uOttawa

L'Université canadienne
Canada's university

Faculty of Engineering
School of Information Technology and Engineering

CSI 2165B: Prolog Concepts Laboratory

Final Exam

Examiner:
Diana Inkpen

April 27, 2006, 14:00
Total marks: **52**
Duration: **3** hours
Total number of pages: **14**

Last Name:		**First Name:**	
Student Number:			

Important Regulations:
1. **No calculators are allowed.**
2. **A student identification cards (or another photo ID and signature) is required.**
3. **An attendance sheet shall be circulated and should be signed by each student.**
4. **<u>Please answer all questions on this paper, in the indicated spaces.</u>**

A	B	C	D	E	F	G	Total
5	**10**	**5**	**20**	**4**	**4**	**4**	**52**

A. [5 marks]
Which of the following are syntactically correct Prolog objects? If yes, identify the types of object they are (atom, number, variable, structure). If not, use a “No” as an answer.

	Term	Type
1.	5a	No
2.	_L	variable
3.	var1	atom
4.	f([1, a, 3])	structure
5.	Likes(John, Mary)	No

B. [10 marks]

Show the variable instantiations (the values of X, Y and Z) if the matching between the term in the first column and the term in the second column succeeds. Use “No” otherwise.

	Term1	Term2	Instantiations
Example	[1, 2, 3]	[X | Y]	X = 1 Y = [2, 3]
1.	[f(a), f(b)]	[X | Y]	X = f(a) Y = [f(b)]
2.	[f(a), f(b), c]	[X, Y]	X = No Y = No
3.	[3, 2, 1]	[g(X), Y, Z]	X = No Y = No
4.	[g(Y), 2, f(1)]	[X, Y, Z]	X = g(2) Y = 2 Z = f(1)
5.	[1, [2, [4]], 3]	[X,Y,Z]	X = 1 Y = [2, [4]] Z = 3
6.	[1, 2]	[_ | [X| [Y]]]	X = No Y = No
7.	[1, 2, 3]	[X | [Y | Z]]	X =1 Y = 2 Z = [3]
8.	[3, Z, 1]	[X, a | Y]	X = 3 Y = [1] Z = a
9.	f(g(X, a), Y)	f(Y, g(b, Z))	X = b Y = g(b, a) Z = a
10.	p(f(X), g(b, Y))	p(f(g(c)), g(X, a))	X = No Y = No

C. [5marks]

Indicate which one of the following choices is correct. (Note: Only **one** of the answers is correct.) **Use “Answer Sheet: Part C” to indicate your answer.**

What is the SWI-Prolog answer to the following queries?

1. `?- f(a, a) =.. [X, Y, Y].`

a) X=a, Y=a

b) X=f, Y=a

c) X=a, Y=f

d) No

2.
```
?- assert(a(3)),assert(a(2)), assert(a(1)),
    retract(a(X)),write(X),write(' '),fail.
```

a) 1 2 3
Yes

b) 1 2 3
No

c) 3 2 1
Yes

d) 3 2 1
No

3. `?- X is 5, Y is X + 1, fail.`

a) X = 5, Y = 6
No

b) No

c) X = 5, Y = 6
Yes

d) none of the above

4.
```
?- assert(a(3)),assert(a(2)),retract(a(X)),!,write(X),
    write(' '),fail.
```

a) 3 2
Yes

b) 2 3
No

c) 2 3
Yes

d) 3
No

5. `?- member(X, [a, b]), !, write(X), write(' '), fail.`

a) No

b) a
No

c) a b
No

d) a b
Yes

Answer Sheet: Part C

Question	**Answer**
1.	**B**
2.	**D**
3.	**B**
4.	**D**
5.	**B**

D. [20 marks]

Read the following programs and provide the answer to the query (only the first answer that SWI-Prolog would produce).

1.
 	
Program
	`prog_1([X, _], [X, X]).`
	
	`prog_1([H|T], [H|T1]):- prog_1(T, T1).`

Query	`?- prog_1([a,b,c,d],P).`
Answer	`P =` **`[a, b, c, c]`**

2.
 	
Program
	`prog_2(L,R) :- acc(L, 1, R).`
	
	`acc([_],A,A).`
	
	`acc([H|T],A,R):- A1 is A*H, acc(T,A1,R).`

Query	`?- prog_2([1,2,3,4],R).`
Answer	`R =` **`6`**

3.
 	
Program
	`prog_3([], 0, 0).`
	
	`prog_3([X|L], N, M):- X >= 0,`
	`                     prog_3(L, NL, M),`
	`                     N is NL + 1.`
	
	`prog_3([X|L], N, M):- X < 0,`
	`                      prog_3(L, N, ML),`
	`                      M is ML + 1.`

Query	`?- prog_3([3, -1, 1, 5], X, Y).`
Answer	X = **3,** Y = **1**

4.
 	
