Monkey Puzzle

A monkey lives in a tree with a branch off which hangs 10 vines. The vines hang in a row at exact distances of 1 metre apart. Somewhat perturbed by this unnatural degree of regularly, the monkey wonders how to hang his four bananas so that each banana is hung in a different vine. Furthermore, the money wants to hang his bananas so that for any three bananas, x, y and z, of his four bananas, the distance between x and y is different from the distance between y and z, which are both different from the distance between x and z. For example, the monkey can place the bananas in vines 1, 2, 4 and 8; 1, 2, 4, and 9; 3, 1, 10 and 7, etc. The monkey observes that are many solutions to this problem and therefore wonders how many ways he has available for arranging his bananas.

How to model the problem

Consider the solution where the bananas are placed in vines 1, 2, 4 and 9. We can observe that this is a solution by using a 2 dimensional grid (matrix) that **records the distance between each pair of bananas** (see the first diagram below). The cells in bold (5), for instance, indicates that the distance between the third banana, located in vine 4, and the fourth banana, located in vine 9, is 5. Note that the cells of the matrix are reflected along the diagonal; it is symmetric. Therefore there is no need to record both the top-right and bottom-left portion of the matrix (see the second diagram). Now we observe that 1, 2, 4 and 9 is a solution because the values in each portion of the matrix are different from one another. Moreover, the top-right can be represented as 3 lists of lengths 3, 2 and 1 respectively (see the third diagram), which can be concatenated to give the single list on the right (see the fourth diagram). Hence 1, 2, 4 and 9 is a solution because each value in the final list is different. This gives a strategy for solving problem, which you will follow by providing definitions of each the predicates detailed below:

$$\begin{bmatrix} 0 & 1 & 3 & 8 \\ 1 & 0 & 2 & 7 \\ 3 & 2 & 0 & \mathbf{5} \\ 8 & 7 & \mathbf{5} & 0 \end{bmatrix} \quad \begin{bmatrix} - & 1 & 3 & 8 \\ - & - & 2 & 7 \\ - & - & - & 5 \\ - & - & - & - \end{bmatrix} \quad \begin{matrix} [1,3,8] \\ [2,7] \\ [5] \end{matrix} \quad [1,3,8,2,7,5]$$

1. Write a predicate `distances(Bs, B, Ds)` where `Bs` and `Ds` are lists of variables such that the ith element of the `Ds` is the absolute difference between the variable `B` and the ith element of `Bs`. Hint: here is an illustrative query.

```
?- distances([X, Y], B, Ds).
Ds = [_G632, _G635],
_G647+X#=B,
_G632#=abs(_G647),
_G632 in 0..sup,
_G682+Y#=B,
_G635#=abs(_G682),
_G635 in 0..sup.
```

Note too that `_G632`, `_G635`, etc are internal Prolog variables which are used to express the relationships (constraints) between `X`, `Y`, `B` and `Bs`. The constraint `_G632 in 0..sup` merely expresses that the variable `_G632` takes an integer value between 0 and ∞.

2. Write a predicate `triangle(Bs, Ds)` where `Bs` a list of the positions of the bananas and `Ds` is the (single) list of differences in position. Hint: you can make use the built-in predicate `append` and the your own `distances` predicate. Further hint: here is an illustrative query for when `Bs` is of length 3 (to make the answer to the query easier to interpret).

```
?- triangle([X, Y, Z], Ds).
Ds = [_G442, _G445, _G448],
_G460+Z#=X,
_G472+Y#=X,
_G445#=abs(_G460),
_G445 in 0..sup,
_G507+Z#=Y,
_G448#=abs(_G507),
_G448 in 0..sup,
_G442#=abs(_G472),
_G442 in 0..sup.
```

3. Write a predicate `bananas(Bs)` which instantiates the list `Bs` such that the ith element of `Bs` is the number of the vine that holds the ith banana. Hint: here is an illustrative query, with some answers.

```
 ?- bananas(Bs).
Bs = [1, 2, 4, 8]
Bs = [1, 2, 4, 9]
Bs = [1, 2, 4, 10]
Bs = [1, 2, 5, 7]
Bs = [1, 2, 5, 10]
...
```

Note that the solution `Bs = [1, 2, 4, 8]` is considered to be distinct from `Bs = [2, 1, 4, 8]` which, in turn, is distinct from `Bs = [4, 2, 1, 8]`, etc.

4. Write a predicate `main(Solns)` which instantiates `Solns` to the number of ways to hang the bananas in the prescribed manner on the vines. Hint: use `findall`.

Submission details and deadline

Your solution must be a SWI-Prolog program, that uses the `clpfd` library. Just 16 marks will be awarded for this assessment; you should aim to make your code short and clear. The assessment is designed to reinforce your learning, but also to be lightweight. Your solution file should be named `bananas.pl`, to distinguish it from any working file that you retain in your submission directory. The first line of the file of `bananas.pl` must be a comment which gives your full name and login. You need to submit a .pl file;

not a .pdf or Word document. You need to place your solutions in: `/proj/co884c/monkey/`xyz on raptor where xyz is your own personal login. Permissions have been set so that only xyz can access files in the directory xyz. The deadline for submission is midnight on the Friday of week 24 (April 5th). After midnight, access to the submission directories will be removed. Section 2.2.1.1 of Annex 9 of the Credit Framework states that, "Academic staff may not accept course work submitted after the applicable deadline except in concessionary circumstances."