28-12-20

AI LAB EXAM VENKATA KRISHNA GIRI KONERU
1BM18CS123

```
4. import re
    def is variable (x):
      return len(x) == 1 and x. is lower() and
      x. is alpha()

    def get attributes (string):
     expr = '\([^)]+\)'
     matches = re. findall (expr, string)
     return matches

    def get predicates (string):
      expr = '([a-z ~]+)\([^&|] + \)'
      return re. findall (expr, string)

    class fact:
     def_ init_ (self, expression):
      self. expression= expression
      predicate , params =
    self. split Expression (expression)
      self. predicate = predicate
      self. params = params
      self. result = any (self. get constants ())

    def split Expression ( self, expression):
      predicate = get predicates ( expression) [0]
      params = get Attributes ( expression)
```

1

K.V. Krishna giri

Scanned with CamScanner

VENKATA KRISHNA GIRI KONERU
1BM18CS123

```
[0].strip("()").split(',')
    return [predicate, params]

def get result(self):
    return self.result

def get constants(self):
    return [None if is variable(c) else c for c
    in self.params]

def get variables(self):
    return [V if is variable (V) else None for V
    in self.params]

def substitute (self, constants):
    c = constants.copy()
    f = f"{self.predicate}
    ({','join([constants.pop(0) if is variable (P) else
    P for P in self.params])})"
    return fact(f)

class implication:
    def _init_(self, expression):
        self.expression = expression)
        l = expression.split('=>')
        self.lhs = [Fact(f) for f in l[0].split('&')]
        self.rhs = Fact (l[1])
```

R.V. Krishna giri

VENKATA KRISHNA GIRI KONERU
1BM19CS123

```
def evaluate(self, facts):
  constants = {}
  new_lhs = []
   for fact in facts:
    for val in self.lhs:
      if val.predicate == fact.predicate:
        for i, v in
enumerate(val.getVariables()):
        if v:
           constants[v] =
 fact.getConstants()[i]
           new_lhs.append(fact)
   predicate, attributes =
   getPredicates(self.rhs expression)[0],
   str(get attributes(self.rhs expression)[0])
   for key in constants:
      if constants[key]:
       attributes= attributes..replace(key, constants[key])
    expr = f'{predicate}{attributes}'
   return fact(expr) if len(new_lhs) and
 all([f.getResult() for f in new_lhs]) else None
```

K.V. Krishna giri

VENKATA KRISHNA GIRI KONERU
1BM19CS123

```
class KB:

  def _init_(self):
    self.facts = set()
    self.implications = set()

  def tell(self, e):
   if '=>' in e:
      self.implications.add(implication(e))
   else:
     self.facts.add(fact(e))
   for i in self.implications:
     res = i.evaluate(self.facts)
   if res:
       self.facts.add(res)

   def query(self, e):
    facts = set([f.expression for f in
       self.facts])
       i=1
      print(f'Querying {e}:')
     for f in facts:
         if Fact(f).predicate ==
    fact(e).predicate
          print(f'\t{i}. {f}')

   def display(self):
   print("All facts:")
   for i, f in enumerate(set([f.expression for
     f in self.facts])):
       print(f'\t{i+1}.{f}')
```

4

K.V. Krishna giri