

Séance 8: RÉSEAU SOCIAL

Université Paris-Diderot

Objectifs:

- Manipulation des listes de listes d'entiers
- Boucles imbriquées

Nous voulons dans ce TP modéliser un réseau social dans lequel plusieurs utilisateurs ayant les numéros $0, 1, 2 \dots n$ sont inscrits. Pour ce faire, nous allons représenter le réseau par une liste de listes d'entiers R où chaque case $R[i][j]$ vaut :

- 1 si l'utilisateur i est ami avec l'utilisateur j
- 0 sinon.

Par exemple, un réseau R dans lequel il y a trois utilisateurs peut valoir la liste de listes suivante :

```
1 [[0, 1, 0], [1, 0, 0], [0, 0, 0]]
```

Où l'utilisateur 0 est ami avec l'utilisateur 1 (car $R[0][1]=1$ et $R[1][0]=1$) et où l'utilisateur 2 n'est ami avec personne.

Notez qu'un utilisateur n'est jamais ami avec lui-même et que pour chaque couple d'utilisateurs i et j : si i est l'ami de j alors j est aussi l'ami de i .

Exercice 1 (Les fonctionnalités de base, * – **)

1. Écrivez une fonction `CreateGraph(n)` qui retourne un réseau sous la forme de liste de listes comme vu ci-dessus. Il y aura dans cette liste un nombre n d'utilisateurs et l'amitié entre deux utilisateurs est aléatoire. Rappelez-vous la fonction `randrange(a,b)` des TP précédents.
2. Écrivez une fonction `friends_nbr(R,a)` qui, pour un réseau R et un utilisateur a , retourne le nombre d'amis que possède a dans le réseau.
3. Écrivez une fonction `friends(R,a)` qui, pour un réseau R et un utilisateur a , retourne une liste contenant tous les amis de a dans le réseau et une liste vide s'il n'en possède aucun.
4. Écrivez une fonction `popular(R)` qui retourne la liste contenant les numéros des utilisateurs les plus populaires du réseau. Un utilisateur est populaire s'il possède le plus grand nombre d'amis dans le réseau.
5. Écrivez une fonction `common_friends_nbr(R,a,b)` qui retourne pour un réseau R et deux utilisateurs a et b le nombre d'amis qu'ils ont en commun.
6. Écrivez une fonction `common_friends(R,a,b)` qui retourne pour un réseau R et deux utilisateurs a et b une liste contenant les amis qu'ils ont en commun.
7. Écrivez une fonction `add_user(R,l)` qui ajoute un utilisateur dans le réseau R et le lie d'amitié avec les utilisateurs qui sont dans la liste l . Dans le nouveau réseau on aura donc un utilisateur en plus.

8. Considérons une liste `noms` qui contient des chaînes de caractères. Pour chaque indice i la case `noms[i]` contient le nom de l'utilisateur i . Par exemple, si `noms` vaut :

```
1 noms = ['Evan Spiegel', 'Mark Zuckerberg', 'Jack Dorsey']
```

Alors l'utilisateur numéro 1 est `noms[1]` et donc `'Mark Zuckerberg'`. Quelle est la longueur de cette liste? Modifiez alors la fonction `popular` pour qu'elle retourne les noms des utilisateurs les plus populaires au lieu de leurs numéros. Attention il faudra bien sûr changer aussi les arguments de la fonction.

□

Exercice 2 (Évènement, ***)

Dans cette partie on souhaite utiliser notre réseau social afin d'améliorer l'alchimie de votre pendaison de crémaillère à venir.

1. Pour cela vous devez programmer une fonctionnalité `invite(R,a)` qui, pour un utilisateur a , retourne tous ses amis sur le réseau mais aussi les amis de ses amis. Bien entendu cette fonction ne doit pas retourner a .
2. Certains sont stricts et veulent que chaque invité connaisse au moins deux amis distincts de a .
Pour vous aider dans cette tâche on vous demandera de calculer la liste de listes P où chaque case de P est obtenue de R de la manière suivante :

$$P[i][j] = \sum_{k=0}^{n-1} R[i][k] * R[k][j]$$

Où n est le nombre d'utilisateurs du réseau. Une fois P calculée, pour chaque couple d'utilisateurs i,j si $P[i][j] \geq 2$ alors i connaît deux amis distincts de j .

Écrivez la fonction `strict_invite(R,a)` qui retourne la liste d'invités selon les conditions strictes.

3. Modifiez les fonctions ainsi que la liste de leurs paramètres pour que les valeurs retournées soient les noms des utilisateurs au lieu de leurs numéros.

□