CÉGEP STE-FOY – INTRODUCTION À LA PROGRAMMATION

EXERCICE 18B – TABLEAUX 1D – PARTIE 3

Préparé par Benjamin Lemelin et Pierre Poulin le 22 février 2023

1 Travail à effectuer

Ouvrez le projet fourni avec cet énoncé. Il devrait déjà contenir un fichier **`Library.cs`**. Il contient aussi des tests unitaires pour chaque fonction à créer. Ces tests sont en commentaires actuellement. Il ne vous restera qu'à les décommenter lorsque nécessaire et à les compléter selon les directives ci-dessous.

1.1 Restreindre des valeurs

Créez une fonction nommée **`Clamp`** dont l'objectif est de s'assurer que les valeurs d'un tableau reçu en paramètre se situent entre deux bornes (minimum et maximum inclusivement).

Par exemple, supposons le tableau suivant :

0	1	2	3	4	5
25	22	56	78	99	45

Nous désirons que les valeurs de ce tableau soient entre 25 et 75. Le tableau qui en résulte est donc :

0	1	2	3	4	5
25	25	56	75	75	45

Cette fois, votre fonction doit modifier le tableau original, mais pas en créer un autre. Vous pouvez assumer que la valeur minimale est plus petite ou égal à la valeur maximale.

Vous devez finalement vous assurer que le tableau reçu en entrée est non **`null`** et lancer une exception sinon.

1.2 Insérer dans un tableau

Créez une fonction nommée **`Insert`** permettant d'insérer un entier dans un tableau. Pour ce faire, elle créé un nouveau tableau incluant l'entier à insérer et le retourne. Elle nécessite trois paramètres : le tableau où insérer l'entier, l'index ou placer l'entier et l'entier à insérer.

Par exemple, si nous désirons insérer le nombre 33 à la position 4 du tableau suivant :

0	1	2	3	4	5
25	22	56	78	99	45

La fonction retournera ce nouveau tableau :

0	1	2	3	4	5	6
25	22	56	78	**33**	99	45

Faits importants à noter :

- L'insertion remplace l'élément à la position indiquée, et l'élément qui était à cette position est tout simplement poussé une case plus loin, de même que les cases subséquentes.
- Il est possible d'insérer à la fin du tableau si la position d'insertion est égale à la taille du tableau.
- Vous devez vous assurer que la position fournie est valide et lancer une exception sinon.
- Vous devez finalement vous assurer que le tableau reçu en entrée est non **`null`** et lancer une exception sinon.

1.3 Dédupliquer

Créez une fonction nommée **`Dedup`** permettant de retirer tous les doublons d'un tableau. Elle prend en paramètre un tableau et retourne un nouveau tableau sans les doublons. Notez qu'au moins une valeur de chaque doublon doit rester.

Par exemple, avec le tableau suivant:

0	1	2	3	4	5
4	2	4	2	1	3

La fonction retournera ce nouveau tableau :

0	1	2	3
4	2	1	3

Ce numéro est plus difficile et nécessitera que vous réfléchissiez à une stratégie avant de commencer. Il est préférable de découper ce problème en des problèmes plus petits.

Vous aurez aussi peut-être besoin de fonctions que vous avez codées dans les exercices précédents.

Vous devez vous assurer que le tableau reçu en entrée est non **`null`** et lancer une exception sinon.

2 Modalités de remise

Remettez votre projet Visual Studio sur LÉA, dans la section travaux, à l'intérieur d'une archive *Zip*. Supprimez tous les dossiers temporaires, à savoir les dossiers **`.vs`**, **`TestResults`**, **`bin`** et **`obj`**.