L3 Miage
PRGA

TP1 : Mots croisés

Dans ce TP vous développerez la classe ***MotsCroises*** pouvant servir de support à un jeu de mots croisés interactif. Chaque instance de cette classe représente un jeu exprimé par une grille de solution, une grille de proposition, une grille de définitions pour les mots horizontaux et une grille de définitions pour les mots verticaux.

Exercice 1 : classe *Grille*

Sachant qu'un jeu de mots croisés nécessite quatre grilles de contenus différents mais de même structure, à la différence lettre/mot près, commencez par réaliser la classe ayant pour nom qualifié **`nom1binome.nom2binome.tp1.Grille`**, dotée des attributs, constructeur et méthodes suivants :

```
package nom1binome.nom2binome.tp1;
public class Grille {
      // Variables d'instance
      // hauteur = nombre de lignes
      // largeur = nombre de colonnes
      // tab = tableau de chaînes de caractères à deux dimensions,
      // avec taille = hauteur x largeur

      // Constructeur permettant d'obtenir une grille
      // dotée d'un tableau de dimensions conformes aux valeurs
      // respectives de hauteur et de largeur, dont tous les
      // éléments contiennent la valeur null.
      // Précondition (assert) : hauteur ≥ 0 et largeur ≥ 0
      public Grille (int hauteur, int largeur) {…}

      // Accesseurs (getters)
      public int getHauteur() {…}
      public int getLargeur() {…}

      // Validité des coordonnées
      // Resultat : vrai si et seulement si lig (resp. col)
      // est compris entre 1 et getHauteur() (resp. getlargeur())
      public boolean coordCorrectes(int lig, int col) {...}

      // Valeur de la cellule ayant pour coordonnées (lig, col)
      // Précondition (assert) : coordCorrectes(lig, col)
      public String getCellule(int lig, int col) {…}

      // Modification de la cellule de coordonnées (lig, col)
      // Précondition (assert) : coordCorrectes(lig, col)
      public void setCellule(int lig, int col, String ch) {…}
```

```
	// Texte sur "hauteur" lignes, colonnes séparées par des |
	// (voir exemple plus loin)
	@Override
	public String toString()  {…}
}
```

Voici un exemple d'utilisation de cette classe :

```
Grille maGrille = new Grille(3,5) ;
for (int l=1; l<=maGrille.getHauteur(); l++)
{
	String texteLigne = Integer.toString(l);
	for (int c=1; c<=maGrille.getLargeur(); c++)
	{
		maGrille.setCellule
			(l, c, texteLigne + ',' + Integer.toString(c));
	}
}
System.out.println(maGrille) ;
```

Résultat affiché :

```
1,1|1,2|1,3|1,4|1,5
2,1|2,2|2,3|2,4|2,5
3,1|3,2|3,3|3,4|3,5
```

En outre, votre classe devra se conformer au test JUnit fourni dans **G:\l3miage\PRGA\TP1\Grilletest.java**

Exercice 2 : classe *MotsCroises*

Dans le même paquetage, écrire une classe *MotsCroises* servant de base à une application de mots croisés (cf. http://fortissimots.com parmi beaucoup d'autres), et utilisant quatre instances de la classe *Grille* :

```
	// Variables d'instance
	…solution…
	…proposition…
	…horizontal…
	…vertical…

	// Constructeur créant une instance de MotsCroises
	// dotée de 4 instances de Grille, suivant les
	// spécifications données ci-dessous :
	public MotsCroises (int hauteur, int largeur) {…}

	// Nombre de rangées
	public int getHauteur(){…}

	// Nombre de colonnes
	public int getLargeur(){…}
```

```
// Validité des coordonnées
// Resultat : vrai si et seulement si (lig, col)
// désignent une cellule existante de la grille
public boolean coordCorrectes(int lig, int col) {...}

// Accesseurs aux cases noires
// Précondition (assert) : coordCorrectes(lig, col)
public boolean estCaseNoire(int lig, int col) {…}
public void setCaseNoire(int lig, int col, boolean noire) {…}

// Accesseurs à la grille de solution
// Préconditions (assert) :
      coordCorrectes(lig, col)
      !estCaseNoire(lig, col)
public char getSolution(int lig, int col) {…}
public void setSolution(int lig, int col, char sol) {…}

// Accesseurs à la grille du joueur
// Préconditions : idem
public char getProposition(int lig, int col) {…}
public void setProposition(int lig, int col, char prop) {…}

// Accesseurs aux définitions.
// Le paramètre "horiz" est "true" pour les définitions
// horizontales, "false" pour les définitions verticales.
// Préconditions : idem
public String getDefinition
      (int lig, int col, boolean horiz) {…}
public void setDefinition
      (int lig, int col, boolean horiz, String def) {…}
```

Les quatre instances de la classe *Grille* permettent de gérer respectivement :

- la grille de la solution, dont les cellules sont soit des chaînes de caractères limitées à un seul caractère, soit **null** s'il s'agit d'une case noire. Moyennant une bonne programmation et surtout une bonne utilisation des méthodes *estCaseNoire()* et *estCaseNoire()*, il ne sera pas nécessaire de « recopier » les cases noires dans les trois autres grilles.
- la grille remplie par le joueur, suivant les mêmes règles que ci-haut, mais pouvant avoir des cases vides (et non noires) représentées par une chaîne limitée au caractère « espace », si le joueur n'a pas encore proposé de lettre pour cette case.
- la grille contenant les définitions des mots horizontaux : chaque définition a les mêmes coordonnées (rangée, colonne) que la première lettre du mot (horizontal) à trouver ;
- même principe pour les définitions des mots verticaux.