TD IV :
.NET

a) Télécharger et ajouter "TD4.cs" au projet TD_console

b) Ajouter les lignes suivantes dans Main() de "Program.cs"

```
TD4 tD4 = new TD4() ;
TD4.Test() ;
```

c) Compléter la méthode "River_unique" qui retourne les rivières uniques entre 1 inclus et le paramètre inclus. Essayer avec 100 et afficher "1 ; 3 ; 5 ; 7 ; 9 ; 20 ; 31 ; 42 ; 53 ; 64 ; 75 ; 86 ; 97."

- Exemple : la rivière est unique parce que 20 n'est la séquence d'aucun nombre avant lui.

d) Compléter la méthode "Encore_RL" qui retourne le paramètre encodé selon les règles suivante :

- A chaque itération, on prend i + 1 caractère qu'on place à droite puis à gauche.
- Exemple :
  - Pour "abcdefghijk" on prend à chaque itération "a", puis "bc", puis "def", puis "ghij", et le reste "k".
  - On place "a" à droite, "a" est la première itération donc on obtient "a".
  - On place "bc" à gauche, on possède "a" qui devient "bca".
  - On place "def" à droite, on possède "bca" qui devient "bcadef".
  - On continu jusqu'à obtenir "ghijbcadefk".

e) Compléter la méthode "Decode_RL" qui retourne le paramètre décodé selon les mêmes règles que Encore_RL. Essayer avec "ghijbcadefk" et afficher "abcdefghijk".

f) Compléter la méthode "Fibonacci_is_sequence" qui retourne si le paramètre est une séquence de la suite de Fibonacci. Essayer avec 8 et afficher true.

- La suite de Fibonacci est la séquence suivante : 0 ; 1 ; 1 ; 2 ; 3 ; 5 ; 8 ; 13 ; 21 ; 34 ; 55 ; 89 ; etc.
- Sauf pour les deux premiers cas, chaque élément est la sommes des deux éléments avant lui.

g) Compléter la méthode "Fibonacci_previous" qui retourne l'élément précédent de la suite Fibonacci du paramètre. Essayer avec 8 et afficher 5.

h) Compléter la méthode "Fibonacci_next" qui retourne l'élément suivant de la suite Fibonacci du paramètre. Essayer avec 8 et afficher 13.

i) Compléter la méthode "Fibonacci_max_sequence" qui retourne la suite de Fibonacci jusqu'à être supérieur ou égal au paramètre. Essayer avec 20 et afficher "0 ; 1 ; 1 ; 2 ; 3 ; 5 ; 8 ; 13 ; 21."

j) Compléter la méthode "Fibonacci_n_sequence" qui retourne un nombre de séquence de la suite de Fibonacci égal au paramètre. Essayer avec 5 et afficher "0 ; 1 ; 1 ; 2 ; 3."

k) Compléter la méthode "Fibonacci_delimiter" qui retourne les éléments du premier paramètre au second paramètre de la séquence de la suite de Fibonacci. Essayer avec 3 et 6 et afficher "1 ; 2 ; 3 ; 5."

l) Compléter la méthode "Conway_previous" qui retourne l'élément précédent de la suite de Conway du paramètre. Essayer avec "111221" et afficher "1211".

m) Compléter la méthode "Syracuse_is_sequence" qui retourne si le second paramètre est une séquence de la suite de la conjecture de Syracuse du premier paramètre. Essayer avec 14 et 54 et afficher false.

- La suite de la conjecture de Syracuse de 14 est la séquence suivante : 14 ; 7 ; 22 ; 11 ; 34 ; 17 ; 52 ; 26 ; 13 ; 40 ; 20 ; 10 ; 5 ; 16 ; 8 ; 4 ; 2 ; 1.
- Si l'élément est pair, on le divise par 2. Si l'élément est impair, on le multiplie par 3 et on ajoute 1. La suite s'arrête si on obtient 1.

n) Compléter la méthode "Syracuse_previous" qui retourne les éléments possibles précédents de la suite de la conjecture de Syracuse du paramètre. Essayer avec 22 et afficher "7 ; 44".

o) Compléter la méthode "Syracuse_next" qui retourne l'élément suivant de la suite de la conjecture de Syracuse du paramètre. Essayer avec 42 et afficher 21.

p) Compléter la méthode "Syracuse_sequence" qui retourne la suite de la conjecture de Syracuse du paramètre. Essayer avec 20 et afficher "20 ; 10 ; 5 ; 16 ; 8 ; 4 ; 2 ; 1."

q) Compléter la méthode "Syracuse_n_sequence" qui retourne un nombre de séquence égal au second paramètre de la suite de la conjecture de Syracuse du premier paramètre. Essayer avec 14 et 5 afficher "14 ; 7 ; 22 ; 11 ; 34."

r) Compléter la méthode "Syracuse_delimiter" qui retourne les éléments du second paramètre au troisième paramètre de la séquence de la suite de la conjecture de Syracuse du premier paramètre. Essayer avec 14, 3 et 6 et afficher "22 ; 11 ; 34 ; 17."

s) Compléter la méthode "Morse_to_morse" qui retourne le paramètre encodé en Morse. Essayer avec "a" et afficher ". _".

- Utiliser des points et des underscores (_).
- Un point se dit "ti" et un underscore se dit "taah", un "taah" vaut 3 "ti"
- L'espacement entre les "ti" et les "taah" d'une lettre à la longueur d'un "ti" et est représenter par 1 espace.
- Morse_to_morse prend un char en paramètre.
- Encoder lettres (a à z), chiffres (0 à 9) et symboles (. , ? ' ! / ( ) & : ; = + - _ " $ @).

t) Compléter la méthode "Morse_to_char" qui retourne le paramètre décodé du Morse. Essayer avec ". _" et afficher "a".

- Morse_to_char retourne un char.

u) Compléter la méthode "Morse_encode" qui retourne la séquence en paramètre encodé en Morse. Essayer avec "Hello World !" et afficher ". . . . . . _ . . . _ . . _ _ _ / . _ _ _ _ _ . _ . . _ . . _ . . / _ _ _ .".

- L'espacement entre les lettres d'un mot à la longueur d'un "taah" et est représenter par 3 espaces.
- L'espacement entre les mots est représenter par un slash (/) délimité par des espaces : " / ".
- Le point d'exclamation à 2 variantes, on utiliseras la version américaine : "_ _ _ .".

v) Compléter la méthode "Morse_decode" qui retourne la séquence en paramètre décodé du Morse. Essayer avec ". . . . . . _ . . . _ . . _ _ _ / . _ _ _ _ _ . _ . . _ . . _ . . / _ _ _ ." et afficher "Hello World !".