

Faire une solution Visual Studio avec les 5 projets suivants (zipped)

Exercice 1 struct

Ecrire une *application console* avec

- La définition d'un type structure Point3D avec les champs réels X, Y et Z, un constructeur avec 3 paramètres et une méthode DistanceToOrigin() qui renvoie un réel
- Une méthode SwapPoints, qui échange les coordonnées de deux points passés en paramètre
- Des instructions pour instancier deux Point3D distincts, afficher leurs coordonnées et distance à l'origine, puis les échanger et les afficher à nouveau

Exercice 2 System.DateTime, boucles, System.Math

Ecrire une *application console* qui demande à l'utilisateur d'introduire un nombre réel positif A, puis calcule sa racine carrée par approximations successives, en utilisant la relation de récurrence suivante: $X_{j+1} = (X_j + A / X_j) / 2$ avec $X_1 = A$.

Les approximations successives seront affichées dans la console

comme suit:

```
approximation de la racine carrée de 8.5 est 4.75
approximation de la racine carrée de 8.5 est 3.27
```

Le processus continuera dans une boucle sans fin, jusqu'à ce que l'erreur passe en dessous de $A \cdot 10^{-9}$

Un récapitulatif affichera alors

- le nombre d'itérations effectuées
- le temps écoulé (en secondes)
- l'erreur résiduelle (par comparaison avec la fonction de la classe Math.Sqrt)

Exercice 3 System.IO

Réaliser une application console, qui à l'aide d'une classe FileStream ou StreamWriter, récupère les données du fichier « mesures.txt » et les affiche à l'écran à raison de 10 par ligne :

```
file:///P:/Formation/010-Bachelor/030-Niveau-3/030-Professeur:
```

```
295, 92, 100, 306, 310, 119, 378, 102, 383, 115
199, 350, 167, 167, 449, 350, 143, 143, 350, 143
143, 350, 350, 143, 143, 350, 143, 143, 350, 350
143, 143, 350, 143, 143, 350, 143, 143, 350, 350
```

Note : Essayer de faire un affichage « Verbatim » (@) et une string interpolation (C#6 avec \$)

Exercice 4 Tableaux

Ecrire un programme qui sépare un tableau de 20 valeurs entières aléatoires (0-99) en deux tableaux, un avec les valeurs paires, l'autre avec les valeurs impaires. Le tableau sera généré dans le programme principal. Factoriser la séparation dans une méthode statique PairImpair qui accepte un tableau à une dimension en paramètre. L'affichage des deux tableaux partiels créés par PairImpair sera réalisé dans le Main.

Main()

PairImpair()

```
Valeurs à séparer
1 17 24 90 8 65 73 49 26 4 55 74 32 67 6 38 93 80 20 46
(appel de PairImpair)
Pair : 24 90 8 26 4 74 32 6 38 80 20 46
Impair: 1 17 65 73 49 55 67 93
```

Exercice 5 classe String

Comparer les 3 chaînes suivantes

```
string s1 = "Hello World" ;
string s2 = "Hello World" ;
string s3 = s1 ;
```

avec les 3 méthodes suivantes : .Equals, .CompareTo et System.ReferenceEquals puis modifier s3 :

```
s3 += '!' ;
```

et refaire les 3 comparaisons. **Que concluez-vous ?**