I2010 : langage C (TP10)

Chaînes de caractères: concaténation

Attention : *Le programme de cette séance est similaire à celui de la séance précédente. Cependant, il est suffisamment différent pour que cela vaille la peine de repartir d'un code vierge.*

Vous allez écrire un programme qui lit une suite de chaînes de caractères représentant des contenus de coffres forts. Elles sont structurées de la façon suivante :

1. les cinq premiers caractères représentent l'identifiant du coffre ;
2. les caractères suivants représentent le contenu du coffre.

Ex: AZ3ER Diamant 4K 1000 Euros Doudou d'enfance

L'application maintiendra deux tables qui seront gérées dynamiquement et auxquelles une taille logique et une taille physique seront associées :

1. une table de chaînes de caractères contenant le **nom** des coffres connus et
2. une table de chaînes de caractères contenant le **contenu** des coffres connus.

Le programme fonctionnera de la façon suivante :

- tant que la fin de fichier n'est pas rencontrée :
  - lit un mot d'au plus 254 caractères sur *stdin*,
  - parcourt la table des **noms** afin de regarder si le coffre est déjà contenu (cf. fonction `strncmp`) :
    - si non, il est inséré à la fin des deux tables en mettant à jour la taille logique et éventuellement la taille physique associées aux deux tables.
    - si oui, la partie de la nouvelle chaîne représentant le contenu est insérée dans le coffre existant à l'aide d'une simple concaténation (il peut donc y avoir plusieurs fois le même objet dans le coffre) (cf. function `strcat`).
- ensuite, la liste des coffres ainsi que leur contenu est affiché.

Ex:

Input

```
ZRE43 pomme poire pèche
QS1SD boulon vis clous
az#fg copies d'examen
ZRE43 fraise yaourt pomme
az#fgPAS_MIS_D_ESPACES_EN_EXPRES
```

Output

```
ZRE43 pomme poire pèche fraise yaourt pomme
QS1SD boulon vis clous
az#fg copies d'examenPAS_MIS_D_ESPACES_EN_EXPRES
```

Notez que l'on ne se soucie pas de l'insertion d'espaces pour rendre le contenu plus agréable à lire : on effectue une « bête » concaténation. Par contre, le *return* de fin de ligne est bien retiré.

Pour cet exercice, utilisez la fonction `strncmp` pour comparer uniquement les 5 premiers caractères ; utilisez `strcat` pour concaténer une partie d'une string dans une autre.