Prolog

Exercises form previous exams

Exercise 1

Provide a Prolog program that defines a predicate *couples(L,C,N)* that, given a list of integers L, returns a list of couples C that contains all the pairs of adjacent numbers in L , where the second element of the couple is greater or equal than the first one. The counter N must return the number of pairs which have been discarded.

For example, given the query

? -couples([0, 0, 12, 3, 11, 5, 7],C, N).

the program return the answer

C = [[0, 0], [0, 12], [3, 11], [5, 7]], N = 2

(the two discarded couples are [12,3] and [11,5]).

Exercise 1 - Solution

```
couples([], [],  0) :- !.

couples([_|[]], [], 0) :- !.

couples([A, B | T], [[A,B]| Res], M) :- A =< B, !, couples([B|T], Res, M).

couples([_, B | T], Res, M) :- couples([B|T], Res, N), M is N+1.
```

Exercise 2

Write a program that defines the predicate sum and prod(L,S,P) that, given a list of integers L, computes the product P and sum S of the numbers in the list.

Examples:

? - sum_and_prod ([5] , S , P). outputs P =5 , S =5.

? - sum_and_prod ([4 , 5] , S , P). outputs P =20 , S =9.

? - sum_and_prod ([3 , 4 , 5] , S , P). outputs P =60 , S =12.

Exercise 2 - Solution

```
sum_and_prod(L, S, P) :- sum_all(L, S), mul_all(L, P).

sum_all([], 0).
sum_all([H|T], S) :- sum_all(T, S1), S is H + S1.

mul_all([], 1).
mul_all([H|T], P) :- mul_all(T, P1), P is H*P1.
```

Exercise 3

Write a Prolog program that defines a predicate selectgreater(L1, L2, R, S), that given 2 lists L1 and L2 compares their elements pairwise and returns the list R of the greater elements, along with the sum S of all the element in the list R. If one of the two lists has more elements than the other, the elements in such a list must be included in R.

Examples:

? - selectgreater ([1 , 4 , 5] , [2 , 5 , 3] , R , S). outputs R =[2 , 5 , 5] , S =12.

? - selectgreater ([5 , 4 , 5] , [2 , 5 , 3] , R , S). outputs R =[5 , 5 , 5] , S =15.

? - selectgreater ([4 , 5] , [2 , 5 , 3] , R , S). outputs R =[4 , 5 , 3] , S =12.

Exercise 3 - Solution

```
selectgreater([], [], [], 0) :- !.

selectgreater([], [H|T], [H|Res], S) :- selectgreater([], T, Res, S1), S is S1 + H.

selectgreater([H|T], [], [H|Res], S) :- selectgreater(T, [], Res, S1), S is S1 + H.

selectgreater([H1|T1], [H2|T2], [H1|Res], S) :- H1 > H2, !, selectgreater(T1, T2, Res, S1), S is S1 + H1.

selectgreater([_|T1], [H2|T2], [H2|Res], S) :- selectgreater(T1, T2, Res, S1), S is S1 + H2.
```

CLP and Minizinc

Exercises form previous exams

Exercise 1

Anton, Beth, Carlos, and Daniel are eating a cake and must divide the 9 slices between each other. Anthon cooked the cake, so he wants more slices than anyone else. Beth has done her workout in the morning, so she deserves a treat and wants at least 3 slices. Carlos is on a diet, so he will eat less than 3 slices. Daniel wants to feel unique, so he will eat a number of slices that is different from anyone else, and at least 1. They want to save the remaining slices in the fridge, but the fridge is almost full, so only 1 slice can remain.

