Web Sémantique (Pratique)

L'objectif du sujet proposé est de développer une bibliothèque Python mettant en oeuvre les éléments vus en cours; ceci en utilisant la méthode KISS (Keep It Simple, Stupid).

I. Données initiales

1. Pour cela, commencer par définir un fichier data.csv contenant les données suivantes:

```
bob,studies,ensisa
ensisa,located,mulhouse
```

2. Définir ensuite un fichier sem.py puis répondre à chacune des questions suivantes à l'aide de sous-programmes.

3. Définir ainsi un programme (load) pour lire le contenu le fichier csv.

4. Faire ensuite un sous-programme qui découpe (split) les données obtenues suivant '0 (et retourne donc un tableau) puis pour chaque élément suivant ','. Le résultat est donc un tableau de triplets.

5. Proposer aussi 2 programmes pour reconstruire les données initiales avec: interspace(s) qui fusionne un tableau en intercalant le symbole s entre chaque valeurs, et write(f,d) qui écrit les données d dans le fichier f.

II. Opérateurs ensemblistes

Le web sémantique travaille sur des ensembles et non des tableaux, et il faut disposer des opérateurs qui caractérisent les ensembles.

1. Définir un programme (contains) permettant de savoir si une valeur est dans un tableau/ensemble.
2. En considérant 2 ensembles s1 et s2, proposer un programme pour: trouver les éléments commun entre s1 et s2 (intersection), trouver les éléments de s1 qui ne sont pas dans s2 (différence), et faire l'union de s1 et s2.
3. (option). Il est possible d'utiliser ce qui précède pour traiter du langage naturel et non du csv: des phrases sont séparer par '.', des mots par ' ' et il faut enlever les mots inutiles (différence de 2 ensembles). Par exemple, "bob etudie à l ensisa." => ["bob","etudie","à","l","ensisa"] - ["à","le","la","les","l"]; ce qui correspond à la première ligne du fichier de données.

III. Compréhensions

1. Définir un opérateur map(f) permettant d'appliquer une fonction f à tous les éléments d'un ensemble.
2. Montrer alors comment ajouter le préfixe "http://sem.org#" à toutes les données chargées précédemment.
3. Proposer aussi un programme permettant de retirer les préfixes.
4. En définissant une fonction ret(x)=[x] (qui signifie return) et une constante fail=[], proposer un programme filter(p,t) qui pour chaque valeur x du tableau t retourne x si p(x) est vrai, sinon échoue.

Par exemple, si l'on recherche des informations sur bob, alors p(x)=x[0]=="bob".

5. Définir un programme permettant de concaténer un tableau de tableaux.
6. On définit l'opérateur bind(t,f) = concat(map(f,t)) et on notera que celui-ci est prédéfini en Python et est associé à une notation particulière:

   bind(t,lambda x:y) == [y for x in t]

   De même, bind(filter(p,t),lambda x:y) == [y for x in t if p(x)]

   Comment traduire une requête sparql en utilisant cette notation ? Par exemple, comment trouver toutes les informations sur bob ?
7. Cette notation peut être imbriquée avec par exemple [(x,y) for x in t for y in t] Sachant cela, comment trouver dans quelle ville étudie bob ? Qui étudie à mulhouse ?

IV. Inférence

Les résultats des requêtes précédentes peuvent être formatées sous forme de triplets. Par exemple, à la requête "qui étude ?", la réponse [bob] peut être transformée en [bob,rdf:type,Personne]

Ainsi, la réponse peut être ajoutée à la base de données initiale et on a une forme d'inférence.

1. Montrer alors comment traduire les inférences suivante: [?x,rdf:type,?y] => [?y,rdf:type,owl:Class] [?x,?r,?y],[?x,rdf:type,?c] => [?r,rdf:domain,?c] On ajoutera aussi les inférences pour les NamedIndividual et ObjectProperty.
2. Une règle d'inférence peut être définie par une fonction prenant des données en entrée (db) et retournant de nouvelles données.

   Ces fonctions peuvent être placées dans un tableau et exécutées lorsqu'on le souhaite. Définir ainsi un programme eval(rules,db) qui retourne toutes les informations à ajouter à la base si l'on applique les règles.
3. Les informations inférées peuvent activer d'autres règles qui vont générer des informations qui vont activer d'autres règles qui ...

   Il y a donc une forme de récursion qui doit s'appliquer tant que toutes les données inférées ne sont pas dans la base. Ceci s'appelle l'algorithme du point fixe et comment le mettre en oeuvre en Python ?
4. Combien de données ont été ajoutées à la base initiale ?

V. Génération de pages html

L'intérêt d'utiliser un langage de programmation généraliste comme Python est de pouvoir intéragir avec d'autres domaine de l'informatique.

1. Ainsi, une page web (code html) est simplement un texte, et il est possible de définir des fonctions (templates) qui génèrent ce texte. Proposer alors un programme qui génère une page web affichant les informations disponibles sur une personne.
2. De la même façon, on peut afficher les valeurs d'une relation. Proposer un programme qui affiche les valeurs de la relation studies.