EX. NO: 1

DATE :

IMPLEMENTATION OF CANDIDATE –ELIMINATION ALGORITHM

AIM:

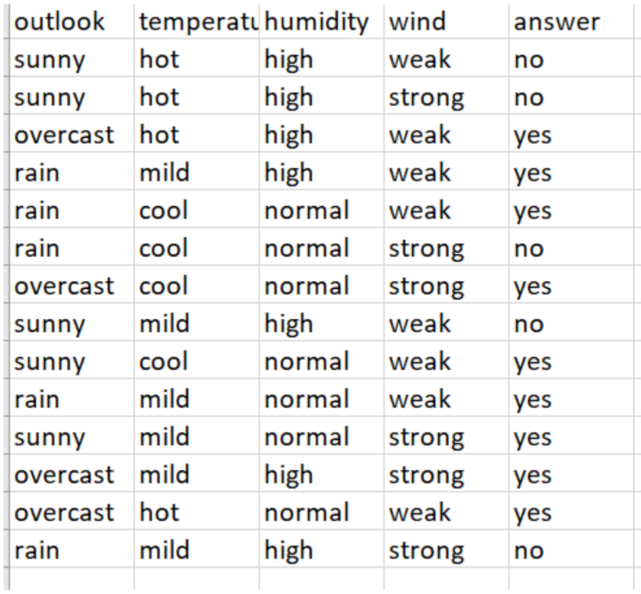
To implement and demonstrate the Candidate-Elimination algorithm, for a given set of training data examples stored in a .CSV file, to output a description of the set of all hypotheses consistent with the training examples.

ALGORITHM:

```
1. Load Data set.
2. Initialize General Hypothesis and Specific Hypothesis.
3. For each training example
4. If example is positive example
       if attribute_value == hypothesis_value:
          Do nothing
       else:
          replace attribute value with '?' (Basically generalizing it)
5. If example is Negative example
       Make generalize hypothesis more specific.
```

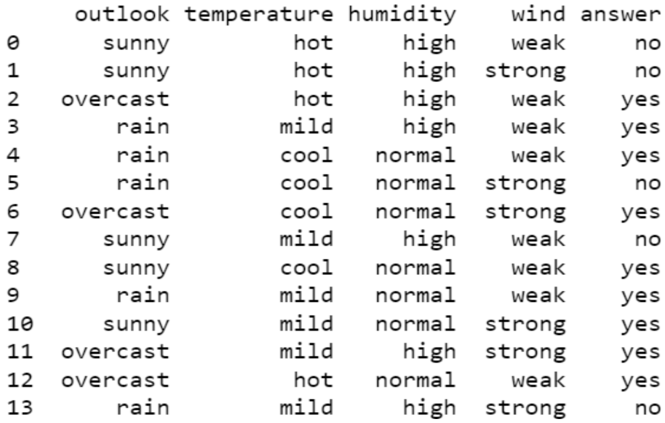
PROGRAM:

dataset.csv



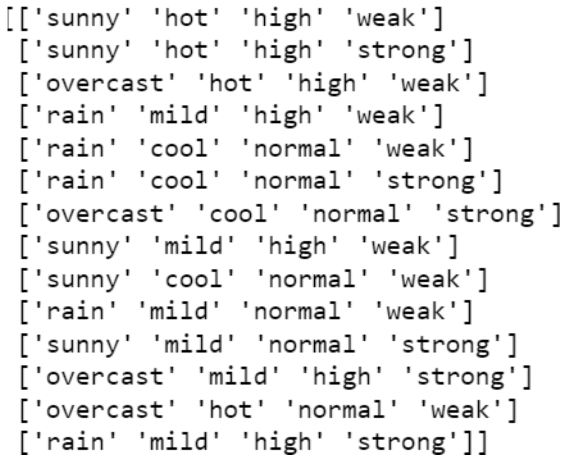
outlook	temperatu	humidity	wind	answer
sunny	hot	high	weak	no
sunny	hot	high	strong	no
overcast	hot	high	weak	yes
rain	mild	high	weak	yes
rain	cool	normal	weak	yes
rain	cool	normal	strong	no
overcast	cool	normal	strong	yes
sunny	mild	high	weak	no
sunny	cool	normal	weak	yes
rain	mild	normal	weak	yes
sunny	mild	normal	strong	yes
overcast	mild	high	strong	yes
overcast	hot	normal	weak	yes
rain	mild	high	strong	no

```
import numpy as np
import pandas as pd
 # Loading Data from a CSV File
 data = pd.DataFrame(data=pd.read_csv('E:\BALA\AI\Lab programs\pgms\dataset.csv'))
 print(data)
```



