[illegible]

6th program - 5

## Unification in first order Logic

### Algorithm

Step 1: Initialize: start with an empty substitution 'theta = { }'

Step 2 = Comparison of atoms compare the two atoms ["P", "x", "y"] and ["P", "A", "B"]
The first elements "P" are same
The second elements "x" and "A" are different, so we need to unify them.

Step 3: Unify variables: unify the variable "x" with the term "A"
Update 'theta' with this substitution
theta = {"x": "A"}

Step 4 - Recursive unification: Now, recursively unify the remaining elements:

- For the third elements "y" and "B", unify the variable "y" with the term "B". Update theta. theta = {"x": "A", "y": "B"}

Step 5: Result. The final substitution
theta is {"x": "A", "y": "B"}
The two expressions ["P", "x", "y"] and ["P", "A", "B"] can be unified

Code

```
class UnificationError(Exception):
    pass

def print_step(step, atom1, atom2, theta):
    print(f"\n step {step}:")
    print(f" Unifying {atom1} and
                {atom2}")
    print(f" Current Substitution Theta:
                {theta}")

def unify_var(var, x, theta):
    if var in theta:
        return unify(theta[var], x, theta)
    else x in theta:
        return unify(var, theta[x], theta)
    else:
        theta[var] = x
        return theta

def unify(atom1, atom2, theta=None, step=1):
    if theta is None:
        theta = {}

print_step(step, atom1, atom2, theta)

if atom1 == atom2:
    return theta
elif isinstance(atom1, str) and
    atom1.isalpha():
return unify_var(atom1, atom2,
            theta)
```

```
if len(atom1) != len(atom2):
    raise UnificationError("Lists have
        different lengths")
    for a1, a2 in zip(atom1, atom2):
        theta = unify(a1, a2, theta, step+1)
    return theta
  else:
    raise UnificationError("Cannot unify")

try:
    theta = unify(["P", "x", "y"],
                  ["P", "A", "B"])
    print("\n Unification successful, theta
except UnificationError as e
    print("\n Unification failed:", e)
```

Output

```
step 1 : Unifying ['P', 'x', 'y'] and
                  ['P', 'A', 'B']
   current substitution Theta : { }

Step 2 : Unifying P and P
   Current substitution Theta : { }

Step 3 : Unifying x and A
   current substitution Theta : { }

Step 4 : Unifying y and B
   current substitution Theta : {'x', 'A'

Unification successful
   substitution theta : {'x' : 'A', 'y'
   : 'B' }
```