**EXPERIMENT – 3**

**PROGRAM:**

```
import math


def entropy(data):
    if len(data) == 0:
        return 0
    pos = sum(1 for row in data if row[-1] == 'Yes')
    neg = len(data) - pos
    if pos == 0 or neg == 0:
        return 0
    p_pos = pos / len(data)
    p_neg = neg / len(data)
    return -p_pos * math.log2(p_pos) - p_neg * math.log2(p_neg)

def info_gain(data, attribute_index):
    total_entropy = entropy(data)
    values = set(row[attribute_index] for row in data)
    weighted_entropy = 0

    for value in values:
        subset = [row for row in data if row[attribute_index] == value]
        weighted_entropy += (len(subset) / len(data)) * entropy(subset)

    return total_entropy - weighted_entropy

def id3(data, attributes, target_values):
    if all(row[-1] == target_values[0] for row in data):
        return target_values[0]
```

```
    if all(row[-1] == target_values[1] for row in data):
        return target_values[1]

    if len(attributes) == 0:
        pos = sum(1 for row in data if row[-1] == target_values[0])
        neg = len(data) - pos
        return target_values[0] if pos >= neg else target_values[1]

    gains = [info_gain(data, i) for i in range(len(attributes))]
    best_index = gains.index(max(gains))
    best_attribute = attributes[best_index]

    tree = {best_attribute: {}}
    values = set(row[best_index] for row in data)

    remaining_attributes = attributes[:best_index] + attributes[best_index+1:]

    for value in values:
        subset = [row[:best_index] + row[best_index+1:] for row in data if row[best_index] == value]
        tree[best_attribute][value] = id3(subset, remaining_attributes, target_values)

    return tree

def classify(tree, sample, attributes):
    if not isinstance(tree, dict):
        return tree

    attribute = list(tree.keys())[0]
    attribute_index = attributes.index(attribute)
    value = sample[attribute_index]
```

```
    if value in tree[attribute]:
        subtree = tree[attribute][value]
        return classify(subtree, sample, attributes)
    else:
        return "Unknown"

n = int(input())
m = int(input())

attributes = [input() for _ in range(m)]

data = []
for _ in range(n):
    row = input().split(',')
    data.append([x.strip() for x in row])

target_values = list(set(row[-1] for row in data))

tree = id3(data, attributes, target_values)

print(tree)

sample = []
for i in range(m):
    value = input()
    sample.append(value)

result = classify(tree, sample, attributes)
print(result)
```