4-IF-ALIA-PR - Prolog - Liste de notions et compétences

Basiques	Faits règles et requêtes	Acquis
Définitions	Base, faits, requêtes, interpréteur et solutions	
Syntaxe	Termes, variables, atomes, nombres	
Logique	Retrouver les notions de logique déjà connues :	
	Modus ponens, connecteurs logiques (implication, et, ou)	
	Quantificateur universel, existentiel, non déterminisme, …..	
Principe	Utiliser des variables nommées et anonymes	
	Utiliser un prédicat de différentes manières/instanciations param	
Technique	Prise en main de l'interpréteur swi-prolog	
	Charger, écrire une base simple et l'interroger	
	Utiliser listing/0. consult/1. halt/0. apropos/1. help/0.	
	Variables anonymes, prédicats à effet de bord/sans effet de bord	

Intermédiaires	Unification, recherche de preuve et récursivité	Acquis
Principe de base	Ecrire un prédicat récursif (points d'arrêt et règles de récursion)	
Mécanisme	Comprendre le mécanisme de résolution	
	Construire l'arbre de recherche	
	Comprendre substitution/unification et l'algorithme de Robinson	
Technique	Analyser une trace d'exécution. trace/0. (*fail, exit, redo)*	
Principe de base	Ecrire, évaluer, comparer des expressions arithmériques	
Structure	Manipuler des listes	

Avancées	Notions avancées de Prolog	Acquis
Technique	Savoir utilise la coupure pour la recherche de première solution,	
	exprimer le déterminisme, les tests "si", la négation	
	Comprendre la différence/risques entre coupure verte et rouge	
Principe	Comprendre la négation par l'échec	
Structure	Utiliser des arbres, graphes, parcours en profondeur et largeur	
Principe	Mémoisation : programme dynamique et non monotonicité	
Technique	Ecrire de nouveaux opérateurs (op/3)	
Technique	Utiliser setof/3. bagof/3. findall/3.	

Autres	Notions utiles pour le projet	Acquis
Développement	XPCE : librairie native de SWI-Prolog pour IHM	
Développement	Utiliser des méta-prédicats, gérer les entrées/sorties	
Développement	Savoir gérer les exceptions throw/1. catch/3.	
Développement	Utiliser des tests unitaires. Profiling (profile/1)	
Développement	Organiser son code Prolog en modules	

Extra	Pour aller plus loin …	Acquis
Accumulateurs	Optimiser la récursivité (recursif en queue ou non)	
Occur check	Comprendre le problème du test d'occurrence (occur-check)	
Contraintes	Solveur de contraintes (générer et tester, retour-arrière simple)	