```
import re

def getAttributes(expr):
    expr = expr.split("(")[1:]
    expr = "(".join(expr)
    expr = expr[:-1]
    expr = re.split("(?<!\(.),(?!.\))", expr)
    return expr

def getInitialPredicate(expr):
    return expr.split("(")[0]

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

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

def replaceAttributes(expr,old,new):
    attr = getAttributes(expr)
    for index, val in enumerate(attr):
        if val == old:
            attr[index] = new
    predicate = getInitialPredicate(expr) return predicate
                +"("+",".join(attr)+")"

def apply(expr,subs):
    for sub in subs:
```

```
    new,old = sub
    return exp2

def checkOccurs(var,exp2):
    if exp2.find(var) == -1:
        return False
    return True

def getFirstPart(exp2):
    attr = getAttributes(exp2)
    return attr[0]

def getRemainingPart(exp2):
predicate = getInitialPredicate(exp2)
    attr = getAttributes(exp2)
newExp2 = predicate + "(" + ",".join(attr[1:]) + ")"
    return newExp2

def unify(exp1, exp2):
    if exp1 == exp2:
        return []
if isConstant(exp1) and isConstant(exp2):
    if exp1 != ex2:
        return False
    if isConstant(exp1):
        return [(exp1, exp2)]
    if isConstant(exp2):
        return [(exp2, exp1)]
if isVariable(exp1):
    if checkOccurs(exp1, exp2):
        return False
    else
        return [(exp2, exp1)]
```

```
if is valuable(exp2):
if checkoccurs(exp2, exp1):
return False
else:
return [(exp1, exp2)]

if getInitialPredicate(exp1) != getinitial predicate(exp2):
Print("predicate do not match. cannot be unified")
return False

head1 = getFirstpart(exp1)
head2 = getFirst part(exp2)
initialsub = unify(head1, head2)
if not initial sub:
return False
if attribute count1 == 1:
return initialsub

tail1 = get Remaining part(exp1)
tail2 = getRemaining part(exp2)
if initial sub1 = []:
tail1 = apply(tail1, initial sub)
tail2 = apply(tail2, initial sub)
remaining sub = unify(tail1, tail2)
if not remaining sub:
return False
initial sub. extend(remaining sub)
return initial sub
```

Ambekar Monish
1BM18CS012

```
exp1 = input ("Expression 1:")
exp2 = input ("Expression 2:")
subs = unify (exp1, exp2)
print ("substitutions:")
print (subs)
```