Name : Mohammad Dayyan

Branch: MCA(AIML)

Section: A

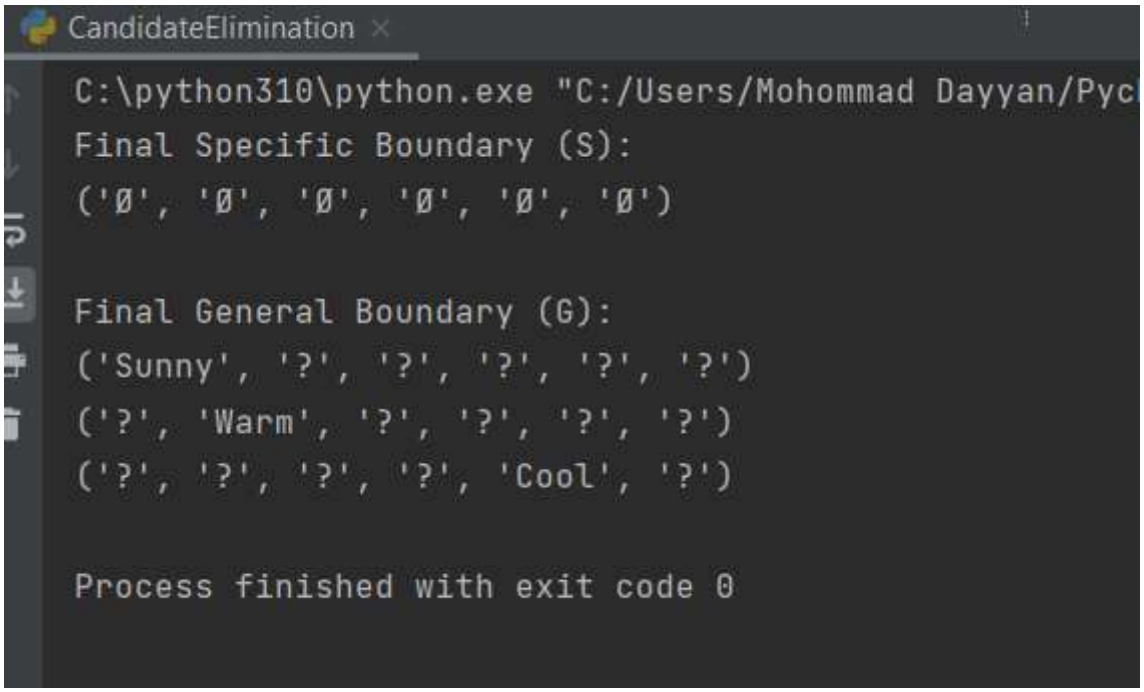
Batch: A1

Code:

```
def is_more_general(h1, h2):
    return all(x == '?' or (x != 'Ø' and (x == y or y == 'Ø')) for x, y in zip(h1, h2))

def min_generalizations(h, x):
    new_h = list(h)
    for i in range(len(h)):
        if h[i] == 'Ø':
            new_h[i] = x[i]
        elif h[i] != x[i]:
            new_h[i] = '?'
    return [tuple(new_h)]

def min_specializations(h, domains, x):
    results = []
    for i in range(len(h)):
        if h[i] == '?':
            for val in domains[i]:
                if val != x[i]:
                    new_h = list(h)
                    new_h[i] = val
                    results.append(tuple(new_h))
        elif h[i] != 'Ø':
            new_h = list(h)
            new_h[i] = 'Ø'
            results.append(tuple(new_h))
    return results

def candidate_elimination(examples, domains):
    S = [('Ø',) * len(domains)]
    G = [('?',) * len(domains)]

    for x, label in examples:
        if label == 'yes':
            G = [g for g in G if is_more_general(g, x)]
            new_S = []
            for s in S:
                if not is_more_general(x, s):
                    new_generalizations = min_generalizations(s, x)
                    new_S.extend(new_generalizations)
                else:
                    new_S.append(s)
            S = [s for s in new_S if any(is_more_general(g, s) for g in G)]
```

```
        else:
            S = [s for s in S if not is_more_general(s, x)]
            new_G = []
            for g in G:
                if is_more_general(g, x):
                    specializations = min_specializations(g, domains, x)
                    new_G.extend(s for s in specializations if any(is_more_general(s, s2) for s2 in S))
                else:
                    new_G.append(g)
            G = new_G

    return S, G

# Example Dataset
examples = [
    (('Sunny', 'Warm', 'Normal', 'Strong', 'Warm', 'Same'), 'yes'),
    (('Sunny', 'Warm', 'High', 'Strong', 'Warm', 'Same'), 'yes'),
    (('Rainy', 'Cold', 'High', 'Strong', 'Warm', 'Change'), 'no'),
    (('Sunny', 'Warm', 'High', 'Strong', 'Cool', 'Change'), 'yes')
]

# Attribute domains
domains = [
    ['Sunny', 'Rainy'],
    ['Warm', 'Cold'],
    ['High', 'Normal'],
    ['Strong', 'Weak'],
    ['Warm', 'Cool'],
    ['Same', 'Change']
]

# Run algorithm
S_final, G_final = candidate_elimination(examples, domains)

# Print results
print("Final Specific Boundary (S):")
for s in S_final:
    print(s)

print("\nFinal General Boundary (G):")
for g in G_final:
    print(g)
```

Output:


CandidateElimination
C:\python310\python.exe "C:/Users/Mohommad Dayyan/Pyc
Final Specific Boundary (S):
('Ø', 'Ø', 'Ø', 'Ø', 'Ø', 'Ø')
Final General Boundary (G):
('Sunny', '?', '?', '?', '?', '?')
('?', 'Warm', '?', '?', '?', '?')
('?', '?', '?', '?', 'Cool', '?')
Process finished with exit code 0