TD7 : éléments de correction

Exercice 1

```
Fonction : CaractereADroite
Précondition : c appartient à l'ensemble {'A', ... , 'Z', ' '}
Entrée :        - c : caractère
Local :         /
Sortie :        - droite : caractère
Début
|       Soit droite = c
|       Si c == ' ' Alors
|       |       Soit droite = 'A' // retour au début de l'ensemble
|       Sinon Si c == 'Z' Alors
|       |       Soit droite = ' '
|       Sinon
|       |       Soit droite = c + 1 // en C# : droite = (char) ((int) c + 1)
|       Fin Si
|       Retourner droite
Fin
```

```
Fonction : CaractereAGauche
Précondition : c appartient à l'ensemble {'A', ... , 'Z', ' '}
Entrée :        - c : caractère
Local :         /
Sortie :        - gauche : caractère
Début
|       Soit gauche = c
|       Si c == 'A' Alors
|       |       Soit gauche = ' ' // retour à la fin de l'ensemble
|       Sinon Si c == ' ' Alors
|       |       Soit gauche = 'Z'
|       Sinon
|       |       Soit gauche = c - 1 // en C# : gauche = (char) ((int) c - 1)
|       Fin Si
|       Retourner gauche
Fin
```

```
Fonction : DecalerCaractere
Précondition : c appartient à l'ensemble {'A', ... , 'Z', ' '}
Entrée :        - c : caractère
                - offset : entier
Local :         - i : entier
Sortie :        - caractereDecale : caractère
Début
|       Soit caractereDecale = c
|       // décalage vers la droite
|       Si offset > 0 Alors
|       |       Pour i de 1 à offset avec un pas de +1 Faire
|       |       |       Soit caractereDecale = CaractereADroite(c)
|       |       Fin Pour
|       // décalage vers la gauche
|       Sinon Si offset < 0 Alors
|       |       Pour i de 1 à -offset avec un pas de +1 Faire
|       |       |       Soit caractereDecale = CaractereAGauche(c)
|       |       Fin Pour
|       Fin Si
|       Retourner caractereDecale
Fin
```

```
Fonction : DecalerCaracteres
Précondition : la chaîne ne contient que des caractères appartenant à l'ensemble {'A', ... , 'Z', ' '}
Entrée :        - chaine : chaîne de caractères
                - longueur : entier
                - offset : entier
Local :         - i : entier
Sortie :        - chaineDecale : chaîne de caractères
Début
|       Soit chaineDecalee = ""
|       Pour i de 0 à longueur - 1 avec un pas de +1 Faire
|       |       Soit chaineDecalee = chaineDecalee + DecalerCaractere(chaine[i], offset)
|       Fin Pour
|       Retourner chaineDecalee
Fin


Fonction : CodeDeCesar
Précondition : la chaîne ne contient que des caractères appartenant à l'ensemble {'A', ... , 'Z', ' '}
Entrée :        - message : chaîne de caractères
                - longueur : entier
                - clef : entier
                - coder : booléen          // indique s'il faut coder ou décoder la chaîne
Local :         - i : entier
Sortie :        - messageCode : chaîne de caractères
Début
|       Soit messageCode = ""
|       Si coder == vrai Alors
|       |       Soit messageCode = DecalerCaracteres(message, clef)
|       Sinon
|       |       Soit messageCode = DecalerCaracteres(message, -clef)
|       Fin Si
|       Retourner messageCode
Fin


Procédure : Main
Entrée :        void
Local :         - messageU : chaîne de caractères
                - longueur : entier
                - chefChiffrement : entier
                - messageChiffre : chaîne de caractères
                - messageDechiffre : chaîne de caractères
Sortie :        void
Début
|       Soit messageU, longueur = SaisirChaîne()
|       Soit clefChiffrement = -1
|       Tant Que clefChiffrement ≤ 0 Faire
|       |       Soit clefChiffrement = SaisirEntier()
|       Fin Tant Que
|       Soit messageChiffre = CodeDeCesar(messageU, longueur, clefChiffrement, vrai)
|       AfficherChaîne(messageChiffre)
|       Soit messageDechiffre = CodeDeCesar(messageChiffre, longueur, clefChiffrement, faux)
|       AfficherChaine(messageDechiffre)
Fin
```