Lab – 7 : Forward Reasoning Algorithm

```
import re

# Define the Knowledge Base class
class KnowledgeBase:
    def __init__(self):
        self.facts = set()
        self.rules = []

    def add_fact(self, fact):
        self.facts.add(fact)

    def add_rule(self, rule):
        self.rules.append(rule)

    def apply_rules(self):
        new_facts = set()
        for rule in self.rules:
            # Check if the rule can be applied based on the facts
            if rule[0](self.facts):
                # If the rule is triggered, add the result (consequent) to new facts
                new_facts.add(rule[1])
        return new_facts

    def get_facts(self):
        return self.facts

# Define a simple function to parse sentences into predicates
```

```
def parse_sentence(sentence):
    # Define regex patterns to match simple sentences
    patterns = [
        (r"(.*) is (.*)", "is"),  # "Robert is American"
        (r"(.*) sells (.*) to (.*)", "sells"),  # "Robert sells weapon to A"
        (r"(.*) owns (.*)", "owns"),  # "A owns missile"
        (r"(.*) is an enemy of (.*)", "enemy"),  # "A is an enemy of America"
        (r"(.*) is hostile", "hostile"),  # "A is hostile"
        (r"(.*) is a missile", "missile"),  # "M1 is a missile"
    ]

    # Try to match the sentence with each pattern
    for pattern, type_ in patterns:
        match = re.match(pattern, sentence)
        if match:
            entities = match.groups()
            if type_ == "is":
                return f"Is({entities[0].strip()}, {entities[1].strip()})"
            elif type_ == "sells":
                return f"Sells({entities[0].strip()}, {entities[1].strip()}, {entities[2].strip()})"
            elif type_ == "owns":
                return f"Owns({entities[0].strip()}, {entities[1].strip()})"
            elif type_ == "enemy":
                return f"Enemy({entities[0].strip()}, {entities[1].strip()})"
            elif type_ == "hostile":
                return f"Hostile({entities[0].strip()})"
            elif type_ == "missile":
                return f"Missile({entities[0].strip()})"
    return None

# Example of reasoning rules
```

```
def rule_1(facts):
    # If an American sells weapons to a hostile country, it's a crime
    for fact in facts:
        if "American" in fact and "Weapon" in fact and "Sells" in fact and "Hostile" in fact:
            return f"Criminal({fact.split('(')[1].split(')')[0]})"
    return None

def rule_2(facts):
    # Missiles are weapons
    for fact in facts:
        if "Missile" in fact:
            return fact.replace("Missile", "Weapon")
    return None

def rule_3(facts):
    # If a country is an enemy of America, it is hostile
    for fact in facts:
        if "Enemy" in fact and "America" in fact:
            return fact.replace("Enemy", "Hostile")
    return None

# Initialize Knowledge Base
knowledge_base = KnowledgeBase()

# Predefine some rules in the knowledge base
knowledge_base.add_rule((rule_1, "Criminal"))
knowledge_base.add_rule((rule_2, "Weapon"))
knowledge_base.add_rule((rule_3, "Hostile"))

# Function to reason from a given sentence
def reason_from_sentence(sentence):
```

```
    fact = parse_sentence(sentence)
    if fact:
        print(f"Adding fact: {fact}")
        knowledge_base.add_fact(fact)
        # Apply rules to deduce new facts
        new_facts = knowledge_base.apply_rules()
        print("New inferences:")
        for new_fact in new_facts:
            print(new_fact)
    else:
        print("Couldn't parse the sentence.")

# Test the reasoning system with a sample sentence
test_sentences = [
    "Robert is American",
    "A sells weapon to B",
    "A owns missile",
    "A is an enemy of America",
    "A is hostile"
]

for sentence in test_sentences:
    reason_from_sentence(sentence)

# Final facts in the knowledge base
print("\nFinal facts in the knowledge base:")
for fact in knowledge_base.get_facts():
    print(fact)
```

Output:

Adding fact: Is(Robert, American)

New inferences:

Adding fact: Sells(A, weapon, B)

New inferences:

Adding fact: Owns(A, missile)

New inferences:

Adding fact: Is(A, an enemy of America)

New inferences:

Adding fact: Is(A, hostile)

New inferences:

Final facts in the knowledge base:

Owns(A, missile)

Sells(A, weapon, B)

Is(Robert, American)

Is(A, an enemy of America)

Is(A, hostile)