UE ALGO5 — TD2 — Séance 5 et 6 : Tables de hachage

On s'intéresse à l'implémentation (en C) d'une table de hachage représentant un ensemble d'éléments de type `T`.

On suppose :
– l'existence d'une fonction de hachage `h` sur le type `T` ;
– que deux éléments de type `T` peuvent être comparé à l'aide de l'opérateur habituel <.

1 Chaînage

On considère que les éléments en collision sont placés dans une liste chaînée.

La table de hachage est déclarée de la manière suivante :

```
typedef T;

typedef struct cell {
    T elem;
    struct cell *suiv;
} Cellule;

typedef Liste * Cellule;

#define TAILLE_TABLE ...

Liste Table[TAILLE_TABLE];
```

Exercice 1. Écrire les primitives permettant d'**insérer**, **rechercher** et **supprimer** un élément dans la table.

2 Adressage ouvert

L'adressage ouvert consiste à placer les éléments en collision non pas dans une liste chaînée mais dans des cases alternatives du tableau. On définit donc une fonction $h'(x, i)$ permettant de trouver une case libre, en essayant successivement $h'(x, 0)$, $h'(x, 1)$, ...

Exercice 2. En supposant h' donnée, implémenter les primitives précédentes avec le principe d'adressage ouvert.

Exercice 3. Donner la définition de h' dans les cas suivants :
– les cases alternatives sont les cases adjacentes à $h(x)$
– on dispose maintenant non plus d'une mais de deux fonctions de hachages différentes h_1 et h_2.