Sudoku génétique.

Bruno Malenfant

9 septembre 2012

- Peut être fait seul(e) ou en équipe de deux (2).
- Date de remise : 27 septembre 2012 avant 23:55.

1 Introduction

Pour le dernier tp vous allez construire un solveur de sudoku utilisant un algorithme génétique. Votre solveur va lire les informations sur le problème dans un fichier choisi par l'utilisateur. Ensuite un algorithme génétique sera utilisé pour trouver des solutions. Votre logiciel doit afficher la meilleur solution par génération.

2 Description

Lors du démarrage l'utilisateur entre le nom du fichier d'entrées sur la ligne de commande unix (à côté du nom de l'exécutable). Ensuite votre programme lit l'info sur le problème dans ce fichier.

2.1 Entrées

Le nom du fichier d'entrées doit être écrit sur la ligne de commande. Par exemple, si votre exécutable porte le nom `tp2` et les entrées sont dans le fichier `sudoku1` alors la commande de lancement sur unix sera :

```
./tp2 sudoku1
```

Le fichier contient la description du sudoku à résoudre. il y aura 9 lignes de 9 chiffres dans ce fichier. Chaque chiffres sont séparés par un ou des espaces. Ce sont des valeurs de 0 à 9 où la valeur 0 indique une case sans valeur. Voici un exemple :

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

Ce fichier représente le sudoku suivant :

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

Pour tout problème rencontré vous affichez un message d'erreur et l'application se termine (`exit(-1)`). Problème posible : fichier inexistant ou illisible, valeur illégale dans la description, pas assez de valeur dans la description.

2.2 Analyse

Ensuite votre logiciel doit appliquer un algorithme génétique pour résoudre le problème. Un algorithme génétique contruit une populations de base et ensuite génère des populations subséquentes en utilisant des mutations. À chaque génération une sous populations est choisi pour passer à la génération suivante.

2.2.1 Population de base

La population de base est composée de 100 solutions possibles. Cette génération (génération zéro) est construite avec des valeurs au hazards. Une valeur aléatoire est placée dans chaque case vide. La librairie `stdlib.h` contient un générateur aléatoire. La fonction `int rand( void )` retourne une valeur entière aléatoire entre 0 et `MAX_RAND`. Donc, pour avoir une valeur entre 1 et 9 nous pouvons écrire :

```
int x = ( rand() % 9 ) + 1;
```

Par exemple :

1	4	2	1	2	4	**8**	**1**	2
2	**4**	2	**8**	2	**6**	2	1	6
3	5	**5**	8	2	**2**	3	**9**	6
8	**7**	5	**4**	2	**1**	**9**	2	6
5	6	4	8	**7**	4	1	2	7
6	3	**4**	**5**	2	**9**	2	**8**	**3**
7	**6**	6	**1**	2	7	**4**	3	9
8	5	6	**6**	1	**5**	1	**2**	8
7	**2**	**8**	9	5	5	3	9	7

2.2.2 Construction de la population suivante

Pour construire la population suivante nous engendrons des sudokus à partir de la population précédante. Nous avons deux (2) techniques de génération pour notre système. Nous allons construire une génération de 1000 solutions. à chaque fois (pour chacun des 1000) nous choisissons une technique au hazard. 70 % du temps se sera un couplage et 30 % du temps ce sera une mutation. Il est possible de générer une valeur entre 0 et 1 de la façon suivante :

```
double x = (double) rand() / (double) MAX_RAND;
```

1. Couplage : Cette technique de génération consiste à choisir au hazard deux (2) solutions (s_1 et s_2) parmis la génération précédante. ensuite un point de coupure est choisi au hazard, cela correspond à une séparation entre deux cases dans le sudoku. Un nouveau sudoku est construit en utilisans les valeurs avant le point de coupure dans le sudoku s_1 et après le point de coupure dans le sudoku s_2.

