TD15 : éléments de correction

Exercice : Algorithme du Soundex

```
Fonction : FormaterChaine
Entrée :        - chaine : chaîne de caractères
Local :         - c : caractère
Sortie :        - chaineFormatee : chaîne de caractères
Début
|       Soit chaineFormatee = ""
|       Pour Chaque caractère c dans chaine Faire
|       |       Si EstUneLettreOuUnChiffre(c) est VRAI ET c ≠ ' ' Alors
|       |       |       Soit chaineFormatee = chaineFormatee + c
|       |       Fin Si
|       Fin Pour Chaque
|       Retourner chaineFormatee.EnMajuscules()
Fin
```

```
Fonction : SupprimerLettres (supprime les lettres A, E, I, O, U, Y, H et W)
Entrée :        - chaine : chaîne de caractères
Local :         - c : caractère
Sortie :        - chaineFormatee : chaîne de caractères
Début
|       Soit chaineFormatee = ""
|       Pour Chaque caractère c dans chaine Faire
|       |       Si c ≠ 'A' OU C ≠ 'E' OU c ≠ 'I' OU c ≠ 'O' OU c ≠ 'U' OU c ≠ 'Y' OU c ≠ 'H' OU
|       |       c ≠ 'W' Alors
|       |       |       Soit chaineFormatee = chaineFormatee + c
|       |       Fin Si
|       Fin Pour Chaque
|       Retourner chaineFormatee
Fin
```

```
Fonction : SupprimerRepetitions (supprime les caractères répétés consécutivement)
Entrée :        - chaine : chaîne de caractères
Local :         - c : caractère
                - index : entier
Sortie :        - chaineFormatee : chaîne de caractères
Début
|       Soit chaineFormatee = ""
|       Soit index = 0
|       Tant Que index < chaine.Longueur() Faire
|       |       Soit prev = chaine[index]
|       |       Soit index = index + 1
|       |       // ajouter le caractère (1er occurrence)
|       |       Soit chaineFormatee = chaineFormatee + prev
|       |       // ignorer le caractère jusqu'à en rencontrer un différent
|       |       Tant Que index < chaine.Longueur() ET chaine[index] == prev Faire
|       |       |       Soit index = index + 1
|       |       Fin Tant Que
|       Fin Tant Que
|       Retourner chaineFormatee
Fin
```

```
Fonction : CreerDictionnaireFrancais
Entrée :        void
Local :         /
Sortie :        - dico : dictionnaire (clef : caractère, valeur : entier)
Début
|       Soit dico = dictionnaire (clef : caractère, valeur : entier) vide
|       dico.Ajouter('B', 1)
|       dico.Ajouter('P', 1)
|       dico.Ajouter('C', 2)
|       dico.Ajouter('K', 2)
|       dico.Ajouter('Q', 2)
|       dico.Ajouter('D', 3)
|       dico.Ajouter('T', 3)
|       dico.Ajouter('L', 4)
|       dico.Ajouter('M', 5)
|       dico.Ajouter('N', 5)
|       dico.Ajouter('R', 6)
|       dico.Ajouter('G', 7)
|       dico.Ajouter('J', 7)
|       dico.Ajouter('S', 8)
|       dico.Ajouter('X', 8)
|       dico.Ajouter('Z', 8)
|       dico.Ajouter('F', 9)
|       dico.Ajouter('V', 9)
|       Retourner dico
Fin
```

```
Fonction : CreerDictionnaireAnglais
Entrée :        void
Local :         /
Sortie :        - dico : dictionnaire (clef : caractère, valeur : entier)
Début
|       Soit dico = dictionnaire (clef : caractère, valeur : entier) vide
|       dico.Ajouter('B', 1)
|       dico.Ajouter('F', 1)
|       dico.Ajouter('P', 1)
|       dico.Ajouter('V', 1)
|       dico.Ajouter('C', 2)
|       dico.Ajouter('G', 2)
|       dico.Ajouter('J', 2)
|       dico.Ajouter('K', 2)
|       dico.Ajouter('Q', 2)
|       dico.Ajouter('S', 2)
|       dico.Ajouter('X', 2)
|       dico.Ajouter('Z', 2)
|       dico.Ajouter('D', 3)
|       dico.Ajouter('T', 3)
|       dico.Ajouter('L', 4)
|       dico.Ajouter('M', 5)
|       dico.Ajouter('N', 5)
|       dico.Ajouter('R', 6)
|       Retourner dico
Fin
```

```
Fonction : CalculerCodeSoundexAnglais
Entrée :        - chaine : chaîne de caractères
Local :         - dico : dictionnaire (clef : caractère, valeur : entier)
Sortie :        - codeSoundex : chaîne de caractères
Début
|       Soit dico = CreerDictionnaireAnglais()
|       Soit codeSoundex = ""
|       Pour Chaque caractère c dans chaine Faire
|       |       Soit codeSoundex = codeSoundex + dico[c]
|       Fin Pour Chaque
|       Retourner codeSoundex
Fin
```

```
Fonction : CalculerCodeSoundexFrancais
Entrée :        - chaine : chaîne de caractères
Local :         - dico : dictionnaire (clef : caractère, valeur : entier)
Sortie :        - codeSoundex : chaîne de caractères
Début
|       Soit dico = CreerDictionnaireFrancais()
|       Soit codeSoundex = ""
|       Pour Chaque caractère c dans chaine Faire
|       |       Soit codeSoundex = codeSoundex + dico[c]
|       Fin Pour Chaque
|       Retourner codeSoundex
Fin
```

```
Fonction : CalculerSoundex
Entrée :        - chaine : chaîne de caractères
                - langue : caractère
Local :         ...
