Faculté des sciences Jean Perrin **2007/2008**
Master 1 Math Info

TD (Méthode d'adressage dispersé : table de hachage)

On veut implanter les opérations de recherche, d'adjonction, et de suppression de mots dans une table de hachage.
Un mot est une chaîne de caractères. Chaque caractère est codé par son code ASCII. La valeur de hachage est la somme des codes des caractères du mot.

- Ecrire une fonction qui étant donnée un mot m calcule une valeur de hachage associé à ce mot.

On considérera deux types de méthodes d'adressages :

a) Adressage dispersé par chaînage
Les collisions sont résolues par chaînage, en utilisant les listes chaînés.

1. Définir les types de données nécessaire à la mise en œuvre d'une telle table.
2. Ecrire une fonction de hachage h, qui étant donnée un mot m, calcule une entrée dans le tableau.
3. Ecrire les fonction suivantes :
   - *InitHashTable* qui initialise la table de hachage à vide.
   - *InsererMot* qui insère un mot dans une table
   - *rechercherMot* permettant de rechercher un mot dans une table
   - *supprimerMot* : suppression d'un mot de la table

b) Adressage dispersé par continuité : stockage linéaire
Les mots sont stockés dans la table, et les collisions sont résolues par calcul (méthodes directes) grâce à une méthode d'essais successives.

On considère pour cela une table de type :

```
Type état = (vide, occupé, libéré)
      doublet = structure
                    marque : état
                    val : mot
                  Fin structure
     Table = tableau[0..k-1] de doublet
```

- Ecrire une fonction des essais successifs qui étant donnée un mot m et un essai i calcul les valeurs de 0 à m-1 (*essai(m, i)*) . Pour se faire on utilisera le double hachage :
  1. Choisir m un nombre premier
  2. Choisir h1 et h2 avec respectivement k et k-1 classes : h1(x) = x mod k et h2(x) = 1 + (x mod (k-1)) où x est la valeur de hachage associée à un mot m.
  3. Ecrire la fonction essai(x, i) = (h1(x) + i *h2(x)) mod k

- Ecrire une fonction initTableVide qui étant donnée une table l'initialise à vide.
- Ecrire les fonctions permettant de tester si une case du tableau est vide, libérée et occupée.
- Ecrire une fonction permettant de *rechercher* si un mot m est présent dans la table ; retourne l'indice de m si $m \in T$; -1 sinon
- Ecrire une procédure permettant de *supprimer* un mot m de la table et à marquer sa place libérée.
- Ecrire une procédure ajouter un mot dans la table, retourne vrai si le mot est ajouté et faux sinon