   Par exemple si s_1 =

1	4	2	1	2	4	**8**	**1**	2
2	**4**	2	**8**	2	**6**	2	1	6
3	5	**5**	8	2	**2**	3	**9**	6
8	**7**	5	**4**	2	**1**	**9**	2	6
5	6	4	8	**7**	4	1	2	7
6	3	**4**	**5**	2	**9**	2	**8**	**3**
7	**6**	6	**1**	2	7	**4**	3	9
8	5	6	**6**	1	**5**	1	**2**	8
7	**2**	**8**	9	5	5	3	9	7

   et s_2 =

4	4	3	4	1	2	**8**	**1**	3
5	**4**	2	**8**	5	**6**	3	3	5
7	5	**5**	3	4	**2**	5	**9**	4
8	**7**	4	**4**	7	**1**	**9**	5	6
2	2	5	1	**7**	5	7	2	1
3	3	**4**	**5**	3	**9**	4	**8**	**3**
7	**6**	7	**1**	4	8	**4**	3	2
5	1	8	**6**	1	**5**	2	**2**	1
4	**2**	**8**	9	5	3	4	5	5

et nous avons trouvez un point de coupure aléatoirement à la quatrième ligne et entre la sixième et septième colonne nous obtiendrons :

1	4	2	1	2	4	**8**	**1**	2
2	**4**	2	**8**	2	**6**	2	1	6
3	5	**5**	8	2	**2**	3	**9**	6
8	**7**	5	**4**	2	**1**	**9**	5	6
2	2	5	1	**7**	5	7	2	1
3	3	**4**	**5**	3	**9**	4	**8**	**3**
7	**6**	7	**1**	4	8	**4**	3	2
5	1	8	**6**	1	**5**	2	**2**	1
4	**2**	**8**	9	5	3	4	5	5

Cette solution est placée dans la prochaine génération.

2. Mutation : dans ce cas une solution de la génération précédante est choisi aléatoirement. Nous choisissons aléatoirement une case dans cette solution. Cette case doit être une case que nous essayons de résoudre, pas une case dont la solution était dans le problème de départ. Une valeur aléatoire différente de la valeur de cette case est choisi et va remplacer celle déjà présente. Cette nouvelle solution est placée dans la prochaine population.

   Par exemple, si nous avons le sudoku s_1 plus haut, et que la la case de la ligne 3 et colonne 7 est choisi. Cette case contient un 7 en ce moment. Nous trouvons une valeur entre 1 et 9 au hazard, il faut rouler à nouveau si un 7 est trouvé. Disons que nous roulons 4. Alors la nouvelle solution sera :

1	4	2	1	2	4	**8**	**1**	2
2	**4**	2	**8**	2	**6**	2	1	6
3	5	**5**	8	2	**2**	3	**9**	6
8	**7**	5	**4**	2	**1**	**9**	2	6
5	6	4	8	**7**	4	1	2	7
6	3	**4**	**5**	2	**9**	2	**8**	**3**
7	**6**	4	**1**	2	7	**4**	3	9
8	5	6	**6**	1	**5**	1	**2**	8
7	**2**	**8**	9	5	5	3	9	7

Finalement nous ajoutons aux 1000 nouvelles solutions les solutions de la génération précédante qui n'ont pas été choisies pour un couplage ou une mutation.

2.2.3 Sélection

Ensuite nous devons choisi 100 solutions parmi les nouvelles solutions construites. Nous devons construire une métrique représentant le niveau de succès d'une solution. Dans notre cas une métrique basse représente un bon succès, si elle est à zéro (0) c'est une solution correct du sudoku.

Nous devons premièrement partitionner notre grille de sudoku en 3 partitions différentes. Une première partition sera les lignes : chaque ligne représente un ensemble $(L_i, i \in [1..9])$ de la partition. Une deuxième partition sera les colonnes : chaque colonne représente un ensemble $(C_i, i \in [1..9])$ de la partition. Une dernière partition est de diviser la grille en carré de 3×3 :

P_1	P_2	P_3
P_4	P_5	P_6
P_7	P_8	P_9

Notre formule pour la métrique devient :

$$\forall v \in [1..9] \qquad s_l = \sum_{i=1}^{9} (\mid \{x \mid x \in L_i, x = v\} \mid -1)^2 \tag{1}$$

$$s_c = \sum_{j=1}^{9} (\mid \{y \mid y \in C_j, y = v\} \mid -1)^2 \tag{2}$$

$$s_p = \sum_{k=1}^{9} (\mid \{z \mid z \in C_k, z = v\} \mid -1)^2 \tag{3}$$

$$s_v = s_l + s_c + s_p \tag{4}$$

$$m = \sum_{v=1}^{9} s_v \tag{5}$$

où si une des cardinalité d'ensemble est zéro alors elle est remplacée par 1.

