Programmation
Licence 1 UPEC 2022/2023
Travaux Machine 9 : Encore fonctions

Exercices

1 Scrutin Préférentiel

On va implémenter le scrutin préférentiel à la Québecoise qui est expliqué dans cette vidéo (vous devez bien regarder cette vidéo avant d'attaquer les exos) :

`https://www.youtube.com/watch?v=Igm7a9_QtY4`

Toutes les fonctions sont à tester aussi sur `moodle.caseine.org`.

Les choix de vote des électeurs seront représentés par un tableau à deux dimensions d'entiers : `votes[i][j]` dénotera la $j+1$-ème préférence de l'électeur $i+1$ (car on commence toujours par 0). Par exemple, si `votes[3][0]` est un nombre $n > 0$, cela dénote que la première préférence de l'électeur 4 est pour le candidat numéro n. Si `votes[i][j]` est 0, cela dénote le fait que l'électeur n'a pas donné toutes ses préférences. Le nombre de préférences autorisées n'est pas limité à 3 comme dans la video. Notre logiciel devra marcher pour n'importe quel nombre (raisonnable) de préférences.

Exercice 1 On va d'abord tester qu'un tableau de préférences est correct.

1. Écrire une fonction `boolean candidatsEx(int[][] votes, int nbcandidats)`, qui vérifie qu'aucune case de tableau ne contient un candidat inexistant ;
2. Écrire une fonction `boolean uniquePref(int[][] votes, int candidat)`, qui vérifie que chaque électeur a donné au plus une préférence pour le candidat numéro *candidat*
3. Écrire une fonction `boolean prefCorrect(int[][] votes)`, qui vérifie que chaque fois que la case `votes[i][j]` contient un 0 alors la case `votes[i][j+1]` contient aussi un 0
4. Écrire une fonction `boolean verifieVote(int[][] votes, int nbcandidats)`, qui combine les propriétés décrites ci-dessus, et qui vérifie si un tableau de préférences est correct.

Exercice 2 On va maintenant implémenter la procédure d'élection.

1. On commencera pour compter les premières préférences. Écrire une fonction `int compteVote(int[][] votes, int candidat)`, qui compte les premières préférences du candidat *candidat*.
2. Écrire une fonction `int gagnant(int[][] votes, int nbcandidats)`, qui renvoie le candidat gagnant, s'il y en a un, et qui renvoie 0 sinon.
3. On va maintenant faire des manipulations sur le tableau de votes. Écrire une fonction `void efface(int[][] votes, int candidat, int electeur)`, qui, pour l'électeur numéro *electeur*, regarde s'il a donné une préférence pour le candidat `candidat`. Si c'est le cas, il déplace toutes les préférences suivantes de cet électeur vers le haut. Par exemple si l'electeur avait donné sa troisième prérérence au candidat, sa quatrième devient maintenant la troisième, la cinquième la quatrième et ainsi de suite). Finalement la fonction met à 0 la dernière préference de l'électeur.
4. Écrire une fonction `void efface(int[][] votes, int candidat)`, qui applique la fonction précédente pour chaque électeur.

5. Écrire une fonction `void effaceZero(int[][] votes, int nbcandidats)`, qui applique la fonction précédente à tous les candidats qui ont zéro premières préferences.
6. Écrire une fonction `int perdant(int[][] votes, int nbcandidats)` qui renvoie un candidat qui a reçu le plus petit nombre de première préferences, mais pas 0 premières préferences (pour simplifier, s'il y en a plusieurs, on en renvoie un seul, même si ce n'est pas comme ça qu'on fait en pratique).
7. Écrire une fonction `int election(int[][] votes, int nbcandidats)` qui combine les fonctions décrites ci-dessus, pour élire le président. D'abord on efface tous les zéros, et ensuite, un par un, tous les candidats perdants jusqu'à ce qu'il y ait un gagnant.