UNIVERSITY OF CALCUTTA

**B.Sc. (HONOURS), CBCS SEMESTER VI**

**2022**

**COMPUTATIONAL INTELLIGENCE**

**(PRACTICAL)**

**C.U. Roll No. : 193016-21-0029**

**C.U. Registration No. : 016-1112-0281-19**

**Paper - DSE-B-3-P**

**Subject Code - CMSA**

**Semester-VI**

# TABLE OF CONTENTS

|  |  |  |  |
| --- | --- | --- | --- |
| **Serial No.** | **Contents** | **Date** | **Signature** |
| **1** | Write rules to define the following relationships: brother(), sister(), aunt(), uncle(), grandfather(), granddaughter(), ancestor(), descendant(), and unrelated(). | 06/05/2022 |  |
| **2** | Write a Prolog query to find all the surgeons who live in Texas and make over $100,000/yr. | 06/05/2022 |  |
| **3** | Write a prolog function to remove duplicates from a list. | 06/05/2022 |  |
| **4** | Implement prime factorization in Prolog. | 06/05/2022 |  |
| **5** | Calculate GCD of two Number. | 21/06/2022 |  |
| **6** | Check if given integer number is prime or not. | 21/06/2022 |  |
| **7** | Write a prolog program to reverse a list. | 21/06/2022 |  |
| **8** | Check given list is Palindrome | 21/06/2022 |  |

# ASSIGNMENT - 1

* **Name of the Assignment:**

Write rules to define the following relationships: brother(), sister(), aunt(), uncle(), grandfather(), granddaughter(), ancestor(), descendant(), and unrelated(). Use the convention that relation(X,Y) means "the relation of X is Y)". For example, uncle(bart,herb) means the uncle of bart is herb.

* **Rules:**

parent(bart,homer).

parent(bart,marge).

parent(lisa,homer).

parent(lisa,marge).

parent(maggie,homer).

parent(maggie,marge).

parent(homer,abraham).

parent(herb,abraham).

parent(tod,ned).

parent(rod,ned).

parent(marge,jackie).

parent(patty,jackie).

parent(selma,jackie).

female(maggie).

female(lisa).

female(marge).

female(patty).

female(selma).

female(jackie).

male(bart).

male(homer).

male(herb).

male(burns).

male(smithers).

male(tod).

male(rod).

male(ned).

male(abraham).

**brother(X,Name):-**

**parent(X,Y),parent(Name,Y),male(Name),Name \== X.**

**sister(X,Name):-**

**parent(X,Y),parent(Name,Y),female(Name),Name \== X.**

**uncle(X,Name):-**

**parent(X,Y),brother(Y,Name).**

**aunt(X,Name):-**

**parent(X,Y),sister(Y,Name).**

**grandfather(C,Name):-**

**parent(C,X),parent(X,Name),male(Name).**

**granddaughter(Y,X):-**

**parent(X,Z),parent(Z,Y).**

**ancestor(X, Y) :-**

**parent(X, Y).**

**ancestor(X, Y) :-**

**parent(X, Z), ancestor(Z, Y).**

**descendant(X, Y) :-**

**parent(Y, X).**

**descendant(X, Y) :-**

**parent(Z, X), descendant(Z,Y).**

* **Input/Output:**

% c:/users/abhisek/documents/prolog/q1 compiled 0.02 sec, -2 clauses

**?- brother(rod,X).**

**X = tod ;**

**?- sister(marge,X).**

**X = patty ;**

**X = selma.**

**?- aunt(X,patty).**

**X = bart ;**

**X = lisa ;**

**X = maggie .**

**?- uncle(bart,X).**

**X = herb .**

**?- grandfather(maggie,X).**

**X = abraham .**

**?- granddaughter(jackie,lisa).**

**true.**

**?- ancestor(bart,X).**

**X = homer ;**

**X = marge ;**

**X = abraham ;**

**X = jackie .**

**?- descendant(jackie,X).**

**X = marge ;**

**X = patty ;**

**X = selma ;**

**X = bart ;**

**X = lisa ;**

**X = maggie .**

# ASSIGNMENT - 2

* **Name of the Assignment:**

Write a Prolog query to find all the surgeons who live in Texas and make over $100,000/yr. You will have to add some additional data, such as about different types of surgeons, or city-state relationships.

* **Rules:**

occupation(joe,oral\_surgeon).

occupation(sam,patent\_laywer).

occupation(bill,trial\_laywer).

occupation(cindy,investment\_banker).

occupation(joan,civil\_laywer).

occupation(len,plastic\_surgeon).

occupation(lance,heart\_surgeon).

occupation(frank,brain\_surgeon).

occupation(charlie,plastic\_surgeon).

occupation(lisa,oral\_surgeon).

address(joe,houston).

address(sam,pittsburgh).

address(bill,dallas).

address(cindy,omaha).

address(joan,chicago).

address(len,college\_station).

address(lance,los\_angeles).

address(frank,dallas).

address(charlie,houston).

address(lisa,san\_antonio).

salary(joe,50000).

salary(sam,150000).

salary(bill,200000).

salary(cindy,140000).

salary(joan,80000).

salary(len,70000).

salary(lance,650000).

salary(frank,85000).

salary(charlie,120000).

salary(lisa,190000).

