In [5]:

```
def find_entropy(df):
    Class=df.keys()[-1]
    entropy=0
    values=df[Class].unique()
    for value in values:
        fraction=df[Class].value_counts()[value]/len(df[Class])
        entropy+=-fraction*np.log2(fraction)
    return entropy
def find_entropy_attribute(df,attribute):
    Class=df.keys()[-1]
    target_variables=df[Class].unique()
    variables=df[attribute].unique()
    entropy2=0
    for variable in variables:
        entropy=0
        for target_variable in target_variables:
            num=len(df[attribute][df[attribute]==variable][df[Class]==target_variable])
            den=len(df[attribute][df[attribute]==variable])
            fraction=num/(den+eps)
            entropy+=-fraction*log(fraction+eps)
            fraction2=den/len(df)
            entropy2+=-fraction2*entropy
    return abs(entropy2)
def find_winner(df):
    Entropy_att=[]
    IG=[]
    for key in df.keys()[:-1]:
        IG.append(find_entropy(df)-find_entropy_attribute(df,key))
    return df.keys()[:-1][np.argmax(IG)]
def get_subtable(df,node,value):
    return df[df[node]==value].reset_index(drop=True)
def buildTree(df,tree=None):
    Class=df.keys()[-1]
    node=find_winner(df)
    attValue=np.unique(df[node])
    if tree is None:
        tree={}
        tree[node]={}
    for value in attValue:
        subtable=get_subtable(df,node,value)
        clValue,counts=np.unique(subtable['PLAY'],return_counts=True)
        if len(counts)==1:
            tree[node][value]=clValue[0]
        else:
            tree[node][value]=buildTree(subtable)
        return tree
import pandas as pd
import numpy as np
eps=np.finfo(float).eps
from numpy import log2 as log
df=pd.read_csv("C:\\Users\\jyothi\\Desktop\\one.csv")
print("\n Given Play tennis data set:\n\n",df)
tree=buildTree(df)
import pprint
pprint.pprint(tree)
test={'SKY':'sunny','AIRTEMP':'cold','HUMIDITY':'normal','WIND':'strong','WATER':'warm','FORECAST':'sample'}
def func(test,tree,default=None):
    attribute=next(iter(tree))
    print(attribute)
    if test[attribute]in tree[attribute].keys():
        print(tree[attribute].keys())
        print(test[attribute])
        result=tree[attribute][test[attribute]]
        if isinstance(result,dict):
            return func(test,result)
        else:
            return result
    else:
        return default
ans=func(test,tree)
print(ans)
```

```
 Given Play tennis data set:

      SKY AIRTEMP HUMIDITY    WIND WATER FORECAST PLAY
0  sunny    warm   normal  strong  warm     same  yes
1  sunny    warm     high  strong  warm     same  yes
2  rainy    cold     high  strong  warm   change   no
3  rainy    cold     high  strong  cool   change  yes
{'SKY': {'rainy': {'WATER': {'cool': 'yes'}}}}
SKY
None
```

In [ ]: