# AI-lab Test 2

Nitish.N.Banaka
1BM18CS065

Program 1 :-

Unification

```
import re

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

def getinitialPredicate(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 replaceAttribute(exp, old, new):
    attributes = getAttributes(exp)
    predicate = getInitialPredicate(exp)
    for i, val in enumerate(attributes):
            if val == old
                attributes[i] = new
    return predicate + "(" + ",".join(attributes) + ")"
```

Nitish

Nitish N B
1BM18CS089

```
def apply(exp, subs):
    for sub in subs:
        new, old = sub
        exp = replaceAttributes(exp, old, new)
    return exp

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

def getFirstPart(exp):
    attributes = getAttributes(exp)
    return attributes[0]

def getRemainingPart(exp):
    predicate = getInitialPredicate(exp)
    attributes = getAttribubes(exp)
    newexp = predicate + "(" + ",".join(attributes[1:]
                                       + ")"
    return newexp
def Unify(exp1, exp2):
    if exp1 == exp2:
        return []
    if isConstant(exp1) and isConstant(exp2):
        if (exp1 != exp2):
            print("Expressions are constants:
                   Cannot be unified")
            return []
```

```
if isConstant (exp1):
    return [(exp1 ,exp2)]
if is Constant (exp2):
    return [(exp2, exp1)]
if isVariable(exp1):
    return [(exp2,exp1)] if not
                         checkOccurs (exp1,exp2) else []
if is Variable(exp2):
    reutrun [(exp1 ,exp2)] if not
                  cedhk Occurs (exp2 ,exp1) else []
if getInitialPredicate (exp1) != getInitialPredicart(exp2):
    print ("Cannot be unified.
                Predicates do not match")
    return []
attributecount1 = len (getAttributes(exp1))
attributecount2 = len(get Attributes(exp2))
if attributecount1 != attributecount2:
    print ("Length of attributes does not match
               Cannot be unified")
    return []
head1 = get FirstPart (exp1)
head2 = get First Part (exp2)
initialSubstitution = unify(head1, head2)
if not initialSubstitution:
    return []
if attribute count1 ==1:
    return initialSubstibution
```

Nitish

```
tail1= getRemainingpart(exp1)
tail2= getRemainingPart (exp2)
if initialSubstitution != []
      tail1  = apply (tail1 ,intialsubstitution)
      tail2 =apply(tail2, initialsubstitution)

   remainingSubstitution = unify (tail1, tail2)
   if not remainingsubstitution:
          return []
   return initialSubstitution + remainingsubstition

if __name__ ="__main__":
      print(" Enter the first expression")
      exp1 = input ()
      print (" Enter the second expression")
      exp2 = input()
      subs = unify (e1,e2)
      print(" The substitutions are:")
      print([' / '.join(sub) for sub in subs])
```

Nitish