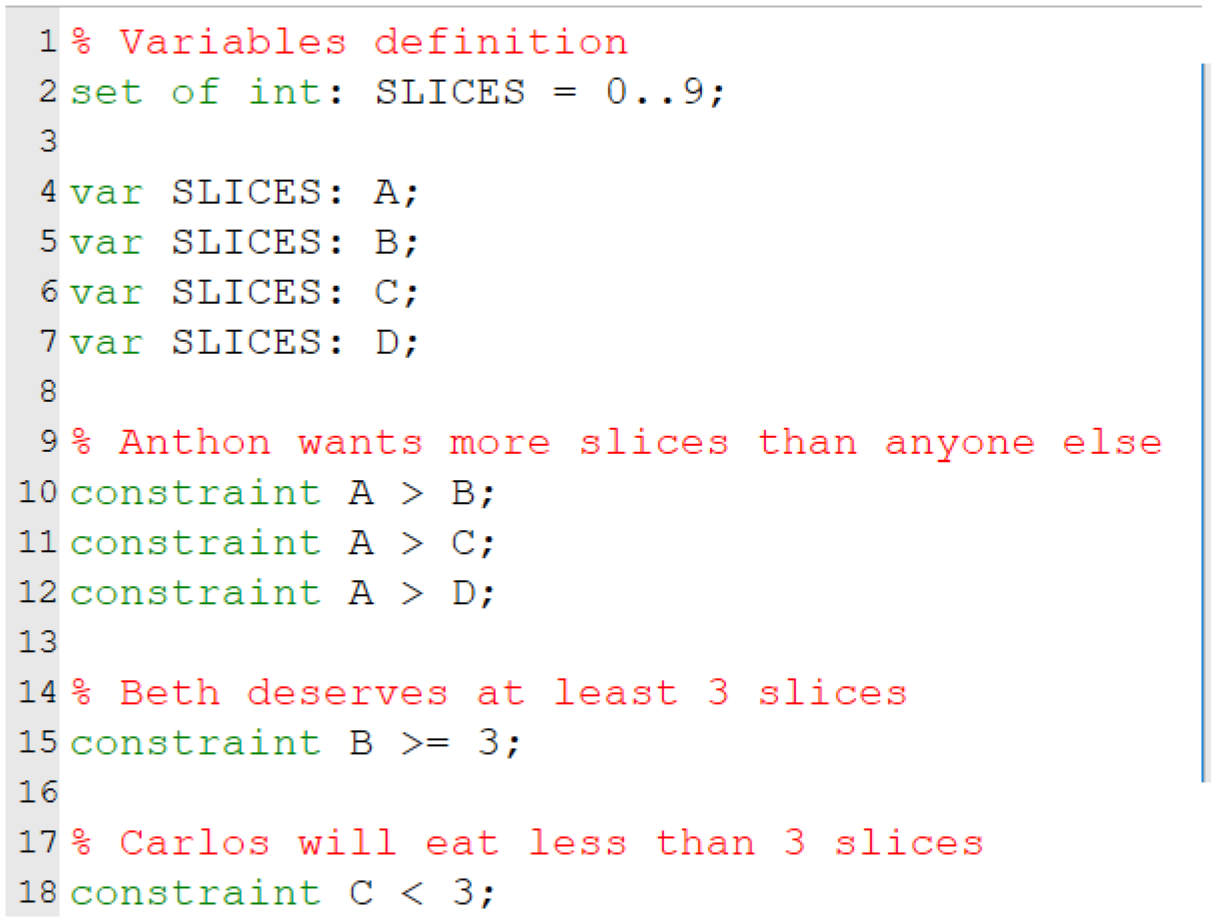
Write a CLP or minizinc program to compute how they can divide the slices. Please use comments so to make clear what is your reasoning and which variables will contain the final results.

Exercise 1 – CLP Solution

```
cake(VARIABLES) :-
                length(VARIABLES, 5),
                VARIABLES = [A,B,C,D,R],
                VARIABLES ins 0..9,
                R #=< 1,
                D #\= A, D #\= B, D #\= C,
                D #>= 1,
                C #< 3,
                B #>= 3,
                A #> B, A #> C, A #> D,
                A + B + C + D + R #= 9.


/** <examples> Your example queries go here, e.g.
?- cake(V), labeling([],V), V=[A,B,C,D,R].
**/
```

