Madhura Nagle
PD 15

Assignment 4

* Aim : To implement Unification algorithm.

* Objective : To study and implement Unification algorithm

* Theory :

1) Unification Algorithm :

→ In logic and computer science unification is an algorithmic process of solving equations between symbolic expressions. A unification algorithm should compute for a given problem a complete and minimal substitution set that is a set covering all its solutions and containing no ~~reduda~~ redundant members.

2) Resolution as Proof procedure:

→ Resolution is a theorem proving technique that proceeds by building refutation proofs, i.e. proofs by contradict~~a~~ions. It was invented by a mathematician John Alan in ~~196~~ 1965. Resolution is used if there are various statements are given, and we need to prove a conclusion of those statements. Unification is a key concept in proofs by ~~e~~resolution. Resolution is a single interence rule which can efficiently operate on the conjuctive normal form or clausal form.

→ Steps :

→ Conversion of facts into first order logic

→ Convert FOL statements into CNF

→ Negate the statements which needs to prove.

→ Draw resolution graph (unification.)

* Input : Two literals L1 & L2

* Output : A set of substitutions

* Algorithm : Unification algorithm.

* FAQ

Q Why resolution is required?

→ Resolution is used if there are various statements are given and we need to prove a conclusion of those statement. Unification is a key concept in proofs by resolutions.

Q What are the pre-requisites for applying unification algorithm?

→ ① Predicate symbol must be same, atoms or expression with different predicate symbol can never be unified.

② Number of arguments in both expression must be ~~ind~~ identical.

③ Unification will fail if there are two similar variables present inthe same expression.

Q What are the applications of unification algorithm?

→ ① Logical programming

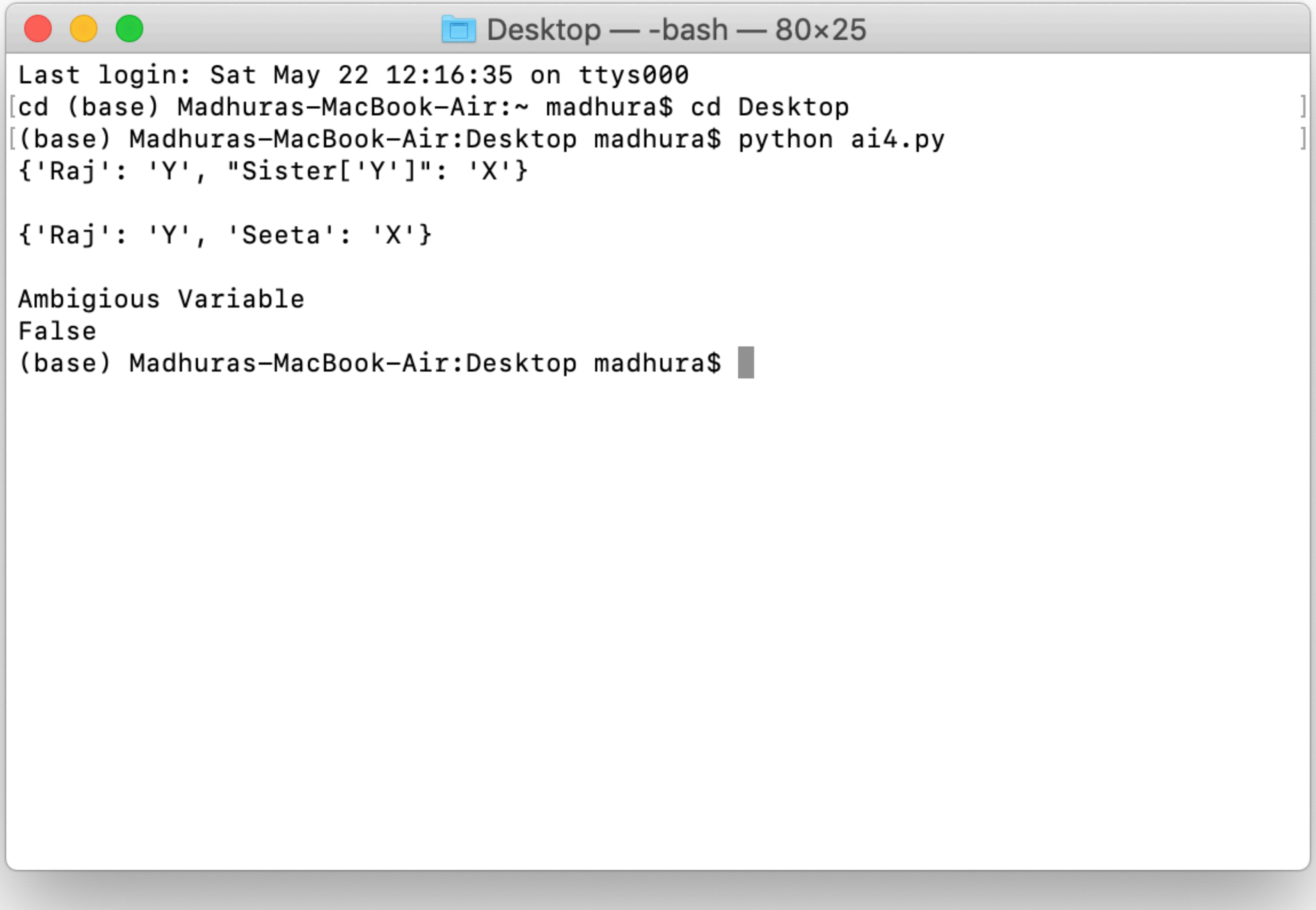
② Programming language type system implemention.

