AI LAB

Mohammed Izhan Mehkri
1BM18CS137

① Unification

```
def getAttributes(expr):
expr = expr.split("(")[1]
expr = "(".join(expr)
expr = expr[:-1]
expr = re.split
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
     expr = replaceAttributes(expr,old,new)
        return expr

  def checkoccurs(var,expr):
    if (expr.find(var)==-1:
       return False
    return True.
```

Mohammed Izhan Mehkri
IBM18CS137

```
def getfirstpart(expr):
    attr = getAttributes(expr)
    return attr[0]

def getremainingpart(expr):
    predicate = getInitialpredicate(expr)
    attr = getattributes(expr)
    newexpr = predicate + "(" + ",".join(attr[1:]) + ")"
    return newexpr.

def unify(exp1, exp2):
    if exp1 == exp2:
        return []
    if isconstant(exp1) and isconstant(exp2):
        if exp1 != exp2
            return False
    if isconstant(exp1):
        return [(exp1, exp2)]
    if is constant(exp2):
        return [(exp2, exp1)]
    If isVaricable(exp1):
        if checkoccurs(exp1, exp2)
        return False
        else:
        return[(exp2, exp1)]
    if getinitialpredicate(exp1) != getinitialpredicate(expr2)
        print("cannot be unified")
            return false
    attribute count1 = len(getattributes(exp1))
    attribute count2 = len(getattributes(exp2))
    if attributecount1 != attributecount2
        return false
```


Mohammed Izhan M

```
head1 = getfirstpart(exp1)
head2 = getfirstpart(exp2)
initialsub = unify(head1, head2)
if not initialsub
    return False
if attributecount1 == 1:
    return initialsub
tail1 = getRemainingpart(exp1)
tail2 = getRemainingpart(exp2)

if initialsub != []:
tail1 = apply(tail1, initialsub)
tail2 = apply(tail2, initialsub)
remainingsub = unify(tail1, tail2)
if not remainingsub
    return false.
initialsub.extend(remainingsub)

    res = []
    for tup in initialsub:
    st = ','.join(tup)
    res.append(st)
    return res.
```