2.D)Implementation of Forward Chaining

AIM:

To implement Forward Chaining to derive all possible conclusions (facts) from a given knowledge base of rules and initial known facts.

CODE:

```
def forward_chaining(rules, facts, goal):
   inferred_facts = set(facts)
   agenda = list(facts)
   path = {}

   while agenda:
       current_fact = agenda.pop(0)

       for rule in rules:
           if all(antecedent in inferred_facts for antecedent in
rule['antecedents']):
               if rule['consequent'] not in inferred_facts:
                   inferred_facts.add(rule['consequent'])
                   agenda.append(rule['consequent'])
                   path[rule['consequent']] = rule['antecedents']
                   print(f"Inferred: {rule['consequent']} from
{rule['antecedents']}")

   return goal in inferred_facts, path


def construct_path(path, goal):
 if goal not in path:
   return [goal]

 full_path = [goal]
 for antecedents in path[goal]:
     full_path.extend(construct_path(path,antecedents))
 return full_path


rules = [
   {'antecedents': ['A'], 'consequent': 'B'},
   {'antecedents': ['B'], 'consequent': 'C'},
   {'antecedents': ['C', 'D'], 'consequent': 'E'},
   {'antecedents': ['F'], 'consequent': 'D'},
]

facts = ['A', 'F']
goal = 'E'

result, path = forward_chaining(rules, facts, goal)

if result:
   print(f"Goal '{goal}' can be proven.")
   print("Inference Path:", construct_path(path, goal))
else:
   print(f"Goal '{goal}' cannot be proven.")
```

OUTPUT:

```
Inferred: B from ['A']
Inferred: C from ['B']
Inferred: D from ['F']
Inferred: E from ['C', 'D']
Goal 'E' can be proven.
Inference Path: ['E', 'C', 'B', 'A', 'D', 'F']
```

RESULT:

The code is executed as expected and the output have been verified successfully.