③ Cryptographic Protocol analysis

④ Term rewriting algorithms

⑤ SMT solvers.

```
import random
class Variable:
    def __init__(self,value):
        self.value = value
    def __eq__(self, other):
        return self.value == other.value
class Constant:
    def __init__(self,value):
        self.value = value
    def __eq__(self, other):
        return self.value == other.value
class Rel:
    def __init__(self,name,args):
        #This is a list
        self.name = name
        self.value = str(self.name)+str([i.value for i in args])
        self.args = args



def Unify(L1,L2,testset):
    '''
    L1 and L2 are Rel types, variables or constants
    '''
    #If both are variable or constants
    if(isinstance(L1,Variable) or isinstance(L2,Variable) or 
isinstance(L1,Constant) or isinstance(L2,Constant)):
        if L1 == L2:
            return None
        elif isinstance(L1,Variable):
            if isinstance(L2,Variable):
                print("Both missmatching variables")
                return False
            else:
                if L1.value not in testset.values():
                    return [L2,L1]
                else:
                    print("Ambigious Variable")
                    return False
        elif isinstance(L2,Variable):
            if isinstance(L1,Variable):
                print("Both missmatching variables")
                return False
            else:
                if L2.value not in testset.values():
                    return [L1,L2]
                else:
                    print("Ambigious Variable")
                    return False
        else:
            print("Missmatch")
            return False

    #Ensuring the functions are the same
    elif L1.name != L2.name:
        print("Relation Missmatch")
        return False
    #Ensuring the functions have the same number of arguments
    elif len(L1.args) != len(L2.args):
        print("length does not match")
        return False

    SUBSET = {}

    for i in range(len(L1.args)):
        S = Unify(L1.args[i],L2.args[i],SUBSET)
        if S==False:
            return False
        if S != None:
            SUBSET[S[0].value] = S[1].value

    return SUBSET


if __name__ == "__main__":

    print(Unify(Rel("Knows",
[Constant("Raj"),Variable("X")]),Rel("Knows",
[Variable("Y"),Rel("Sister",[Variable("Y")])]),{}))
    print()
    print(Unify(Rel("Knows",
[Constant("Raj"),Variable("X")]),Rel("Knows",
[Variable("Y"),Constant("Seeta")]),{}))
    print()
    print(Unify(Rel("Knows",
[Constant("Raj"),Variable("X")]),Rel("Knows",
[Variable("X"),Constant("Seeta")]),{}))
```



```
Last login: Sat May 22 12:16:35 on ttys000
cd (base) Madhuras-MacBook-Air:~ madhura$ cd Desktop
(base) Madhuras-MacBook-Air:Desktop madhura$ python ai4.py
{'Raj': 'Y', "Sister['Y']": 'X'}

{'Raj': 'Y', 'Seeta': 'X'}

Ambigious Variable
False
(base) Madhuras-MacBook-Air:Desktop madhura$ 
```