Exp : 8 (Knowledge Representations)

Aim : Implementation of knowledge representation schemes - use cases.

Initial State :

? (make a guess)

Final State

Yes or
Learn a new concept.

Problem Solving

Given some classification rule & some predefined clauses, guess an animal & let your machine predict it, if the machine is unable to predict the animal, it will ask the answer & store it in its knowledge base.

Problem Solving

Imagine an animal (you are only allowed to answer yes or no for the next set of questions).

→ Does it have a fur?
- Yes
→ Does it have a dark spot?
- Yes
→ Is it the fastest animal?
- no

were you thinking of a leopard?
↓ Yes
I knew it !!

Algorithm :

1. Start
2. The user answers is expected to think of an animal & answer the ques shown.
3. The user answers the set of question & the inference rule is drawn from it.
4. If a conclusion to the premises result true it would display the name of the animal otherwise the machine learns from the given set of Input.
5. Repeat Step 2 or 4 if the user wants to make the guess again otherwise go to step 6.
6. Stop.

Identification of animal:

cheetah :- mammal,
carnivore,
verify(has_tawny_color),
verify(has_dark_spots).
tiger :- mammal,
carnivore,
verify(has_tawny_color),
verify(has_black_stripes).
giraffe :- ungulate,
verify(has_long_neck),
verify(has_long_legs).
zebra :- ungulate,
verify(has_black_stripes).
Classification rules:
mammal :- verify(has_hair), !.
mammal :- verify(gives_milk).
bird :- verify(has_feathers), !.
bird :- verify(flys),
verify(lays_eggs).
carnivore :- verify(eats_meat), !.
carnivore :- verify(has_pointed_teeth),

verify(has_claws),
verify(has_forward_eyes).
ungulate :- mammal,
verify(has_hooves), !.
ungulate :- mammal,
verify(chews_cud).

CODE:-

```
import sys
def definiteNoun(s):
s = s.lower().strip()
if s in ['a', 'e', 'i', 'o', 'u', 'y']:
return "an " + s
else:
return "a " + s
def removeArticle(s):
"Remove the definite article 'a' or 'an' from a noun."
```

```
s = s.lower().strip()
if s[0:3] == "an ": return s[3:]
if s[0:2] == "a ": return s[2:]
return s
def makeQuestion(question, yes, no):
return [question, yes, no]
def isQuestion(p):
"Check if node is a question (with answers), or a plain answer."
return type(p).__name__ == "list"
def askQuestion(question):
print ("\r%s " % question,)
return sys.stdin.readline().strip().lower()
def getAnswer(question):
if isQuestion(question):
return askQuestion(question[0])
else:
return askQuestion("Were you thinking about %s?" %
definiteNoun(question))
def answeredYes(answer):
if len(answer) > 0:
return answer.lower()[0] == "y"
return False
def gameOver(message):

global tries
print ("")
print ("\r%s" % message)
print ("")
def playAgain():
return answeredYes(askQuestion("Do you want to play again?"))
def correctGuess(message):
global tries
gameOver(message)
if playAgain():
print ("")
tries = 0
return Q
else:
sys.exit(0)
def nextQuestion(question, answer):
global tries
tries += 1
if isQuestion(question):
if answer:
```

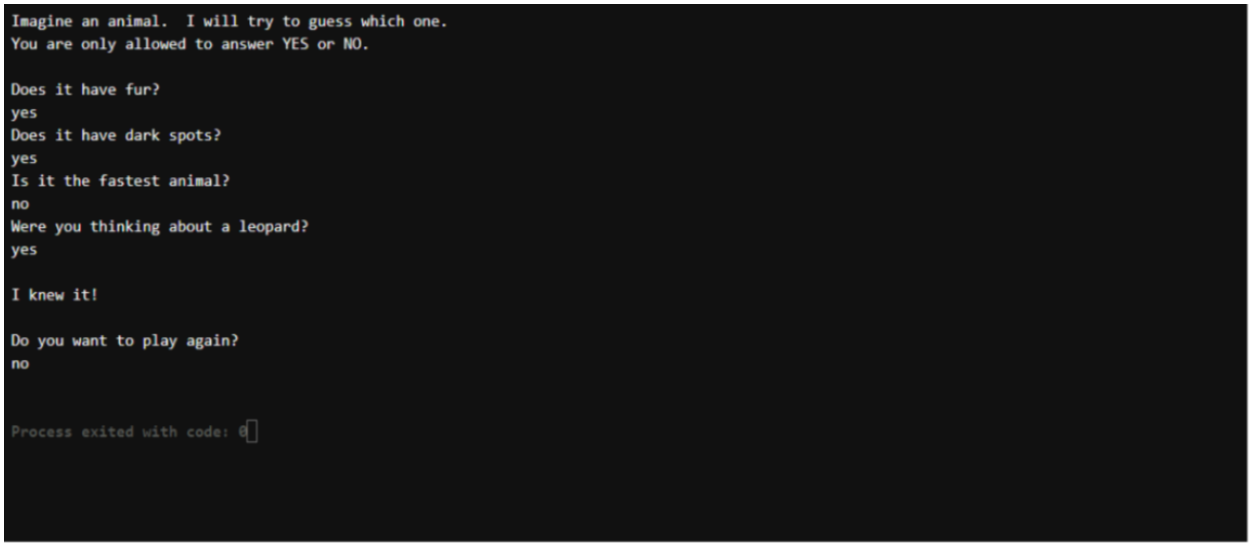
```
return question[1]
else:
return question[2]
else:
if answer:
return correctGuess("I knew it!")
else:
return makeNewQuestion(question)
def replaceAnswer(tree, find, replace):
if not isQuestion(tree):
if tree == find:
return replace
else:
return tree
else:
return makeQuestion(tree[0],
replaceAnswer(tree[1], find, replace),
replaceAnswer(tree[2], find, replace))
def makeNewQuestion(wrongAnimal):

global Q, tries
correctAnimal = removeArticle(askQuestion("I give up. What did you
think about?"))
newQuestion = askQuestion("Enter a question that would distinguish %s
from %s:"
% (definiteNoun(correctAnimal),
definiteNoun(wrongAnimal))).capitalize()
yesAnswer = answeredYes(askQuestion("If I asked you this question " +
"and you thought about %s, what would the correct answer be?" %
definiteNoun(correctAnimal)))
# Create new question node
if yesAnswer:
q = makeQuestion(newQuestion, correctAnimal, wrongAnimal)
else:
q = makeQuestion(newQuestion, wrongAnimal, correctAnimal)
Q = replaceAnswer(Q, wrongAnimal, q)
tries = 0
return Q
def addNewQuestion(wrongAnimal, newques, correct):
global Q
q = makeQuestion(newques, correct, wrongAnimal)
Q = replaceAnswer(Q, wrongAnimal, q)
return Q
tries = 0
```

```
Q = (makeQuestion('Does it have fur?', 'Tiger', 'Penguin'))
q = addNewQuestion('Tiger', 'Does it have dark spots?', 'Leopard')
q = addNewQuestion('Leopard', 'Is it the fastest animal?', 'Cheetah')
q = addNewQuestion('Penguin', 'Can it fly?', 'Parrot')
q = Q
print ("Imagine an animal. I will try to guess which one.")
print ("You are only allowed to answer YES or NO.")
print ("")
try:
while True:
ans = answeredYes(getAnswer(q))
q = nextQuestion(q, ans)
except KeyboardInterrupt:
sys.exit(0)
except Exception:

sys.exit(1)
```

OUTPUT:



RESULT:-

Hence, the Implementation of rule based inference system is done successfully.