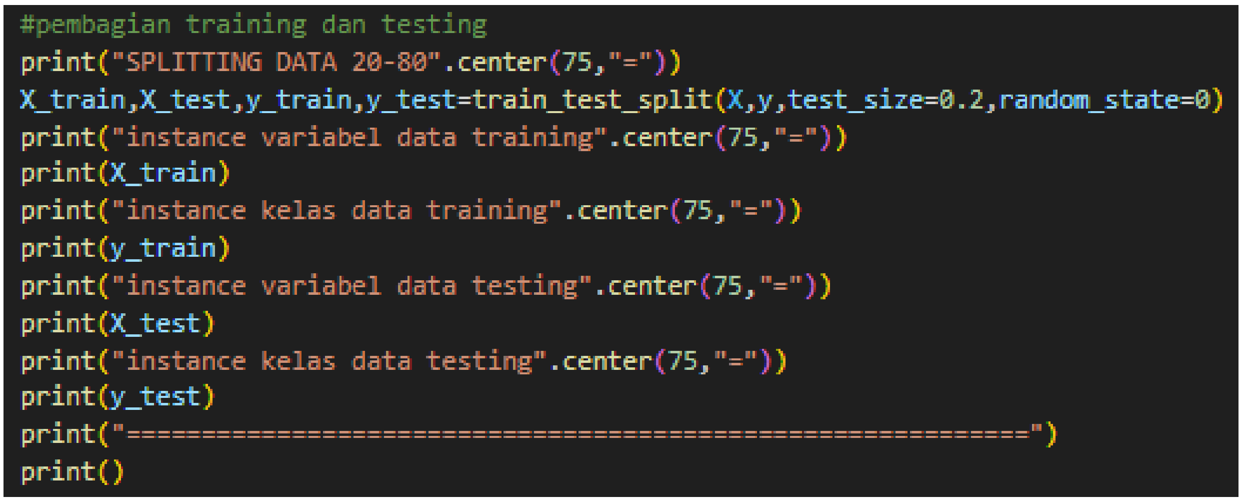
```
#grouping yang dibagi menjadi dua
print("GROUPING VARIABEL".center(75,"="))
X = data.iloc[:, 0:13].values
y = data.iloc[:, 13].values
print("data variabel".center(75,"="))
print(X)
print("data kelas".center(75,"="))
print(y)
print("=================================================================")
```

Hasil:



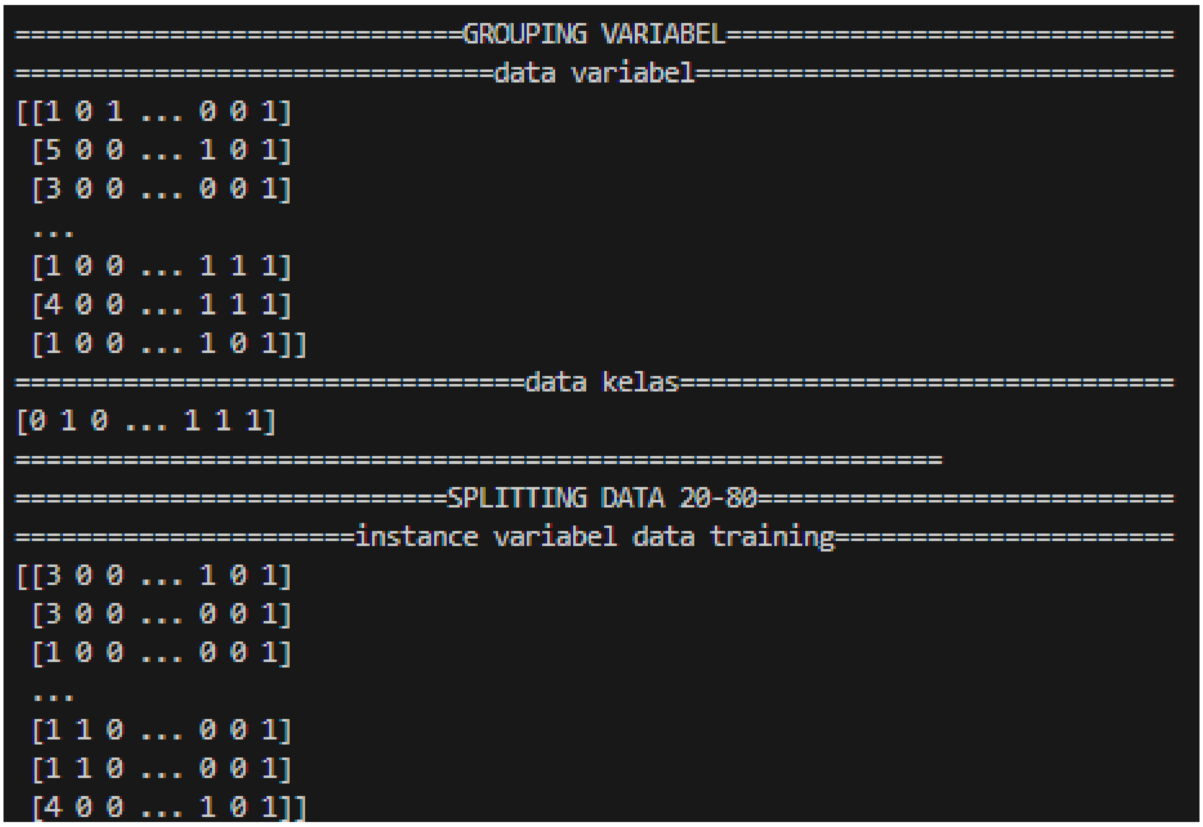
```
=============================GROUPING VARIABEL=============================
===============================data variabel===============================
[[1 0 1 ... 0 0 1]
 [5 0 0 ... 1 0 1]
 [3 0 0 ... 0 0 1]
 ...
 [1 0 0 ... 1 1 1]
 [4 0 0 ... 1 1 1]
 [1 0 0 ... 1 0 1]]
================================data kelas=================================
[0 1 0 ... 1 1 1]
============================================================
```