Vous évaluez la métrique pour chaque solution de la nouvelle génération. ensuite vous gardez les 100 sudoku qui ont la métrique la plus basse. Vous devez aussi afficher la solution qui à la métrique la plus basse. Si cette métrique est à zéro alors l'application se termine après cette affichage. Sinon vous demandez à l'utilisateur s'il veut construire une nouvelle génération, la question doit être répondu par un caractère 'o' ou 'n' (en minuscule). Si la réponse est oui ('o') alors vous évaluez une nouvelle génération.

Exemple de calcul de métrique pour s_1 :

1	4	2	1	2	4	**8**	**1**	2
2	**4**	2	**8**	2	**6**	2	1	6
3	5	**5**	8	2	**2**	3	**9**	6
8	**7**	5	**4**	2	**1**	**9**	2	6
5	6	4	8	**7**	4	1	2	7
6	3	**4**	**5**	2	**9**	2	**8**	**3**
7	**6**	6	**1**	2	7	**4**	3	9
8	5	6	**6**	1	**5**	1	**2**	8
7	**2**	**8**	9	5	5	3	9	7

$$
\begin{aligned}
v = 1 \qquad & s_l = \sum_{i=1}^{9} (\mid \{x \mid x \in L_i, x = 1\} \mid -1)^2 \\
& s_l = (\mid \{x \mid x \in L_1, x = 1\} \mid -1)^2 + \\
& (\mid \{x \mid x \in L_2, x = 1\} \mid -1)^2 + \\
& (\mid \{x \mid x \in L_3, x = 1\} \mid -1)^2 + \\
& (\mid \{x \mid x \in L_4, x = 1\} \mid -1)^2 + \\
& (\mid \{x \mid x \in L_5, x = 1\} \mid -1)^2 + \\
& (\mid \{x \mid x \in L_6, x = 1\} \mid -1)^2 + \\
& (\mid \{x \mid x \in L_7, x = 1\} \mid -1)^2 + \\
& (\mid \{x \mid x \in L_8, x = 1\} \mid -1)^2 + \\
& (\mid \{x \mid x \in L_9, x = 1\} \mid -1)^2 \\
& s_l = (\mid \{1, 1, 1\} \mid -1)^2 + \\
& (\mid \{1\} \mid -1)^2 + \\
& (\mid \{\} \mid -1)^2 + \\
& (\mid \{1\} \mid -1)^2 + \\
& (\mid \{1\} \mid -1)^2 + \\
& (\mid \{\} \mid -1)^2 + \\
& (\mid \{1\} \mid -1)^2 + \\
& (\mid \{1, 1\} \mid -1)^2 + \\
& (\mid \{\} \mid -1)^2
\end{aligned}
$$

$$
\begin{aligned}
s_l = &(3-1)^2 + \\
&(0-1)^2 + \\
&(0-1)^2 + \\
&(1-1)^2 + \\
&(1-1)^2 + \\
&(0-1)^2 + \\
&(1-1)^2 + \\
&(2-1)^2 + \\
&(0-1)^2
\end{aligned}
$$

remplacer les cardinalités à 0 par des 1.

$$
\begin{aligned}
s_l = &(3-1)^2 + \\
&(1-1)^2 + \\
&(1-1)^2 + \\
&(1-1)^2 + \\
&(1-1)^2 + \\
&(1-1)^2 + \\
&(1-1)^2 + \\
&(2-1)^2 + \\
&(1-1)^2 \\
s_l = &2^2 + 0^2 + 0^2 + 0^2 + \\
&0^2 + 0^2 + 0^2 + 1^2 + 0^2 \\
s_l = &4 + 1 \\
s_l = &5
\end{aligned}
$$