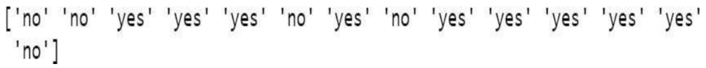
```
     outlook temperature humidity    wind answer
0      sunny         hot     high    weak     no
1      sunny         hot     high  strong     no
2   overcast         hot     high    weak    yes
3       rain        mild     high    weak    yes
4       rain        cool   normal    weak    yes
5       rain        cool   normal  strong     no
6   overcast        cool   normal  strong    yes
7      sunny        mild     high    weak     no
8      sunny        cool   normal    weak    yes
9       rain        mild   normal    weak    yes
10     sunny        mild   normal  strong    yes
11  overcast        mild     high  strong    yes
12  overcast         hot   normal    weak    yes
13      rain        mild     high  strong     no
```

```
 # Separating concept features from Target
 concepts = np.array(data.iloc[:,0:-1])
 print(concepts)
```



```
[['sunny' 'hot' 'high' 'weak']
 ['sunny' 'hot' 'high' 'strong']
 ['overcast' 'hot' 'high' 'weak']
 ['rain' 'mild' 'high' 'weak']
 ['rain' 'cool' 'normal' 'weak']
 ['rain' 'cool' 'normal' 'strong']
 ['overcast' 'cool' 'normal' 'strong']
 ['sunny' 'mild' 'high' 'weak']
 ['sunny' 'cool' 'normal' 'weak']
 ['rain' 'mild' 'normal' 'weak']
 ['sunny' 'mild' 'normal' 'strong']
 ['overcast' 'mild' 'high' 'strong']
 ['overcast' 'hot' 'normal' 'weak']
 ['rain' 'mild' 'high' 'strong']]
```

```
 # Isolating target into a separate DataFrame
 # copying last column to target array
 target = np.array(data.iloc[:,-1])
 print(target)
```



```
['no' 'no' 'yes' 'yes' 'yes' 'no' 'yes' 'no' 'yes' 'yes' 'yes' 'yes' 'yes'
 'no']
```

```
def learn(concepts, target):
    '''
    learn() function implements the learning method of the Candidate elimination algorithm.
    Arguments:
        concepts - a data frame with all the features
        target - a data frame with corresponding output values
    '''
    # Initialise S0 with the first instance from concepts
    # .copy() makes sure a new list is created instead of just pointing to the same memory location
    specific_h = concepts[0].copy()
    print("\nInitialization of specific_h and general_h")
    print(specific_h)
    #h=["#" for i in range(0,5)]
    #print(h)

    general_h = [["?" for i in range(len(specific_h))] for i in range(len(specific_h))]
    print(general_h)
    # The learning iterations
    for i, h in enumerate(concepts):
        # Checking if the hypothesis has a positive target
        if target[i] == "Yes":
            for x in range(len(specific_h)):
                # Change values in S & G only if values change
                if h[x] != specific_h[x]:
                    specific_h[x] = '?'
                    general_h[x][x] = '?'
        # Checking if the hypothesis has a positive target
        if target[i] == "No":
            for x in range(len(specific_h)):
                # For negative hyposthesis change values only in G
                if h[x] != specific_h[x]:
                    general_h[x][x] = specific_h[x]
                else:
                    general_h[x][x] = '?'
        print("\nSteps of Candidate Elimination Algorithm",i+1)
        print(specific_h)
        print(general_h)
    # find indices where we have empty rows, meaning those that are unchanged
    indices = [i for i, val in enumerate(general_h) if val == ['?', '?', '?', '?', '?', '?']]
    for i in indices:
        # remove those rows from general_h
        general_h.remove(['?', '?', '?', '?', '?', '?'])
    # Return final values
    return specific_h, general_h
s_final, g_final = learn(concepts, target)
print("\nFinal Specific_h:", s_final, sep="\n")
print("\nFinal General_h:", g_final, sep="\n")
```