4. Training dan Testing



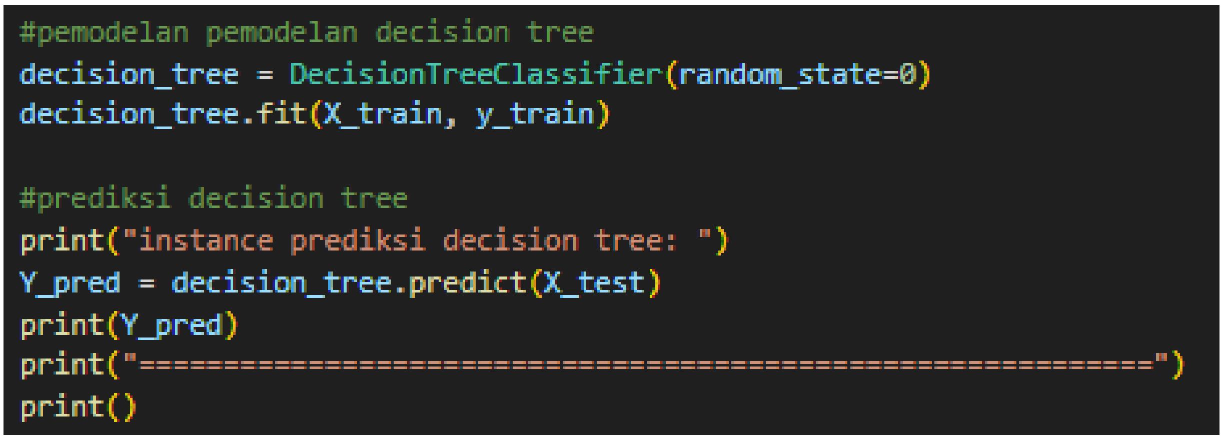
```
#pembagian training dan testing
print("SPLITTING DATA 20-80".center(75,"="))
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=0)
print("instance variabel data training".center(75,"="))
print(X_train)
print("instance kelas data training".center(75,"="))
print(y_train)
print("instance variabel data testing".center(75,"="))
print(X_test)
print("instance kelas data testing".center(75,"="))
print(y_test)
print("============================================================")
print()
```

Hasil:



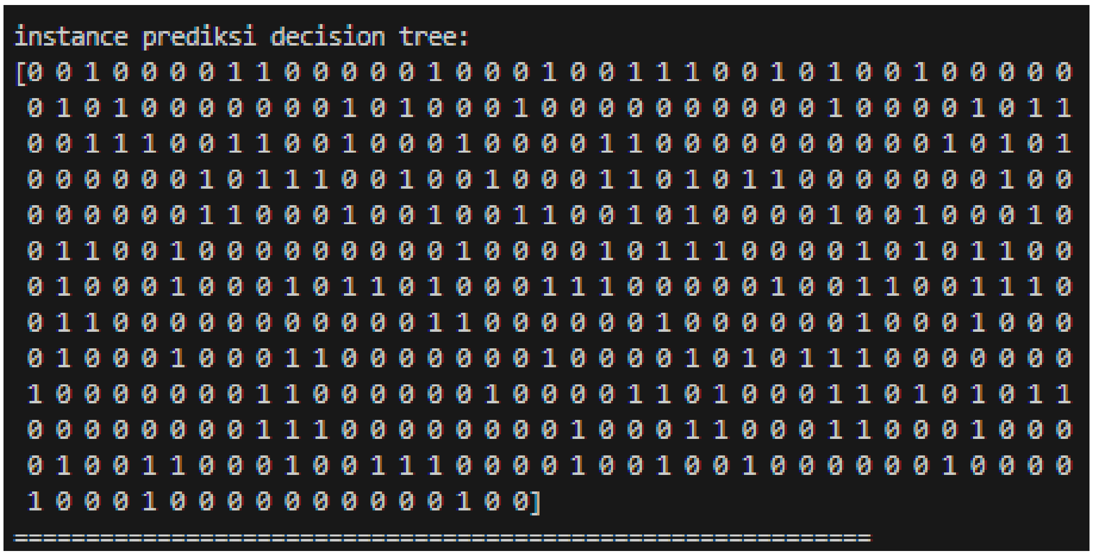
```
=============================GROUPING VARIABEL=============================
===============================data variabel===============================
[[1 0 1 ... 0 0 1]
 [5 0 0 ... 1 0 1]
 [3 0 0 ... 0 0 1]
 ...
 [1 0 0 ... 1 1 1]
 [4 0 0 ... 1 1 1]
 [1 0 0 ... 1 0 1]]
=================================data kelas================================
[0 1 0 ... 1 1 1]
============================================================
============================SPLITTING DATA 20-80===========================
======================instance variabel data training======================
[[3 0 0 ... 1 0 1]
 [3 0 0 ... 0 0 1]
 [1 0 0 ... 0 0 1]
 ...
 [1 1 0 ... 0 0 1]
 [1 1 0 ... 0 0 1]
 [4 0 0 ... 1 0 1]]
```

5. **Decision Tree**



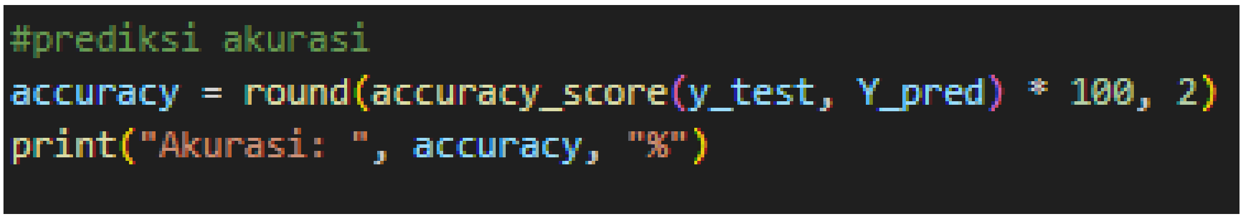
```
#pemodelan pemodelan decision tree
decision_tree = DecisionTreeClassifier(random_state=0)
decision_tree.fit(X_train, y_train)

#prediksi decision tree
print("instance prediksi decision tree: ")
Y_pred = decision_tree.predict(X_test)
print(Y_pred)
print("============================================================")
print()
```

Hasil:



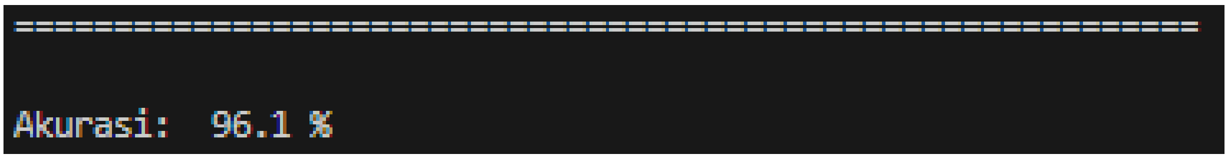
```
instance prediksi decision tree:
[0 0 1 0 0 0 0 1 1 0 0 0 0 0 1 0 0 0 1 0 0 1 1 1 0 0 1 0 1 0 0 1 0 0 0 0 0
 0 1 0 1 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 1
 0 0 1 1 1 0 0 1 1 0 0 1 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1
 0 0 0 0 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 0 1 0 1 1 0 0 0 0 0 0 0 1 0 0
 0 0 0 0 0 0 1 1 0 0 0 1 0 0 1 0 0 1 1 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 1 0
 0 1 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 1 1 0 0 0 0 1 0 1 0 1 1 0 0
 0 1 0 0 0 1 0 0 0 1 0 1 1 0 1 0 0 0 1 1 1 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 0
 0 1 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0
 0 1 0 0 0 1 0 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 1 1 1 0 0 0 0 0 0 0
 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 1 1 0 1 0 0 0 1 1 0 1 0 1 0 1 1
 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 0 0 0
 0 1 0 0 1 1 0 0 0 1 0 0 1 1 1 0 0 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 1 0 0 0 0
 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0]
============================================================
```

6. **Prediksi Akurasi**



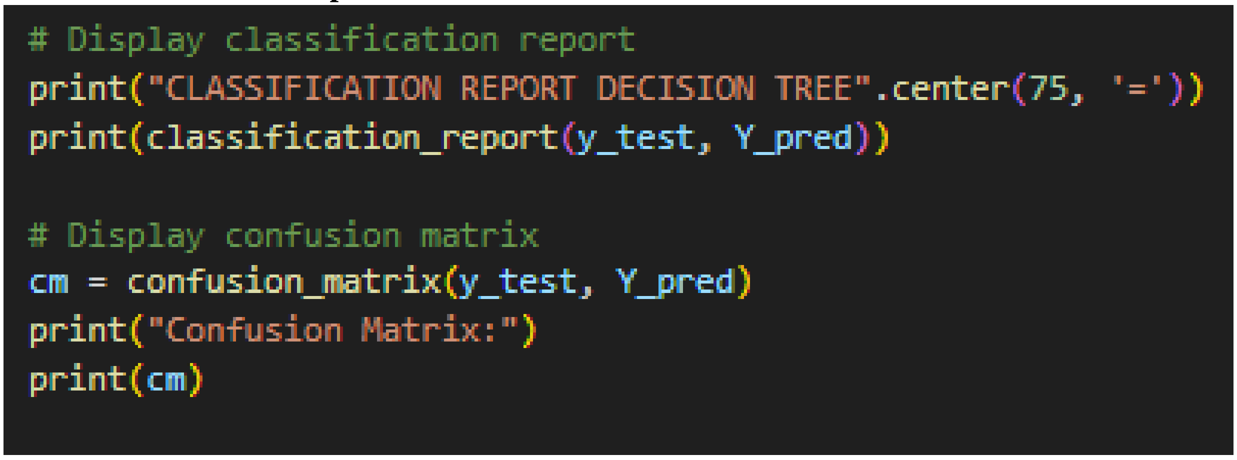
```
#prediksi akurasi
accuracy = round(accuracy_score(y_test, Y_pred) * 100, 2)
print("Akurasi: ", accuracy, "%")
```

Hasil:



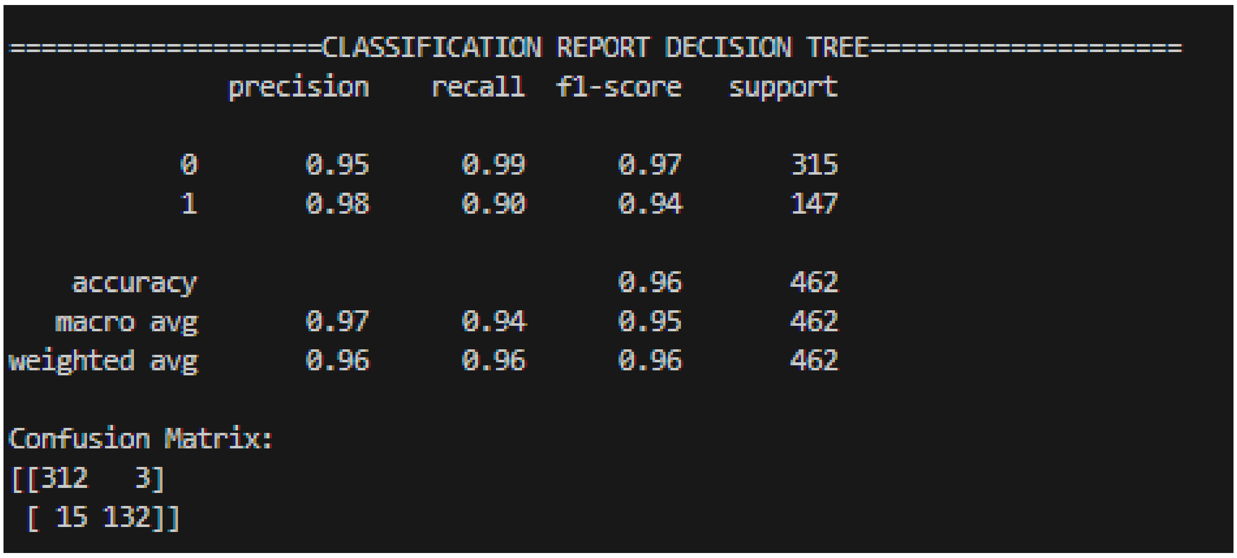
```
============================================================

Akurasi:  96.1 %
```

