Université d'Ottawa
Faculté de génie

École de science
d'informatique
et de génie électrique

uOttawa
L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering

School of Electrical
Engineering
and Computer Science

Assignment 2
CSI2120 Programming Paradigms

Winter 2019

Due on March 22nd before 11:00 pm in Virtual Campus

6 marks

There are [10 points] in this assignment. The assignment is worth 6% of your final mark.

For Questions 1 and 2 please refer to the database on the last page of this assignment.

Question 1. [2.5 points]

Find the query to obtain the following answer:

a) Obtain a list of all basketball players:
`L = [jane, joe, robert].`

b) Obtain a list of Ontario teams
`L = [ggs, gryphons, queens, ravens].`

c) Obtain the list of players that play more than one sport:
`L = [annie, suzy].`

d) Use **`findall`** obtain the list of players for Ottawa teams and their sport:
```
L = [(paul, crosscountry),  (joe, basketball),  (marie,
crosscountry),  (marie, crosscountry),  (simon, lacrosse)].
```

e) The previous list contains twice the name of `marie`, why? Show how one can use `setof` to eliminate this problem.
```
L = [(joe, basketball),  (marie, crosscountry),  (paul,
crosscountry),  (simon, lacrosse)].
```

Use your exact five queries from a)-e) above and create a predicate `myAnswer` that prints using write the above answers, i.e., the following query is to work:

```
?- myAnswer.
[jane,joe,robert]
[ggs,gryphons,queens,ravens]
[annie,suzy]
[(paul,crosscountry),(joe,basketball),(marie,crosscountry),(marie,cro
sscountry),(simon,lacrosse)]
[(joe,basketball),(marie,crosscountry),(paul,crosscountry),(simon,lac
rosse)]
true.
```

Aside : This will help us to mark your assignment in a timely fashion.

<u>Question 2.</u> **[2 points]**

The predicate below could read as follows:
"*A player is of interest if the sport is* `ski` *or the player plays for a* `quebec` *team*".

```
interest(X) :-
     sport(X, ski).

interest(X) :-
     sport(X,S),
     S \= ski,
     player(X, T),
     team(T, C),
     city(C, quebec).
```

Draw the complete search tree for the following query:

```
?- interest(X).
```

The Prolog search tree shows, for each node, the current goals to be proven. The branches of the tree must show the unification performed in order to pass from a parent node to its child node.

Question 3. **[1.5 points]**

Give a predicate `area/2` that calculates the area of a triangle in two dimensions with the determinant formula. Example:

```
?- area( [[4,0],[7,2],[2,3]], A).
A = 6.5.
```

This area can be found as follows: $A_T = \frac{1}{2}\begin{vmatrix} b_x - a_x & c_x - a_x \\ b_y - a_y & c_y - a_y \end{vmatrix}$

Note that the predicate uses two parameters: The corners or vertices of the triangle are specified in a list of lists, while the second parameter is the result.

Question 4. **[2 points]**

a) Write a predicate `skip` that returns a list which skips elements as toggled by elements in a second list. In particular, the elements of the first list are skipped until an element is encountered that is in the second list. All elements after this element are included in the result list. When another element of the first list is encountered that is also in the second list, then the elements of the first list are skipped again. In other words, the predicate toggles back-and-forth from skipping to insertion into the output list.

```
?- skip([1,2,3,4,5],[3],L).
L=[4, 5].

?- skip([b,3,5,hello,5,7,z],[5,b,a],L).
L=[3, 5, 7, z].
```

b) Write a predicate `turn` that returns a list where the order of elements is reversed as toggled by elements in a second list. In particular, the elements of the first list are reversed until an element is encountered that is in the second list. All elements after this element are included in the result list in unchanged order. When another element of the first list is encountered that is also in the second list, then the elements of the first list are reversed again. In other words, the predicate toggles back-and-forth from reverse to forward insertion into the output list.