Sortie :        - code : chaîne de caractères
Début
|       Si langue.EnMinuscules().Egal("f") OU langue.EnMinuscules().Egal("a") Alors
|       |       Si chaine.Longueur() < 1 Alors
|       |       |       Exception(chaine vide!)
|       |       Fin Si
|       |       // suppression des espaces et mise en majuscules
|       |       Soit chaineFormatee = FormaterChaine(chaine)
|       |       Soit premiereLettre = chaineFormatee[0]
|       |       Soit autresLettres = chaineFormatee.SousChaine(1, chaineFormatee.Longueur() - 1)
|       |       // Suppression des lettres A, E, I, O, U, Y, H et W
|       |       Soit chaineAvecSuppressions = SupprimerLettres(autresLettres)
|       |       // obtention des chiffres du soundex à l'aide du dictionnaire
|       |       Soit chiffres = ""
|       |       Si langue.EnMinuscules().Egal("f") Alors
|       |       |       Soit chiffres = CalculerCodeSoundexFrancais(chaineAvecSuppressions)
|       |       Sinon Si langue.EnMinuscules().Egal("a") Alors
|       |       |       Soit chiffres = CalculerCodeSoundexAnglais(chaineAvevSuppressions)
|       |       Fin Si
|       |       // supprimer les caractères répétés consécutivement
|       |       Soit chiffresSansRepetition = SupprimerRepetitions(chiffres)
|       |       // code final = première lettre + trois premiers chiffres sans répétitions
|       |       Soit code = premiereLettre + chiffresSansRepetition
|       |       Si code.Longueur() == 4 Alors
|       |       |       Retourner code
|       |       Sinon Si code.Longueur() > 4 Alors
|       |       |       Retourner code.SousChaine(0, 4)
|       |       Sinon
|       |       |       Retourner code.PaddingDroite(4, ' ')
|       |       Fin Si
|       Sinon
|       |       Exception(langue non supportée !)
|       Fin Si
Fin
```

```
Fonction : AfficherCorrespondances
Entrée :        - noms : liste de chaînes de caractères
                - nom : chaîne de caractères
                - langue : chaîne de caractères
Local :         - n : chaîne de caractères
                - codeNoms, codeN : chaîne de caractères
Sortie :        void
Début
|       Soit codeNom = CalculerSoundex(nom, langue)
|       Afficher("Liste des correspondances pour le nom {nom} avec la langue {langue} :\n")
|       Pour Chaque chaîne de caractères n dans noms Faire
|       |       Soit codeN = CalculerSoundex(n, langue)
|       |       Si codeN.Egal(codeNom) est VRAI Alors
|       |       |       Afficher("{n}\n")
|       |       Fin Si
|       Fin Pour Chaque
Fin


Fonction : AfficherCodes
Entrée :        - noms : liste de chaînes de caractères
Local :         - n : chaîne de caractères
                - codeN : chaîne de caractères
Sortie :        void
Début
|       Afficher("Nom – code Français – code Anglais\n")
|       Pour Chaque chaîne de caractères n dans noms Faire
|       |       Soit codeF = CalculerSoundex(n, "f")
|       |       Soit codeA = CalulerSoundex(n, "a")
|       |       Afficher("{n} - {codeF} - {codeA}\n")
|       Fin Pour Chaque
Fin


Procédure : Main
Entrée :        void
Local :         - quitter : booléen
                - choix : entier
                - nom : chaîne de caractères
                - noms : liste de chaînes de caractères
Sortie :        void
Début
|       Soit noms = liste de chaînes de caractères initialisée avec des noms
|       Soit quitter = FAUX
|       Tant Que quitter est FAUX Faire
|       |       Afficher("Menu\n1 – Chercher un nom\n2 – Lister les codes (Fr/En)\n3 – Quitter\n")
|       |       Soit choix = 0
|       |       Tant Que choix ≠ 1 ET choix ≠ 2 ET choix ≠ 3 Faire
|       |       |       Soit choix = LireEntier()
|       |       Fin Tant Que
|       |       Si choix == 1 Alors
|       |       |       Afficher("Nom à chercher : ")
|       |       |       Soit nom = LireChaine()
|       |       |       Afficher("Correspondances en Français :\n")
|       |       |       AfficherCorrespondances(noms, nom, "f")
|       |       |       Afficher("Correspondances en Anglais :\n")
|       |       |       AfficherCorrespondances(noms, nom, "a")
|       |       Sinon Si choix == 2 Alors
|       |       |       AfficherCodes(noms)
|       |       Sinon Si choix == 3 Alors
|       |       |       Soit quitter = VRAI
|       |       Fin Si
|       Fin Tant Que
Fin
```