

## S07 - Prolog (Introduction)

### Instructions

- Date butoir : Jeudi prochain à 05h00
- Travail à rendre : un fichier `.zip` contenant tout votre travail pour la série
- N'oubliez pas de mettre votre nom au début du fichier

### SWI-Prolog

Installez l'interpréteur SWI-Prolog sur votre ordinateur.

- Le manuel est accessible depuis le menu Help ou en ligne :  
`http://www.swi-prolog.org/pldoc/doc\_for?object=manual`
- Pour charger les clauses d'un fichier source dans l'interpréteur, utilisez le Menu File -> Consult (si vous utilisez le GUI) ou le prédicat prédéfini `consult/1`.
- Une ligne de commentaire commence par `%`. Les commentaires sur plusieurs lignes se notent entre `/*` et `*/`
- Pour éviter les warning “ Singleton variables ” de l'interpréteur, renommer les variables concernées par `_`
- Quand vous interrogez l'interpréteur, il commence par afficher la première réponse puis il attend une action. Tapez `;` ou `space` pour demander la prochaine réponse ou tapez `enter` pour arrêter la recherche de solution
- Une clause telle que `:- [file1, file2, ...]` placée au début de votre texte source indique à l'interpréteur de charger également les clauses des fichiers `file1.pl`, `file2.pl`, etc.

Vous pouvez aussi utiliser SWISH pour faire certains exercices.

## Exercice 1 : Prédicats prédéfinis

Recherchez dans la documentation les prédicats suivants, testez-les et expliquez-les en quelques lignes.

1. =
2. \=
3. ==
4. \==
5. =\=
6. =:=
7. is

## Exercice 2 : Relations familiales

Développez les prédicats qui suivent en utilisant les prédicats `parent/2`, `male/1` et `female/1` du fichier `s07_annexe.pl`.

- `child(X,Y)` : vrai si X est un enfant de Y
- `father(X,Y)` : vrai si X est le père de Y
- `mother(X,Y)` : vrai si X est la mère de Y
- `grandparent(X,Y)` : vrai si X est un des grand-parents de Y
- `grandfather(X,Y)` : vrai si X est un grand-père de Y
- `aunt(X,Y)` : vrai si X est une tante de Y
- `ancestor(X,Y)` : vrai si X est un ascendant de Y. Un ascendant est soit un des deux parents, un des grands-parents, un des arrière-grands-parents, etc.
  - Faites 2 variantes du prédicat `ancestor` en échangeant l'ordre des sous-buts.
  - Observer les différences de résolution pour la question ouverte : `ancestor(X,Y)`.

Expliquez votre raisonnement pour chaque prédicat en commentaire (%).

## Exercice 3 : Tours de Hanoï

1. Résolvez le problème de Hanoï avec 3 disques et 3 piquets (sur papier). Dessinez toutes les étapes à suivre pour bouger les 3 disques du piquet de gauche au piquet de droite.
2. Implémentez le prédicat `hanoi(N,D,I,G)` où `N` est le nombre de disques à déplacer, `D` est le piquet de départ, `I` le piquet intermédiaire et `G` est le piquet d'arrivée. Utilisez le prédicat prédéfini `format(Format,Arguments)` pour afficher les déplacements de disques.

Il permet d'afficher sur la sortie standard du texte formaté de manière similaire à `printf` en C (cf. manuel en ligne). Il s'utilise comme suit :

`format("dep de ~q vers ~q \n",[p1,p3])` est évalué à vrai et affiche sur la sortie standard : `dep de p1 vers p3`.

- Exemple : la question `hanoi(3,p1,p2,p3)` affichera :

```
dep de d vers g
dep de d vers i
dep de g vers i
dep de d vers g
dep de i vers d
dep de i vers g
dep de d vers g
```

3. Dessinez l'exécution complète de la fonction `hanoi` pour 3 piquets : `hanoi(3,p1,p2,p3)`.

Recommandation : dessinez l'exécution sous forme d'arbre, avec les prédicats et arguments comme nœuds. Notez aussi les exécutions des `format/2`.