In [1]:

```
#Write Python Code to demonstrate implementation of Decision Trees Using Python.Use BREAST CANCER Dataset
```

In [13]:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_breast_cancer
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn import tree

```

In [14]:

```
data = load_breast_cancer()

```

In [15]:

```
data.feature_names # The names of the features
```

Out[15]:

```
array(['mean radius', 'mean texture', 'mean perimeter', 'mean area',
       'mean smoothness', 'mean compactness', 'mean concavity',
       'mean concave points', 'mean symmetry', 'mean fractal dimension',
       'radius error', 'texture error', 'perimeter error', 'area error',
       'smoothness error', 'compactness error', 'concavity error',
       'concave points error', 'symmetry error',
       'fractal dimension error', 'worst radius', 'worst texture',
       'worst perimeter', 'worst area', 'worst smoothness',
       'worst compactness', 'worst concavity', 'worst concave points',
       'worst symmetry', 'worst fractal dimension'], dtype='<U23')
```

In [16]:

```
data.target_names #The names of the classes
```

Out[16]:

```
array(['malignant', 'benign'], dtype='<U9')
```

In [17]:

```
data.data
```

Out[17]:

```
array([[1.799e+01, 1.038e+01, 1.228e+02, ..., 2.654e-01, 4.601e-01,
        1.189e-01],
       [2.057e+01, 1.777e+01, 1.329e+02, ..., 1.860e-01, 2.750e-01,
        8.902e-02],
       [1.969e+01, 2.125e+01, 1.300e+02, ..., 2.430e-01, 3.613e-01,
        8.758e-02],
       ...,
       [1.660e+01, 2.808e+01, 1.083e+02, ..., 1.418e-01, 2.218e-01,
        7.820e-02],
       [2.060e+01, 2.933e+01, 1.401e+02, ..., 2.650e-01, 4.087e-01,
        1.240e-01],
       [7.760e+00, 2.454e+01, 4.792e+01, ..., 0.000e+00, 2.871e-01,
        7.039e-02]])
```

In [33]:

```
# Split into training and test
X_train, X_test, y_train, y_test = train_test_split(data.data[:,0:9],data.target,shuffle=True,
                                                    test_size=0.3, random_state=42)
```

In [30]:

```
# DECISION TREE
# initialize the model with standard parameters
clf_dt = DecisionTreeClassifier(criterion="entropy")
# train the model
clf_dt.fit(X_train,y_train)
```

Out[30]:

```
DecisionTreeClassifier(criterion='entropy')
```

In [31]:

```
# Evaluating on the test data
y_test_pred = clf_dt.predict(X_test);
a_dt_test = accuracy_score(y_test, y_test_pred);
```

In [32]:

```
print("Training data accuracy is : " )
a_dt_test
```

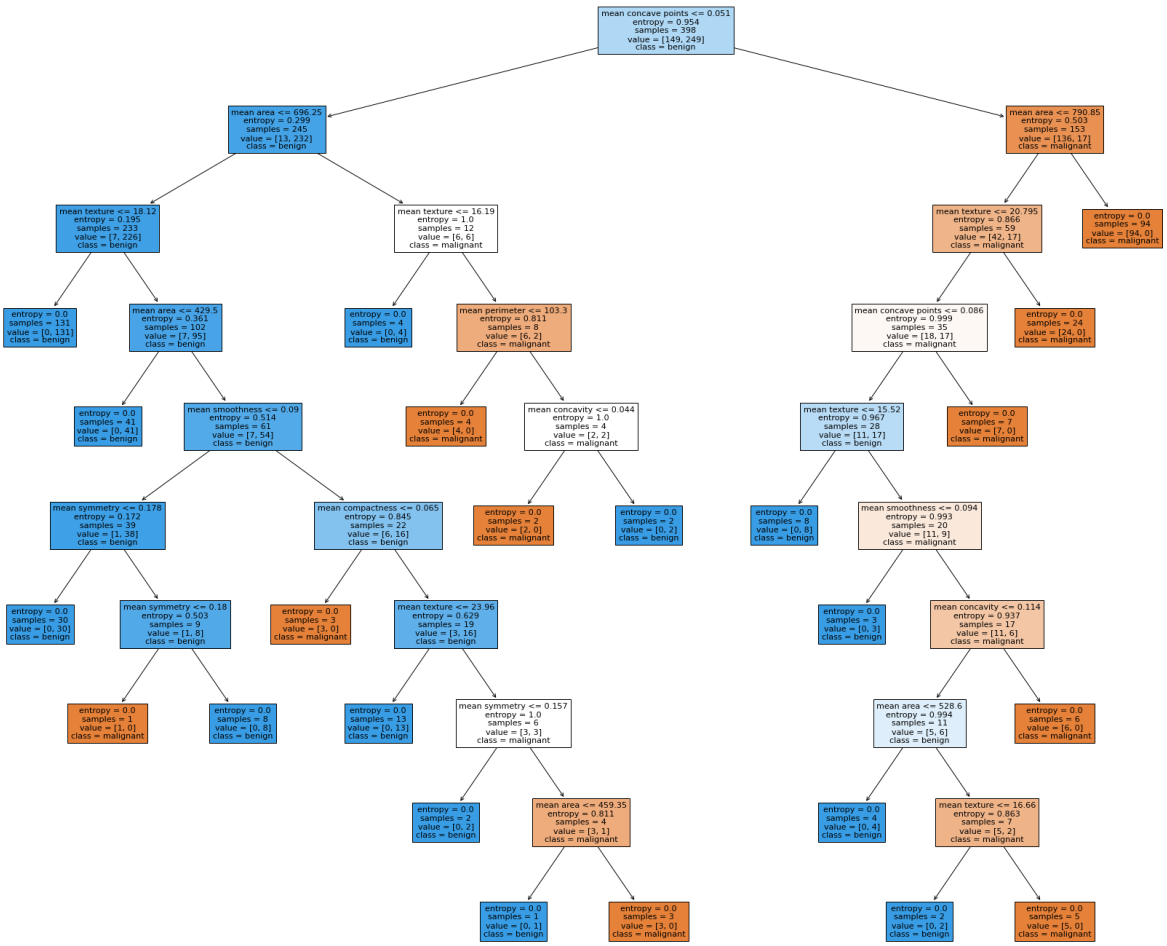
```
Training data accuracy is :
```

Out[32]:

```
0.9649122807017544
```

In [28]:

```
# Visualizing the decision tree (optional)
fig = plt.figure(figsize=(30,25))
_ = tree.plot_tree(clf_dt,feature_names=data.feature_names,
                   class_names=data.target_names,
                   filled=True)
```



In [ ]:

```
1
```

In [ ]:

```
1
```