IMPORT LIBRARY

```
import pandas as pd
import numpy as np
```

LOAD DATASET

```
df = pd.read_csv('kidney_disease.csv')
df.head()
```

```
   id   age    bp     sg   al   su     rbc        pc         pcc          ba  ...  pcv    wc   rc  htn   dm  cad  appet   pe  ane  classifica
0   0  48.0  80.0  1.020  1.0  0.0     NaN    normal  notpresent  notpresent  ...   44  7800  5.2  yes  yes   no   good   no   no
1   1   7.0  50.0  1.020  4.0  0.0     NaN    normal  notpresent  notpresent  ...   38  6000  NaN   no   no   no   good   no   no
2   2  62.0  80.0  1.010  2.0  3.0  normal    normal  notpresent  notpresent  ...   31  7500  NaN   no  yes   no   poor   no  yes
3   3  48.0  70.0  1.005  4.0  0.0  normal  abnormal     present  notpresent  ...   32  6700  3.9  yes   no   no   poor  yes  yes
4   4  51.0  80.0  1.010  2.0  0.0  normal    normal  notpresent  notpresent  ...   35  7300  4.6   no   no   no   good   no   no

5 rows × 26 columns
```

DROP FIELD

```
df = df.drop(['id','age'], axis=1)
```

```
df.head()
```

	bp	sg	al	su	rbc	pc	pcc	ba	bgr	bu	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classifi
0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	121.0	36.0	...	44	7800	5.2	yes	yes	no	good	no	no	
1	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	NaN	18.0	...	38	6000	NaN	no	no	no	good	no	no	
2	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	423.0	53.0	...	31	7500	NaN	no	yes	no	poor	no	yes	
3	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	117.0	56.0	...	32	6700	3.9	yes	no	no	poor	yes	yes	

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 24 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   bp              388 non-null    float64
 1   sg              353 non-null    float64
 2   al              354 non-null    float64
 3   su              351 non-null    float64
 4   rbc             248 non-null    object
 5   pc              335 non-null    object
 6   pcc             396 non-null    object
 7   ba              396 non-null    object
 8   bgr             356 non-null    float64
 9   bu              381 non-null    float64
 10  sc              383 non-null    float64
 11  sod             313 non-null    float64
 12  pot             312 non-null    float64
 13  hemo            348 non-null    float64
 14  pcv             330 non-null    object
 15  wc              295 non-null    object
 16  rc              270 non-null    object
 17  htn             398 non-null    object
 18  dm              398 non-null    object
 19  cad             398 non-null    object
 20  appet           399 non-null    object
 21  pe              399 non-null    object
 22  ane             399 non-null    object
 23  classification  400 non-null    object
dtypes: float64(10), object(14)
memory usage: 75.1+ KB
```

BREAKDOWN 1 DATATYPE

```
numerical = []
catgcols = []

for col in df.columns:
    if df[col].dtype=="float64":
        numerical.append(col)
    else:
        catgcols.append(col)

for col in df.columns:
    if col in numerical:
        df[col].fillna(df[col].median(), inplace=True)
    else:
        df[col].fillna(df[col].mode()[0], inplace=True)
```

```
numerical
```

```
['bp', 'sg', 'al', 'su', 'bgr', 'bu', 'sc', 'sod', 'pot', 'hemo']
```

```
catgcols
```

```
['rbc',
 'pc',
 'pcc',
 'ba',
 'pcv',
 'wc',
 'rc',
 'htn',
 'dm',
 'cad',
 'appet',
 'pe',
 'ane',
 'classification']
```

BREAKDOWN CLASSIFICATION TYPE NUMERIC

```
df['classification'].value_counts()
```

```
ckd       248
notckd    150
```

```
ckd\t       2
Name: classification, dtype: int64
```

DATA ANOMALI OUTLIER ckd\t

```
df['classification'] = df['classification'].replace(['ckd\t'], 'ckd')
```

```
df['classification'].value_counts()
```

```
ckd       250
notckd    150
Name: classification, dtype: int64
```

TRANSFORM INDEX & LABEL DATASET

```
index_col = [col for col in df.columns if col != 'classification']
label_col = 'classification'
```