Exercise 1 – MiniZinc Solution



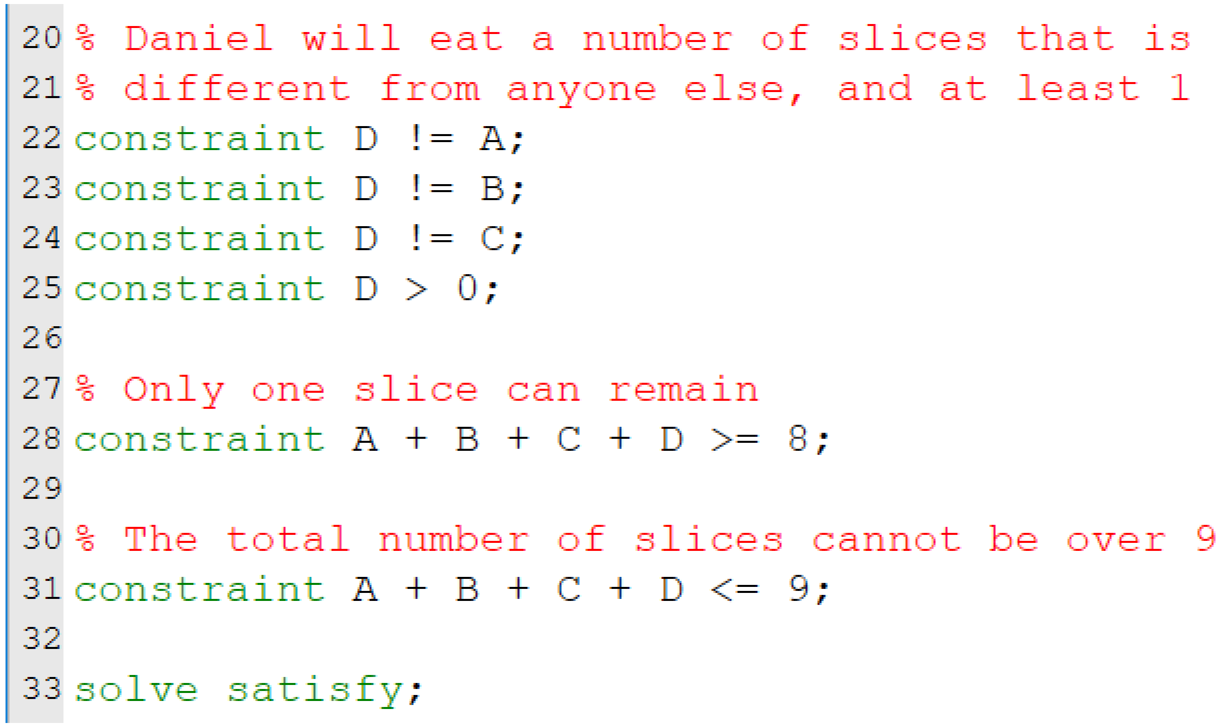
```
% Variables definition
set of int: SLICES = 0..9;

var SLICES: A;
var SLICES: B;
var SLICES: C;
var SLICES: D;

% Anthon wants more slices than anyone else
constraint A > B;
constraint A > C;
constraint A > D;

% Beth deserves at least 3 slices
constraint B >= 3;

% Carlos will eat less than 3 slices
constraint C < 3;
```



```
% Daniel will eat a number of slices that is
% different from anyone else, and at least 1
constraint D != A;
constraint D != B;
constraint D != C;
constraint D > 0;

% Only one slice can remain
constraint A + B + C + D >= 8;

% The total number of slices cannot be over 9
constraint A + B + C + D <= 9;

solve satisfy;
```

Exercise 2

A thief is stealing sculptures in an art gallery. Each sculpture has a specific value and a weight. The thief wants to steal sculptures for more than 600$ of total value, but can only have 11 kilos of sculptures in the sack. The thief must therefore decide what to take and what to leave.

Write a CLP or Minizinc program to compute which choices thief has, and for each choice, what is the final value of the stolen sculptures. The sculptures are 4 (A, B, C, D): A weights 10kg and it is worth 500$, B weights 4kg and it is worth 600$, C weights 2kg and it is worth 100$, D weights 2kg and it is worth 200$.