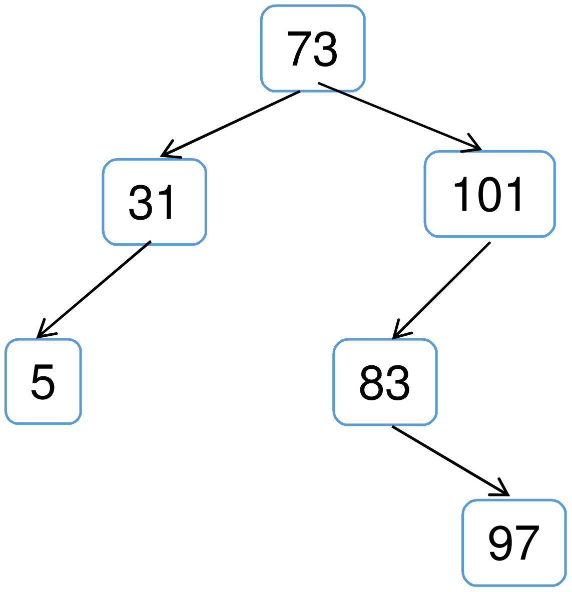
```
?- turn([1,2,3,4,5],[3],L).
L = [3, 2, 1, 4, 5].

?- turn([1,b,3,5,hello,5,7,z],[5,b,a],L).
L = [b, 1, 3, 5, 5, hello, 7, z].
```

Question 5. **[2 points]**

Consider a binary tree representation as discussed in class.

```
treeEx(X) :-
 X = t(73,t(31,t(5,nil,nil),nil),t(101,t(83,nil,t(97,nil,nil)),nil)).
```



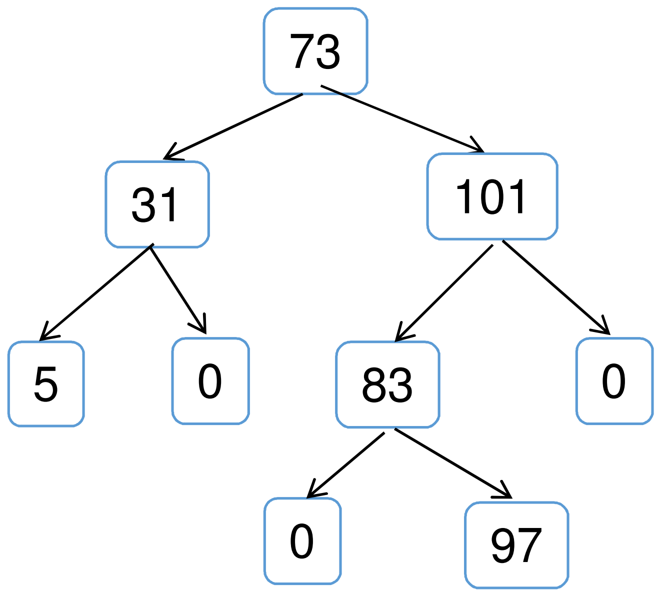
a) Code a predicate `single(tree,L)` that finds all nodes in the tree with exactly one single child node.

```
?- treeEx(T),single(T,L).
T = … , % Omitted
L = [31, 101, 83]
```

b) Code a predicate `singleFill(tree,V,ntree)` that add a new sibling node to every single node in the tree.

```
?- treeEx(T),single(T,0,L).
T = t(73, t(31, t(5, nil, nil), nil), t(101, t(83, nil, t(97,
nil, nil)), nil)),
L = t(73, t(31, t(5, nil, nil), t(0, nil, nil)), t(101, t(83,
t(0, nil, nil), t(97, nil, nil)), t(0, nil, nil)))
```

(see next page)


73
31
101
5
0
83
0
0
97

Appendix Database for Question 1 and 2.

```
city(ottawa,ontario).
city(guelph,ontario).
city(kingston,ontario).
city(gatineau,quebec).
city(montreal,quebec).
team(ravens,ottawa).
team(ggs,ottawa).
team(gryphons,guelph).
team(queens,kingston).
team(torrents,gatineau).
team(stingers,montreal).
sport(annie,lacrosse).
sport(paul,crosscountry).
sport(suzy,ski).
sport(robert,basketball).
sport(tom,lacrosse).
sport(tim,ski).
sport(annie,ski).
sport(joe,basketball).
sport(robert,basketball).
sport(jane,basketball).
sport(marie,crosscountry).
sport(suzy,crosscountry).
sport(jack,ski).
sport(simon,lacrosse).
player(annie,gryphons).
player(tom,torrents).
player(jane,stingers).
player(marie,ggs).
player(joe,ravens).
player(jack,queens).
player(simon,ravens).
player(suzy,torrents).
player(paul,ggs).
player(marie,ggs).
player(simon,gryphons).
```