$$s_c = \sum_{j=1}^{9}(\mid \{y \mid y \in C_j, y = 1\} \mid -1)^2$$
$$s_c = (\mid \{x \mid x \in C_1, x = 1\} \mid -1)^2 +$$
$$(\mid \{x \mid x \in C_2, x = 1\} \mid -1)^2 +$$
$$(\mid \{x \mid x \in C_3, x = 1\} \mid -1)^2 +$$
$$(\mid \{x \mid x \in C_4, x = 1\} \mid -1)^2 +$$
$$(\mid \{x \mid x \in C_5, x = 1\} \mid -1)^2 +$$
$$(\mid \{x \mid x \in C_6, x = 1\} \mid -1)^2 +$$
$$(\mid \{x \mid x \in C_7, x = 1\} \mid -1)^2 +$$
$$(\mid \{x \mid x \in C_8, x = 1\} \mid -1)^2 +$$
$$(\mid \{x \mid x \in C_9, x = 1\} \mid -1)^2$$
$$s_c = (\mid \{1\} \mid -1)^2 +$$
$$(\mid \{\} \mid -1)^2 +$$
$$(\mid \{\} \mid -1)^2 +$$
$$(\mid \{1, 1\} \mid -1)^2 +$$
$$(\mid \{1\} \mid -1)^2 +$$
$$(\mid \{1\} \mid -1)^2 +$$
$$(\mid \{1, 1\} \mid -1)^2 +$$
$$(\mid \{1, 1\} \mid -1)^2 +$$
$$(\mid \{\} \mid -1)^2$$
$$s_c = (1 - 1)^2 +$$
$$(0 - 1)^2 +$$
$$(0 - 1)^2 +$$
$$(2 - 1)^2 +$$
$$(1 - 1)^2 +$$
$$(1 - 1)^2 +$$
$$(2 - 1)^2 +$$
$$(2 - 1)^2 +$$
$$(0 - 1)^2$$

$$
\begin{aligned}
s_c &= (1-1)^2 + \\
&(1-1)^2 + \\
&(1-1)^2 + \\
&(2-1)^2 + \\
&(1-1)^2 + \\
&(1-1)^2 + \\
&(2-1)^2 + \\
&(2-1)^2 + \\
&(1-1)^2 \\
s_c &= 0^2 + 0^2 + 0^2 + 1^2 + \\
&0^2 + 0^2 + 1^2 + 1^2 + 0^2 \\
s_c &= 1 + 1 + 1 \\
s_c &= 3 \\
s_p &= \sum_{k=1}^{9} (\mid \{z \mid z \in C_k, z = 1\} \mid -1)^2 \\
s_p &= (\mid \{x \mid x \in P_1, x = 1\} \mid -1)^2 + \\
&(\mid \{x \mid x \in P_2, x = 1\} \mid -1)^2 + \\
&(\mid \{x \mid x \in P_3, x = 1\} \mid -1)^2 + \\
&(\mid \{x \mid x \in P_4, x = 1\} \mid -1)^2 + \\
&(\mid \{x \mid x \in P_5, x = 1\} \mid -1)^2 + \\
&(\mid \{x \mid x \in P_6, x = 1\} \mid -1)^2 + \\
&(\mid \{x \mid x \in P_7, x = 1\} \mid -1)^2 + \\
&(\mid \{x \mid x \in P_8, x = 1\} \mid -1)^2 + \\
&(\mid \{x \mid x \in P_9, x = 1\} \mid -1)^2 \\
s_p &= (\mid \{1\} \mid -1)^2 + \\
&(\mid \{1\} \mid -1)^2 + \\
&(\mid \{1, 1\} \mid -1)^2 + \\
&(\mid \{\} \mid -1)^2 + \\
&(\mid \{1\} \mid -1)^2 + \\
&(\mid \{1\} \mid -1)^2 + \\
&(\mid \{\} \mid -1)^2 + \\
&(\mid \{1, 1\} \mid -1)^2 + \\
&(\mid \{1\} \mid -1)^2
\end{aligned}
$$

$$
\begin{aligned}
& s_p = (1-1)^2 + \\
& (1-1)^2 + \\
& (2-1)^2 + \\
& (0-1)^2 + \\
& (1-1)^2 + \\
& (1-1)^2 + \\
& (0-1)^2 + \\
& (2-1)^2 + \\
& (1-1)^2 \\
& s_p = (1-1)^2 + \\
& (1-1)^2 + \\
& (2-1)^2 + \\
& (1-1)^2 + \\
& (1-1)^2 + \\
& (1-1)^2 + \\
& (1-1)^2 + \\
& (2-1)^2 + \\
& (1-1)^2 \\
& s_p = 0^2 + 0^2 + 1^2 + 0^2 + \\
& 0^2 + 0^2 + 0^2 + 1^2 + 0^2 \\
& s_p = 1 + 1 \\
& s_p = 2 \\
& s_1 = s_l + s_c + s_p \\
& s_1 = 5 + 3 + 2 \\
& s_1 = 10
\end{aligned}
$$

v_2

$$
\begin{aligned}
& s_l = \sum_{i=1}^{9} (\mid \{x \mid x \in L_i, x = 2\} \mid -1)^2 \\
& s_l = 4 + 9 + 1 + 1 + 0 + 1 + 0 + 0 + 0 \\
& s_l = 16 \\
& s_c = \sum_{j=1}^{9} (\mid \{y \mid y \in C_j, y = 2\} \mid -1)^2 \\
& s_c = 0 + 0 + 1 + 0 + 25 + 0 + 1 + 4 + 0 \\
& s_c = 31
\end{aligned}
$$

$$s_p = \sum_{k=1}^{9}(\mid \{z \mid z \in C_k, z = 2\} \mid -1)^2$$
$$s_p = 4 + 9 + 1 + 0 + 1 + 4 + 0 + 0 + 0$$
$$s_p = 19$$
$$s_2 = 16 + 31 + 19$$
$$s_2 = 66$$

v_3
$$s_l = 0 + 0 + 1 + 0 + 0 + 1 + 0 + 0 + 0$$
$$s_l = 2$$
$$s_c = 0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 + 0$$
$$s_c = 1$$
$$s_p = 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 1$$
$$s_p = 1$$
$$s_3 = 2 + 1 + 1$$
$$s_3 = 4$$

v_4
$$s_l = 1 + 0 + 0 + 0 + 1 + 0 + 0 + 0 + 0$$
$$s_l = 2$$
$$s_c = 1 + 1 + 0 + 0 + 1 + 0 + 0 + 0$$
$$s_c = 3$$
$$s_p = 1 + 0 + 0 + 1 + 1 + 0 + 0 + 0 + 1$$
$$s_p = 4$$
$$s_4 = 2 + 3 + 4$$
$$s_4 = 9$$

v_5
$$s_l = 0 + 0 + 1 + 0 + 0 + 0 + 0 + 1 + 1$$
$$s_l = 3$$
$$s_c = 0 + 0 + 1 + 0 + 0 + 1 + 0 + 0 + 0$$
$$s_c = 2$$
$$s_p = 1 + 0 + 0 + 1 + 0 + 0 + 0 + 4 + 0$$
$$s_p = 6$$
$$s_5 = 3 + 2 + 6$$
$$s_5 = 11$$

v_6
$$s_l = 0 + 1 + 0 + 0 + 0 + 0 + 1 + 1 + 0$$
$$s_l = 3$$
$$s_c = 0 + 1 + 1 + 0 + 0 + 0 + 0 + 0 + 4$$
$$s_c = 6$$

$$
\begin{aligned}
& & s_p &= 0+0+1+1+0+0+1+0+0 \\
& & s_p &= 3 \\
& & s_6 &= 3+6+3 \\
& & s_6 &= 12 \\
v_7 & & s_l &= 0+0+0+0+1+0+1+0+1 \\
& & s_l &= 3 \\
& & s_c &= 1+0+0+0+0+0+0+0+1 \\
& & s_c &= 2 \\
& & s_p &= 0+0+0+0+0+0+1+0+0 \\
& & s_p &= 1 \\
& & s_7 &= 3+2+1 \\
& & s_7 &= 6 \\
v_8 & & s_l &= 0+0+0+0+0+0+0+1+0 \\
& & s_l &= 1 \\
& & s_c &= 1+0+0+4+0+0+0+0+0 \\
& & s_c &= 5 \\
& & s_p &= 0+1+0+0+0+0+1+0+0 \\
& & s_p &= 2 \\
& & s_8 &= 1+5+2 \\
& & s_8 &= 8 \\
v_9 & & s_l &= 0+0+0+0+0+0+0+0+1 \\
& & s_l &= 1 \\
& & s_c &= 0+0+0+0+0+0+0+1+0 \\
& & s_c &= 1 \\
& & s_p &= 0+0+0+0+0+0+0+0+1 \\
& & s_p &= 1 \\
& & s_9 &= 1+1+1 \\
& & s_9 &= 3 \\
m & = & \sum_{v=1}^{9} s_v & \\
& = & 10+66+4+9+11+12+6+8+3 & \\
& = & 129 &
\end{aligned}
$$

Le sudoku à donc une métrique de 129.

2.3 Sorties

à la fin de chaque génération vous devez afficher la meilleure solution obtenue. Premièrement il faut afficher le numéro de génération. Vous afficher neuf (9) lignes de neuf (9) colonnes séparées par un espace. Chaque case contient une valeur de 1 à 9. Ensuite vous affichez la métrique obtenue pour cette solution.

Exemple d'affichage :

```
génération 7 :

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

métrique : 129

une autre génération ? (o/n)
```

3 Conception

Chacune de vos routines doit avoir des commentaires. Ces commentaires doivent avoir l'information suivantes :

- La routine est-elle une fonction ou une procédure ?
- La cohésion de la routine : fonctionnelle, séquentielle, communicationnelle ou temporelle.
- Rôle de la routine : ce qu'elle fait, ce qu'elle retourne.
- Argument : Description des arguments, leurs rôles dans la routine, les valeurs permisent, sont-ils des entrées ou des sorties ?, ...
- Les erreurs possibles.

Utilisez des assertions pour les entrées. (Ce qui ne vous enpêche pas de les utiliser pour les autres cas.)

Chaque module doit avoir un commentaire donnant l'informations suivantes sur le module :

- Nom du module.

- Sorte de module : bibliothèque de routine, classe d'objets, collection de valeurs ou machine abstraite.
- Une description sommaire du rôle du module.

Votre programme doit être dans plusieurs fichiers, vous pouvez utiliser mes modules de Séquences et de Dictionnaires disponible sur le site (dans ce cas veuillez les inclures avec votre projet). Vous devez construire au minimum un module de structure de données pour ce tp, avec le fichier .h et .c. Vous devez fournir un makefile, les .h et les .c. Quand je vais recevoir votre logiciel, je vais le décompresser avec la ligne suivante sur chicoree :

```
gunzip  nomDeLArchive.tar.gz
```

Ensuite je vais désarchiver l'archive avec la commande suivante :

```
tar xvf  nomDeLArchive.tar
```

et finalement je compile le projet avec la commande suivante, toujours sur chicoree :

```
make
```

Je ne fais aucune exception.

4 Remise

N'oubliez pas d'écrire votre nom. Remettez votre programme en utilisant Moodle. Vous n'avez le droit d'utiliser que les librairies standards de C pour votre logiciel :

assert.h, complex.h, ctype.h, errno.h, fenv.h, float.h, inttypes.h, iso646.h, limits.h, locale.h, math.h, setjmp.h, signal.h, stdarg.h, stdbool.h, stddef.h, stdint.h, stdio.h, stdlib.h, string.h tgmath.h, time.h, wchar.h, wctype.h.

5 Évaluation

- Fonctionnalité (18 pts).
  - Votre programme doit compiler avec le compilateur 'gcc' sur la machine rayon. Si le programme ne compile pas, aucun point n'est donné pour la fonctionnalité.
- Structure (6 pts).
  - Le respect des critères pour les commentaires.
  - Bonne utilisation des structures de contrôle : if, while, ...
  - Bonne modularité.

- Lisibilité (6 pts).
  - Bonne indentation.
  - Identificateur significatif.
- Autres critères :
  - Pas de 'goto'.
  - Un seul 'return' par routine, à la fin de celle-ci.
  - Pas de variables globales.
  - Pas de ligne contenant des commentaires et du code.