AI Lab Test-2

ABHAVA V
1BM18CS001
29/12/20

Q2) Unification

```
def getAttributes(expression):
    expression = expression.split("(")[1:]
    expression = "(".join(expression)
    expression = expression.split(")")[:-1]
    expression = ")".join(expression)
    attributes = expression.split(',')
    return attributes

def getPredicate(expression):
    return expression.split("(")[0]

def isConstant(char):
    return char.isupper() and len(char)==1

def isVariable(char):
    return char.islower() and len(char)==1

def substitute(exp,old,new):
    attributes = getAttributes(exp)
    predicate = getPredicate(exp)
    for index,val in enumerate(attributes):
        if val==old
            attributes[index]=new
    return predicate+"("+",".join(attributes)+")"
```


Abhava V

AKHILA Y.V
1BM18CS001
29/12/20

```
def apply(exp, substitutions):
    for substitution in substitutions:
        new, old = substitution
        exp = substitute(exp, old, new)
    return exp

def checkOccurs(var, exp):
    if exp.find(var) == -1:
        return False
    return True

def getFirstAttribute(expression):
    attributes = getAttributes(expression)
    return attributes[0]

def getRemainingExpression:
    predicate = getPredicate(expression)
    attributes = getAttributes(expression)
    newExpression = predicate + "(" + ",".join(attributes[1:]) + ")"
    return newExpression.
```


Akhila Y

AI LabTest-2

ABHAYA V
[BM18CS001]

29/12/20

```
def unify(exp1, exp2):
    if exp1 == exp2:
        return []

    elif isConstant(exp1) and isConstant(exp2):
        if exp1 != exp2:
            print(f"{exp1} and {exp2} are constants.
                  Cannot be unified")
            return []

    elif isConstant(exp1):
        return [(exp1, exp2)]

    elif isConstant(exp2):
        return [(exp2, exp1)]

    elif isVariable(exp1):
        return [(exp2, exp1)] if not checkOccurs(exp1, exp2)
                              else []

    elif isVariable(exp2):
        return [(exp1, exp2)] if not checkOccurs(exp2, exp1)
                              else []

    elif getPredicate(exp1) != getPredicate(exp2):
        print(f"Predicates {getPredicate(exp1)} and
              {getPredicate(exp2)} do not match. Cannot be unified")
        return []

    elif len(getAttributes(exp1)) != len(getAttributes(exp2)):
        print(f"Length of Attributes {len(getAttributes(exp1))}
              and {len(getAttributes(exp2))} do not match.
              Cannot be unified")
        return []
```


Abhaya

AI Lab test-2

ABHAYA.V
1BM18CS001

29/12/20

head1 = getFirstAttribute(exp1)

head2 = getFirstAttribute(exp2)

initial substitution = unify(head1, head2)

if not initial substitution:

    return {}

if len(getAttributes(exp1)) == 1:

    return initial substitution

tail1 = getRemaining(exp1)

tail2 = getRemaining(exp2)

if initial substitution != {}:

    tail1 = apply(tail1, initial substitution)

    tail2 = apply(tail2, initial substitution)

remaining substitution = unify(tail1, tail2)

if not remaining substitution:

    return {}

return initial substitution + remaining substitution

Abhaya