MDOP Exam 1

Sunday, 13 April 2025 11:32 AM

In a darts game one has to throw a sequence of at most *n* darts to reach a *target* score. The dartboard is made of zones each of which have different score. The last dart thrown to reach the target score must be a final zone (in usual darts a double or triple score zone). Given a MiniZinc data file describing the game to be played of the form

```
enum ZONE;
ZONE: miss; % this zone represents missing the board or an unthrown dart
array[ZONE] of int: score; % the score for hitting each zone
constraint assert(score[miss] = 0,"miss zone must have 0 score");
array[ZONE] of bool: final; % true if this zone is a correct final zone
int: target; % target score to reach
int: n; % maximum number  of darts to throw;
```

enum Zone;
Makes a holder that can stand for a finite set of named vals

Zone: Miss;
declares a constant that is one of the Members Zone

array[Zone] of int: score;
makes an array that takes the enum Zone and Forces each element of the enum to an int

Constraint assert(score[Miss]=0, "Miss Zone must be eq to 0");
Catching Constraint that throws an error if the Constraint fails

array[Zone] of bool: final;
Makes a separate array that takes the enum where each element is a boolean

int: target;
Makes a Constant that is known before Solving

This differs from var int: target Which tells Solver that it needs to Find a value