Centrale 2002 — épreuve d'informatique — corrigé

Laurent Chéno, lycée Louis-le-Grand, Paris

mai 2002

On choisit dans ce corrigé le langage Caml.

Première partie
Systèmes de pièces

1 Fonction de validation

On propose le programme 1.

Programme 1 validation d'un système

```
let rec est_un_systeme = function
  | [] -> false
  | [ x ] -> x = 1
  | a :: b :: q -> a > b && est_un_systeme (b :: q) ;;
```

Deuxième partie
Représentations de poids minimal

2 Un encadrement

On peut toujours, puisque $c_m = 1$, rendre la monnaie en pièces de 1 : cette représentation est de poids x, donc $M(x) \leqslant x$.
Soit k une représentation de $x = \sum_{i=1}^{m} k_i c_i$. On a $x \leqslant c_1 \sum_{i=1}^{m} k_i = c_1 \|k\|$ puisque c_1 est le plus grand entier du système c. Donc $\|k\| \geqslant \frac{x}{c_1}$. Mais $\|k\|$ est entier donc finalement, on a toujours $\|k\| \geqslant \left\lceil \frac{x}{c_1} \right\rceil$. Ce qui est vrai pour toute représentation vaut pour une représentation minimale, et donc on a bien $\left\lceil \frac{x}{c_1} \right\rceil \leqslant M(x) \leqslant x$.

3 Non-unicité de la représentation

Par exemple dans le système Euro, 100 admet au moins deux représentations : $(0, 1, 0, 0, 0, 0, 0, 0)$ et $(0, 0, 2, 0, 0, 0, 0, 0)$.

4 Une majoration

Soit k' une représentation minimale de $x - c_j$, $k = k' + e_j$ est clairement une représentation de x, et donc $M(x) \leqslant \|k\| = \|k' + e_j\| = 1 + \|k'\| = 1 + M(x - c_j)$.

5 Une condition nécessaire et suffisante

Supposons que $M(x) = 1 + M(x - c_j)$ où $c_j \leqslant x$. Soit k' une représentation minimale de $x - c_j$. La représentation $k = k' + e_j$ de x vérifie $\|k\| = \|k' + e_j\| = 1 + M(x - c_j) = M(x)$ donc k est une représentation minimale de x faisant intervenir c_j.
Réciproquement, supposons qu'il existe une représentation minimale k de x faisant intervenir c_j : $k' = k - e_j$ est bien alors une représentation de $x - c_j$, et on dispose de $M(x) = \|k\| = 1 + \|k'\|$.
Si k' n'était pas une représentation minimale de $x - c_j$, il existerait une représentation k'' de $x - c_j$ telle que $\|k''\| \leqslant \|k'\| - 1 = M(x) - 2$. Alors $k'' + e_j$ serait une représentation de x de poids $\|k'' + e_j\| = 1 + \|k''\| \leqslant M(x) - 1$ ce qui contredit la définition de $M(x)$.
Donc k' est une représentation minimale de $x - c_j$ et $M(x) = 1 + \|k'\| = 1 + M(x - c_j)$.

6 Une égalité

D'après la question 3, pour tout indice $i \geqslant s$, $M(x) \leqslant 1 + M(x - c_i)$, donc $M(x) \leqslant 1 + \min\limits_{s \leqslant i \leqslant m} M(x - c_i)$.
Mais si k est une représentation minimale de x qui fait figurer un certain c_j, on a bien sûr $c_j \leqslant x$ donc $j \geqslant s$, et, d'après la question 4, $M(x) = 1 + M(x - c_j) \geqslant 1 + \min\limits_{s \leqslant i \leqslant m} M(x - c_i)$.
On a bien établi l'égalité demandée.

7 Un algorithme

Remarquons qu'on a toujours $M(1) = 1$, quelle que soit la représentation choisie.
On décrit l'algorithme suivant, où M est un tableau où l'on peut ranger les résultats consécutifs :

1. on initialise $M(1) := 1$, $s := m$ et $y := 1$;
2. si $y = x$ on a terminé ;
3. $y := y + 1$;
4. tant que $s > 1$ et $c_{s-1} \leqslant y$, faire $s := s - 1$;
5. $M(y) := 1 + \min_{s \leqslant i \leqslant m} M(y - c_i)$;
6. retourner à l'étape 2.

À l'étape 4, on évalue le nouveau s en observant que quand y augmente, s ne peut que diminuer.
À l'étape 5, on doit faire le calcul d'un minimum parmi des éléments déjà calculés du tableau M.
Le coût de cet algorithme est clairement un $O(mx)$, puisque y prend x valeurs consécutives, et que, pour chaque valeur de y, l'étape 4 se déroule en au plus $s \leqslant m$ opérations, et l'étape 5 en $m - s + 1 \leqslant m$ opérations.

8 Un programme

On traduit l'algorithme de la question précédente. Pour coller au plus près à cet algorithme, on indexe les tableaux à partir de 1, contrairement à l'habitude : on ajoute pour cela des éléments fictifs à l'indice 0 aux tableaux `M` et `cv` (qui contient les valeurs du système).
On obtient le programme 2 page suivante.

Troisième partie
L'algorithme glouton

9 Programmation

L'écriture récursive de `glouton` est naturelle et ne présente guère de difficulté : on obtient le programme 3.

Programme 2 recherche des poids minimaux

```
let poids_minimaux x c =
  let s = ref (list_length c)
  and m = list_length c
  and M = make_vect (x+1) 0
  and cv = vect_of_list (0 :: c)
  in
    M.(1) <- 1;
    for y = 2 to x do
      while !s > 1 && cv.(!s - 1) <= y do decr s done;
      M.(y) <- 1 + M.(y - cv.(!s)) ;
      for i = !s + 1 to m do
        let x = 1 + M.(y - cv.(i)) in if x < M.(y) then M.(y) <- x
      done
    done;
    tl (list_of_vect M) ;;
```

Programme 3 l'algorithme glouton

```
let rec glouton x = function
  | [ 1 ] -> [ x ]
  | c :: q -> (x/c) :: glouton (x - (x/c)*c) q
  | [] -> failwith "systeme non valide" ;;
```

10 (c_1, c_2) est canonique

Notons simplement $c = (a, 1)$ où $a > 1$.
Soit $x \geqslant 1$ et $x = aq + r$ sa division euclidienne par $a : 0 \leqslant r \leqslant a - 1$. Alors (q, r) est la représentation gloutonne de x, et $G_c(x) = q + r$.
Une autre représentation de x est de la forme $k = (q', x - aq')$ avec $q' < q$: alors $\|k\| - G_c(x) = q' + x - aq' - q - r = q' - q + a(q - q') = (a - 1)(q - q') \geqslant 0$.
On a bien montré que la représentation gloutonne était minimale, et donc que le système est canonique.

11 Un système non canonique

Le système $(6, 4, 1)$ n'est pas canonique.
En effet, $x = 8$ y a pour représentation gloutonne $(1, 0, 2)$, de poids 3, alors que la représentation $k = (0, 2, 0)$ est de poids moindre $\|k\| = 2 < 3$.

12 $(q^n, q^{n-1}, \ldots, q^2, q, 1)$ est canonique

Raisonnons par l'absurde : si le système n'est pas canonique, il existe un entier x qui admet une représentation minimale k de poids strictement inférieur à celui de sa représentation gloutonne g.
Soit alors $i = \min\{j, 1 \leqslant j \leqslant n + 1, g_j \neq k_j\}$. Quitte à remplacer x par $x - \sum_{j=1}^{i-1} g_j e_j$ et à tronquer le système par la gauche de $i - 1$ valeurs, on se ramène au cas où $g_1 \neq k_1$.
Comme $g_1 = \left\lfloor \frac{x}{q^n} \right\rfloor$, $(1 + g_1)q^n > x$ et on a donc $k_1 < g_1$. Quitte à remplacer x par $x - k_1 q^n$, on peut même supposer que $k_1 = 0$ et $g_1 \geqslant 1$.
Remarquons que $\forall i, k_i \leqslant q - 1$. En effet, sinon, $k' = (k_1, \ldots, k_{i-1} + 1, k_i - q, k_{i+1}, \ldots, k_{n+1})$ serait une représentation de poids $\|k\| + 1 - q$ strictement inférieur à $\|k\|$ ce qui contredit sa minimalité.
Alors $x = \sum_{i=2}^{n+1} k_i q^{n+1-i} \leqslant (q - 1)(1 + q + \cdots + q^{n-1}) = q^n - 1$ alors que $g_1 \geqslant 1 \Rightarrow x \geqslant q^n$: c'est la contradiction souhaitée, et le système est bien canonique.

13 Le système Euro est canonique

Soit $x \geqslant 2$ et $k = (k_1, \ldots, k_8)$ une représentation minimale de x.
$k_8 \leqslant 1$ car sinon $(k_1, \ldots, k_7 + 1, k_8 - 2)$ serait une représentation moins lourde.
Alors $2k_7 + k_8 \leqslant 4$ car sinon $2k_7 \geqslant 5 - k_8 \geqslant 4$ donc $k_7 = 2$ et $k_8 \geqslant 1$ mais alors $(k_1, \ldots, k_6 + 1, 0, k_8 - 1)$ serait une représentation moins lourde, ou alors $k_7 \geqslant 3$ mais alors $(k_1, \ldots, k_7 + 1, k_7 - 3, k_8 + 1)$ serait une représentation moins lourde.
Alors $5k_6 + 2k_7 + k_8 \leqslant 9$ car sinon $5k_6 \geqslant 10 - (2k_7 + k_8) \geqslant 6$ donc $k_6 \geqslant 2$ mais alors $(k_1, \ldots, k5 + 1, k_6 - 2, k_7, k_8)$ serait une représentation moins lourde.
Alors $10k_5 + 5k_6 + 2k_7 + k_8 \leqslant 19$ car sinon $10k_5 \geqslant 20 - (5k_6 + 2k_7 + k_8) \geqslant 11$ donc $k_5 \geqslant 2$ mais alors $(k_1, \ldots, k_4 + 1, k_5 - 2, k_6, k_7, k_8)$ serait une représentation moins lourde.
Alors $20k_4 + 10k_5 + 5k_6 + 2k_7 + k_8 \leqslant 49$ car sinon $20k_4 \geqslant 50 - (10k_5 + 5k_6 + 2k_7 + k_8) \geqslant 31$: ou bien $k_4 \geqslant 3$ mais alors $(k_1, k_2, k_3 + 1, k_4 - 3, k_5 + 1, k_6, k_7, k_8)$ serait une représentation moins lourde ; ou bien $k_4 = 2$ et $10k_5 + 5k_6 + 2k_7 + k_8 \geqslant 10$ donc, puisque $5k_6 + 2k_7 + k_8 \leqslant 9$, $k_5 \geqslant 1$, mais alors $(k_1, k_2, k_3 + 1, 0, k_5 - 1, k_6, k_7, k_8)$ serait une représentation moins lourde.
Alors $50k_3 + 20k_4 + 10k_5 + 5k_6 + 2k_7 + k_8 \leqslant 99$ car sinon $50k_3 \geqslant 100 - (20k_4 + 10k_5 + 5k_6 + 2k_7 + k_8) \geqslant 51$ donc $k_3 \geqslant 2$ mais alors $(k_1, k_2 + 1, k_3 - 2, k_4, \ldots, k_8)$ serait une représentation moins lourde.
Alors $100k_2 + 50k_3 + 20k_4 + 10k_5 + 5k_6 + 2k_7 + k_8 \leqslant 199$ car sinon $100k_2 \geqslant 200 - (50k_3 + 20k_4 + 10k_5 + 5k_6 + 2k_7 + k_8) \geqslant 101$ donc $k_2 \geqslant 2$ mais alors $(k_1 + 1, k_2 - 2, k_3, \ldots, k_8)$ serait une représentation moins lourde.
On a quasiment terminé la preuve : en effet, $0 \leqslant x - 200k_1 = k.c - 200k_1 \leqslant 199$ donc k_1 est bien le quotient de x par 200. Puis notant $x_1 = x - 200k_1$, $0 \leqslant x_1 - 100k_2 \leqslant 99$ donc k_2 est bien le quotient de x_1 par 100, puis, de la même manière, k_3 est le quotient de $x_2 = x_1 - 100k_2$ par 50, k_4 le quotient de $x_3 = x_2 - 50k_3$ par 20, k_5 le quotient de $x_4 = x_3 - 20k_4$ par 10, k_6 le quotient de $x_5 = x_4 - 10k_5$ par 5, k_7 le quotient de $x_6 = x_5 - 5k_6$ par 2 et enfin $k_8 = x_7 = x_6 - 2k_7$: la représentation k coïncide avec la représentation gloutonne.

14 Le système britannique impérial

48 a pour représentation gloutonne $(1, 0, 1, 1, 0, 0)$, de poids 3, alors que la représentation $(0, 2, 0, 0, 0, 0)$ est de poids moindre : le système britannique impérial n'est pas canonique.

Quatrième partie
L'algorithme de Kozen et Zaks

15 Plus petit contre-exemple

Montrons que pour tout contre-exemple x, $x > 1 + c_{m-2}$.
Soit pour cela $x \leqslant 1 + c_{m-2}$ et montrons que sa représentation gloutonne g est minimale, donc que x n'est pas un contre-exemple.
Si $x = c_{m-2} + 1$, $g = (0, \ldots, 0, 1, 0, 1)$ de poids 2 qui est bien minimale sauf si $x = c_{m-2} + 1 = c_{m-1}$, mais dans ce cas on aurait eu $g = (0, \ldots, 0, 1, 0, 0, 0)$ évidemment minimale.
Si $x = c_{m-2}$, $g = e_{m-2}$ est évidemment minimale.
Si $x < c_{m-2}$, toute représentation de x ne peut faire intervenir que c_{m-1} et $c_m = 1$, mais on a vu à la question 10 que $(c_{m-1}, 1)$ est canonique, d'où le résultat.

Remarquons que $c_1 + c_2$ a pour représentation gloutonne $(1, 1, 0, \ldots, 0)$ qui est évidemment minimale, donc n'est pas un contre-exemple.
Soit maintenant x le plus petit des contre-exemples, supposons que $x \geqslant c_1 + c_2 + 1$, et trouvons une contradiction. On sait que si $y < x$, $G_c(y) = M(y)$.
On sait qu'il existe une représentation minimale k de x, vérifiant : $M(x) = \|k\| < G_c(x)$.

Si c_1 figure dans k, alors $G_c(x) = 1+G_c(x-c_1)$ mais $x-c_1$ n'est pas un contre-exemple donc $G_c(x-c_1) = M(x-c_1)$ et on conclut que $G_c(x) = 1 + M(x-c_1) = M(x)$ grâce à la question 5, et donc que x n'est pas un contre-exemple : c'est la contradiction espérée.
Sinon, si c_1 ne figure pas dans k, choisissons $i \geqslant 2$ tel que c_i figure dans k : $x > c_1 + c_2 \geqslant c_1 + c_i$.
On a de même que ci-dessus $G_c(x) = 1 + G_c(x-c_1) = 1 + M(x-c_1)$. En outre, d'après 4,

$$M(x-c_1) \leqslant 1 + M(x-c_1-c_i) = 1 + G_c(x-c_1-c_i) = G_c(x-c_i) = M(x-c_i).$$

Mais la question 5 nous dit ici que $M(x-c_i) = M(x) - 1$, de sorte que l'on dispose de

$$G_c(x) = 1 + M(x-c_1) \leqslant 1 + M(x-c_i) = M(x).$$

x n'est donc pas un contre-exemple : c'est là encore la contradiction attendue.

16 Le système $(q+1, q, 1)$ n'est pas canonique

La question précédente assure que le plus petit contre-exemple est à chercher entre $q+3$ et $2q$ (bornes incluses).
Soit $x = 2q = q + 1 + (q-1)1 = q + q$. Sa représentation gloutonne est $(1, 0, q-1)$, de poids $q \geqslant 3$, sa représentation minimale est $(0, 2, 0)$ de poids 2 : le système n'est pas canonique.
Si $q + 3 \leqslant x < 2q$, sa représentation gloutonne est $g = (1, 0, x-q-1)$, de poids $x-q$. Soit k une représentation minimale de x : si c_1 y figure, alors $k_1 = 1$ (car $x < 2(q+1)$) et, comme $x - (q+1) \leqslant q-2 < c_2$, c'est que $k = g$; sinon, si c_2 y figure, alors $k_1 = 0$, $k_2 = 1$ (car $x < 2q$) et donc $k = (0, 1, x-q)$ de poids $x-q+1$ trop lourd ; enfin, si $k = (0, 0, x)$, k n'est évidemment pas minimale.
On a bien montré que $2q$ est le plus petit contre-exemple.

17 Le système $(\alpha(q), q, 1)$

Notons que si le système n'est pas canonique, son plus petit contre-exemple est au moins égal à $\alpha(q)+2$.
La représentation gloutonne de $x = \alpha(q)+2$ est $g = (1, 0, 2)$ de poids 3. Une représentation k de x telle que $k_1 = 1$ coïncide nécessairement avec g. Si $k = (0, k_2, k_3)$ permet de prouver que x est un contre-exemple, c'est que $k_2 q + k_3 = \alpha(q) + 2$ et $k_2 + k_3 \leqslant 2$.
Il y a 5 cas à étudier.
$k = (0, 0, 1)$: $x = 1$ donc $\alpha(q) = -1 < q$, c'est exclu.
$k = (0, 0, 2)$: $x = 2$ donc $\alpha(q) = 0 < q$, c'est exclu.
$k = (0, 1, 0)$: $x = q$ donc $\alpha(q) = q - 2 < q$, c'est exclu.
$k = (0, 1, 1)$: $x = q + 1$ donc $\alpha(q) = q - 1 < q$, c'est exclu.
$k = (0, 2, 0)$: $x = 2q$ donc $\alpha(q) = 2q - 2 = q + q - 2 > q$ est le seul cas acceptable.
On peut énoncer que $(2q-2, q, 1)$ n'est pas canonique, et que le plus petit contre-exemple qui le prouve est $x = 2q$.

18 Une remarque judicieuse

On vient de montrer que les bornes obtenues à la question 15 peuvent être atteintes et qu'elles sont donc optimales.

19 Les témoins sont des contre-exemples

D'après la question 5, si x est un témoin, et $c_i < x$ avec $G_c(x-c_i) < G(x)-1$, on a $M(x) = 1+M(x-c_i) \leqslant 1 + G_c(x-c_i) < G_c(x)$, et donc x est un contre-exemple.

20 Réciproque ?

Dans le système $(5, 4, 1)$, on vérifie que $x = 12$ est un contre-exemple, mais pas un témoin.
En effet, sa représentation gloutonne est $g = (2, 0, 2)$, une représentation minimale est $k = (0, 3, 0)$, de poids moindre, mais $12 - 5 = 7$ est de représentation gloutonne $(1, 0, 2)$ de poids 3, $12 - 4 = 8$ est de représentation gloutonne $(1, 0, 3)$ de poids 4, $12 - 1 = 11$ est de représentation gloutonne $(1, 1, 2)$ de poids 4 : on n'a jamais $G_c(x - c_i) < G_c(x) - 1 = 3$.

21 Le plus petit contre-exemple est un témoin

Soit x le plus petit contre-exemple : $G_c(x) > M(x) = \|k\|$, où k est une représentation minimale.
Soit i tel que c_i figure dans k : la question 5 fournit $M(x) = 1 + M(x - c_i) = 1 + G_c(x - c_i)$ puisque $x - c_i$ n'est pas un contre-exemple. On en déduit que $G_c(x - c_i) = M(x) - 1 < G_c(x) - 1$ et que x est bien un témoin.

22 Des exemples d'utilisation de l'algorithme de Kozen et Zaks

22.1 Le système $(q, 2, 1)$ où $q \geqslant 3$

On devrait tester les entiers x tels que $q+2 \leqslant x \leqslant q+1$: il n'y a rien à faire, et le système est canonique.

22.2 Le système $(7, 4, 1)$

On doit tester les entiers x tels que $9 \leqslant x \leqslant 10$.
Le tableau suivant indique les représentations gloutonnes et leurs poids, pour les x et les $x - c_i$.

y	repr. gloutonne	$G_c(y)$
9	$(1, 0, 2)$	$3 = 1 + 2$
$9 - 7 = 2$	$(0, 0, 2)$	$2 \geqslant 2$
$9 - 4 = 5$	$(0, 1, 1)$	$2 \geqslant 2$
$9 - 1 = 8$	$(1, 0, 1)$	$2 \geqslant 2$
10	$(1, 0, 3)$	$4 = 1 + 3$
$10 - 7 = 3$	$(0, 0, 3)$	$3 \geqslant 3$
$10 - 4 = 6$	$(0, 1, 2)$	$3 \geqslant 3$
$10 - 1 = 9$	$(1, 0, 2)$	$3 \geqslant 3$

Le système est bien canonique.

22.3 Le système $(6, 5, 1)$

On teste de même les entiers entre 8 et 10.

y	repr. gloutonne	$G_c(y)$
8	$(1, 0, 2)$	$3 = 1 + 2$
$8 - 6 = 2$	$(0, 0, 2)$	$2 \geqslant 2$
$8 - 5 = 3$	$(0, 0, 3)$	$3 \geqslant 2$
$8 - 1 = 7$	$(1, 0, 1)$	$2 \geqslant 2$
9	$(1, 0, 3)$	$4 = 1 + 3$
$9 - 6 = 3$	$(0, 0, 3)$	$3 \geqslant 3$
$9 - 5 = 4$	$(0, 0, 4)$	$4 \geqslant 3$
$9 - 1 = 8$	$(1, 0, 2)$	$3 \geqslant 3$
10	$(1, 0, 4)$	$5 = 1 + 4$
$10 - 6 = 4$	$(0, 0, 4)$	$4 \geqslant 4$
$10 - 5 = 5$	$(0, 1, 0)$	$\mathbf{1 < 4}$
$10 - 1 = 9$	$(1, 0, 3)$	$4 \geqslant 4$

Le témoin 10 permet d'affirmer que le système n'est pas canonique.

23 Un programme Caml

Le programme 4 page suivante reprend l'algorithme de Kozen et Zaks.

24 Discussion sur le coût

Dans le cas du système $(q^{m-1}, q^{m-2}, \ldots, q^2, q, 1)$ dont nous avons vu, dans la question 12, qu'il est canonique, l'algorithme de Kozen et Zaks va devoir examiner les entiers x compris au sens large entre q^2+2 et $q^{m-2}+q^{m-1}-1$ à la recherche d'un témoin qu'il ne trouvera pas. Le coût de l'algorithme sera alors clairement exponentiel en m.
Le coût du calcul de $G_c(x)$ est linéaire en m, pour tout entier x. Vérifier si x est un témoin a donc un coût en $O(m^2)$. L'utilisation de l'algorithme de Kozen et Zaks pour prouver qu'un système n'est pas canonique est par conséquent de coût exponentiel quand la plage des témoins possibles est de taille exponentielle en m et que le plus petit des contre-exemples (c'est-à-dire le premier témoin trouvé) est maximal dans cette plage.
Considérons ainsi le système $c = (1+2^n, 2^n, 2^{n-1}, \ldots, 2^2, 2, 1)$. Comme $(2^n, \ldots, 2, 1)$ est canonique, il n'y a pas de contre-exemple inférieur ou égal à 2^n.
$4+2^n$ est un contre-exemple : sa représentation gloutonne est $(1, 0, \ldots, 0, 1, 1)$. Donc $G_c(2.2^n) = 3 > 2 = \|k\|$ où $k = (0, 1, 0, \ldots, 0, 1, 0, 0)$ est une autre représentation de $4+2^n$ (c'en est d'ailleurs évidemment une représentation minimale).
Montrons pour terminer notre démonstration qu'il n'y a pas de contre-exemple entre 2^n+1 et 2^n+3 : 2^n+4 sera le plus petit contre-exemple et donc le premier témoin trouvé, ce qui aura eu un coût exponentiel.
$c_1 = 2^n + 1$ n'est évidemment pas un contre-exemple.
$2^n + 2$ a pour représentation gloutonne $(1, 0, \ldots, 0, 1)$ de poids 2, qui est clairement minimale.
$2^n + 3$ a pour représentation gloutonne $(1, 0, \ldots, 0, 1, 0)$ de poids 2, qui est clairement minimale.
La démonstration est achevée, le problème et le correcteur aussi...

Programme 4 l'algorithme de Kozen et Zaks

```
(* for_all : ('a -> bool) -> 'a list -> bool                          *)
(* for_all predicat liste renvoie true si et seulement si            *)
(* le predicat est verifie par tous les elements de la liste         *)
let rec for_all p = function
  | [ ] -> true
  | t :: q -> p t && for_all p q ;;

(* poids : int list -> int                                           *)
(* poids k renvoie le poids d'une representation k                   *)
let rec poids = function
  | [ ] -> 0
  | t :: q -> t + (poids q) ;;

(* gc : int list -> int -> int                                       *)
(* gc c x renvoie le poids de la representation gloutonne de x       *)
(* dans le systeme c                                                 *)
let gc c x = poids (glouton x c) ;;

(* teste : int list -> int -> int -> bool                            *)
(* teste c gx x utilise le systeme c,                                *)
(* gx est le poids de la representation gloutonne de x,              *)
(* la fonction renvoie true si et seulement si pour tout ci<x        *)
(* le poids de la representation gloutonne de x-ci est superieur a gx-1 *)
let teste c gx x =
  for_all (function ci -> ci >= x || gc c (x - ci) + 1 >= gx) c ;;

(* pourtous : int list -> (int * int) -> bool                        *)
(* pourtous c (a,b)  applique le test a tout entier x entre a et b   *)
let rec pourtous c (a,b) =
  if a <= b then
    teste c (gc c a) a && pourtous c (a+1,b)
  else true ;;

(* antepenultien : int list -> int                                   *)
(* antepenultien l renvoie son element antepenultien,                *)
(* l est supposee de longueur au moins egale a 3                     *)
let rec antepenultien l = match l with
  | [a;_;_] -> a
  | _ -> antepenultien (tl l) ;;

(* bornes : int list -> int * int                                    *)
(* bornes c renvoie les bornes de la plage des entiers a tester      *)
let bornes c =
  let c1 :: c2 :: _ = c in
    (antepenultien c + 2, c1 + c2 + 1) ;;

let kozen_zaks c =
  if list_length c < 3 then true
  else pourtous c (bornes c) ;;
```