Class SimplesTableau

java.lang.Object
    SimplesTableau

```
public class SimplesTableau
extends java.lang.Object
```

Cette classe effectue des opérations élémentaires sur un ou plusieurs tableaux d'entiers. La taille d'un tableau est par définition le nombre TOTAL de cases = tab.length. Un tableau d'entiers créé possède nbElem élements qui est nécessairement inférieur ou égal à la taille du tableau : nbElem <= tab.length (= taille).

Author:

J-F. Kamp - octobre 2016

Constructor Summary

Constructors

Constructor and Description
`SimplesTableau()`

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method and Description
(package private) void	`afficherTab(int[] leTab, int nbElem)` Affiche le contenu des nbElem cases d'un tableau une par une.
(package private) void	`afficherTabLgn(int[] leTab, int nbElem, int nbLgn)` Affiche le contenu d'un tableau case par case et par ligne de nbLgn éléments.
(package private) int[]	`copier(int[] tabToCopy, int nbElem)` Renvoie la copie exacte (clone) du tableau passé en paramètre.
(package private) void	`decalerGche(int[] leTab, int nbElem, int ind)` Décale de une case de la droite vers la gauche toutes les cases d'un tableau à partir d'un indice "ind" et jusque nbElem-1 ([ind]<-[ind+1]<-[ind+2]<-...
(package private) void	`echange(int[] leTab, int nbElem, int ind1, int ind2)` Echange les contenus des cases du tableau passé en paramètre, cases identifiées par les indices ind1 et ind2.
(package private) boolean	`egalite(int[] tab1, int[] tab2, int nbElem1, int nbElem2)` Renvoie vrai si les 2 tableaux passés en paramètre sont exactement les mêmes en nombre d'éléments et en contenu (case par case).
(package private) boolean	`inclusion(int[] tab1, int[] tab2, int nbElem1, int nbElem2)` Renvoie vrai ssi le tableau tab1 est inclus dans tab2.
(package private) int[]	`inverse(int[] leTab, int nbElem)` Renvoie un nouveau tableau qui est l'inverse de celui passé en paramètre.
(package private) int	`leMax(int[] leTab, int nbElem)` Renvoie le maximum parmi les éléments du tableau.
(package private) int	`leMin(int[] leTab, int nbElem)` Renvoie le minimum parmi les éléments du tableau.
(package private) int[]	`melange(int[] leTab, int nbElem)` Retourne un nouveau tableau qui a la même taille et les mêmes occurrences d'élements que le tableau passé en paramètre mais ces éléments sont répartis selon des indices aléatoires (0 <= indice <= nbElem-1).
(package private) int	`nbOccurrences(int[] leTab, int nbElem, int elem)` Renvoie le nombre d'occurrences d'un entier dans un tableau.

(package private) void	**principal**() Le point d'entrée du programme.
(package private) void	**remplirAleatoire**(int[] leTab, int nbElem, int min, int max) A partir d'un tableau créé, remplit aléatoirement le tableau de nbElem valeurs comprises entre min et max.
(package private) void	**saisir**(int[] leTab, int nbElem) Saisie de nbElem valeurs dans un tableau par boite de dialogue.
(package private) int	**supprimerUneValeur**(int[] leTab, int nbElem, int valeur) Supprime du tableau la première case rencontrée dont le contenu est égale à "valeur".
(package private) void	**testAfficherTab**() Test de la méthode afficherTab
(package private) void	**testCopier**() Test de la méthode copier
(package private) void	**testDecalerGche**() Test de la méthode decalerGche
(package private) void	**testEchange**() Test de la méthode echange
(package private) void	**testEgalite**() Test de la méthode egalite
(package private) void	**testInclusion**() Test de la méthode inclusion
(package private) void	**testInverse**() Test de la méthode inverse
(package private) void	**testLeMinEtLeMax**() Test des méthodes leMin et leMax
(package private) void	**testMelange**() Test de la méthode melange
(package private) void	**testNbOccurrences**() Test de la méthode nbOccurrences
(package private) void	**testRemplirAleatoire**() Test de la méthode remplirAleat
(package private) void	**testSaisirEtAfficherTabLgn**() Test des méthodes saisir et afficherTabLgn
(package private) void	**testSupprimerUneValeur**() Test de la méthode supprimerUneValeur
(package private) void	**testTirerAleatoire**() Test de la méthode tirerAleatoire
(package private) int	**tirerAleatoire**(int min, int max) Renvoie un entier aléatoire compris entre min et max (min <= valeur <= max).