**surgeon(oral\_surgeon).**

**surgeon(plastic\_surgeon).**

**surgeon(heart\_surgeon).**

**surgeon(brain\_surgeon).**

**city(houston,texas).**

**city(pittsburgh,pennsylvania).**

**city(dallas,texas).**

**city(omaha,nebraska).**

**city(chicago,lllinois).**

**city(college\_station,texas).**

**city(los\_angeles,california).**

**city(san\_antonio,texas).**

**relation(Name):-**

**occupation(Name,X),**

**surgeon(X),**

**address(Name,Y),**

**city(Y,texas),**

**salary(Name,Z),**

**Z>100000.**

* **Input/Output:**

% c:/users/abhisek/documents/prolog/q2 compiled 0.00 sec, 0 clauses

**?- relation(Name).**

**Name = charlie ;**

**Name = lisa.**

# ASSIGNMENT - 3

* **Name of the Assignment:**

Write a prolog function to remove duplicates from a list.

* **Rules:**

**remdups([], []).**

**remdups([H|T], [H|T1]) :-**

**subtract(T, [H], T2), remdups(T2, T1).**

* **Input/Output:**

% c:/users/abhisek/documents/prolog/q3 compiled 0.00 sec, 0 clauses

**?- remdups([1,3,4,2,4,3,6,8,6,5,4,2,3,4,9],X).**

**X = [1, 3, 4, 2, 6, 8, 5, 9].**

# ASSIGNMENT - 4

* **Name of the Assignment:**

Implement prime factorization in Prolog.

* **Rules:**

**prime\_factors(N,L) :-**

**N > 0, prime\_factors(N,L,2).**

**prime\_factors(1,[],\_) :- !.**

**prime\_factors(N,[F|L],F) :-**

**R is N // F, N =:= R \* F, !, prime\_factors(R,L,F).**

**prime\_factors(N,L,F) :-**

**next\_factor(N,F,NF), prime\_factors(N,L,NF).**

**next\_factor(\_,2,3) :- !.**

**next\_factor(N,F,NF) :- F \* F < N, !, NF is F + 2.**

**next\_factor(N,\_,N).**

* **Input/Output:**

% c:/users/abhisek/documents/prolog/q4 compiled 0.00 sec, 0 clauses

**?- prime\_factors(120, R).**

**R = [2, 2, 2, 3, 5].**

**?- prime\_factors(7, R).**

**R = [7].**

# ASSIGNMENT - 5

* **Name of the Assignment:**

Calculate GCD of two Number.

* **Rules:**

**gcd(X,0,X) :- X > 0.**

**gcd(X,Y,G) :- Y > 0, Z is X mod Y, gcd(Y,Z,G).**

* **Input/Output:**

**% c:/users/abhisek/documents/prolog/gcd compiled 0.00 sec, 0 clauses**

**?- gcd(10,52).**

**Gcd is 2**

**true .**

# ASSIGNMENT - 6

* **Name of the Assignment:**

Check if given integer number is prime or not.

* **Rules:**

**is\_prime(2).**

**is\_prime(3).**

**is\_prime(P) :- integer(P), P > 3, P mod 2 =\= 0, \+ has\_factor(P,3).**

**has\_factor(N,L) :- N mod L =:= 0.**

**has\_factor(N,L) :- L \* L < N, L2 is L + 2, has\_factor(N,L2).**

* **Input/Output:**

**?-**

**% c:/users/abhisek/documents/prolog/given integer number is prime compiled 0.00 sec, 0 clauses**

**?- is\_prime(151).**

**true.**

**?- is\_prime(152).**

**false.**

# ASSIGNMENT - 7

* **Name of the Assignment:**

Write a prolog program to reverse a list.

* **Rules:**

**my\_reverse(L1,L2) :- my\_rev(L1,L2,[]).**

**my\_rev([],L2,L2) :- !.**

**my\_rev([X|Xs],L2,Acc) :- my\_rev(Xs,L2,[X|Acc]).**

* **Input/Output:**

**?-**

**% c:/users/abhisek/documents/prolog/reverse a list compiled 0.00 sec, -0 clauses**

**?- my\_reverse([10,20,30,40],X).**

**X = [40, 30, 20, 10].**

# ASSIGNMENT - 8

* **Name of the Assignment:**

**Check given list is Palindrome.**

* **Rules:**

**is\_palindrome(L) :- reverse(L,L).**

* **Input/Output:**

**?-**

**% c:/users/abhisek/documents/prolog/ex/find out whether a list is a palindrome compiled 0.00 sec, -2 clauses**

**?- is\_palindrome([10,20,30,20,10]).**

**true.**

**?- is\_palindrome([10,20,30,10,10]).**

**false.**