Mots de Lukasiewicz (d'après Centrale 2008)

Durée : 2 heures

On utilise dans tout ce problème l'alphabet $\Sigma = \{-1, +1\}$. Un *mot* sur cet alphabet est une suite finie $u = (u_1, u_2, \ldots, u_n)$ telle que pour tout $k \in [\![1, n]\!]$, $u_k \in \Sigma$. L'entier n est la *longueur* du mot u. On appelle *poids* d'un mot, noté $p(u)$, la somme des lettres qui le composent : $p(u) = \sum_{k=1}^{n} u_k$.

On notera avec un point $\cdot$ la concaténation de deux mots. Par exemple, si $u = (u_1, \ldots, u_p)$ et $v = (v_1, \ldots, v_q)$ sont deux mots, on désignera par $u \cdot v$ le mot $(u_1, \ldots, u_p, v_1, \ldots, v_q)$.

On appelle *mot de Lukasiewicz* toute mot $u = (u_1, u_2, \ldots, u_n)$ vérifiant les deux propriétés suivantes :

$$\sum_{i=1}^{n} u_i = -1 \qquad \text{et} \qquad \sum_{i=1}^{k} u_i \geqslant 0 \quad \text{pour } 1 \leqslant k \leqslant n-1.$$

Les mots sur l'alphabet Σ seront représentés en Caml par le type *int list* ; pour cette raison on définit l'abréviation suivante :

```
type mot == int list ;;
```

Partie I. Quelques propriétés

Question 1. Donner tous les mots de Lukasiewicz de longueur 1, 2 et 3, puis tous ceux de longueur paire.

Question 2. Écrire une fonction **luka** qui prend en argument un mot et qui renvoie une valeur booléenne indiquant si ce mot est de Lukasiewicz. La fonction proposée devra impérativement avoir une complexité en $O(n)$, où n est la longueur du mot d'entrée.

```
luka : mot -> bool
```

Question 3. Montrer que si u et v sont des mots de Lukasiewicz, alors $(+1) \cdot u \cdot v$ est un mot de Lukasiewicz.

Question 4. Réciproquement, montrer que tout mot de Lukasiewicz de longueur supérieure ou égale à 3 admet une décomposition *unique* de la forme $(+1) \cdot u \cdot v$, où u et v sont des mots de Lukasiewicz.

Question 5. Écrire une fonction **decompose** qui prend pour argument un mot de Lukasiewicz de longueur supérieure ou égale à 3 et renvoie le couple (u, v) défini de manière unique à la question précédente. *On ne demande pas de traiter les cas où le mot fourni en entrée ne serait pas de Lukasiewicz.*

```
decompose : mot -> mot * mot
```

Question 6. On souhaite calculer tous les mots de Lukasiewicz d'une longueur donnée. Comparer les avantages d'une solution récursive appliquant le principe de la décomposition suggérée à la question 4, et celle d'une solution appliquant le même principe, mais pour laquelle on tabulerait les résultats intermédiaires.

Question 7. Compte tenu de la question précédente, écrire une fonction *efficace* **obtenir_luka** qui calcule la liste des mots de Lukasiewicz de taille inférieure ou égale à un entier donné.

```
obtenir_luka : int -> mot list
```

Question 8. On considère l'ensemble des arbres binaires définis en Caml par le type :

```
type arbre = Vide | Noeud of arbre * arbre
```

À l'aide de la question 4, établir que l'ensemble des mots de Lukasiewicz est en bijection avec l'ensemble des arbres binaires, puis rédiger deux fonctions **arbre_of_luka** et **luka_of_arbre** qui réalisent effectivement cette bijection.

```
arbre_of_luka : mot -> arbre
luka_of_arbre : arbre -> mot
```

Partie II. Dénombrement

Question 9. Soit $u = (u_1, \dots, u_n)$ un mot tel que $p(u) = -1$. Démontrer qu'il existe un *unique* entier $i \in [\![1, n]\!]$ tel que $(u_i, u_{i+1}, \dots, u_n, u_1, \dots, u_{i-1})$ soit un mot de Lukasiewicz. Ce mot est appelé le *conjugué* de u.

Question 10. Écrire une fonction **conjugue** qui calcule le conjugué d'un mot $u = (u_1, \dots, u_n)$ vérifiant $p(u) = -1$.

```
conjugue : mot -> mot
```

Question 11. En utilisant les résultats précédents, déterminer le nombre de mots de Lukasiewicz de longueur $2n+1$.
On pourra utiliser le résultat admis : *si u et v sont deux mots non vides, les deux propositions suivantes sont équivalentes :*

- $u \cdot v = v \cdot u$;
- *il existe un mot w et deux entiers $k, \ell \geqslant 1$ tels que $u = w^k$ et $v = w^\ell$.*

Partie III. Capsules

On appelle *capsule* d'un mot u tout facteur de u de la forme $(+1, -1, -1)$. En d'autres termes, si $u = (u_1, \dots, u_n)$ admet une capsule au rang $i \in [\![1, n-2]\!]$ lorsque $(u_i, u_{i+1}, u_{i+2}) = (+1, -1, -1)$.
On définit sur l'ensemble des mots une fonction ρ dite de *décapsulage* :

$$\rho(u) = u \quad \text{si } u \text{ ne contient pas de capsule ;}$$
$$\rho(u) = (u_1, \dots, u_{i-1}, u_{i+2}, \dots, u_n) \quad \text{si } (u_i, u_{i+1}, u_{i+2}) = (+1, -1, -1) \text{ est la première capsule de } u.$$

Question 12. Justifier le fait que la suite $\left(\rho^n(u)\right)_{n \in \mathbb{N}}$ est constante au delà d'un certain rang. La valeur limite de cette suite sera notée $\rho^*(u)$.

Question 13. Écrire une fonction **rho** qui prend pour argument un mot u et renvoie $\rho(u)$.

```
rho : mot -> mot
```

Question 14. Ecrire une fonction **rholim** qui prend pour argument un mot u et renvoie $\rho^*(u)$.

```
rholim : mot -> mot
```

Question 15. Démontrer enfin que u est un mot de Lukasiewicz si et seulement si $\rho^*(u) = (-1)$.