Université de Bourgogne
janvier 2005

Examen informatique I1a

Durée 2h, tous documents autorisés. Le barème est donné à titre indicatif

Exercice 1. (5 pts. Evaluation d'une fonction mathématique)
1. Ecrire le programme qui permet de saisir deux entiers `x` et `n` puis de calculer :
$x^0 + x^1 + x^2 \ldots x^n$.
2. Ecrire le programme qui permet de saisir deux entiers `x` et `n` puis de calculer :
$a_0x^0 + a_1x^1 + a_2x^2 \ldots a_nx^n$. Les coefficients a_i doivent être saisis par l'utilisateur au fur et à mesure du calcul.

Exercice 2. (4 pts. Vérification d'une liste triée) Soit un tableau d'entiers `T` à une dimension. Ecrire un morceau de programe Java permettant de vérifier que les valeurs stockées dans `T` sont en ordre croissant (c'est à dire que pour tout `i` compris entre `0` et `T.length-2`, `T[i+1] >= T[i]`). Ce morceau de programme devra afficher une seule fois le message "oui" si les valeurs de `T` sont en ordre croissant et devra afficher une seule fois le message "non" dans le cas contraire.

Exercice 3. (5 pts. Détection d'une phrase) On souhaite faire un programme permettant de reconnaître si une chaîne de caractère donnée par l'utilisateur est une phrase correcte de la langue française. On dira qu'une chaîne de caractères est une phrase de langue française si elle a les caractéristiques suivantes :

- Elle possède trois mots : un sujet, un verbe et un complément.
- Deux mots sont séparés avec un seul espace.
- Un sujet est un mot de l'ensemble {*je*, *tu*, *il*, *nous*, *vous*, *ils*}.
- Un verbe ou un complément est une succession de lettres (sans espaces).

Questions :
1. Faire un programme permettant de vérifier si la chaîne de caractères est une phrase de la langue française.
2. Faire un programme qui permet de vérifier si le verbe a la bonne conjugaison. Pour que ce soit le cas, il est nécessaire d'avoir les conditions suivante :

Si le sujet est *je* ou *il*, le verbe doit se terminer par la lettre *e*.
Si le sujet est *tu*, le verbe doit se terminer par les lettres *es*.
Si le sujet est *nous*, le verbe doit se terminer par les lettres *ons*.
Si le sujet est *vous*, le verbe doit se terminer par les lettres *ez*.
Si le sujet est *ils*, le verbe doit se terminer par les lettres *ent*.
Exemple{*je parle*, *tu parles*, *il parle*, *nous parlons*, *vous parlez*, *ils parlent*}

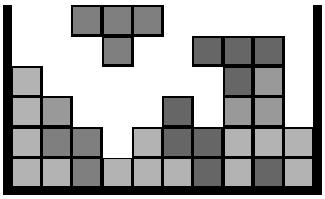
Exercice 4 (8 pts.Tetris) Le jeu Tetris se déroule dans un puits (à seulement deux dimensions). Des pièces, formées de quatre petits carrés, tombent lentement dans le puits, sous contrôle du joueur, puis finissent par se poser au fond, ou sur les autres pièces déjà présentes. Quand une rangée horizontale est complètement remplie, elle disparaît, et les pièces qui étaient au dessus se décalent d'un cran vers le bas.

Le but de cet exercice est de programmer quelques fonctions utiles pour ce jeu. On représente le terrain de jeu par un tableau d'entiers:

```
int tetris[][] = new int[20][10]
```

Les lignes sont numérotées **de bas en haut** de 0 à 19, les colonnes de gauche à droite de 0 à 9. Chaque case du tableau contient:

- 0 si la case est vide;
- un nombre entre 1 et 9 si la case contient une pièce posée (la valeur indique la couleur de la pièce, pour des raisons uniquement décoratives);
- un nombre entre 11 et 19 si la case contient une pièce en train de tomber (avec la même convention pour les couleurs).



5	0	0	16	16	16	0	0	0	0	0
4	0	0	0	16	0	0	4	4	4	0
3	1	0	0	0	0	0	0	4	5	0
2	1	5	0	0	0	7	0	5	5	0
1	1	3	3	0	2	7	7	1	1	2
0	1	1	3	2	2	2	7	1	7	2
	0	1	2	3	4	5	6	7	8	9

Questions

1. Écrire une fonction qui teste si la pièce en train de tomber peut encore descendre ou pas. Cette fonction prend en argument un tableau d'entiers de 20 sur 10, et renvoit comme résultat:

- `true` si toutes les cases du tableau qui contiennent un nombre entre 11 et 19 sont immédiatement au dessus d'une case qui contient soit elle aussi un nombre entre 11 et 19, soit 0;
- `false` sinon.

Attention au fond du puits!

La fonction doit s'écrire:

```
public static boolean peut_tomber(int [][]tetris) { ... }
```

2. Écrire une fonction qui fait descendre d'un cran la pièce en train de tomber. Cette fonction prend en argument un tableau d'entiers de 20 sur 10, et modifie le tableau ainsi: toutes les cases qui contiennent nombre entre 11 et 19 sont copiées dans les cases immédiatement en dessous, et leur contenu est remplacé par 0.

On peut supposer, en écrivant cette fonction, qu'on est sûr que la pièce peut descendre d'un cran. Faites attention au sens dans lequel vous faites le déplacement: il ne faut pas effacer une case avant de l'avoir traitée.

La fonction doit s'écrire:

```
public static void descend(int [][]tetris) { ... }
```

3. Écrire une fonction qui pose la pièce tombante et supprime les lignes complètes. Cette fonction prend en argument un tableau d'entiers de 20 sur 10, et modifie le tableau ainsi:

- toutes les cases du tableau qui contiennent un nombre de 11 à 19 sont remplacées par le nombre de 1 à 9 correspondant;
- si une ligne entière du tableau ne contient aucun 0, toutes les lignes situées au dessus sont décalées d'un cran vers le bas, et la ligne tout en haut remplie de 0.

Attention à ce qui se passe si plusieurs lignes doivent être supprimées.

La fonction doit s'écrire:

```
public static void pose_et_supprime(int [][]tetris) { ... }
```