U23AI021

Lab assignment 5
Artificial Intelligence

Harshil Andhariya
u23ai021@coed.svnit.ac.in

Q1)

```python
from pyswip import Prolog


prolog = Prolog()


prolog.assertz("mammal(tom)")
prolog.assertz("mammal(jerry)")
prolog.assertz("animal(tom)")
prolog.assertz("animal(jerry)")


prolog.assertz("has_fur(X) :- mammal(X)")
prolog.assertz("alive(X) :- animal(X)")

queries = ["has_fur(tom)", "has_fur(jerry)", "alive(tom)",
"alive(jerry)"]


for query in queries:
    result = list(prolog.query(query))
    if result:
        print(f"{query} is proven")
    else:
        print(f"{query} is not proven")
```

```
PS D:\ai_lab\assignment5> python -u "d:\ai_lab\assignment5\q1.py"
has_fur(tom) is proven
has_fur(jerry) is proven
alive(tom) is proven
alive(jerry) is proven
PS D:\ai_lab\assignment5>
```

Q2

```python
from pyswip import Prolog


prolog = Prolog()

prolog.assertz("human(socrates)")

prolog.assertz("mortal(X) :- human(X)")


query = "mortal(socrates)"
result = list(prolog.query(query))


if result:
    print(f"{query} is proven")
else:
    print(f"{query} is not proven")
```

```
PS D:\ai_lab\assignment5> python -u "d:\ai_lab\assignment5\q2.py"
mortal(socrates) is proven
PS D:\ai_lab\assignment5>
```

Q3)

```python
from pyswip import Prolog

prolog = Prolog()

prolog.assertz("loves(john, mary)")
prolog.assertz("loves(mary, john)")
prolog.assertz("kind(john)")

prolog.assertz("loves(X, everyone) :- kind(X)")
```

```
prolog.assertz("happy(X) :- loves(X, Y)")
prolog.assertz("smiles(X) :- happy(X)")
prolog.assertz("friendly(X) :- smiles(X)")
prolog.assertz("not(sad(X)) :- friendly(X)")
prolog.assertz("not(happy(X)) :- sad(X)")


query = "happy(john)"


result = list(prolog.query(query))


if result:
    print(f"{query} is proven")
else:
    print(f"{query} is not proven")
```

```
PS D:\ai_lab\assignment5> python -u "d:\ai_lab\assignment5\q1.py"
has_fur(tom) is proven
has_fur(jerry) is proven
alive(tom) is proven
alive(jerry) is proven
PS D:\ai_lab\assignment5>
```