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

SimplesTableau

```
public SimplesTableau()
```

Method Detail

principal

```
void principal()
```

Le point d'entrée du programme.

testAfficherTab

```
void testAfficherTab()
```

Test de la méthode afficherTab

testSaisirEtAfficherTabLgn

```
void testSaisirEtAfficherTabLgn()
```

Test des méthodes saisir et afficherTabLgn

testTirerAleatoire

```
void testTirerAleatoire()
```

Test de la méthode tirerAleatoire

testRemplirAleatoire

```
void testRemplirAleatoire()
```

Test de la méthode remplirAleat

testEgalite

```
void testEgalite()
```

Test de la méthode egalite

testCopier

```
void testCopier()
```

Test de la méthode copier

testNbOccurrences

```
void testNbOccurrences()
```

Test de la méthode nbOccurrences

testLeMinEtLeMax

```
void testLeMinEtLeMax()
```

Test des méthodes leMin et leMax

testInverse

```
void testInverse()
```

Test de la méthode inverse

testEchange

```
void testEchange()
```

Test de la méthode echange

testMelange

```
void testMelange()
```

Test de la méthode melange

testDecalerGche

```
void testDecalerGche()
```

Test de la méthode decalerGche

testSupprimerUneValeur

```
void testSupprimerUneValeur()
```

Test de la méthode supprimerUneValeur

testInclusion

```
void testInclusion()
```

Test de la méthode inclusion

afficherTab

```
void afficherTab(int[] leTab,
                 int nbElem)
```

Affiche le contenu des nbElem cases d'un tableau une par une. Tenir compte du cas particulier où le tableau n'est pas créé.

Parameters:

`leTab - le tableau à afficher`

`nbElem - le nombre d'entiers que contient le tableau`

afficherTabLgn

```
void afficherTabLgn(int[] leTab,
                    int nbElem,
                    int nbLgn)
```

Affiche le contenu d'un tableau case par case et par ligne de nbLgn éléments. Tenir compte du cas particulier où le tableau n'est pas créé.

Parameters:

`nbLgn - le nombre d'éléments par ligne`

`leTab - le tableau à afficher`

`nbElem - le nombre d'entiers que contient le tableau`

saisir

```
void saisir(int[] leTab,
            int nbElem)
```

Saisie de nbElem valeurs dans un tableau par boite de dialogue. Tenir compte du cas particulier où le tableau n'est pas créé. Vérifier que nbElem <= taille avant de commencer la saisie sinon afficher un message d'erreur.

Parameters:

`leTab - le tableau à remplir par saisies successives de l'utilisateur`

`nbElem - le nombre d'entiers que le tableau contiendra réellement (<= taille)`

egalite

```
boolean egalite(int[] tab1,
                int[] tab2,
                int nbElem1,
                int nbElem2)
```

Renvoie vrai si les 2 tableaux passés en paramètre sont exactement les mêmes en nombre d'éléments et en contenu (case par case). Vérifier que les 2 tableaux sont créés sinon afficher un message d'erreur.

Parameters:

`tab1 - le 1er tableau à comparer`

`tab2 - le 2ème tableau à comparer`

`nbElem1 - le nombre d'entiers présents dans le 1er tableau`

`nbElem2 - le nombre d'entiers présents dans le 2ème tableau`

Returns:

true si égalité parfaite sinon false

remplirAleatoire

```
void remplirAleatoire(int[] leTab,
                      int nbElem,
                      int min,
                      int max)
```

A partir d'un tableau créé, remplit aléatoirement le tableau de nbElem valeurs comprises entre min et max. Tenir compte du cas particulier où le tableau n'est pas créé. Vérifier que nbElem <= taille sinon afficher une erreur. Vérifier que min <= max, sinon afficher une erreur.
Utiliser obligatoirement la méthode "int tirerAleatoire (int min, int max)".

Parameters:

leTab - le tableau à remplir de valeurs tirées aléatoirement

nbElem - le nombre d'entiers que contiendra le tableau

min - la valeur de l'entier minimum

max - la valeur de l'entier maximum

tirerAleatoire

```
int tirerAleatoire(int min,
                   int max)
```

Renvoie un entier aléatoire compris entre min et max (min <= valeur <= max).

Parameters:

min - la valeur de l'entier minimum

max - la valeur de l'entier maximum

Returns:

l'entier aléatoire

copier

```
int[] copier(int[] tabToCopy,
             int nbElem)
```

Renvoie la copie exacte (clone) du tableau passé en paramètre.

Parameters:

tabToCopy - le tableau à copier

nbElem - le nombre d'entiers présents dans le tableau

Returns:

le nouveau tableau qui est la copie du tableau passé en paramètre

nbOccurrences

```
int nbOccurrences(int[] leTab,
                  int nbElem,
                  int elem)
```

Renvoie le nombre d'occurrences d'un entier dans un tableau.