7. Classification Report



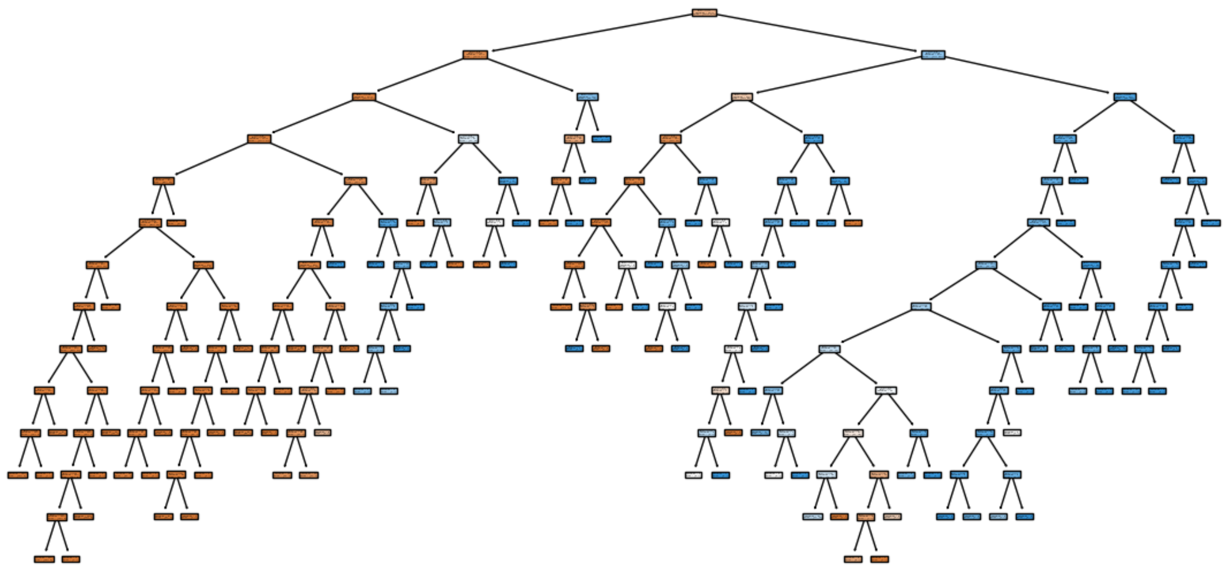
```
# Display classification report
print("CLASSIFICATION REPORT DECISION TREE".center(75, '='))
print(classification_report(y_test, Y_pred))

# Display confusion matrix
cm = confusion_matrix(y_test, Y_pred)
print("Confusion Matrix:")
print(cm)
```

Hasil:

