

PRINCIPLES OF ARTIFICIAL INTELLIGENCE LAB – EXPERIMENT 7:BACKWARD CHAINING

```
# Knowledge Base (Rules in IF-THEN format) knowledge_base = {
```

```
    "flu": [{"cough", "fever"}],
```

```
    "fever": [{"sore_throat"}],
```

```
}
```

```
# Known facts facts =
```

```
{"sore_throat", "cough"}
```

```
# Backward chaining
```

```
function def
```

```
backward_chaining(goal):
```

```
    if goal in facts: # If the goal is a known fact, return True
```

```
    return True    if goal in knowledge_base: # If the goal has rules in
```

```
KB        for conditions in knowledge_base[goal]: # Check each
```

```
rule        if all(backward_chaining(cond) for cond in conditions): #
```

```
Recursively verify
```

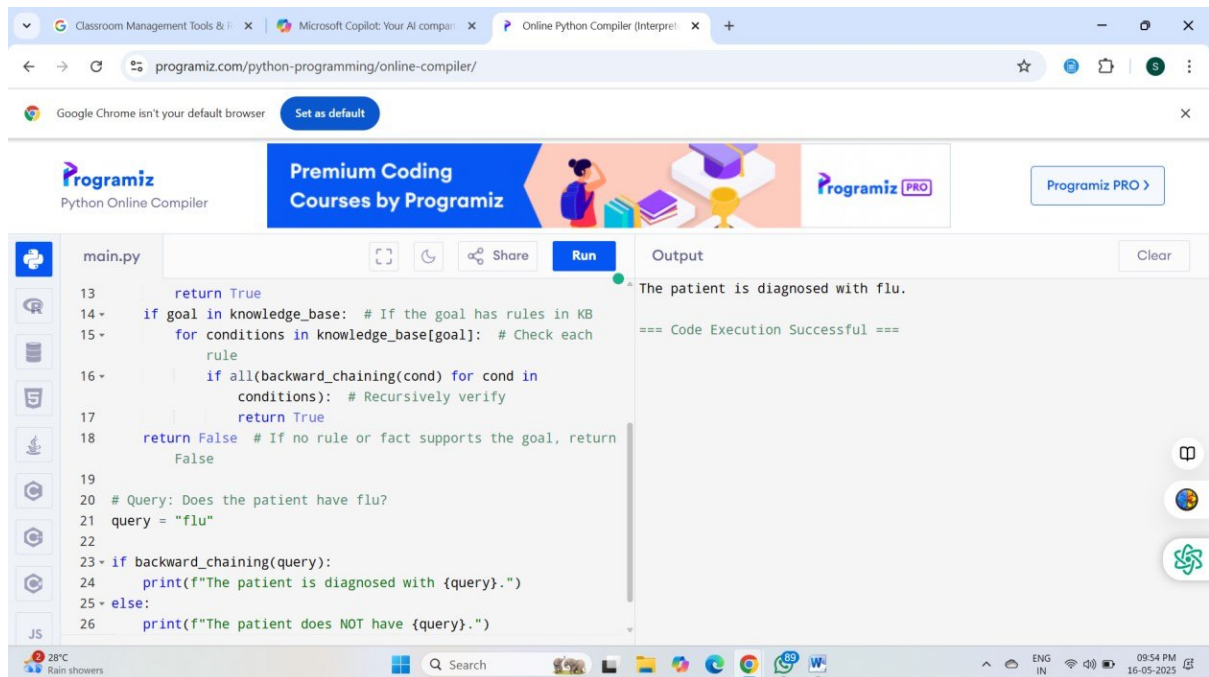
```
        return True    return False # If no rule or fact
```

```
supports the goal, return False
```

```
# Query: Does the patient have flu?
```

```
query = "flu"
```

```
if backward_chaining(query):  
    print(f"The patient is diagnosed with {query}.")  
  
else:    print(f"The patient does NOT  
have {query}.")
```



The screenshot shows a web browser window with the Programiz Python Online Compiler. The code in the editor is as follows:

```
13     return True  
14     if goal in knowledge_base: # If the goal has rules in KB  
15         for conditions in knowledge_base[goal]: # Check each  
16             rule  
17                 if all(backward_chaining(cond) for cond in  
18                     conditions): # Recursively verify  
19                         return True  
20     return False # If no rule or fact supports the goal, return  
21     False  
22  
23     # Query: Does the patient have flu?  
24     query = "flu"  
25     if backward_chaining(query):  
26         print(f"The patient is diagnosed with {query}.")  
27     else:  
28         print(f"The patient does NOT have {query}.")
```

The output on the right shows:

```
The patient is diagnosed with flu.  
=== Code Execution Successful ===
```

NAME-SUJIT R

REG - 241801280