MOOC Init Prog Java
Exercices semaine 4

Exercice 12 : produits scalaire (Tableaux)

Écrivez un programme `Scalaire.java` qui calcule le produit scalaire de deux vecteurs, **implémentés au moyen de tableaux unidimensionnels**. Votre programme devra utiliser (entre autre) les éléments suivants :

- Déclarations dans la méthode `main()` :
  - une variable `nMax` représentant la taille maximale des vecteurs (inutile de lui donner une valeur trop élevée... 10 suffit amplement)
  - deux variables `v1` et `v2`, de type tableau de réels de taille `nMax`.
- Méthode :
  - demander à l'utilisateur d'entrer `n`, la taille effective des vecteurs.
  - vérifier que `n` est compris entre 1 et `nMax` et demander à l'utilisateur d'entrer à nouveau une valeur tant que ce n'est pas le cas
  - demander à l'utilisateur les composantes ($v1_0 \ldots v1_{n-1}$, $v2_0 \ldots v2_{n-1}$) des vecteurs `v1` et `v2`.
  - calculer le produit scalaire de `v1` et `v2`.
  - afficher le résultat.

Rappel :
Le produit scalaire de *a* par *b* est: $a \cdot b = a_1 * b_1 + a_2 * b_2 + \ldots + a_n * b_n$

Exemple: a = (5, 3, -1) b = (2, 1, 2) $a \cdot b = 11$

Exercice 13 : multiplication de matrices (tableaux)

On cherche ici à écrire un programme `MulMat.java` qui calcule la multiplication de deux matrices (rappel ci-dessous).

Vous utiliserez pour représenter la matrice un tableau de tableaux de double.

Déclarations :

- Dans la méthode `main`, déclarez deux matrices `mat1` et `mat2`.

Traitements :

- Lire depuis le clavier les éléments de chacune des deux matrices (après avoir demandé leurs dimensions).
- Multiplier les deux matrices et stocker le résultat dans une nouvelle matrice `prod`.
- Afficher le contenu de cette nouvelle matrice ligne par ligne.

Méthode :

- Lire depuis le clavier les dimensions `lignes` (nombre de lignes) et `colonnes` (nombre de colonnes) de la première matrice `mat1`
- Lire le contenu de `mat1`.
- De même, lire les dimensions puis le contenu de la seconde matrice `mat2`.
- Vérifier que le nombre de lignes de `mat2` est identique au nombre de colonnes de `mat1`. Dans le cas contraire, afficher un message d'erreur "`Multiplication de matrices impossible !`". (rappel: si l'on multiplie deux matrices M = M1*M2, les dimensions de M sont "nombre de lignes de M1" et "nombre de colonnes de M2", et l'élément $M_{i,j}$ est défini par

$$M_{i,j} = \sum_{k=1}^{c1} M1_{i,k} \cdot M2_{k,j}$$

- afficher le résultat ligne par ligne.

Exemple d'utilisation:

```
Saisie de la 1ère matrice :
  Nombre de lignes : 2
  Nombre de colonnes : 3
  M[1,1]=1.0
  M[1,2]=2.0
  M[1,3]=3.0
  M[2,1]=4.0
  M[2,2]=5.0
  M[2,3]=6.0
Saisie de la 2ème matrice :
  Nombre de lignes : 3
```

```
  Nombre de colonnes : 4
  M[1,1]=1.0
  M[1,2]=2.0
  M[1,3]=3.0
  M[1,4]=4.0
  M[2,1]=5.0
  M[2,2]=6.0
  M[2,3]=7.0
  M[2,4]=8.0
  M[3,1]=9.0
  M[3,2]=0.0
  M[3,3]=1.0
  M[3,4]=2.0
Résultat :
38.0 14.0 20.0 26.0
83.0 38.0 53.0 68.0
```

Exercice 14 : triangle de Pascal (tableaux)

Le nombre de combinaisons de `p` objets parmi `n` est donné par le triangle de [Blaise] Pascal ci-dessous:

1									
1	1								
1	2	1							
1	3	3	1						
1	4	6	4	1					
1	5	10	10	5	1				
1	6	15	20	15	6	1			
1	...	...	...	...	...	...	1		
1	...	...	...	C_{n-1}^{p-1}	C_{n-1}^{p}	...	...	1	
1	...	...	...	...	C_{n}^{p}	...	...	...	1

Les relations suivantes y sont définies :

$$\forall n \geq 0 \, C_n^0 = C_n^n = 1$$
$$\forall p \in]0, n[\, C_n^p = C_{n-1}^{p-1} + C_{n-1}^p$$

Écrire un programme qui demande à l'utilisateur d'entrer un nombre `n>=1`, qui stocke le triangle de Pascal de taille `n` dans un tableau de tableaux et qui l'affiche. Le tableau de tableaux devra avoir la structure triangulaire ci-dessus, ce qui signifie qu'on ne réservera pas de place mémoire inutile pour des éléments indéfinis du triangle.

Méthode :

Le programme doit afficher le triangle de Pascal ligne par ligne. Notez que pour calculer la $i^{ème}$ ligne, il suffit de connaître la ligne i-1.