```
df[label_col].value_counts()
```

```
ckd       250
notckd    150
Name: classification, dtype: int64
```

TRANSFORM DATASET MODELING

```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

for col in catgcols:
    df[col] = le.fit_transform(df[col])
```

```
df['classification'] = le.fit_transform(df['classification'])
```

```
x = df[index_col]
y = df[label_col]
```

DATASET CLEAN

```
df.head()
```

	bp	sg	al	su	rbc	pc	pcc	ba	bgr	bu	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	80.0	1.020	1.0	0.0	1	1	0	0	121.0	36.0	...	32	72	34	1	4	1	0	0	0	0
1	50.0	1.020	4.0	0.0	1	1	0	0	121.0	18.0	...	26	56	34	0	3	1	0	0	0	0
2	80.0	1.010	2.0	3.0	1	1	0	0	423.0	53.0	...	19	70	34	0	4	1	1	0	1	0
3	70.0	1.005	4.0	0.0	1	0	1	0	117.0	56.0	...	20	62	19	1	3	1	1	1	1	0
4	80.0	1.010	2.0	0.0	1	1	0	0	106.0	26.0	...	23	68	27	0	3	1	0	0	0	0

5 rows × 24 columns

EXPORT DATASET CLEAN

```
df.to_csv('kidney_disease_updated.csv')
```

MODELING BUILD

```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)
```

```
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree

dtc = DecisionTreeClassifier(
    ccp_alpha=0.0, class_weight=None, criterion='entropy',
    max_depth=4, max_features=None, max_leaf_nodes=None,
    min_impurity_decrease=0.0,
    min_samples_split=2,min_weight_fraction_leaf=0.0,
    random_state=42, splitter='best'
)

model = dtc.fit(x_train, y_train)

dtc_acc = accuracy_score(y_test, dtc.predict(x_test))

print(f"Akurasi Data Training = {accuracy_score(y_train, dtc.predict(x_train))}")
print(f"Akurasi Data Testing = {dtc_acc} \n")

print(f"Confusion Matrix : \n{confusion_matrix(y_test, dtc.predict(x_test))}\n")
confusion = confusion_matrix(y_test, dtc.predict(x_test))
tn,fp,fn,tp = confusion.ravel()
print(f"classification_report : \n{classification_report(y_test, dtc.predict(x_test))}\n")
```

```
Akurasi Data Training = 0.99375
Akurasi Data Testing = 1.0

Confusion Matrix :
[[52  0]
 [ 0 28]]

classification_report :
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        52
           1       1.00      1.00      1.00        28

