TP Java : les figures géométriques en mode texte

M.C. Lagasquie

10 décembre 2019

But du TP : Définition et développement d'un ensemble de classes, dans le but de créer et d'afficher un schéma composé de figures géométriques.
Les figures géométriques peuvent être des carrés, triangles, rectangles, cercles, ovales, mais cette liste n'est pas limitative et peut comprendre des losanges, polygones, etc.
A chaque figure est associé en tant qu'attribut un "petit" dessin, qui permet de représenter la figure sous forme d'un tableau de caractères.

1. Définir et tester une classe DESSIN qui crée un dessin (matrice de caractères) associé à une figure. Un dessin est créé à une taille donnée :
   — `Dessin (int NbLigMax, int NbColMax)`.
   Il est initialement rempli avec un caractère par défaut (le caractère '.', juste pour être visible à l'écran). Il doit disposer des méthodes :
   — `void setPoint(int nl, int nc, char c)` pour mettre le caractère `c` dans la case de la matrice en position `(nl,nc)` (si les positions sont erronées, la méthode ne doit pas avoir d'impact) ;
   — `char getPoint(int nl, int nc)` pour récupérer la valeur qui est dans la case de la matrice en position `(nl,nc)` (si les positions sont erronées, la méthode doit retourner une exception du type IllegalArgumentException définie dans la biliothèque Java) ;
   — `String toString()` pour renvoyer uniquement la matrice de caractères sous la forme d'une chaîne de caractères bien formatée et sans caractère inutile (pour affichage ultérieur).
2. Définir une classe abstraite FIGURE qui possède un DESSIN d'une taille donnée et dont elle stocke la référence. Elle a également une position (en x et y) qui sera nécessaire quand on ajoutera une figure dans un schéma. Et enfin, elle possède aussi un attribut contenant le caractère qui servira à dessiner la figure dans son DESSIN. Le constructeur sera donc de la forme :
   — `Figure (int NbLigMax, int NbColMax, int x, int y, char c)`.
   Elle dispose d'une méthode abstraite `void fill()` dont le rôle est de remplir son DESSIN avec le caractère donné lors de la construction.
   Il faut aussi prévoir une méthode `String toString()` pour renvoyer uniquement la matrice de caractères de son DESSIN sous la forme d'une chaîne de caractères bien formatée et sans caractère inutile (pour affichage ultérieur).
3. Définir et tester des classes héritant de FIGURE (a minima RECTANGLE, CARRÉ, TRIANGLE, mais vous pouvez aussi écrire OVALE, CERCLE, LOSANGE, etc.) qui devront définir la méthode `fill` afin qu'elle effectue le remplissage effectif du DESSIN associé. Créer une hiérarchie de classes là où c'est approprié (c'est-à-dire quand on s'aperçoit qu'on duplique des attributs ou des méthodes) et exploiter au maximum l'héritage. Chaque classe devra avoir un constructeur qui lui est propre et qui n'utilise comme paramètres que ceux qui sont absolument nécessaires pour cette classe. Remarque : après l'appel du constructeur, le DESSIN de la figure devra avoir été mis à jour.

On souhaite maintenant pouvoir créer un schéma rassemblant plusieurs figures à dessiner créées au préalable (on pourra ajouter et/ou enlever des figures). Pour cela, le DESSIN du SCHEMA est "rempli" par recopie du "petit" DESSIN de chaque FIGURE, à la position (x, y) précisée lors de la création de la FIGURE (cette position (x, y) correspond à la position de l'angle haut-gauche du "petit" DESSIN de la FIGURE dans le "grand" DESSIN du SCHEMA, en prenant comme hypothèse que la position $(0, 0)$ est dans l'angle haut-gauche du SCHEMA et la position `(NbLigMax-1,NbColMax-1)` dans l'angle bas-droit, avec l'axe des x qui correspond aux lignes et celui des y aux colonnes).

4. Définir et tester une classe SCHEMA contenant une *Collection* (par exemple une ARRAYLIST) de FIGUREs. On associe également un "grand" DESSIN au SCHEMA. Le constructeur sera donc de la forme suivante :

— `Schema (int NbLigMax, int NbColMax)`.

On aura aussi la méthode :

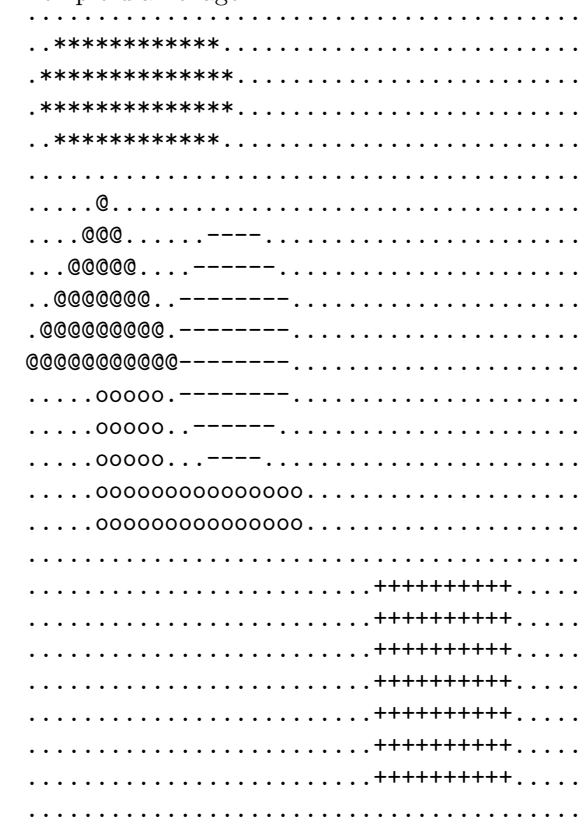
— `String toString()` qui renvoie uniquement la matrice de caractères du DESSIN du SCHEMA sous la forme d'une chaîne de caractères bien formatée et sans caractère inutile (pour affichage ultérieur).

On peut ajouter une FIGURE dans le SCHÉMA avec la méthode : [1]

— `void ajout (Figure f)`.

Remarque : chaque classe devra disposer des accesseurs nécessaires en fonction de ses attributs.

Exemple d'affichage :



Ce schéma de taille (26, 40) contient :

— un ovale de taille (4,14), en position (1,2) dessiné avec une '*',

— deux rectangles (par exemple, celui dessiné avec le caractère 'o' est de taille (5,15) et en position (12,5)),

— un triangle de base 11 dessiné avec le caractère '@' en position (11,0) et

— un cercle de rayon 4, en position (7,11).

On constate qu'en recopiant le petit dessin du cercle tracé avec le caractère '-', on a écrasé en partie le dessin du rectangle tracé avec le caractère 'o'.

Une attention particulière devra être portée aux tests. Pour cela, une classe TESTFIGURE vous est fournie (accessible sous Moodle) que vous installerez dans votre projet Java. Cette classe contient une méthode `main` permettant d'exécuter *automatiquement* un jeu de tests le plus complet et le plus lisible possible. Vous ne devrez pas modifier cette classe ! Et pour qu'elle fonctionne correctement, vous devrez respecter *scrupuleusement* les prototypes du constructeur et des méthodes demandés.

Cela ne vous dispensera pas d'écrire votre propre méthode `main` dans vos classes afin de tester *au fur et à mesure* votre travail.

Nous vous rappelons que, dans le cadre des évaluations, vos classes seront aussi exécutées sur d'autres jeux de tests non fournis.

1. La méthode permettant de retirer une figure du schéma n'est pas demandée.