TP1 de prolog 2019

Utilisation de Swi-Prolog : http://www.swi-prolog.org/

_ Pour lancer l'interprète de Swi-Prolog, tapez swipl dans une console. Le signal d'invite : ?- attend que vous saisissiez un but que le moteur de résolution Prolog tentera de satisfaire.
_ Pour quitter Swi-Prolog, tapez halt. (avec le « . »!).
_ Dans l'interprète, vous pouvez obtenir une aide sur tout prédicat prédéfini à l'aide du prédicat help. Par exemple tester "help(halt).".

1)

Faites fonctionner les programmes ci-dessous pour vous familiariser avec l'environnement de Prolog

A)

```
pere(moha,ali).
pere(ahmed,moha).
grandpere(X,Y) :- pere(X,Z), pere(Z,Y).
```

Vous tapez ce programme dans un éditeur de texte simple et sauvegardez dans un fichier « prprog.pl » (par exemple, ou tout autre nom avec le suffixe .pl).
Lancer SWI-Prolog puis sous l'interpréteur et tapez :

```
consult('c:/prprog.pl').
pere(X,Y).
grandpere(X,Y).
grandpere(ahmed,X).
halt.
```

Pour avoir toutes les réponses possibles, taper un **;** après chaque réponse obtenue.

B) rajouter dans le fichier les prédicats suivants :

```
habite(moha,rabat).
habite(hans,munich).
habite(juan,madrid).
capitale(rabat).
capitale(madrid).
Habite_capitale(Qui) :- habite(Qui,Qqpart),capitale(Qqpart).
```

Lancer SWI-Prolog puis sous l'interpréteur et tapez :

```
consult('c:/prprog.pl').
habite(moha,rabat).
habite(moha,Qqpart).
Habite_capitale(Qui).
habite(Qui,madrid).
habite(_,Qqpart).
habite(Qui,Qqpart),capitale(Qqpart).
```

Pour mieux comprendre ce que fait Prolog (et par la suite, déboguer votre programme), vous pouvez faire précéder une requête du prédicat **trace**. Par exemple :

```
trace, Habite_capitale(Qui).
```

Prolog fait alors du pas à pas. Vous passez au pas suivant en tapant Entrée. Quatre types d'informations sont affichés :
Call : c'est l'appel du prédicat

Exit : Retour avec succès
Fail : Retour avec échec
Redo : Retour en arrière, Prolog essaie une autre branche.
Si vous ne souhaitez pas faire du pas à pas sur un appel particulier de prédicat (parce que vous savez qu'il fonctionne bien), tapez la touche 's' au lieu de Entrée.

2)

Récupérer le fichier **hotel.pl** qui contient des informations sur les chambres des hôtels d'une ville.
Les arguments du prédicat hotel/5 sont les suivants : le nom de l'hôtel, le numéro de la chambre, son prix, la vue, et la situation de la chambre (libre ou réservée).
Par exemple :

```
chambre(hotelDeLaPlage, 1, 460, mer, libre).
chambre(hotelDeLaPlage, 2, 460, mer, libre).
chambre(hotelDeLaPlage, 3, 510, rue, reservee).
chambre(hotelDeLaPlage, 4, 410, rue, libre).
```

a) Par une simple requête, afficher :
¬∧⊏l'ensemble des chambres libres.
¬∧⊏L'ensemble des chambres libres de moins de 500Dh.

b) Ecrire au début du fichier hotel.pl un prédicat **hotelPasCher/2** qui détermine un hôtel ayant au moins une chambre de moins de 500Dh étant donnée une vue. Par exemple :

```
?- hotelPasCher(mer,N).
N= hotelDeLaPlage
```

c) Ajouter au fichier hotel.pl quatre faits qui indiquent que l'hôtel de la Plage est à 100m de la mer, l'hôtel du Nord est à 2km, l'hôtel de la Gare est à 3km et l'hôtel des Dunes est à 300m.

d) Ajouter un prédicat **chambreSympa/2** qui détermine les numéros de chambre (et les noms d'hôtels) des hôtels à moins de 500m de la mer et ayant une chambre libre avec vue sur la mer.

3) Faire le programme et tester l'exemple suivant vu en cours

Ecrire le prédicat **fibonacci/2** qui détermine le Nème élément de la suite de Fibonacci.

fb(0)=1
fb(1)=1
fb(n)=fb(n-2)+fb(n-1). Pour n> 1