

# Travaux pratiques

## séance n°4 - corrigé

### Exercice 1 : les tableaux

On considère qu'un écran d'ordinateur est une matrice de `l` lignes et `c` colonnes. Chaque élément de cette matrice représente un pixel et vaut `false` s'il est éteint et `true` s'il est allumé.

#### Question 1

Ecrire un morceau de code qui permet à un utilisateur de saisir le nombre de lignes et de colonnes de l'écran, représenter cet écran par un tableau puis l'initialiser avec des pixels tous éteints et enfin l'afficher. Les pixels éteints seront représentés par des '0'.

```
import java.util.Scanner;
import static java.lang.System.*;
public class S4E1{
    public static void main( String[] args ){
        Scanner input = new Scanner( System.in );
        out.print(" nombre de lignes : ");
        int nbLignes = input.nextInt();
        out.print(" nombre de colonnes : ");
        int nbColonnes = input.nextInt();
        // déclarer et créer un tableau de booléens de nbLignes et nbColonnes
        .....
        // initialiser ce tableau à false
        .....
        .....
        .....
        // afficher ce tableau.
        // Les valeurs false seront représentées par des '0'
        .....
        .....
        .....
        .....
    }
}
```

#### Solution

```
import java.util.Scanner;
import static java.lang.System.*;
public class S4E1{
    public static void main( String[] args ){
        Scanner input = new Scanner( System.in );
        out.print(" nombre de lignes : ");
        int nbLignes = input.nextInt();
        out.print(" nombre de colonnes : ");
        int nbColonnes = input.nextInt();
        // déclarer et créer un tableau de booléens de nbLignes et nbColonnes
        boolean[][] tableau = new boolean[nbLignes][nbColonnes];
        // initialiser ce tableau à false
        for(int i=0;i<nbLignes;i++)
            for(int j=0;j<nbColonnes;j++)
                tableau[i][j]=false;
        // afficher ce tableau.
```

```

// Les valeurs false seront représentées par des '0'
for(int i=0;i<nbLignes;i++){
    for(int j=0;j<nbColonnes;j++){
        out.print('0');
        out.println();
    }
}
}

```

## Question 2

On veut maintenant donner à l'utilisateur la possibilité de représenter un rectangle sur son écran. Cela revient à faire passer tous les pixels du rectangle de la position " éteint" à la position " allumé" représentée par des '1'.

Un rectangle est caractérisé par les coordonnées (numéro de ligne, numéro de colonne) de son coin supérieur gauche, sa largeur et sa longueur. Ecrire le morceau de code qui demande à l'utilisateur de rentrer les caractéristiques d'un rectangle, puis modifie l'écran et l'affiche.

On suppose que le rectangle est contenu dans l'écran ( on ne prend pas en compte les erreurs de dimensionnement).

Compléter le programme précédent avec le code ci-dessous

```

// saisie des caractéristiques du rectangle
out.print(" abscisse du point haut gauche : ");
int x = input.nextInt();
out.print(" ordonnée du point haut gauche : ");
int y = input.nextInt();
out.print(" largeur du rectangle : ");
int largeur = input.nextInt();
out.print(" longueur du rectangle : ");
int longueur = input.nextInt();
// allumer les pixels du rectangle
// c'est à dire affecter la valeur true aux éléments de l'écran
// correspondant au rectangle
.....
.....
.....
// affichage de l'écran avec son rectangle
.....
.....
.....

```

## Solution

```

// allumer les pixels du rectangle
// c'est à dire affecter la valeur true aux éléments de l'écran
// correspondant au rectangle
for(int i=y;i<y+longueur;i++){
    for(int j=x;j<x+largeur;j++){
        tableau[i][j]=true;
    }
}
// affichage de l'écran avec son rectangle
for(int i=0;i<nbLignes;i++){
    for(int j=0;j<nbColonnes;j++){
        out.print(tableau[i][j]?'1':'0');
        out.println();
    }
}

```

## Question 3

Transformation du programme précédent.

Ecrire une procédure d'affichage du tableau qui sera appelée 2 fois en remplacement des affichages du tableau initial et du tableau après insertion dans l'écran.

Ecrire une fonction qui transforme le tableau en y insérant une fenêtre. On ne tiendra pas compte des erreurs dues à des valeurs de coordonnées inadéquates de la fenêtre interne.

