

Problem Set 5  
TK2ICM: *Logic Programming* (CSH4Y3)  
Second Term 2018-2019

Day, date : Tuesday, April 9, 2019  
Duration : **60 minutes**  
Type : ***open all***, individual (no cooperation between/among class participants)

Instruction:

1. You are not allowed to discuss these problems with other class participants.
2. You may use any reference (books, slides, internet) as well as other students who are not enrolled to this class.
3. Use the predicate name as described in each of the problem. **The name of the predicate must be precisely identical.** Typographical error may lead to the cancellation of your points.
4. Submit your work to the provided slot at CeLoE under the file name PS5-<your\_name>.pl. For example: PS5-Albert.pl. Please see an information regarding your nickname at google classroom.

## 1 Interactive Grade Converter

**Remark 1** This problem is worth 20 points.

Write the predicate `igc/0` that interactively ask the user to input a number between 0 and 100 (inclusive) and outputs the corresponding index grade of that number. The grade is converted using the following rule. Suppose  $G \in [0, 100]$ , then the index of  $G$ , denoted by  $ind(G)$ , is defined as follows

$$ind(G) = \begin{cases} \mathbf{A}, & \text{if } 80 < G \leq 100 \\ \mathbf{AB}, & \text{if } 70 < G \leq 80 \\ \mathbf{B}, & \text{if } 65 < G \leq 70 \\ \mathbf{BC}, & \text{if } 60 < G \leq 65 \\ \mathbf{C}, & \text{if } 50 < G \leq 60 \\ \mathbf{D}, & \text{if } 40 < G \leq 50 \\ \mathbf{E} & \text{if } 0 \leq G \leq 40. \end{cases}$$

The interaction will be halted if the user type the word `end`. In addition, the program outputs a warning Input must be a number between 0 - 100 (inclusive) or a string "end" without quotes. if the input is not a number in  $[0, 100]$  neither a string `end`. An example of an I/O interaction is as follows:

?- igc.

Write a number between 0 - 100 (inclusive) or a string "end" without quotes:

|: 79.99.

The index is AB

Write a number between 0 - 100 (inclusive) or a string "end" without quotes:

|: 90.11.

The index is A

Write a number between 0 - 100 (inclusive) or a string "end" without quotes:

|: 56.57.

The index is C

Write a number between 0 - 100 (inclusive) or a string "end" without quotes:

|: 101.50.

Input must be a number between 0 - 100 (inclusive) or a string "end" without quotes.

Write a number between 0 - 100 (inclusive) or a string "end" without quotes:

|: twenty.

Input must be a number between 0 - 100 (inclusive) or a string "end" without quotes.

Write a number between 0 - 100 (inclusive) or a string "end" without quotes:

|: 0.

The index is E

Write a number between 0 - 100 (inclusive) or a string "end" without quotes:

|: end.

You choose to end the program, thank you!

**true.**

## 2 Monotone List

**Remark 2** This problem is worth 20 points.

Let  $L = [a[0], a[1], \dots, a[n-1]]$  be a list of length  $n$ . We say  $L$  is *monotone* if one of the following condition is satisfied:

- for all  $1 \leq i \leq j \leq n$  we have  $a[i] \leq a[j]$ , formally  $\forall i \forall j (i \leq j \rightarrow a[i] \leq a[j])$ , in this case the list  $L$  is *monotonically increasing*; or
- for all  $1 \leq i \leq j \leq n$  we have  $a[i] \geq a[j]$ , formally  $\forall i \forall j (i \leq j \rightarrow a[i] \geq a[j])$ , in this case the list  $L$  is *monotonically decreasing*.

For instance, we have:

- the list  $L = [-1, 0, 1, 1, 1, 2, 3]$  is *monotonically increasing*,
- the list  $L = [2, 0, -1, -1, -1, -2, -3]$  is *monotonically decreasing*,
- the list  $L = [1, 1, 1, 1, 1, 1]$  is both *monotonically increasing* and *monotonically decreasing*
- the list  $L = [0, 1, 0, 1, 0, 1]$  is neither *monotonically increasing* nor *monotonically decreasing*,
- the list  $L = [1, 2]$  is *monotonically increasing*,
- the list  $L = [2, 1]$  is *monotonically decreasing*,
- the list  $L = [8]$  is both *monotonically increasing* and *monotonically decreasing*.

Write the predicate `monotone/1` that takes a non-empty list  $L$  as input, `monotone(L)` is true whenever  $L$  is a monotone list ( $L$  is monotonically increasing, monotonically decreasing, or both). Several examples:

- `?- monotone([-2,-1,-1,0,0,1,2,7,9]).` returns **true**.
- `?- monotone([100,11,9,9,8,3,1,0,-1]).` returns **true**.
- `?- monotone([9,9,9,9,9,9,9,9,9,9,9]).` returns **true**.
- `?- monotone([1,2]).` returns **true**.
- `?- monotone([3,2]).` returns **true**.
- `?- monotone([3,2,3]).` returns **false**.
- `?- monotone([1,2,1]).` returns **false**.
- `?- monotone([1,2,3,4,1,5,6,7,8,9]).` returns **false**.
- `?- monotone([9,8,8,7,6,5,3,4,1,0]).` returns **false**.

