SL.NO:CAA [illegible]

SRI SIVASUBRAMANIYA NADAR COLLEGE OF ENGINEERING

(An Autonomous Institution, Affiliated to Anna University, Chennai)
Rajiv Gandhi Salai (OMR), Kalavakkam – 603 110

THEORY EXAMINATIONS

Register Number	205001131		
Name of the Student	Yogesh Nimesh ·E		
Degree and Branch	BE CSE -B	Semester	V
Subject code and Name	UCS1524 – Logic Programming		
Assessment Test No.	II	Date	20/10/22

Details of Marks Obtained

Part A		Part B				Part C			
Question No.	Marks	Question No.	(a) Marks	(b) Marks	Total Marks	Question No.	(a) Marks	(b) Marks	Total Marks
1	2	7			6	10			
2	2					11			10
3	2	8			6	12			8½
4	0								
5	2	9			6	13			
6	2								
Total (A)	10	Total (B)			18	Total (C)			18½

Grand Total (A+B+C)				Marks (in words)			
Signature of Faculty	46½/50						

ssn

Part-A

1. Swapping lemma:

The order of swapping the elments does not affect the functionality. The order of exchange usually doesn't matter in swapping lemma.

$\{A_1, \ldots, A_j, \ldots, A_k\}$

$\{C_1, \ldots, C_j, \ldots C_k\}$

$\{A_1, \ldots A_j, C, \ldots A_k\}$

$\{E_1, \ldots, E_j, \ldots E_k\}$

$\{A_1, \ldots A_j, C, E, \ldots A_k\}$

Hence, the swapping lemma is defined.

2. Procedural semantics	Declarative semantics
• $G_{proc}(G, F) = \{H \mid$ the F is obtained by input G to give the H $\}$	• $S_{prod}(G, F) = \{H \mid$ For H being $(A_1 \land \ldots \land A_n)$ it is the base of F $\}$
• It proceeds step by step.	• It considers all the literals at once

PTO

Procedural semantics	Declarative semantics.
• If A_1 is true, do A_2, if A_2 is true, do A_3... do A_n.	• The F is true if all A_i are true
• It follows a more derived approach.	• It follows a ~~dto~~ determined approach. (model theoritic semantics)

SSN 4

3. Type I non-determinism:

If there are several ~~procedure calls~~ goal clause that causes evolution to produce output with a single clause, it is type I non-determinism.

eg: ~~A~~A, B, C → B, C
~~B, C, E, F~~

B
~~A~~ :- B
~~A~~ :- EF
A :- CD

eg: A, B, C → A, C
A, E, F, C
A, D, C

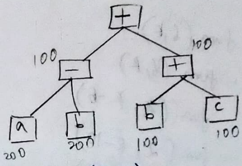
B :- A
B :- EF
B :- CD

SSN 5

4. op (100, xfx, +).
op (200, xfx, -).

precedence of (b+c) in (a-b)+(b+c).

The precedence of (b+c) is 100.



It is given the ~~second~~ last preference ~~in~~ (a-b) + (b+c).

5. (a) X is *(-(4,3), +(2,3)).
⇒ *(1,5) [∵ (4-3=1) & (2+3=5)]
⇒ 5 [∵ 1*5=5]

$\boxed{\text{Output is 5.}}$

(b) X = *(-(4,3), +(2,3)).

$\boxed{\text{The output is } *(-(4,3), +(2,3))}$.

The 'is' keyword calculates and evaluates the given infix expression.
The '=' keyword assigns the string to X.

[P.T.O]

BCD to Hex:

Label

Samp
00
00

(6) fun(0,0).

fun(N,R) :- N>0, N1 is N-2, fun(N1,R1), R is R1+N

- fun(6,R).

|?- fun(6,R)

fun(4, R-6)

fun(2, R-10)

fun(0, R-12)

R=12

Hence the value of $\boxed{R \text{ is } 12.}$

The "fun" represents,

$\boxed{\text{Sum of first } N/2 \text{ even numbers.}}$

$\boxed{\text{Part-B}}$

(7) Model theoritic Semantics:

The declarative semantic follows this particular model theoritic semantics.