    accuracy                           1.00        80
   macro avg       1.00      1.00      1.00        80
weighted avg       1.00      1.00      1.00        80
```

SIMULASI MODEL

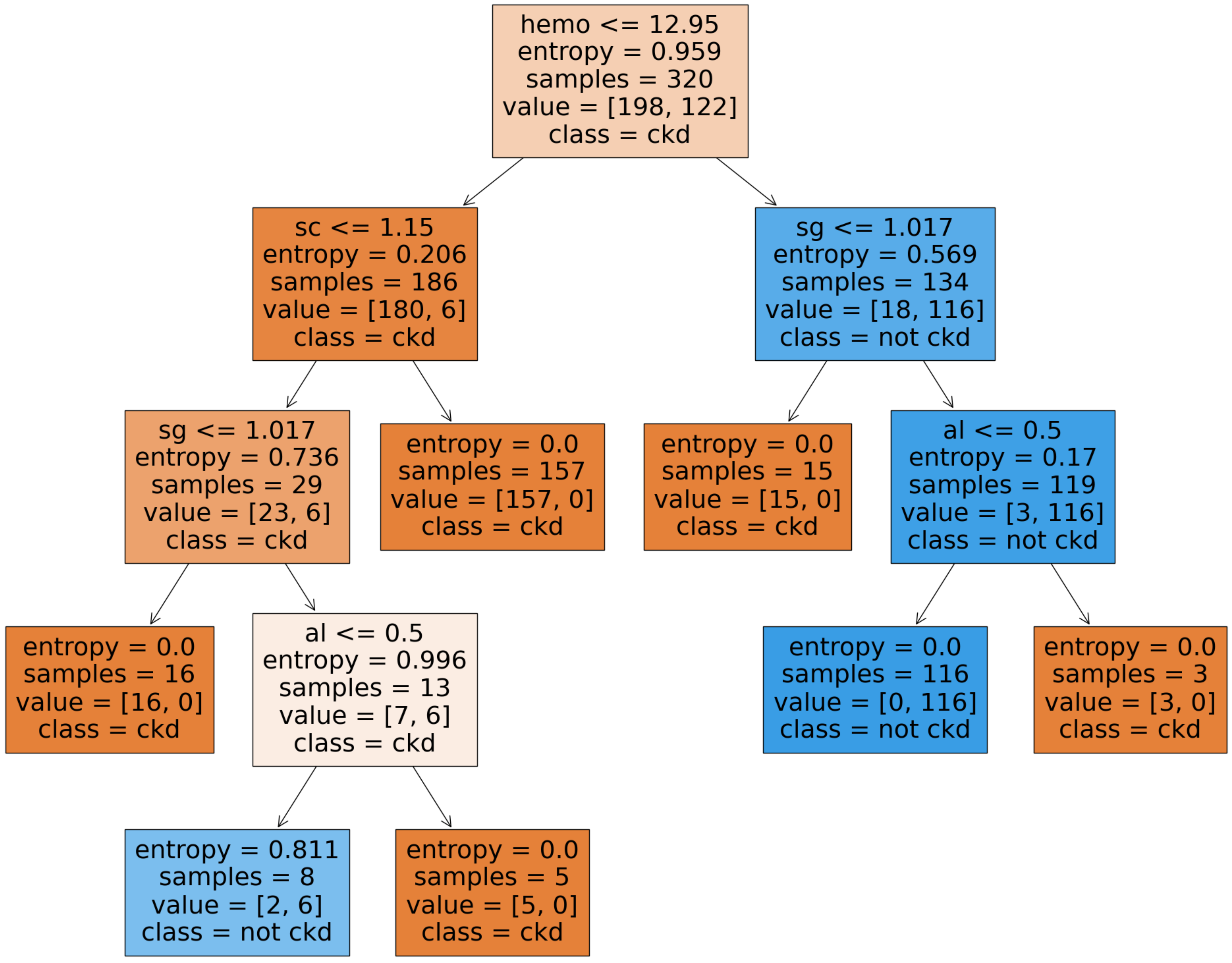
```
input_data = (80,1.02,1,0,1,1,0,0,121,36,1.2,138,4.4,15.4,32,72,34,1,4,1,0,0,0)
```

```
input_data_as_numpy_array = np.array(input_data)

input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = model.predict(input_data_reshaped)

print(prediction)

if (prediction[0] == 0):
    print('Pasien tidak terkena batu ginjal')
else:
    print('Pasien terkena batu ginjal')
```

```
[0]
Pasien tidak terkena batu ginjal
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but Decision
  warnings.warn(
```

VISUALISASI

```
import matplotlib.pyplot as plt
fig = plt.figure(figsize=(25,20))
_ = tree.plot_tree(model,
                   feature_names=index_col,
                   class_names=['ckd', 'not ckd'],
                   filled=True
)
```


hemo <= 12.95
entropy = 0.959
samples = 320
value = [198, 122]
class = ckd
sc <= 1.15
entropy = 0.206
samples = 186
value = [180, 6]
class = ckd
sg <= 1.017
entropy = 0.569
samples = 134
value = [18, 116]
class = not ckd
sg <= 1.017
entropy = 0.736
samples = 29
value = [23, 6]
class = ckd
entropy = 0.0
samples = 157
value = [157, 0]
class = ckd
entropy = 0.0
samples = 15
value = [15, 0]
class = ckd
al <= 0.5
entropy = 0.17
samples = 119
value = [3, 116]
class = not ckd
entropy = 0.0
samples = 16
value = [16, 0]
class = ckd
al <= 0.5
entropy = 0.996
samples = 13
value = [7, 6]
class = ckd
entropy = 0.0
samples = 116
value = [0, 116]
class = not ckd
entropy = 0.0
samples = 3
value = [3, 0]
class = ckd
entropy = 0.811
samples = 8
value = [2, 6]
class = not ckd
entropy = 0.0
samples = 5
value = [5, 0]
class = ckd