# Programmation par contraintes
## TP 1

Ouvrir un terminal et lancer l'interpréteur `gprolog`. Pour compiler le fichier `tp1.pl`, taper `[tp1].` après l'invite de commande `| ?-` au sein de l'interpréteur. On peut ensuite effectuer des requêtes en utilisant les prédicats de `tp1.pl`.

Ouvrir également un navigateur html pour pouvoir consulter la documentation, dont le sommaire est située (en général) dans : `/usr/share/doc/gprolog-doc/gprolog.html/index.html`
Un chapitre est dédié au solveur sur les domaines finis.

Un programme avec contraintes est toujours composé de trois parties :

1. déclarations de variables avec leur domaine (ensemble des valeurs possibles) : `fd_domain(Vars, Inf, Sup)`
2. pose des *contraintes*, les relations que doivent respecter les variables entre elles ;
3. recherche de solutions par énumération des valeurs possibles pour chaque variable (éventuellement combinée avec une optimisation) : e.g. `fd_labeling(Vars)`

### Arithmétique cryptée

Résoudre le problème suivant, dans lequel tous les chiffres doivent être différents, et D, G et R doivent être différents de 0 :

$$\begin{array}{cccccccc} & D & O & N & A & L & D \\ + & G & E & R & A & L & D \\ \hline = & R & O & B & E & R & T \end{array}$$

### Nombre de Hardy-Ramanujan [1]

Retrouver le numéro de taxi de Hardy avec un programme en contraintes. C'est le plus petit entier qui soit somme de deux cubes de deux manières différentes :

$$n = a^3 + b^3 = c^3 + d^3 \qquad (a, b) \neq (c, d) \quad (a, b) \neq (d, c)$$

On supposera que $n$ est inférieur à 1 000 000.

### Monnaie

1. Quelles sont toutes les possibilités de rendre la monnaie sur un billet de 20 € pour un achat de 17,29 € en utilisant au maximum trois fois la même pièce ?
2. Écrire le prédicat `sum(L, S)` qui construit l'expression `S` égale à $X_1 + X_2 + ... + X_n + 0$ si $L = [X_1, X_2, ..., X_n]$, puis définir une nouvelle variable `Cost` contrainte à être égale à cette expression.
3. Minimiser le nombre de pièces rendues en utilisant le prédicat `fd_minimize(Goal, Cost)`.

### Chargement de fret

On veut optimiser la cargaison d'un avion de transport de fret. On dispose d'un ensemble de marchandises à expédier et on doit en choisir un sous-ensemble dont le volume total ne dépasse pas la capacité [2] de l'avion et dont l'utilité est maximale. Une instance de ce problème est spécifiée dans le fichier `fret.pl`. Elle contient :

— la liste des volumes de chaque marchandise ;

1. `mathworld.wolfram.com/Hardy-RamanujanNumber.html`
2. Dans la réalité, on ajoute de nombreuses contraintes telles que le poids, les dimensions, la compatibilité des marchandises entre elles etc.

— une liste de de la même taille contenant leur valeur ;
— la capacité maximale de l'appareil.

Utilisez-les dans votre fichier source, puis :

1. Modéliser et résoudre ce problème d'optimisation. On utilisera une variable à domaine booléen pour chaque objet, instanciée à 1 si l'objet est sélectionné et à 0 sinon. Le coût d'une solution est la somme des valeurs des marchandises choisies.
2. Modifier la stratégie de recherche pour changer l'ordre d'essai des **valeurs** (`value_method`) et améliorer l'efficacité de la recherche. Commenter.