

# BPO

## TP 3 - Océan

### Exercice 1 - artEoz et les tableaux

1. Demandez successivement les dessins des schémas mémoire par le logiciel **artEoz** (depuis l'url [arteoz.loria.fr](http://arteoz.loria.fr), cliquez sur le lien “accès à la version en ligne du logiciel artEoz”) sur les trois exemples donnés ci-dessous :

```
// tableau à une dimension

int[] tab = new int[6] ;
for (int i = 0 ; i < tab.length ; i++) {
    tab[i] = 2 * i ;
}
```


```
// tableau à 2 dimensions

int[][] mat1 = new int[2][4] ;
for (int i = 0 ; i < mat1.length ; i++) {
    for (int j = 0 ; j < mat1[0].length ; j++) {
        mat1[i][j] = 2 * i + j ;
    }
}
```

```
// tableau triangulaire

int[][] mat2 = new int[4][] ;
for (int i = 0 ; i < mat2.length ; i++) {
    mat2[i] = new int[i+1] ;
    for (int j = 0 ; j < mat2[i].length ; j++) {
        mat2[i][j] = 2 * i + j ;
    }
}
```

Vous pouvez ainsi comprendre la représentation de tableaux d'entiers :

- un tableau à une dimension (**tab**),
- un tableau à deux dimensions, c'est-à-dire une matrice (**mat1**)
- une matrice triangulaire un peu particulière (**mat2**) ; pour ce dernier exemple, demander une exécution en mode pas à pas 

2. Quelle est la représentation d'un tableau à 3 dimensions ? Avec **artEoz**, créez un nouveau tableau, prenez des petites valeurs pour les dimensions pour ne pas faire exploser le temps de calcul du schéma.
3. Écrire une séquence d'instructions qui crée et manipule un tableau de **geometrie.Point**. Pour la création, utilisez la notation raccourcie avec { et }.

## Exercice 2 - Les icebergs sur l'océan

Cet exercice devra être déposé sur Arche pour la semaine prochaine.

1. Recopiez la classe **glaces.ArcticImage** donnée sur Arche. Voici la documentation des fonctions :

- **ArcticImage(int largeur, int hauteur)**  
Paramètres :  
    largeur - en nombre de pixels  
    hauteur - en nombre de pixels
- **public void setColors(int[ ][ ] tab)**  
Mise à jour de l'image, en fonction des couleurs fournies  
Paramètres :  
    tab - contient les couleurs des pixels
- **public void fermer()**

2. Dans la fonction **main** de la classe **glaces.tests.TestArcticImage**, écrivez une séquence d'instructions qui crée un tableau d'entiers 300 x 300 avec la valeur 0 sur les lignes paires et la valeur 1 sur les lignes impaires, puis visualise le résultat en utilisant la fonction **setColors** de **ArcticImage**.
3. Programmez et testez la classe **Ocean** en suivant la méthode vue en cours (fonction après fonction).  
Le constructeur par défaut fixe la largeur et la hauteur de l'océan et place 2 icebergs à une position fixe.  
Attendez le prochain CM pour écrire (et tester) la fonction **toString()**.

Il est possible de générer des nombres aléatoires avec la classe **java.util.Random** :

- on déclare et on initialise le générateur de nombres par :

**Random g = new Random() ;**

- on peut ensuite générer autant de nombres que l'on veut, par exemple dans l'intervalle **[0, 100[** par :

**int x = g.nextInt(100) ;**