Piles

Informatique #03

Exercice 1 — les fonctions de base
Dans tout cet exercice, on manipulera des piles à capacité bornée implémentées à l'aide de listes de longueur `c = 10`.

- **Nouvelle pile –** Écrire une fonction `creer_pile()` qui renvoie une pile vide.
- **Pile vide –** Écrire une fonction `est_vide(p)` qui détecte la présence d'une pile vide.
- **Taille –** Écrire une fonction `taille(p)` qui renvoie la taille de la pile `p`.
- **Empiler –** Écrire une fonction `empiler(p, e)` qui empile l'élément `e` dans la liste `p`.
- **Dépiler –** Écrire une fonction `depiler(p)` qui dépile le sommet de la pile `p` et qui renvoie le sommet en question.
- **Égalité entre deux piles –** Écrire une fonction `egalite(p1, p2)` qui retourne `True` si les deux piles sont identiques, `False` sinon.
- **Couper le sommet d'une pile –** Écrire une fonction `decouper(p, i)` qui retourne une nouvelle pile contenant la liste des i-èmes derniers éléments de `p`.
- **Empiler deux piles –** Écrire une fonction `empiler_les_piles(p1, p2)` qui entasse la pile `p2` à la pile `p1`.
- **Échanger deux éléments d'une pile** (♣) **–** Écrire une fonction `echanger(p, i, j)` qui échange les i-ième et j-ième éléments d'une pile.

On manipule désormais des piles à capacité infinie. On s'autorise l'usage des méthodes `append` et `pop` sur les listes.

Exercice 2 — battre les cartes
Écrire une fonction `melanger(p1, p2)` qui mélange les éléments de `p1` et `p2` dans une troisième pile de la façon suivante : tant qu'une pile (au moins) n'est pas vide, on retire aléatoirement un élément au sommet d'une des deux piles et on l'empile sur la pile résultat.

Exercice 3 — conversion base 10 / base 2
Écrire une fonction `conversion(nb10)` qui retourne la représentation en base 2 du nombre `nb10` représenté en base 10 à l'aide de piles.

Exercice 4 — mots bien parenthésés (♣)

1. Écrire une fonction `parenthese(mot)` qui vérifie que la chaîne de caractères `mot`constituée de caractères `(` et `)` est bien parenthésée, à l'aide de piles. Par exemple, '(()())' est bien parenthésée ; '())(' est mal parenthésée.
2. Généraliser la fonction précédente en une fonction `parenthese_g(mot)` qui vérifier le bon parenthésage d'une chaîne de caractères constituée de parenthèses, d'accolades et de crochets. On pourra écrire une fonction intermédiaire `inv(symbole)` qui prend en argument une parenthèse ouvrante, une accolade ouvrante ou un crochet ouvrant et qui retourne le symbole fermant correspondant.

Exercice 5 — évaluer une expression en notation polonaise inversée
Écrire une fonction `eval_polonaise(exp)` qui permet d'évaluer une expression en notation polonaise inversée, ceci à l'aide d'une pile. (cf. les explications au tableau)