Parameters:

leTab - le tableau

nbElem - le nombre d'entiers présents dans le tableau

elem - l'entier à rechercher dans le tableau

Returns:

le nombre d'occurrences

leMin

```
int leMin(int[] leTab,
```

```
          int nbElem)
```

Renvoie le minimum parmi les éléments du tableau.

Parameters:

`leTab - le tableau`

`nbElem - le nombre d'entiers présents dans le tableau`

Returns:

`le minimum des éléments du tableau`

leMax

```
int leMax(int[] leTab,
          int nbElem)
```

Renvoie le maximum parmi les éléments du tableau.

Parameters:

`leTab - le tableau`

`nbElem - le nombre d'entiers présents dans le tableau`

Returns:

`le maximum des éléments du tableau`

inverse

```
int[] inverse(int[] leTab,
              int nbElem)
```

Renvoie un nouveau tableau qui est l'inverse de celui passé en paramètre. Son jème élément est égal au (nbElem+1-j) élément du tableau initial (j=1 signifie premier élément du tableau).

Parameters:

`leTab - le tableau`

`nbElem - le nombre d'entiers présents dans le tableau`

Returns:

`le nouveau tableau qui est l'inverse de leTab sur la plage (0...nbElem-1)`

echange

```
void echange(int[] leTab,
             int nbElem,
             int ind1,
             int ind2)
```

Echange les contenus des cases du tableau passé en paramètre, cases identifiées par les indices ind1 et ind2. Vérifier que les indices ind1 et ind2 sont bien compris entre zéro et (nbElem-1), sinon afficher un message d'erreur.

Parameters:

`leTab - le tableau`

`nbElem - le nombre d'entiers présents dans le tableau`

`ind1 - numéro de la première case à échanger`

`ind2 - numéro de la deuxième case à échanger`

melange

```
int[] melange(int[] leTab,
              int nbElem)
```

Retourne un nouveau tableau qui a la même taille et les mêmes occurrences d'élements que le tableau passé en paramètre mais ces éléments sont répartis selon des indices aléatoires (0 <= indice <= nbElem-1). Une technique simple consiste à utiliser les méthodes "echange" et "tirerAleatoire" pour effectuer le mélange.

Parameters:

`leTab - le tableau`

`nbElem - le nombre d'entiers présents dans le tableau`

Returns:

`le nouveau tableau qui a le même contenu que le tableau initial mais mélangé`

decalerGche

```
void decalerGche(int[] leTab,
                 int nbElem,
                 int ind)
```

Décale de une case de la droite vers la gauche toutes les cases d'un tableau à partir d'un indice "ind" et jusque nbElem-1 ([ind]<-[ind+1]<-[ind+2]<-...<-[nbElem-2]<-[nbElem-1]). Vérifier que ind est compris entre 0 et (nbElem-2) sinon afficher une erreur.

Parameters:

`leTab - le tableau`

`nbElem - le nombre d'entiers présents dans le tableau`

`ind - l'indice à partir duquel commence le décalage à gauche`

supprimerUneValeur

```
int supprimerUneValeur(int[] leTab,
                       int nbElem,
                       int valeur)
```

Supprime du tableau la première case rencontrée dont le contenu est égale à "valeur". La case du tableau est supprimée par décalage à gauche des cases du tableau. L'appel de la méthode "decalerGche" est obligatoire. A l'issue de la suppression (si elle existe) le nombre d'éléments dans le tableau est décrémenté et retourné.

Parameters:

`leTab - le tableau`

`nbElem - le nombre d'entiers présents dans le tableau`

`valeur - le contenu de la première case à supprimer`

Returns:

`le nombre d'éléments dans le tableau (éventuellement inchangé)`

inclusion

```
boolean inclusion(int[] tab1,
                  int[] tab2,
                  int nbElem1,
                  int nbElem2)
```

Renvoie vrai ssi le tableau tab1 est inclus dans tab2. Autrement dit, si tous les éléments de tab1 se retrouvent intégralement dans tab2 (y compris les doublons) mais pas nécessairement dans le même ordre. L'utilisation de méthodes déjà écrites est autorisé.

Parameters:

`tab1 - le premier tableau`

`tab2 - le deuxième tableau`

`nbElem1 - le nombre d'entiers présents dans le tableau1`

`nbElem2 - le nombre d'entiers présents dans le tableau2`

Returns:

`vrai ssi tableau1 est inclus dans tableau2`