amU Faculté des sciences Aix Marseille Université **Année universitaire 2024-2025**

Site : ☒ Luminy ☐ St-Charles ☐ St-Jérôme ☐ Cht-Gombert ☐ Aix-Montperrin ☐ Aubagne-SATIS

Sujet de : ☒ 1er semestre ☐ 2ème semestre ☐ Session 2 Durée de l'épreuve : 2h

Examen de : L3 Nom du diplôme : Licence informatique : parcours math-info

Code du module : SIN5U34 Libellé du module : Initiation au Génie logiciel

Calculatrices autorisées : NON Documents autorisés : OUI, notes de Cours, supports de cours

---

1 Dessins de glyphes

La classe suivante permet de dessiner des glyphes à l'écran :

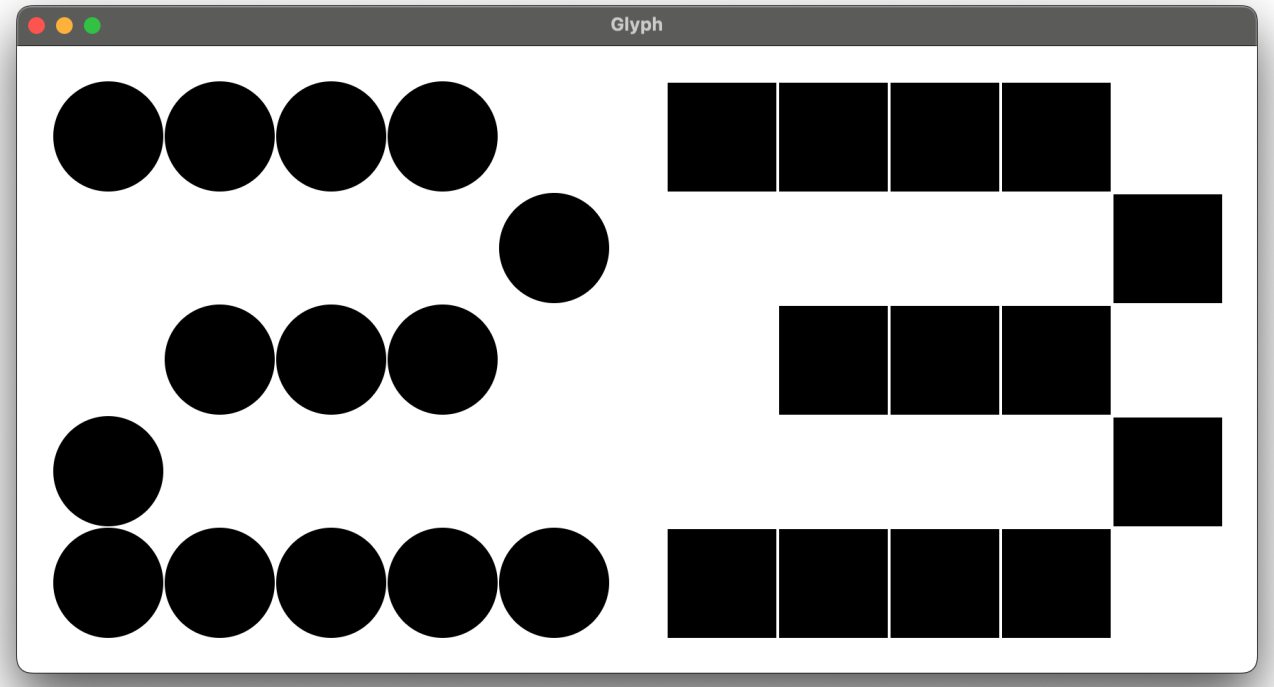
```
public class Glyph {
    public enum DotType {SQUARE, CIRCLE}

    private final DotType type;
    private final char[][] matrix;

    public Glyph(char[][] matrix, DotType type) {
        this.matrix = matrix;
        this.type = type;
    }

    public void draw(GraphicsContext g, int x, int y, int size) {
        int n = matrix.length;
        int ds = size / n;
        for (int px = 0; px < n; px++)
            for (int py = 0; py < n; py++)
                if (matrix[py][px] == '#')
                    drawDot(g, x + px * ds + 1, y + py * ds + 1, ds - 1);
    }

    private void drawDot(GraphicsContext g, int x, int y, int size) {
        switch (type) {
            case SQUARE -> drawSquare(g, x, y, size);
            case CIRCLE -> drawCircle(g, x, y, size);
        }
    }

    private void drawSquare(GraphicsContext g, int x, int y, int size) {
        g.fillRect(x + 1, y + 1, size - 1, size - 1);
    }

    private void drawCircle(GraphicsContext g, int x, int y, int size) {
        g.fillOval(x, y, size, size);
    }
}
```

Un exemple d'utilisation de la classe précédente est donné ci-dessous :

