Discrete Optimisation Exercise: **Cryptarithm**

Problem Statement

A *cryptarithm* is a mathematical puzzle which requires determining the digit for each letter in an equation. The most famous cryptarithm is

		S	E	N	D
+		M	O	R	E
=	M	O	N	E	Y

That is, we need to determine which digit each of the letters represent so that

The rules of cryptarithms are:

1. Each letter represents a different digit
2. The first letter in each word cannot be 0 (otherwise it would not be a proper number)
3. The arithmetic equation must hold

This exercise will require modelling and solving cryptarithm problems. A MiniZinc model for the SEND + MORE = MONEY puzzle would be

```
var 1..9: S;
var 0..9: E;
var 0..9: N;
var 0..9: D;
var 1..9: M;
var 0..9: O;
var 0..9: R;
var 0..9: Y;

constraint                1000 * S + 100 * E + 10 * N + D
           +              1000 * M + 100 * O + 10 * R + E
           = 10000 * M + 1000 * O + 100 * N + 10 * E + Y;

include "all_different.mzn";
constraint all_different([S,E,N,D,M,O,R,Y]);
```

Part 1 - SEND+MORE=MONEY

Simply submit the provided `smm.mzn` model. This will test that your MiniZinc installation works correctly.

Part 2 - SNAKE+SNAKE=RATTLE

Complete the MiniZinc model `snake.mzn` which solves the problem

		S	N	A	K	E
+		S	N	A	K	E
=	R	A	T	T	L	E

You should determine at least one solution.

Part 3 - SEND+MOST=MONEY

Complete the MiniZinc model `most.mzn` to solve the problem

		S	E	N	D
+		M	O	S	T
=	M	O	N	E	Y

and maximise the value of the word MONEY. You will need to add an *objective function* of the form

```
solve maximize <exp>;
```

where you replace `<exp>` with the expression to be maximised.

Part 4 - Attemptation

Attemptation is a more general form of cryptarithm problem where we are given a partially filled codex, showing the set of possible values. Some values may be used more than once. For example

	P	L	E	A	S	E
+		S	O	L	V	E
+	P	U	Z	Z	L	E
=	Q	Q	Q	Q	Q	Q

with codex

0	1	2	3	5	6	7	7	8	9
						Z	Q		

means that Z and Q are both the digit 7, and the remaining letters take different values in the codex. Create the MiniZinc model `attempt.mzn` to solve this puzzle.

Instructions

Edit the provided `mzn` model files to solve the problems described above. The MiniZinc IDE and the online auto-grader will give you feedback on your solution.