Assignment 2

Rules

mother(X,Y):- parent(X,Y),female(X).

father(X,Y):- parent(X,Y),male(X).

has child(X):- parent(X,_).

sister(X,Y):- parent(Z,X),parent(Z,Y),female(X),X\==Y.

brother(X,Y):-parent(Z,X),parent(Z,Y),male(X),X\==Y. grandparent(X,Y):-parent(X,Z),parent(Z,Y).

grandmother(X,Z):-mother(X,Y),parent(Y,Z).

grandfather(X,Z):-father(X,Y),parent(Y,Z).

wife(X,Y):-parent(X,Z),parent(Y,Z),female(X),male(Y).

uncle(X,Z):-brother(X,Y),parent(Y,Z).

Assignment 4

1. True or False!

a-f

b-t

c-t

d-t

e-t

f-f

g-f

2- Given that , Celsius = (Fahrenheit - 32) / 1.8. Write a prolog program to accept a Fahrenheit degree and result the Celsius equivalent.

SOLUTION:

fah_to_cel(Fahrenheit):- celsius is ((Fahrenheit - 32) / 1.8)

Assignment 6

Q1- Write a predicate list_sum(list, sum) that succeeds if 'sum' is the sum of elements of 'list', consisting of numbers. Sample runs:

?- list_sum([1,2,3], 6).

true ?

- list_sum([1,2,3], X).

X=6.

?- list_sum([], X).

X=0.

SOLUTION:

list_sum([], 0).

list_sum([Head|Tail], Sum) :-

is_list(Head), list_sum(Head, 0, Accumulator), list_sum(Tail, Accumulator, Sum).

list_sum([Head|Tail], Sum) :-

\+ is_list(Head), list_sum(Tail, Head, Sum). list_sum([], Accumulator, Accumulator). list_sum([Head|Tail], Accumulator, Sum) :- is_list(Head), list_sum(Head, Accumulator, NextAccumulator), list_sum(Tail, NextAccumulator, Sum).

list_sum([Head|Tail], Accumulator, Sum) :- number(Head), NextAccumulator is Accumulator + Head, list_sum(Tail, NextAccumulator, Sum).

Q1- Write a prolog program to alter every element less than 10 in a list of integers with its Alphabet name. if the number >= 10 do nothing.

Sample run:

?- Alter([1, 3, 21], X).

X= [one, three, 21].

SOLUTION:

alter([],[]). alter([H|T],[H|M]):-

H>9, alter(T,M),!. alter([H|T],[one|M]):-

H=1, alter(T,M),!. alter([H|T],[two|M]):- H=2,

alter(T,M),!. alter([H|T],[three|M]):- H=3,

alter(T,M),!. alter([H|T],[fore|M]):- H=4,

alter(T,M),!. alter([H|T],[five|M]):- H=5,

alter(T,M),!. alter([H|T],[six|M]):- H=6,

alter(T,M),!. alter([H|T],[seven|M]):- H=7, alter(T,M),!.

alter([H|T],[ghit|M]):- H=8, alter(T,M),!.

alter([H|T],[nine|M]):- H=9, alter(T,M)

Assignment 7

B

A

B

D

D

2. True or False!

T

T

T

T

F

F

T

3. What does Prolog return to the following queries?

Error

Y=3 x=2

X=2+3

False

5

Error

X=3+5

X is f(०+३)

False

True

False

X= [b,c,d]

X= a y= b

X= a Y=[b]

False

X= a y=b

True

False

False

True

False

False

True

True

Assignment 8

Q1- Implement predicates that compute

Padovan number.

PADOVAN::

Pad(0) = 1

Pad(1) = 1

Pad(2) = 1

Pad(n) = Pad(n-2) + Pad(n-3) for n>2

SOLUTION:

```
pad( X, 1) :- X<3.! ,
pad( X, Y) :- X>2,
X2 is X-2, X3 is X-3,
pad( X2, Y2), pad( X3, Y3),
Y is Y2 + Y3.
```

Q2- Return the sum of digits recursively.

GOAL:

SUM(123, O)

O = 6

SUM(10, O)

O = 1

SUM(554, O)

O = 14

SOLUTION:

```
sumd(X, X) :- X<10.
sumd(X, Y)-:
X>=10,
X1 is X // 10,
X2 is X mod 10,
sumd(X1, Y1),
Y is Y1 + X2.
```

Assignment 11

Q1- Write these rules in efficient prolog code as

binary relation where prevent the backtracking

once we get the first right answer.

RULES:

Rule 1: if X<2 then Y=1

Rule 2: if 2≤X and X<4 then Y=2

Rule 3: if 4 ≤X then Y=3

SOLUTION:

f(X, 1) :- X < 2.! ,

f(X, 2) :- X >= 2, X < 4.! ,

f(X, 3) :- X >= 4.,

Q2- Trace this predicate by the following queries.

PREDICATES:

classify(0,zero).

classify(N,negative):-N<0.

classify(N,positive).

IMPROVE:

classify(N,negative) :- N<0. !

classify(0,zero). ! -:

classify(N,positive) :- N>0. !

SOLUTION:

i. ?- classify(1,N). N= positive

X classify(N,negative) :- N<0. !

X classify(0,zero). ! -:

O classify(N,positive) :- N>0. !

ii. ?- classify(0,N). N= zero

X classify(N,negative) :- N<0. !

O classify(0,zero). ! -:

iii. ?- classify(-1,N). N= negative

O classify(N,negative) :- N<0 ! .