FIT5216: Modelling Discrete Optimization Problems
Inclass Task 12: Dinner Party

1 Problem Statement

The dinner party problem is as follows: We have a set of n people `PERSON` to seat at a circular dinner table. Each couple must be either seated adjacent or far apart. The interest of the dinner is the number of rivalry pairs which are seated adjacent or far apart. Far apart is considered as more than $\frac{n}{3}$ positions apart.

Data for the problem is defined as follows:

```
enum PERSON;
array[int,1..2] of PERSON: couple;
array[int,1..2] of PERSON: rivals;
```

Build a MiniZinc model `dinner.mzn`. The output should be a solution to the person array defined below

```
int: n = card(PERSON);
set of int: SEAT = 1..n;
array[SEAT] of var PERSON: person;
```

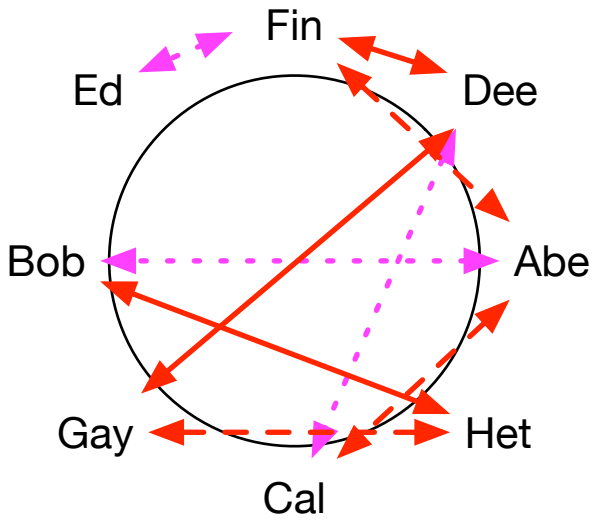
For example a small dataset is

```
PERSON = { Abe, Bob, Cal, Dee, Ed, Fin, Gay, Het };
couple = [| Abe, Bob
          | Cal, Dee
          | Ed, Fin |];
rivals = [| Abe, Cal
          | Abe, Fin
          | Bob, Het
          | Dee, Fin
          | Dee, Gay
          | Gay, Het |];
```

the model returns an answer

```
person = [Fin, Dee, Abe, Het, Cal, Gay, Bob, Ed];
obj = 3;
```

If we examine the seating arrangment in the solution, with 8 people, far apart means at greater than 8/3 or at least 3 seats apart.



We can see that the three dotted (pink) couples pairs are either adjacent or far apart. Of the rivalries 3 are adjacent or far apart shown in full (red), and 3 are not, shown dashed (red).

2 Instructions

Edit the provided `mzn` model files to solve the problems described above. Your implementations can be tested locally by using the *Run* icon in the MiniZinc IDE or by using,

```
minizinc ./modelname.mzn ./datafile.dzn
```

at the command line.