$S_{mod}(P) = \{H \mid H \text{ is } A_1 \wedge A_2 \wedge \ldots A_n\}$ are the base which derives the F}.

It is based on the followingelements,

(i) Herbrand Universe

(ii) Herbrand Base

(iii) Interpretations

(iv) Model

(v) Minimal model.

Herbrand universe:

Let ~~H(B)~~ H(U) be the Herbrand universe, It contains the set of all semantic elements.

eg: ~~person~~ parent (Sibin, Tubin)

parent (Tubin, Mibin)

H(U) = { Sibin, Tubin, Mibin}

Herbrand Base:

Let H(B) be the Herbrand base, it consists of all possible ~~the~~ semantic goals.

eg: parent (Sibin, Tubin)

parent (Tubin, Mibin)

H(B) = {Parent (Sibin, ~~Tubin~~ Sibin), Parent (Tubin, Tubin), parent (Tubin, Sibin), parent (Mibin, Sibin)...}

[PTO

Interpretation:

Let I be all the interpretations,

I is a subset of Herbrand Base.

It assigns based on the truth values of the given clauses.

eg: parent (sibin, Tubin).
parent (Tubin, Mitin)

I = { parent (sibin, Tubin), parent (Tubin, Mitin)}

Minimal model:

It is a set of all the possible values and conclusions along with clauses to form a model.

eg. ~~Genius (X)~~ mathematician (~~[illegible]~~ Ramanujan)

Mathematician (X), Genius (X).

Minimal model is,

{ Mathematician (Ramanujan),
Genius (Ramanujan)}.

8. List operations:

(a) Delete Element from list:

delete1 (X, [X, Tail], Tail).
delete1 (X, [Y, Tail1], [Y, Tail2]) :- delete1 (X, Tail1, Tail2)

The base case is,

If X equals to header of list, the header is deleted.

If not,
recursion takes place by considering the subsequent tail portion of the lists.

eg: ?- delete1 (3, [1, 2, 3, 4], X)

1	1	call :	delete1 (3, [1, 2, 3, 4], _16)?
2	2	call :	delete1 (3, [2, 3, 4], _32)?
3	3	call :	delete1 (3, [3, 4], _64)?
3	3	Exit :	delete1 (3, [3, 4], [4])?
2	2	Exit :	delete1 (3, [2, 3, 4], [2, 4])?
1	1	Exit :	delete1 (3, [1, 2, 3, 4], [1, 2, 4])?

X = [1, 2, 4]

Yes

P.T.O

(b) permutation:


```
perm([],[]).
perm([X|L],P) :- perm(L,L1), insert(X,L1,P).
insert(X,List,BiggerList) :- del(X,BiggerList,List).
```

If the list is Null, the operation stops.
else, the permutation is done with recursion along with insert which is derived from delete().
(delete() is done in (a))

eg: perm([1,2,3])
?-

perm([1,2,3])?
perm([2,3,1])?
perm([3,1,2])?
perm([3,2,1])?
perm([2,1,3])?
perm([1,3,2])?
~~perm([3,1,2~~

No.

9. Family knowledge base:

Unary:

Male(Tom).
Male(Greg).
Male(Mike).
Female(Lisa).
Female(Pat).
Female(Anne).

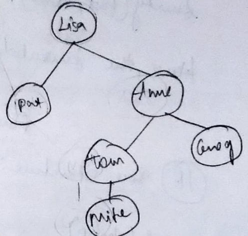


fig: An example of a family knowledge base

Binary:

parent(Lisa, Pat).
parent(Lisa, Anne).
parent(Anne, Tom).
parent(Anne, Greg).
parent(Tom, Mike).

Construction of prolog rules:

• Uncle:

?- Uncle(X,Y) :- Parent(Z,X), Parent(Z,Y1),
Male(Y1).

• Aunty

?- Aunty(X,Y) :- parent(Z,X), parent(Z,Y1),
Female(Y1).

[P.T.O

BCD to Hex:

Label

Uncle (Greg, Mike).
Aunty (Pat, Tom).