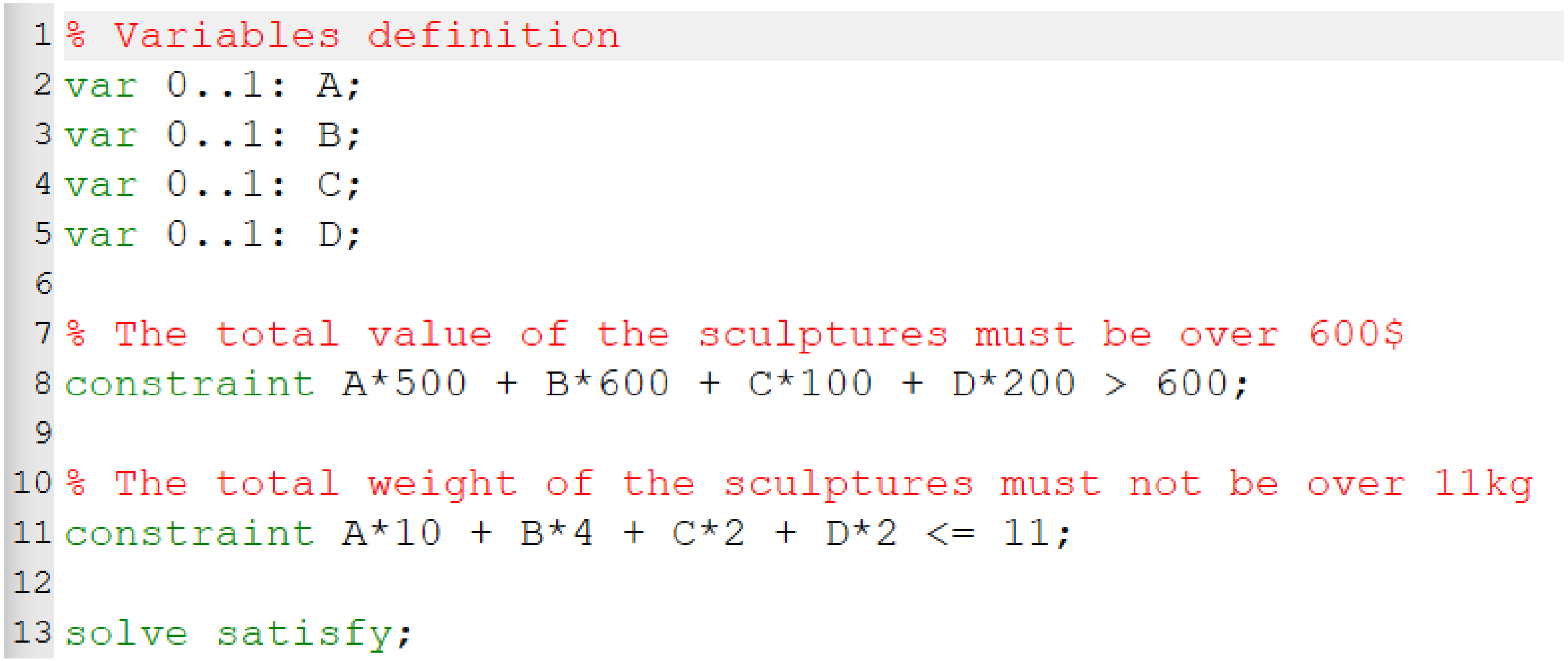
Exercise 2 – CLP Solution

```
bag(VARIABLES) :-
                length(VARIABLES, 6),
                VARIABLES = [A,B,C,D,W,V],
                [A,B,C,D] ins 0..1,

        Aw #= A * 10,
                Bw #= B * 4,
        Cw #= C * 2,
        Dw #= D * 2,
        Av #= A * 500,
        Bv #= B * 600,
        Cv #= C * 100,
        Dv #= D * 200,
        W #= Aw + Bw + Cw + Dw,
        V #= Av + Bv + Cv + Dv,
                W #< 12,
        V #> 600.

% query example
?- bag(V), labeling([],V), V=[A,B,C,D].
```

Exercise 2 – MiniZinc Solution



```
% Variables definition
var 0..1: A;
var 0..1: B;
var 0..1: C;
var 0..1: D;

% The total value of the sculptures must be over 600$
constraint A*500 + B*600 + C*100 + D*200 > 600;

% The total weight of the sculptures must not be over 11kg
constraint A*10 + B*4 + C*2 + D*2 <= 11;

solve satisfy;
```