CSE 341: Homework 8

due 12/8/99

All predicates you are asked to implement and turn in should be typed and tested with CLP(R). With each predicate, turn in the source code listing as well as sample output for a demonstrative set of tests. Do not forget to test *end cases*. Use the `turnin` program to turn in an electronic copy of your source code. Be sure to show how your CLP(R) predicates can be used in both directions.

Part I: (Not to be turned in.)

1. Given definition:

```
length([],0).
length([X|Xs],1+L) :- L >= 0, length(Xs,L).

ones([]).
ones([1|Xs]) :- ones(Xs).
```

   Which of the following CLP(R) goals are satisfiable? What values for unknowns make them satisfiable?

   (a) `X + Y < 10, Y = 3.`
   (b) `length([1,2,3],L).`
   (c) `length([1,2,3],L).`
   (d) `length([1,2|Xs],L).`
   (e) `length(Xs,3).`
   (f) `length(Xs,3).`
   (g) `ones(Xs).`
   (h) `length(Xs,3),ones(Xs).`

2. Define a binary predicate `sum` that takes a list and a number. The predicate is true if the sum of the elements in the list is the number. For example:

```
1 ?- sum([1,2,3],S).

S = 6

*** Yes

2 ?- sum([X,2,3],10).

X = 5

*** Yes
```

Part II: (Not to be turned in.)

These are some sample problems that we will be discussing in section.

1. Define a binary predicate `same_length` that is true if its two arguments are lists that have the same length. For example:

```
1 ?- same_length([1,2],[3]).

*** No

2 ?- same_length([1,2],[red,blue]).

*** Yes

3 ?- same_length([1,2],X).

X = [_h5, _h7]

*** Retry? y

*** No
```

2. Write a ternary predicate `index` that takes as parameters a list, an index, and a value. `index(L,I,X)` should be true if the element `X` is in the list `L` at index `I` (the first element is at index 0). For example:

```
1 ?- index([dog,cat,bird,cat],2,X).

X = bird

*** Retry? y

*** No

2 ?- index([dog,cat,bird,cat],I,cat).

I = 1

*** Retry? y

I = 3

*** Retry? y

*** No
```

```
3 ?- index(L,2,bird).

L = [_h1, _h3, bird | _h6]

*** Retry? y

*** No
```

3. Write a unary predicate `matrix` that is true if its argument is a matrix. We will represent a matrix as a list of rows. Thus

   ```
   M = [[1,2,3],
        [4,5,6]].
   ```

   is a 2×3 matrix (two rows and three columns). To verify that `M` is a matrix we just need to make sure that it has the same number of columns in every row. Thus

   ```
   N = [[1,2,3],
        [4]].
   ```

   is not a matrix. Here are some example usages:

   ```
   4 ?- matrix([[1,2,3],[4,5,6]]).

   *** Retry? y

   *** No

   5 ?- matrix([[1,2,3],[4]]).

   *** No
   ```

Part III: (Due Wednesday, 12/8/99. Turn this in.)

Complete the following. You must turn in output of your code on a demonstrative set of test cases. Show your predicates being used in both directions where applicable.

1. (5 points)

   (a) Write a unary predicate `fiblist` that takes in a list and is satisfiable if the list contains Fibonacci numbers starting from 0. For example:

   ```
   1 ?- fiblist([0,1,1,2,3,5]).

   *** Yes

   2 ?- fiblist([5,6]).
   ```

```
*** No
```

(b) Use `fiblist` to define binary predicate `fib`. `fib(F,N)` is satisfiable if `F` is the `N`th Fibonacci number.

```
1 ?- fib(F,7).

F = 13

*** Yes
```

2. (10 points)

Given a list of word lengths, we would like to format them for printing on a page that is `H` lines and `W` characters wide. Define a predicate `format(Format,Words,H,W)` that is true if `Format` is correctly formatted in height at most `H` and width at most `W` with word lengths given by `Words`. `Words` is a list of word lengths. If we were trying to format the string "If we were trying to format the string" then `Words` would be `[2,2,4,6,2,6,3,6]`. `Format` is a list of lines where each line is a list of word lengths that are on that line. So if we formated our above string as:

```
If we were
trying to
format the
string
```

then `Format` would be `[[2,2,4],[6,2],[6,3],[6]]`. No all lines should be less that `W` characters long including the space between each word. There should be less than `H` lines. CLP(R) would give the following results:

```
1 -? format(Format,[2,2,4,6,2,6,3,6],4,10).

Format = [[2, 2, 4], [6], [2, 6], [3, 6]]

*** Retry? y

Format = [[2, 2, 4], [6, 2], [6], [3, 6]]

*** Retry? y

Format = [[2, 2, 4], [6, 2], [6, 3], [6]]

*** Retry? y

*** No
```

Is it possible to format this above string in at most 5 lines with at most 9 characters per line?