In [1]:
```
import numpy as np
```

In [2]:
```
# Example dataset with Boolean attributes (0 or 1) and the class label as last
column.
# Assume the dataset has features for [Outlook, Temperature, Humidity, Windy]
 and a class label.
# Example dataset with 4 attributes and a binary target (Yes/No).
data = np.array([
    ['sunny', 'hot', 'high', 'weak', 'no'],
    ['sunny', 'hot', 'high', 'strong', 'no'],
    ['overcast', 'hot', 'high', 'weak', 'yes'],
    ['rain', 'mild', 'high', 'weak', 'yes'],
    ['rain', 'cool', 'normal', 'weak', 'yes'],
    ['rain', 'cool', 'normal', 'strong', 'no'],
    ['overcast', 'cool', 'normal', 'strong', 'yes'],
    ['sunny', 'mild', 'high', 'weak', 'no'],
    ['sunny', 'cool', 'normal', 'weak', 'yes'],
    ['rain', 'mild', 'normal', 'weak', 'yes'],
])
```

In [3]:
```
# Function to implement the Find-S algorithm
def find_s(dataset):
    # The hypothesis is initially set to the first positive example
    hypothesis = dataset[0, :-1]  # Excluding the class label

    # Iterate through all examples in the dataset
    for example in dataset:
        if example[-1] == 'Yes':  # Consider only positive examples
            for i in range(len(hypothesis)):  # Check all attributes
                # If the current hypothesis value is different from the exampl
e's attribute value, generalize
                if hypothesis[i] != example[i]:
                    hypothesis[i] = '?'  # '?' represents generalization (any
 value can match)

    return hypothesis
```

In [4]:
```
# Apply the Find-S algorithm to the dataset
hypothesis = find_s(data)
print("Final hypothesis:", hypothesis)
```

```
Final hypothesis: ['sunny' 'hot' 'high' 'weak']
```