codes de Huffman (structures, tableaux, fonctions, ..., niveau 3)

Le but de cet exercice est de réaliser des codes de Huffman de textes. Le principe du code de Huffman (binaire) est assez simple (cf cours ICC): il consiste à regrouper les deux « entités » les moins probables à chaque étape; une « entité » étant soit une lettre de départ, soit un groupe d'entités déjà traitées (une, deux, trois, ... lettres déjà traitées).

Le plus simple est peut être de commencer directement par l'algorithme lui-même à un niveau d'abstraction assez haut (approche descendante) :
à partir de la distribution de probabilité initiale des lettres (leur compte), on va, de proche en proche tant qu'il nous reste plus que deux « entités » à traiter

1. rechercher les deux entités les moins probables ;
2. ajouter le symbole `'0'` devant tous les codes déjà produits pour la première de ces deux entités ;
3. ajouter le symbole `'1'` devant tous les codes déjà produits pour la seconde de ces deux entités ;
4. fusionner ces deux entités en une nouvelle entité, dont on calcule la probabilité : somme des probabilités des deux entités fusionnées;

   pour cela on écrasera une de ces deux entités par leur fusion et on supprimera l'autre.

   Notez qu'on a donc une entité de moins à chaque fois (la fusion supprime les deux entités, mais n'en rajoute qu'une seule nouvelle).

Il nous faut donc représenter ces « entités » (et un ensemble d'entités). Je vous propose de le faire de la façon suivante :

- une « entités » est simplement un groupe de lettres avec sa probabilité ;
- le groupe de lettres peut simplement être représenté par la liste des positions de ces lettres (voir ci-dessous) ;
- au départ, ces entités correspondent simplement aux lettres du mot avec leur probabilité.

Nous aurons aussi besoin de représenter le code lui-même. Un code est une bijection entre les mots à coder et leur code. La bonne structure de données pour représenter cela est ce que l'on appelle des « tables associatives », que nous n'avons pas encore vues (2e semestre). A ce stade du cours, nous vous proposons donc simplement de représenter le code comme un tableau dynamique de structures contenant:

- le mot à coder (type « chaîne de caractères ») ; dans les exemple du cours, c'est simplement une seule lettre, mais on pourrait généraliser ;
- la probabilité correspondante ;
- le mode de code correspondant (type « chaîne de caractères »)).

Prenons un exemple (cf cours ICC) :
supposons que l'on veuille coder la phrase « JE PARS A PARIS »

On commencera par créer un « code » (il n'est pas complet à ce stade, tous ses mots sont vides) contenant par exemple (ordre de lecture ici, mais vous pouvez changer cet ordre, bien sûr):

- `"J"`, probabilité : `1./12.`, mot de code vide;
- `"E"`, probabilité : `1./12.`, ...(mot de code vide);
- `"P"`, probabilité : `2./12.`, ...;
- `"A"`, probabilité : `3./12.`, ...;
- etc.
- `"I"`, probabilité : `1./12.`, mot de code vide.

Une fois ce « code » (mots vides) créé, on pourra créer la version initiale de la liste d'« entités » nécessaires pour l'algorithme décrit plus haut (ordonnée à nouveau ici dans l'ordre de lecture, mais vous pouvez choisir un autre ordre):

- indices : `0`, probabilité : `1./12.`;
  cette entité représente la lettre `"J"`, lettre à la position `0` dans le « code » (mots vides) précédemment créé ;
- indices: `1`, probabilité : `1./12.`;
- indices: `2`, probabilité : `2./12.`;
- etc.
- indices: `7`, probabilité : `1./12.`.

Pour l'instant cette liste d'« entités » ne semble pas très intéressante et semble faire double emploi avec le code : c'est normal : le code de Huffman commence par les lettres seules.

Mais dès la première étape de codage, cela va changer : on commence par regrouper les deux moins probables, disons par exemple le "J" et le "E" (les deux premiers moins probables dans l'ordre de lecture, là encore votre choix peut être différent suivant comment vous écrivez l'algorithme, cela ne change rien au final sur la longueur moyenne du code). On aura alors une nouvelle entité :
indices: `0`, `1`, probabilité : `2./12.`;
et la liste d'« entités » devient (elle ne contient plus que 7 éléments):

- indices: `0`, `1`, probabilité : `2./12.`;
- indices: `2`, probabilité : `2./12.`;
- etc.
- indices: `7`, probabilité : `1./12.`.

Et comme dit au départ de cet exercice : une fois la fusion ci-dessus effectuée (ou même avant), on ajoutera respectivement `'0'` et `'1'` à chacun des mots de code correspondants. Dans cet exemple précis, cela à revient à les ajouter à chacun des deux mots de code, vides, de `"J"` et `"E"`.

Voilà pour les structures de données proposées pour cet algorithme.

Pour l'algorithme lui-même : adoptez (bien sûr !) une démarche très modulaire en introduisant des fonctions pour chacune des tâches élémentaires ; par exemple : compter les lettres, normaliser les probabilités, fusionner 2 entités, rechercher les deux entités les moins probables, ajouter un symbole à un mot de code, construire les entités initiales à partir d'un « code » (mots vides), etc. etc.