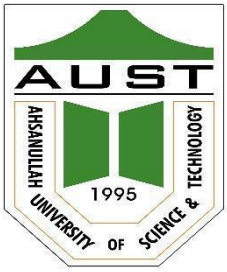


Ahsanullah University of Science & Technology

Department of Computer Science & Engineering

Course No : **CSE4108**
Course Title : **Artificial Intelligence Lab**
Assignment No : **Term Assignment 1**

Date of Performance : **29.01.2023**
Date of Submission : **29.01.2023**

Submitted To : **Mr. Faisal Muhammad Shah & Mr. Raihan Tanvir**

Submitted By-
Group : **C_2**
Name : **Ashiqul Islam**
Id : **190104140**
Section : **C**

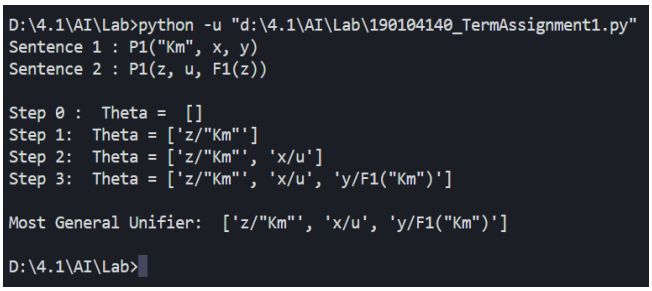
Topic: Unification with Predicate Logic

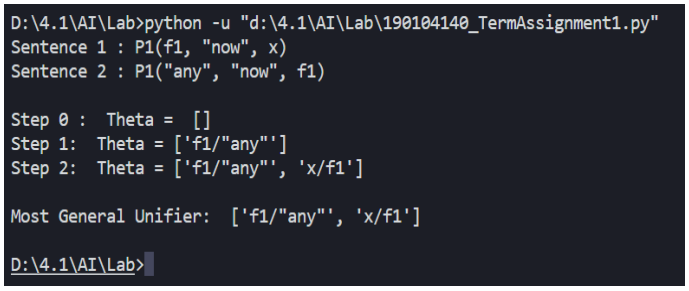
Source Code:

```
theta = list()
def is_function(term):
    if("(" in term):
        function_symbol = term.split("(")[0]
        if(function_symbol.isalnum() and term[-1] == ")"):
            return True
        else:
            return False
    else:
        return False

def is_variable(term):
    if(term[0] != '"' and is_function(term)==False):
        return True
    else:
        return False

def substitute(terms, prev, new):
    for j in range(len(terms)):
        if(is_function(terms[j])):
            terms[j] = terms[j].replace(prev, new)
        elif(terms[j] == prev):
            terms[j] = new
    return terms;

def unify(terms1, terms2):
    step = 0
    print("\nStep", step, ": ", end=" ")
    print("Theta = ", theta)

    while(terms1 != terms2):
        i = 0
        while(i < len(terms1)):
            if(terms1[i] != terms2[i]):
                if(is_variable(terms1[i])):
                    theta.append(terms1[i] + "/" + terms2[i])
                    #substitute terms of sentance 1 that is the same as terms1[i], with the
value of terms2[i]
                    terms1 = substitute(terms1, terms1[i], terms2[i])
                elif(is_variable(terms2[i])):
                    theta.append(terms2[i] + "/" + terms1[i])
                    #substitute terms of sentance 2 that is the same as terms2[i], with the
value of terms1[i]
                    terms2 = substitute(terms2, terms2[i], terms1[i])
                elif((is_function(terms2[i]) and terms1[i] in terms2[i]) or
                     (is_function(terms1[i]) and terms2[i] in terms1[i])):
                        #Stop unifying as a term contains the the other variable
                        return False
                else:
                    theta.append(terms1[i] + "/" + terms2[i])
                    terms1 = substitute(terms1, terms1[i], terms2[i])
                step+=1
                print("Step " + str(step) + ": ", end=" ")
                print("Theta =", theta)
            i+=1
    return True

sentence1 = input("Sentence 1 : ").replace(" ", "")
sentence2 = input("Sentence 2 : ").replace(" ", "")

predicate1 = sentence1.split("(")[0]
predicate2 = sentence2.split("(")[0]

terms1 = sentence1.replace(predicate1, "")
terms1 = terms1[1: len(terms1)-1].split(",")
```

```
terms2 = sentence2.replace(predicate2, "")
terms2 = terms2[1: len(terms2)-1].split(",")

if (predicate1 != predicate2):
    print("Predicates don't match, Not unifiable!")
elif (len(terms1) != len(terms2)):
    print("Number of terms don't match, Not unifiable!")
else:
    if(unify(terms1, terms2) == False):
        print("Not unifiable!")
    else:
        print("\nMost General Unifier: ", theta)
```

Output:





Explanation:

The code takes two sentences as input and checks if the predicates of the two sentences match and if the number of terms in both sentences match. If the predicates and number of terms in the sentences don't match, the code prints "Not unifiable!". Otherwise, the code proceeds to find the Most General Unifier.

The code uses the following functions:

is_function(term): This function checks if the given term is a function by checking if it contains a '(' and if the last character is ')' and the predicate name is valid or not that is it must contain only alpha-numerics.

is_variable(term): This function checks if the given term is a variable by ensuring that it is not a function and checking if it is wrapped by double quotes or not.

substitute(terms, prev, new): This function takes a list of terms, a previous term and a new term as input and replaces all occurrences of the previous term in the list of terms with the new term.

unify(terms1, terms2): This function takes two lists of terms as input and finds the most general unifier of the two lists of terms. It uses a while loop that runs until the terms in both lists exactly match. Inside the while loop, the function uses a nested for loop that iterates through each term in the list of terms. If the terms at a given index in both lists do not match, the function checks if the term is a variable or a function. If the term is a variable, it is replaced with the corresponding term in the other list and added to the unifier list (theta). If the term is a function, the function checks if the term contains the other variable and if it does, it returns False as the terms are not unifiable. If the terms are unifiable, the function continues to the next index. If all terms in both lists match, the function returns True and the most general unifier is stored in the theta list finally. In each step the theta list is printed to show the substitutions in each step that leads to the most general unifier. The sample input provided in the code is:

Sentence 1 : P1("Km", x, y)

Sentence 2 : P1(z, u, F1(z))

The MGU for this input would be:

Most General Unifier: ['x/z', 'y/u', 'F1(z)/F1(z)']

In this example, the MGU is a list of terms that shows how the terms in the two sentences can be made to match. The value of theta in each steps shows the substitutions that should be made.