### 3 Hamming Distance

**Remark 3** This problem is worth 20 points.

Suppose  $L_1$  and  $L_2$  are two lists of the same lengths, the *Hamming distance* between  $L_1$  and  $L_2$  is the number of digits at which the corresponding components of  $L_1$  and  $L_2$  are different. Formally, suppose  $L_1 = [L_1[0], L_1[1], \dots, L_1[n-1]]$  and  $L_2 = [L_2[0], L_2[1], \dots, L_2[n-1]]$ , then the Hamming distance between  $L_1$  and  $L_2$ , denoted by  $\text{dist}_H(L_1, L_2)$ , is defined as

$$\text{dist}_H(L_1, L_2) = |\{i \mid (0 \leq i \leq n-1) \wedge (L_1[i] \neq L_2[i])\}|.$$

For example, the Hamming distance between  $L_1 = [2, 1, 7, 3, 8, 9, 6]$  and  $L_2 = [2, 2, 3, 3, 7, 9, 6]$  is 3, because the different components of  $L_1$  and  $L_2$  occurs at positions 2, 3 and 5.

In this problem your task is to write the predicate `hamming/3` such that `hamming(L1, L2, D)` is true whenever the Hamming distance between L1 and L2 is D. The first two arguments (i.e., L1 and L2) are always instantiated. If the length of L1 and L2 are different, then the program returns the string 'dimension error'. Several examples:

- `?- hamming([k,a,r,o,l,i,n],[k,a,t,h,r,i,n],D).` returns `D = 3`.
- `?- hamming([k,a,r,o,l,i,n],[k,e,r,s,t,i,n],D).` returns `D = 3`.
- `?- hamming([k,a,t,h,r,i,n],[k,e,r,s,t,i,n],D).` returns `D = 4`.
- `?- hamming([1,0,1,1,1,0,1],[1,0,0,1,0,0,1],D).` returns `D = 2`.
- `?- hamming([2,1,7,3,8,9,6],[2,2,3,3,7,9,6],D).` returns `D = 3`.
- `?- hamming([1,2,3],[3,2],D).` returns `D = 'dimension error'`.
- `?- hamming([1,2],[3,2,1],D).` returns `D = 'dimension error'`.
- `?- hamming([a,b,c],[3,2,1],D).` returns `D = 3`.

## 4 Arithmetic List

**Remark 4** This problem is worth 20 points.

A list  $L$  of  $n$  elements,  $L = [L[0], L[1], \dots, L[n-1]]$  is called as an *arithmetic list* if  $L[i+1] - L[i] = c$  for  $c \in \mathbb{R}$  for all  $0 \leq i \leq n-1$ . In other words, the differences between two consecutive elements is a constant and it is always identical for every pair of consecutive elements. Hence, an arithmetic list  $L$  of  $n$  elements can be represented as

$$[a, a+b, a+2b, \dots, a+(n-2)b, a+(n-1)b].$$

For example, we have:

- $[1, 2, 3, 4, 5, 6]$  is an arithmetic list of length 6,
- $[1, 4, 7, 10, 13, 16, 19]$  is an arithmetic list of length 7,
- $[5, 1, -3, -7, -11]$  is an arithmetic list of length 5,
- $[3, 3, 3, 3]$  is an arithmetic list of length 4,
- $[1, 0, 1, 0, 1, 0, 1]$  is not an arithmetic list,
- $[1, 3, 9, 27]$  is not an arithmetic list,
- $[1, 1, 2, 3, 5, 8]$  is not an arithmetic list.

Write a predicate `arithmetic/1` that returns true whenever the input is an arithmetic list. For example:

- `?- arithmetic([1,2,3,4,5,6]).` returns **true**.
- `?- arithmetic([1,4,7,10,13,16,19]).` returns **true**.
- `?- arithmetic([5,1,-3,-7,-11]).` returns **true**.
- `?- arithmetic([3,3,3,3]).` returns **true**.
- `?- arithmetic([1,0,1,0,1,0,1]).` returns **false**.
- `?- arithmetic([1,3,9,27]).` returns **false**.
- `?- arithmetic([1,1,2,3,5,8]).` returns **false**.

## 5 Maximum Odd Number

**Remark 5** This problem is worth 20 points.

Write a predicate `maxodd/2` that takes two argument, the first argument is a list of numbers and the second argument is the maximum odd value of such list. If the list contains no odd values, the predicate returns **false**. For example, we have:

- ?- `maxodd([1,1,2,3,5,8],M)` . returns `M = 3`.
- ?- `maxodd([1,1,2,3,9,10],M)` . returns `M = 9`.
- ?- `maxodd([10,12,2,30,52,88],M)` . returns **false**.