Winter 2013

COSC 2P93 Prolog   Assignment #2

Due date: 12:00 noon Monday February 11.
Lates: 12:00 noon, Thursday February 14 (-25%).
Comments: See Assignment 1 comments for program requirements.
Hand in: Printouts of your programs and adequate examples of their execution. Electronic submission of all source code and files.

1. **Execution trees**: Consider this predicate:

```
member(A, [A|_]).
member(A, [_|B]) :- member(A, B).
```

Draw computation trees for the following queries. Label all tree branches, and refresh variable names when necessary. Indicate the final computed answer to query variables at the success leaf(s) of the tree.

(i) **?- member(b, [a,b,c]).**   (ii) **?- member(X, [a,b]).**

2. List processing: Write Prolog predicates for the following:

(a) grade/2: This takes a numeric grade G between 0 and 100. It then returns a letter grade value for that grade, based on this table:

$80 \le G \le 100$	$\rightarrow$ a
$70 \le G \le 79$	$\rightarrow$ b
$60 \le G \le 69$	$\rightarrow$ c
$50 \le G \le 59$	$\rightarrow$ d
$0 \le G \le 49$	$\rightarrow$ f

If G is out of range, or not a number, then the predicate fails.

(b) classGrade/2: This takes a list of grades as in (a) above, and returns a list with the letter grade matching every integer grade. You can assume that the input list is legal. For example,

```
?-  classGrade([ 2, 55, 67, 99, 25, 71], L).
L = [f, d, c, a, f, b]
```

(c) listSplitter/3:   Given a list InputList, it is split into 2 new lists. The first new list is the 1st, 3rd, 5th,... members of InputList. The second new list are all the even members. For example,

```
?-  listSplitter([ 1, a, b, 2, hello, 2.6, [2, 5]], A, B).
A = [1, b, hello, [2, 5]]
B = [a, 2, 2.6]

?- listSplitter( [a], A, B).
A = [a]
B = [ ]
```

(d) removeDups/2: This takes a list L, and removes all duplicate values from the list. You do not have to recursively remove duplicates from nested lists. For example,

```
?- removeDups([ a, b, a, [a, 2], 5, b], New).
New = [a, b, [a, 2], 5]
```

(e) evenFilter/2: This takes a list of integers List, and returns a new list with all the even integers from List. Furthermore, all the duplicates are removed from this new list. For example,

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

3. Factoring integers

Write a predicate that takes a positive integer Num, and returns a list of its prime factors. For example,

```
?- factor(25, L).
L = [5, 5]

?- factor(24, L).
L = [2, 2, 2, 3]

?- factor(17, L).
L = [17]
```

Hints:

a) You don't have to test any factor having a value greater than the truncated square root of Num.
b) Section 4.7.5 of the Sicstus 4.2.3 manual has a number of arithmetic operators that may be helpful. The builtin predicates number/1 and float/1 might also be of use.
c) You don't have to test any even factor other than 2. (This generalizes; see Sieve of Erastosthenes).

Comments:

Feel free to create additional predicates if they help your solution.

Try to use solutions to previous questions whenever possible. Many questions are designed with this in mind!

Show output of your work on Linux using "script". For example,

```
> script output.txt   (this will create a shell, with a new Linux prompt.
                       Everything to the screen will be copied into the file
                       'output.txt'.)

>  sicstus
          (Do your sicstus stuff).
?- halt. (leave sicstus)

> exit  (leave script)
```

You will now see a text file "output.txt" with all the I/O from your session. Print and hand in this file.