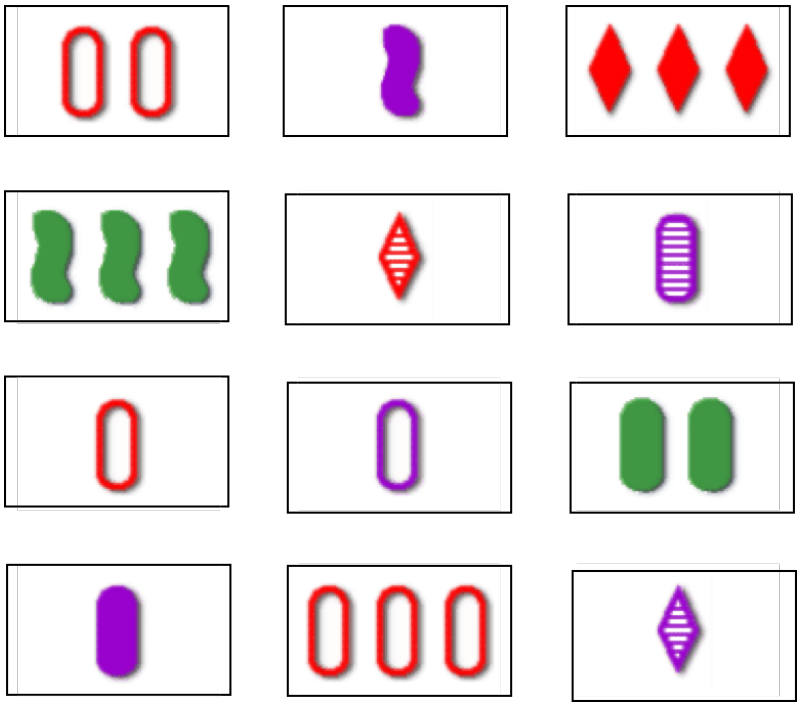
Exercise: The Game of Set

Consider the game of Set: http://en.wikipedia.org/wiki/Set_(game). The objective is to be the first to find a set of three cards in a 3x4 grid that satisfy all of the following conditions:

- They all have the same number, or they have three different numbers
- They all have the same symbol, or they have three different symbols.
- They all have the same shading, or they have three different shadings.
- They all have the same color, or they have three different colors.

For example, consider the following grid:



Some of the possible sets are:

Problem: Set up a Constraint Satisfaction Problem. Give a specification for variables, variable domains, and constraints.

One possible solution:

Variables: S1, S2, S3
Each variable is one of the slots in the set of three. The goal is thus to assign cards to the three slots.

Domains: D1 = D2 = D3 = {A1, A2, A3, B1, B2, B3, C1, C2, C3, D1, D2, D3}
Since we do not know which cards will be dealt, we designate the domain to be locations in the grid of cards (letters indicate column and numbers indicate row). Thus, each variable will be assigned a grid location.

To make this scheme work, we need a way to look at what card is actually in each cell of the grid. I will define a language that contains a number of functions that looks at different attributes of the cards in a given cell:

- number(cell) – returns the number {1, 2, 3} of items on the card in the given cell.
- shape(cell) – returns the shape {diamond, squiggle, oval} of the items on the card in the given cell.
- color(cell) – returns the color {red, green, blue} of the items on the card in the given cell.
- pattern(cell) – returns the pattern {solid, open, shaded} of the items on the card in the given cell

The language also consists of:

- Equality (=) which returns true if two symbols are the same.
- Inequality (<>) which returns true of two symbols are not the same.
- AND
- OR
- Parentheses

Constraints:

S1 <> S2
S1 <> S3
S2 <> S3

(pattern(s1) = pattern(s2) AND pattern(s1) = pattern(s3) AND pattern(s2) = pattern(s3))
OR
(pattern(s1)<>pattern(s2) AND pattern(s1)<>pattern(s3) AND pattern(s2)<>pattern(s3))

(color(s1) = color(s2) AND color(s1) = color(s3) AND color(s2) = color(s3)) OR
(color(s1)<>color(s2) AND color(s1)<>color(s3) AND color(s2)<>color(s3))

(shape(s1) = shape(s2) AND shape(s1) = shape(s3) AND shape(s2) = shape(s3)) OR
(shape(s1)<>shape(s2) AND shape(s1)<>shape(s3) AND shape(s2)<>shape(s3))

(number(s1) = number(s2) AND number(s1) = number(s3) AND number(s2) = number(s3)) OR
(number(s1)<>number(s2) AND number(s1)<>number(s3) AND number(s2)<>number(s3))