Write a program to demonstrate the working of the decision tree based ID3 algorithm. Use an appropriate data set for building the decision tree and apply this knowledge to classify a new sample

Lab3data.csv

```
Outlook,Temperature,Humidity,Wind,PlayTennis
Sunny,Hot,High,Weak,No
Sunny,Hot,High,Strong,No
Overcast,Hot,High,Weak,Yes
Rainy,Mild,High,Weak,Yes
Rainy,Cool,Normal,Weak,Yes
Rainy,Cool,Normal,Strong,No
Overcast,Cool,Normal,Strong,Yes
Sunny,Mild,High,Weak,No
Sunny,Cool,Normal,Weak,Yes
Rainy,Mild,Normal,Weak,Yes
Sunny,Mild,Normal,Strong,Yes
Overcast,Mild,High,Strong,Yes
Overcast,Hot,Normal,Weak,Yes
Rainy,Mild,High,Strong,No
```

Program4.py

```
import csv
import math

def major_class(attrs, data, target):
    freq = {}
    i = attrs.index(target)
    for row in data:
        freq[row[i]] = freq.get(row[i], 0) + 1
    return max(freq, key=freq.get)

def entropy(attrs, data, target):
    freq = {}
    entropy_val = 0
    i = len(attrs) - 1
    for row in data:
        freq[row[i]] = freq.get(row[i], 0) + 1
    for val in freq.values():
        entropy_val += (-val / len(data)) * math.log(val / len(data), 2)
    return entropy_val

def info_gain(attrs, data, attribute, target):
```

```
    freq = {}
    sub_entropy = 0
    i = attrs.index(attribute)
    for row in data:
        freq[row[i]] = freq.get(row[i], 0) + 1
    for key in freq.keys():
        prob = freq[key] / sum(freq.values())
        data_subset = [row for row in data if row[i] == key]
        sub_entropy += prob * entropy(attrs, data_subset, target)
    data_subset = [row for row in data if row[0] != attrs[0]]
    return (entropy(attrs, data_subset, target) - sub_entropy)

def choose_attr(data, attrs, target):
    best = attrs[0]
    max_gain = 0
    for attr in attrs:
        if attr != target:
            new_gain = info_gain(attrs, data, attr, target)
            if new_gain > max_gain:
                max_gain = new_gain
                best = attr
    return best

def get_values(data, attrs, attribute):
    i = attrs.index(attribute)
    values = []
    for row in data:
        if row[i] != attribute and row[i] not in values:
            values.append(row[i])
    return values

def get_data(data, attrs, best, val):
    i = attrs.index(best)
    new_data = [[row[j] for j in range(len(row)) if j != i] for row in data if
row[i] == val]
    return new_data

def build_tree(data, attrs, target):
    vals = [row[attrs.index(target)] for row in data]
    default = major_class(attrs, data, target)
    if not data or (len(attrs) - 1) <= 0:
        return default
    elif vals.count(vals[0]) == len(vals):
        return vals[0]
    else:
```

```
        best = choose_attr(data, attrs, target)
        tree = {best: {}}
        for val in get_values(data, attrs, best):
            new_data = get_data(data, attrs, best, val)
            new_attrs = attrs[:]
            new_attrs.remove(best)
            subtree = build_tree(new_data, new_attrs, target)
            tree[best][val] = subtree
        return tree

def classify(attrs, inst, tree):
    attribute = next(iter(tree))
    i = attrs.index(attribute)
    if inst[i] in tree[attribute].keys():
        result = tree[attribute][inst[i]]
        if isinstance(result, dict):
            return classify(attrs, inst, result)
        else:
            return result
    else:
        return None

file = open('Lab3data.csv')
data = list(csv.reader(file))
attrs = data[0]
tree = build_tree(data[1:], attrs, attrs[-1])

print('Decision Tree: \n', tree)

inst = input("Enter a test instance:").split(',')
print('Output Class: ', classify(attrs, inst, tree))
```

OUTPUT

Decision Tree:

{'Outlook': {'Sunny': {'Humidity': {'High': 'No', 'Normal': 'Yes'}}, 'Overcast': 'Yes', 'Rainy': {'Wind': {'Weak': 'Yes', 'Strong': 'No'}}}}

Enter a test instance:Rainy,cool,Normal,Weak

Output Class: Yes