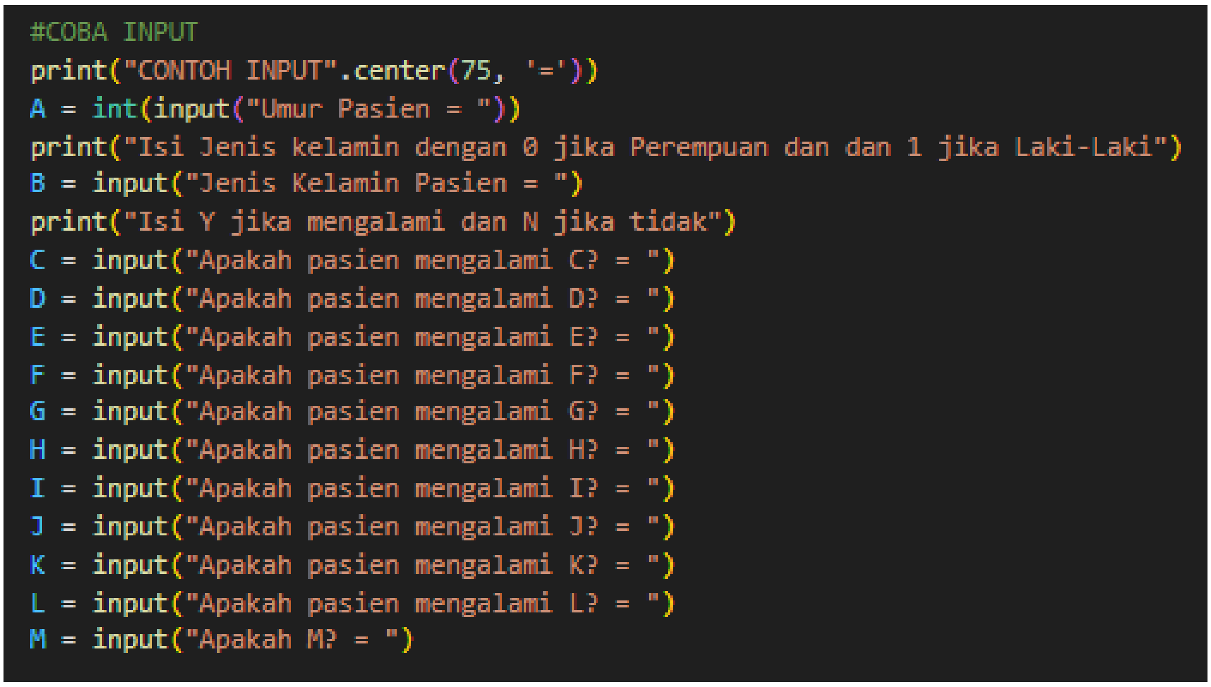


```
====================CLASSIFICATION REPORT DECISION TREE====================
              precision    recall  f1-score   support

           0       0.95      0.99      0.97       315
           1       0.98      0.90      0.94       147

    accuracy                           0.96       462
   macro avg       0.97      0.94      0.95       462
weighted avg       0.96      0.96      0.96       462

Confusion Matrix:
[[312   3]
 [ 15 132]]
```

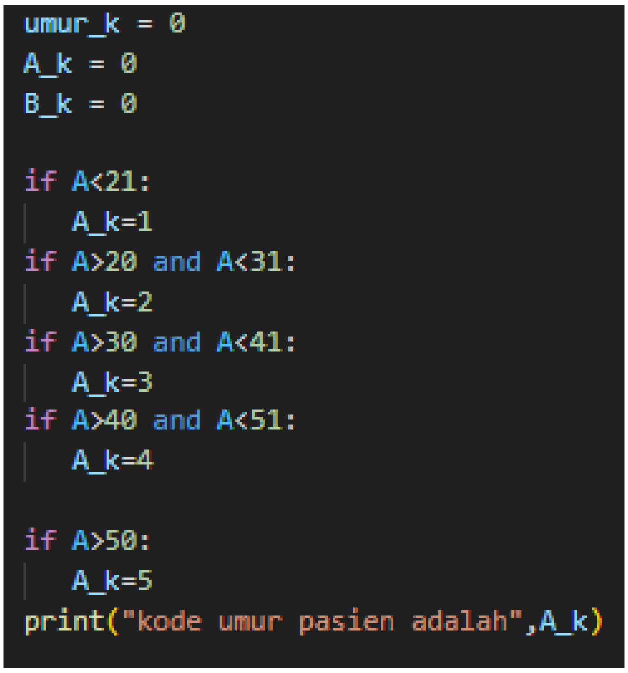
8. Visualisasi Decision Tree



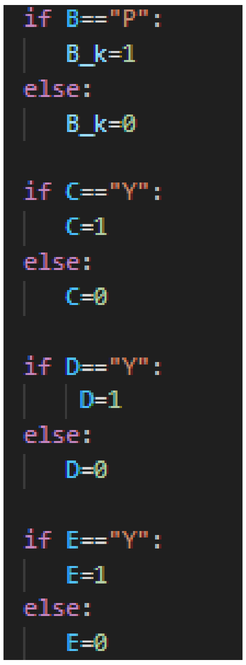
9. Klasifikasi / Prediksi Input



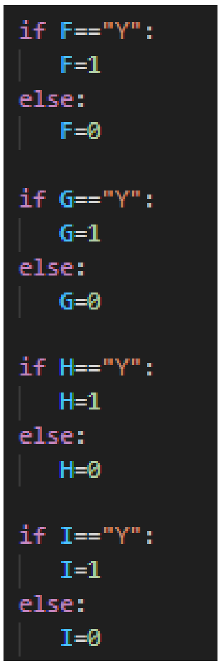
```python
#COBA INPUT
print("CONTOH INPUT".center(75, '='))
A = int(input("Umur Pasien = "))
print("Isi Jenis kelamin dengan 0 jika Perempuan dan dan 1 jika Laki-Laki")
B = input("Jenis Kelamin Pasien = ")
print("Isi Y jika mengalami dan N jika tidak")
C = input("Apakah pasien mengalami C? = ")
D = input("Apakah pasien mengalami D? = ")
E = input("Apakah pasien mengalami E? = ")
F = input("Apakah pasien mengalami F? = ")
G = input("Apakah pasien mengalami G? = ")
H = input("Apakah pasien mengalami H? = ")
I = input("Apakah pasien mengalami I? = ")
J = input("Apakah pasien mengalami J? = ")
K = input("Apakah pasien mengalami K? = ")
L = input("Apakah pasien mengalami L? = ")
M = input("Apakah M? = ")
```



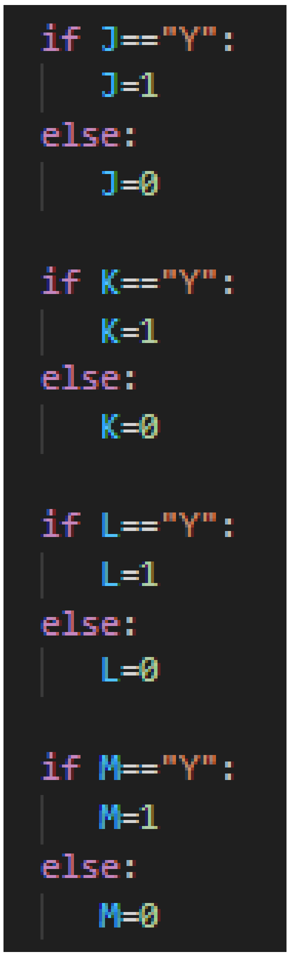
```python
umur_k = 0
A_k = 0
B_k = 0

if A<21:
    A_k=1
if A>20 and A<31:
    A_k=2
if A>30 and A<41:
    A_k=3
if A>40 and A<51:
    A_k=4

if A>50:
    A_k=5
print("kode umur pasien adalah",A_k)
```



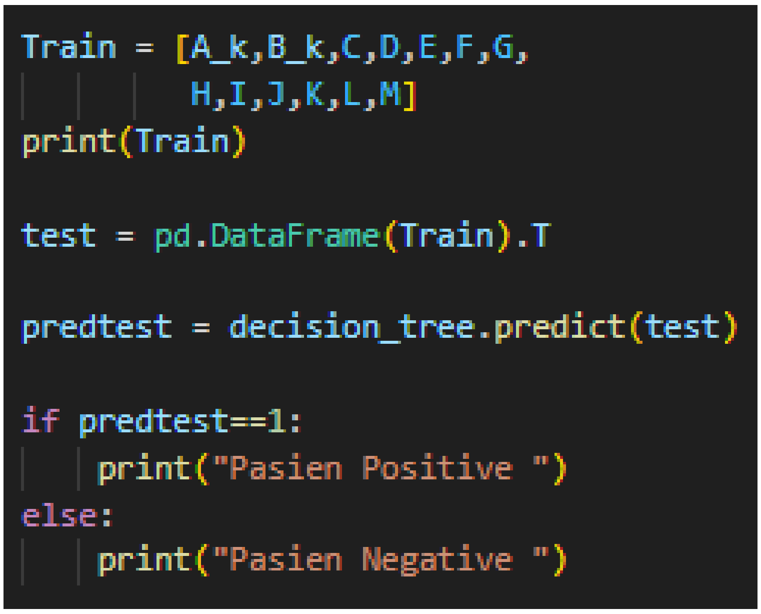
```python
if B=="P":
    B_k=1
else:
    B_k=0

if C=="Y":
    C=1
else:
    C=0

if D=="Y":
    D=1
else:
    D=0

if E=="Y":
    E=1
else:
    E=0
```



```python
if F=="Y":
    F=1
else:
    F=0

if G=="Y":
    G=1
else:
    G=0

if H=="Y":
    H=1
else:
    H=0

if I=="Y":
    I=1
else:
    I=0
```