Program
	`p1(X) :- X is 5,!,fail.`
	
	`p1(_).`
	
	`p2(X) :- p1(X),fail,!.`
	
	`p2(_).`

Query	`?- p2(4).`
Answer	**Yes**

5. Program	`prog_5(nil, []).` `prog_5(t(X,nil,nil), [X]):- !.` `prog_5(t(X,L,R), Y):- prog_5(L, NewL), append([X | NewL], NewR, Y), prog_5(R, NewR).`
Query	`?- prog_5( t(1, t(2, t(4, nil, nil), t(5, nil, nil)), nil), Z).`
Answer	`Z =` **`[1, 2, 4, 5]`**

Note: t(Node, LeftSubTree, RightSubTree) is a structure representing a binary tree;
nil is a constant used to represent empty sub-trees

6. Program	`prog_6(E, t(Node,L,nil), t(Node,L, t(E,nil,nil))).` `prog_6(E, t(Node,L,R), t(Node,L,M)):- R \== nil, prog_6(E, R, M).`
Query	`?- prog_6(3,t(5, t(1, nil, nil), t(4, nil,nil)), M).`
Answer	`M = t(5, t(1, nil, nil), t(4, nil, t(3, nil, nil)))`

7.

Program

```
prog_7([],0).

prog_7([H|T], N) :- is_list(H), !,
                    prog_7(H,N1),
                    prog_7(T,N2),
                    N is N1 + N2.

prog_7([_|T], N) :- prog_7(T,N1), N is N1 + 1.
```

Query

```
prog_7([1, [2,[3]], [4, 5]], R).
```

Answer

```
R = 5
```

8.

Program

```
prog_8([], []) :- !.

prog_8([H|T], L3) :-
              prog_8(H, L1),
              prog_8(T, L2),
              append(L1, L2, L3), !.

prog_8(X, [X]).
```

Query

```
?- prog_8([a, [b, [c, d, [e, f, [g, h]]], i]], L)
```

Answer

`L = [a, b, c, d, e, f, g, h, i]`

9.
 	
Program
	`prog_9(t(Node, _, nil), Node).`
	
	`prog_9(t(Node, LT, _), M) :-`
	`             LT \== nil,`
	`             prog_9(LT, M).`

Query	`?- prog_9(t(25, t(15, t(12, nil, nil), t(17, nil, nil)), t(35, t(29, nil, nil), nil)), N).`
Answer	**`N = 12`**

10.
 	
Program
	`new_stack(stack([])).`
	
	`push(Elem, stack(Stack), stack([Elem | Stack])).`
	
	`pop(stack([Top | Stack]), Top, stack(Stack)).`

Query	`?- new_stack(S1), push(1, S1, S2), push(2, S2, S3),`
	`   pop(S3, E, S4).`

Answer	`S1 = stack([])`
	
	`S2 = stack([1])`
	
	`S3 = stack([2, 1])`
	
	`S4 = stack([1])`
	
	`E = 2`

E. [4 marks]

Write a predicate named `delete` that takes three arguments: a list, a number N, and an output list, and deletes N elements from the end of the list to produce the output list.

Example:
```
?- delete([a,b,c,d,e,f,g], 4, L).
L = [a, b, c] ;
No

?- delete([a,b,c,d,e],1,L).
L = [a, b, c, d] ;
No
```

```
delete(L, N, []):- length(L,N1), N1 =:= N, !.

delete([H|T], N, [H|T1]) :- delete(T, N, T1).
```

F. [4 marks]

Write a predicate `sum` that computes the sum of all the elements in a nested list, at all levels. The first argument is the nested list and the second is the sum. If some elements in the list are not numbers, they are not included in the sum.

Example:

```
?- sum([a,[[b, 5, [1]], d], 3, 2],R).

R = 11
```

```
sum([],0).

sum([H|T], R):- is_list(H), !,
                sum(H, R1), sum(T, R2), R is R1 + R2.

sum([H|T],R):- number(H), !, sum(T,R1), R is R1 + H.

sum([H|T],R):- sum(T,R).
```

G. [4 marks]

Consider the following Prolog database:

```
mushroom(toadstool).
mushroom(morel).
mushroom(shiitake).

poisonous(toadstool).

edible(shiitake).
edible(morel).

likes(john, Z) :- poisonous(Z).

eats(didier, X) :- likes(john, X).
eats(paul, X) :- edible(X).
eats(marie, X) :- mushroom(X).
eats(john, X) :- likes(john, X), edible(X).

dies(X) :- eats(X,Y), poisonous(Y).
```

1. How does Prolog answer the query
   `?- dies(X).`
   assuming that you hit ";" until you get all the possible answers?

2. How does Prolog answer the query
   `?- eats(X, morel).`
   assuming that you hit ";" until you get all the possible answers?

Solution

```
1. ?- dies(X).
X = didier;
X = marie;
No


2. ?- eats(X, morel).
X = paul;
X = marie;
No
```