## Solution

```
import java.util.Scanner;
import static java.lang.System.*;
public class S4E1{
    public static void main( String[] args ){
        Scanner input = new Scanner( System.in );
        out.print(" nombre de lignes : ");
        int nbLignes = input.nextInt();
        out.print(" nombre de colonnes : ");
        int nbColonnes = input.nextInt();
        // déclarer et créer un tableau de booléens de nbLignes et nbColonnes
        boolean[][] tableau = new boolean[nbLignes][nbColonnes];
        // initialiser ce tableau à false
        for(int i=0;i<nbLignes;i++)
            for(int j=0;j<nbColonnes;j++)
                tableau[i][j]=false;

        printTableau(tableau);

        // saisie des caractéristiques du rectangle
        out.print(" abscisse du point haut gauche : ");
        int x = input.nextInt();
        out.print(" ordonnée du point haut gauche : ");
        int y = input.nextInt();
        out.print(" largeur du rectangle : ");
        int largeur = input.nextInt();
        out.print(" longueur du rectangle : ");
        int longueur = input.nextInt();

        // insertion d'une fenêtre dans l'écran
        tableau = insereFenetre(tableau,y,x,longueur,largeur);

        // affichage de l'écran après insertion d'une fenêtre
        printTableau(tableau);
    }

    /**
     * affichage de l'écran sur le terminal
     *
     * @param t    l'écran
     */
    static void printTableau(boolean[][] t){
        for(int i=0;i<t.length;i++){
            for(int j=0;j<t[0].length;j++){
                out.print(t[i][j]?'1':'0');
                out.println();
            }
        }
    }

    /**
     * allumer les pixels du rectangle, c'est à dire affecter la valeur
     * true aux éléments de l'écran correspondant au rectangle
     */
}
```

```

* Les cas d'impossibilité d'insertion de la fenêtre dans l'écran ne
* sont pas traités
* @param t l'écran
* @param a abscisse du point haut et gauche de la fenêtre à
* insérer
* @param b ordonnée du point haut et gauche de la fenêtre à
* insérer
* @param lg longueur de la fenêtre
* @param la largeur de la fenêtre
* @return l'écran dans lequel une fenêtre a été insérée
*/
static boolean[][] insereFenetre(boolean[][] t,int a,int b,int lg,int la){
    for(int i=a;i<a+lg;i++)
        for(int j=b;j<b+la;j++)
            t[i][j]=true;
    return t;
}
}

```

#### Question 4

Modifier la fonction précédente pour prendre en compte les cas d'impossibilité d'insertion de fenêtre dans l'écran.

```

static boolean[][] insereFenetre(boolean[][] t,int a,int b,int lg,int la){
    if(a+la<t[0].length-1 && b+lg<t.length-1)
        for(int i=a;i<a+la;i++)
            for(int j=b;j<b+lg;j++)
                t[i][j]=true;
    return t;
}

```

### Exercice 2 : la courte paille

5 joueurs décident de choisir l'un d'entre eux en tirant à la courte paille.

La paille est représentée par un entier compris entre 1 et 5. Les joueurs sont représentés par les lettres a, b, c, d, e.

On demande de compléter le programme suivant qui tire aléatoirement une paille pour chaque joueur et affiche le nom du gagnant. Il est possible qu'il y ait égalité de pailles gagnantes, dans ce cas, le joueur dont le nom est le plus petit (ordre alphabétique) gagne.

#### Rappel

Pour effectuer un tirage pseudo-aléatoire, il faut créer un générateur d'une séquence pseudo-aléatoire.

Pour cela :

- import de la classe `java.util.Random`
- création d'un objet initialisant la séquence
- application de la méthode `nextInt( int x )` à cet objet. Cette méthode retourne la valeur aléatoire dans l'intervalle `[0, x[` qui suit dans la séquence.

```

import java.util.Random;
public class S4E2{
    public static void main(String[] args ){
        // déclarer et initialiser le tableau représentant les 5 pailles
        .....
        // déclarer et initialiser le tableau représentant les 5 joueurs
        .....
        // création d'un objet initialisant la séquence

```

```

    Random alea = new Random();
    // tirage aléatoire des 5 pailles.
    for( int i=0;i<5;i++ )
        .....
    // détermination de l'indice de la plus courte
    // et donc de l'indice du gagnant
    // il est possible qu'il y ait égalité.
    // l'indice de la paille dans le tableau paille
    // est en correspondance avec l'indice du joueur
    // dans le tableau joueurs.
    int plusCourte = 0;
    for( int i=1;i<5;i++ )
        .....
    // affichage du joueur gagnant
    System.out.println(" le gagnant est : "+ ..... );
}
}

```

### Solution

```

import java.util.Random;
public class CourtePaille{
    public static void main(String[] args ){
        // déclarer et initialiser le tableau représentant les 5 pailles
        int[] paille = new int[5];
        // déclarer et initialiser le tableau représentant les 5 joueurs
        char[] joueurs = {'a','b','c','d','e'};
        // création d'un objet initialisant la séquence
        Random alea = new Random();
        for( int i=0;i<paille.length;i++)
            paille[i] = alea.nextInt(15);
        // détermination de l'indice de la plus courte
        // et donc de l'indice du gagnant
        // il est possible qu'il y ait égalité.
        // l'indice de la paille dans le tableau paille
        // est en correspondance avec l'indice du joueur
        // dans le tableau joueurs.
        int plusCourte = 1;
        for( int i=1;i<paille.length;i++ )
            if( paille[plusCourte]>paille[i] ) plusCourte=i;

        System.out.println(" le gagnant est : "+ joueurs[plusCourte] );
    }
}

```