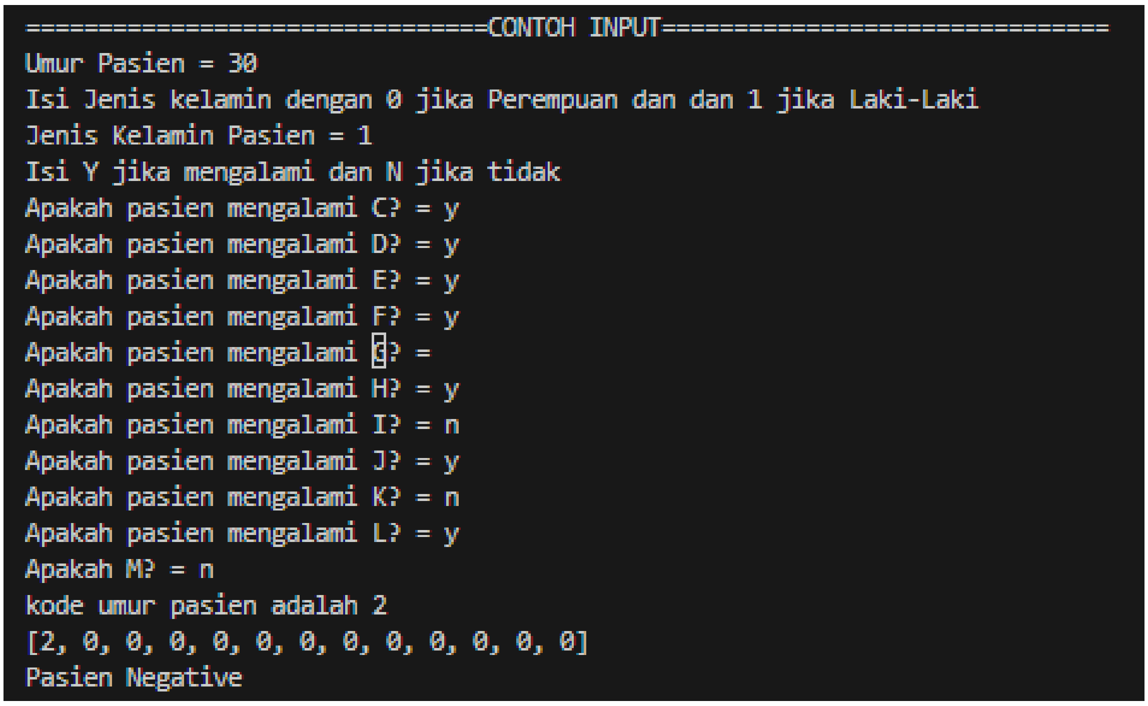


```python
if J=="Y":
    J=1
else:
    J=0

if K=="Y":
    K=1
else:
    K=0

if L=="Y":
    L=1
else:
    L=0

if M=="Y":
    M=1
else:
    M=0
```



```python
Train = [A_k,B_k,C,D,E,F,G,
            H,I,J,K,L,M]
print(Train)

test = pd.DataFrame(Train).T

predtest = decision_tree.predict(test)

if predtest==1:
    print("Pasien Positive ")
else:
    print("Pasien Negative ")
```

Hasil:



```
================================CONTOH INPUT================================
Umur Pasien = 30
Isi Jenis kelamin dengan 0 jika Perempuan dan dan 1 jika Laki-Laki
Jenis Kelamin Pasien = 1
Isi Y jika mengalami dan N jika tidak
Apakah pasien mengalami C? = y
Apakah pasien mengalami D? = y
Apakah pasien mengalami E? = y
Apakah pasien mengalami F? = y
Apakah pasien mengalami G? =
Apakah pasien mengalami H? = y
Apakah pasien mengalami I? = n
Apakah pasien mengalami J? = y
Apakah pasien mengalami K? = n
Apakah pasien mengalami L? = y
Apakah M? = n
kode umur pasien adalah 2
[2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
Pasien Negative
```