**Les dictionnaires**

1. Ecrire une fonction **nbrAlphabet(CH)** qui compte les occurrences de chacune des lettres de l’alphabet dans une chaîne de caractères **CH**. On met le résultat dans un dictionnaire **Nbralphabet**.

**Exemple** : CH = ’les dictionnaires’

**Nbralphabet** = {’a’ : 1, ’c’ : 1, ’b’ : 0, ’e’ : 2, ’d’ : 1, ’g’ : 0, ’f’ : 0, ’i’ : 3, ’h’ : 0,

⇒

’k’ : 0, ’j’ : 0, ’m’ : 0, ’l’ : 1, ’o’ : 1, ’n’ : 2, ’q’ : 0, ’p’ : 0, ’s’ : 1, ’r’ : 1, ’u’ : 0, ’t’ : 1,...}

1. Ecrire une fonction **posAlphabet(CH)** qui analyse une chaîne de caractères **CH**, et mémorise dans un dictionnaire **Posalphabet** l’emplacement exact de chaque caractère. Chaque valeur du dictionnaire **Posalphabet** doit donc être une **liste** d’emplacements. **Exemple** : CH = ’les dictionnaires’

**Posalphabet** = {’a’ : [12], ’c’ : [6], ’e’ : [1, 15], ’d’ : [4], ’i’ : [5, 8, 13], ’l’ : [0], ’o’ :

⇒

[9], ’n’ : [10, 11], ’s’ : [2, 16], ’r’ : [14], ’t’ : [7]}

1. Ecrire une fonction **compterMots(CH)** ayant un argument une chaîne de caractères **CH** et qui renvoie un dictionnaire **Comptermots** qui contient la fréquence de tous les mots de la chaîne **CH**.

**NB** : dans un texte quelconque, les mots ne sont pas seulement séparés par des espaces, mais également par divers signes de ponctuation. Pour simplifier le problème, on peut commencer par remplacer tous les caractères non-alphabétiques par des espaces.

**Exemple** : CH = ’les dictionnaires : sont ...’

⇒ **Comptermots** = {’les’ : 1, ’dictionnaires’ : 1, ’sont’ :1}

1. Ecrire une fonction **construireFr2Ang(Fr, Ang)** qui reçoit en paramètres deux listes de la même taille : **Fr** et **Ang**, et qui retourne un dictionnaire **Fr2Ang**, de tel sorte que dans le dictionnaire **Fr2Ang** l’élément **Fr**[k] est la clé de valeur **Ang**[k] pour tout **k** compris entre 0 et la longueur de Fr et Ang.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Fr** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| ’Un’ | ’Deux’ | ’Trois’ | ’Quatre’ | ’Cinq’ | ’Six’ | ’Sept’ | ’Huit’ | ’Neuf’ | ’Dix’ |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Ang** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| ’One’ | ’Tow’ | ’Three’ | ’Four’ | ’Five’ | ’Six’ | ’Seven’ | ’Eight’ | ’Nine’ | ’Ten’ |

**Fr2Ang** = {’Un’ : ’One’, ’Deux’ : ’Tow’, ’Trois’ : ’Three’, ’Quatre’ : ’Four’, ’Cinq’ :

⇒

’Five’, ’Six’ : ’Six’, ’Sept’ : ’Seven’, ’Huit’ : ’Eight’, ’Neuf’ : ’Nine’, ’Dix’ : ’Ten’}

1. Ecrire une fonction **inverseLangue(Fr2Ang)** qui reçoit en paramètres un dictionnaire **Fr2Ang**, et qui retourne le dictionnaire inverse **Ang2Fr** : (*inverser les clés avec les valeurs*).

**Ang2Fr** = {’One’ : ’Un’, ’Tow’ : ’Deux’, ’Three’ : ’Trois’, ’Four’ : ’Quatre’, ’Cinq’ :

⇒

’Five’, ’Six’ : ’Six’, ’Seven’ : ’Sept’, ’Eight’ : ’Huit’, ’Nine’ : ’Neuf’, ’Ten’ : ’Dix’}

1. Ecrire une fonction **construireAng2Es(Es, Fr2Ang)** qui reçoit en paramètres la liste

**Es** et le dictionnaire **Fr2Ang**, et qui retourne le dictionnaire **Ang2Es**.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Es** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| ’Uno’ | ’Dos’ | ’Tres’ | ’Cuatro’ | ’Cinco’ | ’Seis’ | ’Siete’ | ’Ocho’ | ’Nueve’ | ’Diez’ |

**Ang2Es** = {’One’ : ’Uno’, ’Tow’ : ’Dos’, ’Three’ : ’Tres’, ’Four’ : ’Cuatro’ : ’Five’ :

⇒

’Cinco’, ’Six’ : ’Seis’, ’Seven’ : ’Siete’, ’Eight’ : ’Ocho’, ’Nine’ : ’Nueve’, ’Ten’ : ’Diez’}

1