Hence, the knowledge base is represented.

Part-C

11. Tree structure:

male (philip).
female (Elizabeth).
parent (Elizabeth, Charles).
parent (Elizabeth, anne).
parent (philip, anne).
father (X, Y) :- parent (X, Y), male (X).
mother (X, Y) :- parent (X, Y), female (X).

Goal:
Mother (X, Charles).

On applying procedural semantics,

Mother (X, Charles)
[Y | Charles]
parent (X, ~~Y~~), female (X)
~~[X]~~
parent (Elizabeth, Charles)
female (Elizabeth)

which means,
{mother (X, Y) :- parent (X, Y), female (X), (¬Mother (X, Charles))}
[Y, Charles]
Parent (X, Charles), female (X)
[X, Elizabeth]
parent (Elizabeth, Charles)
female (Elizabeth)
[ ]
parent (Elizabeth, Anne)
X

∴ Elizabeth is the mother of Charles.
X = Elizabeth.

[P.T.O

BCD to Hex:
Label

- Mother(X, Charles) is the goal,
- Mother(X,Y) relations are all checked.
- The matching relation is parent (X, Y), female(X).
- parent (X, Y) are all checked.
- parent (Elizabeth, Charles) is matched.
- Female (Elizabeth) is checked.
- It returns a null value, so, the program is concluded.

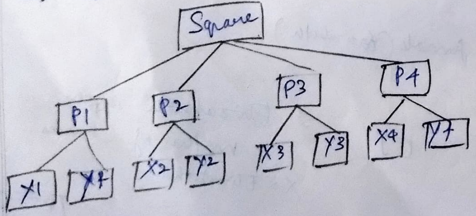
10

---

12. 'point' relation:

point (X1, Y1)
point (X2, Y2)
point (X3, Y3)
point (X4, Y4).

(a) Square:



~~?- Square(P1, P2, P3, P4) :-~~

~~?- Square~~

?- distance (Point1(X1, Y1), Point2 (X2, Y2))
:- Z1 is (X2 - X1)**2, Z2 is (Y2 - Y1)**2,
distance(point1(X1, Y1), point2(X2, Y2), Z4).
Z3 is Z1 + Z2,
Z4 is Z3**1/2,

Hence the distance is defined.

?- Square (P1(X1, Y1), P2 (X2, Y2), P3(X3, Y3),
P4 (X4, Y4)) :- distance (P1, P2) =:= distance (P3, P4),
distance (P2, P3), distance (P2, P3) =:=
distance (P4, P1).
distance (P3, P4) =:= distance (P4, P1).

(b) Rectangle:

?- Rectangle (P1(X1, Y1), P2 (X2, Y2), P3(X3, Y3),
P4(X4, Y4)) :-
distance (P1, P2, Z1) =:= distance (P3, P4, Z1),
distance (P2, P3, Z2) =:= distance (P4, P1, Z2).

[P-70]

BCD to H
Label

ⓒ Isosceles trapezoid :

?- isosceles trap (P1(X1,Y1), P2(X2,Y2), P3(X3,Y3),
P4(X4,Y4)) :-
distance (P1,P2,Z1) =:= distance (P3,P4,Z1),
distance (P2,P3,Z2) =/= distance (P4,P1,Z2)

ⓓ Parallelogram:

?- parallel (P1(X1,Y1), P2(X2,Y2), P3(X3,Y3),
P4(X4,Y4)) :-
distance (P1,P2,Z1) =/= distance (P2,P3,Z1),
distance (P3,P4,Z1) =/= distance (P4,P1,Z2).

ⓔ Rhombus :

?- rhombus (P1(X1,Y1), P2(X2,Y2),
P3(X3,Y3), P4(X4,Y4)) :-
distance (P1 P2 Z1) =:= distance (P2 P3 Z1)
distance (P2, P3, Z2) =:= distance (P4, P1, Z2)
distance (P3, P4, Z3) =:= distance (P1, P2, Z3)
distance (P2, P1, Z4) =:= distance (P4, P3, Z4)