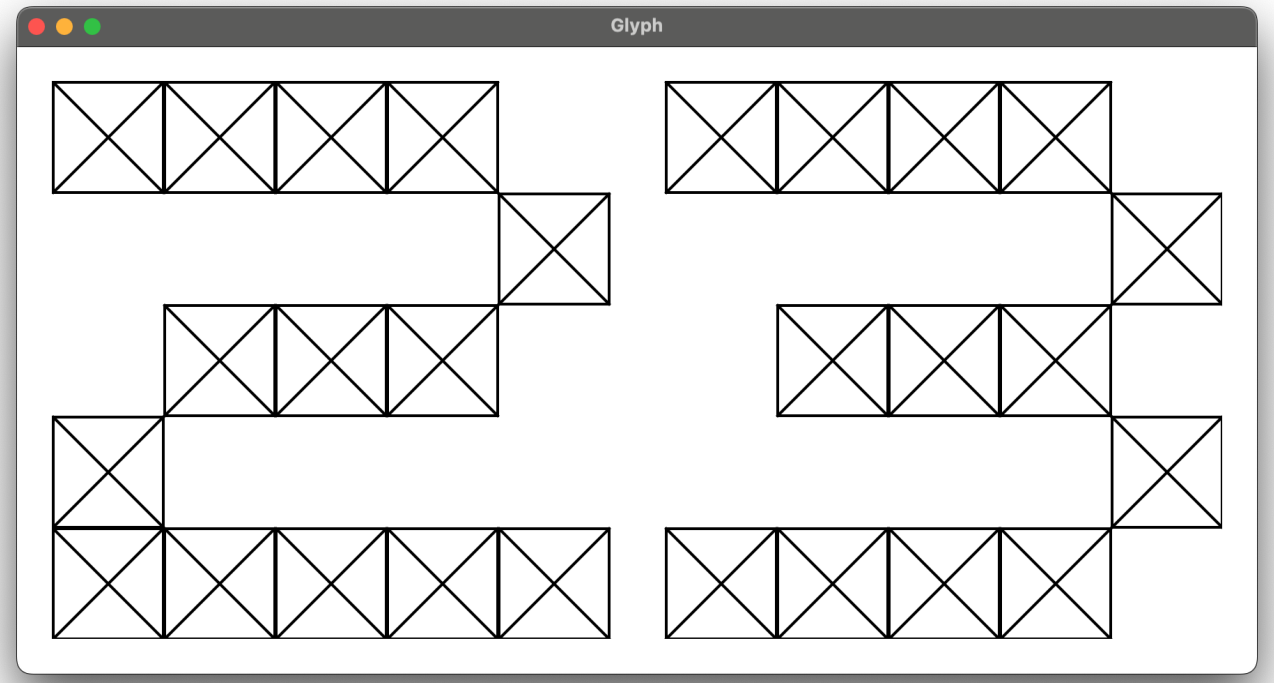
```
public class GlyphApplication extends Application {
    @Override
    public void start(Stage stage) {
        StackPane root = new StackPane();
        Canvas canvas = new Canvas(840, 400);
```

```
        GraphicsContext graphicsContext2D = canvas.getGraphicsContext2D();
        paint(graphicsContext2D);
        root.getChildren().add(canvas);
        stage.setTitle("Glyph");
        stage.setScene(new Scene(root));
        stage.show();
    }

    char[][] c2 =  {{'#','#','#','#',' '},
            {' ',' ',' ',' ','#'},
            {' ','#','#','#',' '},
            {'#',' ',' ',' ',' '},
            {'#','#','#','#','#'}};
    char[][] c3 =  {{'#','#','#','#',' '},
            {' ',' ',' ',' ','#'},
            {' ','#','#','#',' '},
            {' ',' ',' ',' ','#'},
            {'#','#','#','#',' '}};

    private void paint(GraphicsContext graphicsContext) {
        graphicsContext.setFill(Color.BLACK);
        graphicsContext.setLineWidth(2);
        Glyph glyph2 = new Glyph(c2, Glyph.DotType.CROSS);
        Glyph glyph3 = new Glyph(c3, Glyph.DotType.CROSS);
        glyph2.draw(graphicsContext, 0, 0, 400);
        glyph3.draw(graphicsContext, 440, 0, 400);
    }

    public static void main(String[] args) {
        launch();
    }
}
```

L'exécution de la classe `GlyphApplication` donne l'affichage suivant :



Question 1 : Quelles parties du code doivent être modifiées pour ajouter un nouveau type de points en forme de croix ? L'utilisation de ce nouveau type de points dans l'exemple précédent doit produire les glyphes suivants :



Question 2 : Parmi les 5 principes SOLID, lequel est violé ? Justifiez.

Question 3 : Donnez le diagramme de classes d'une nouvelle organisation du code (sans ajouter pour le moment de classes pour le dessin des croix) qui respecte les principes SOLID en utilisant le patron de conception "Patron de méthode" (*template method*).

Question 4 : Implémentez le diagramme de classes que vous avez proposé à la question 3 et réécrivez en conséquence le code de la méthode `paint` de la classe `GlyphApplication` donnée en exemple.

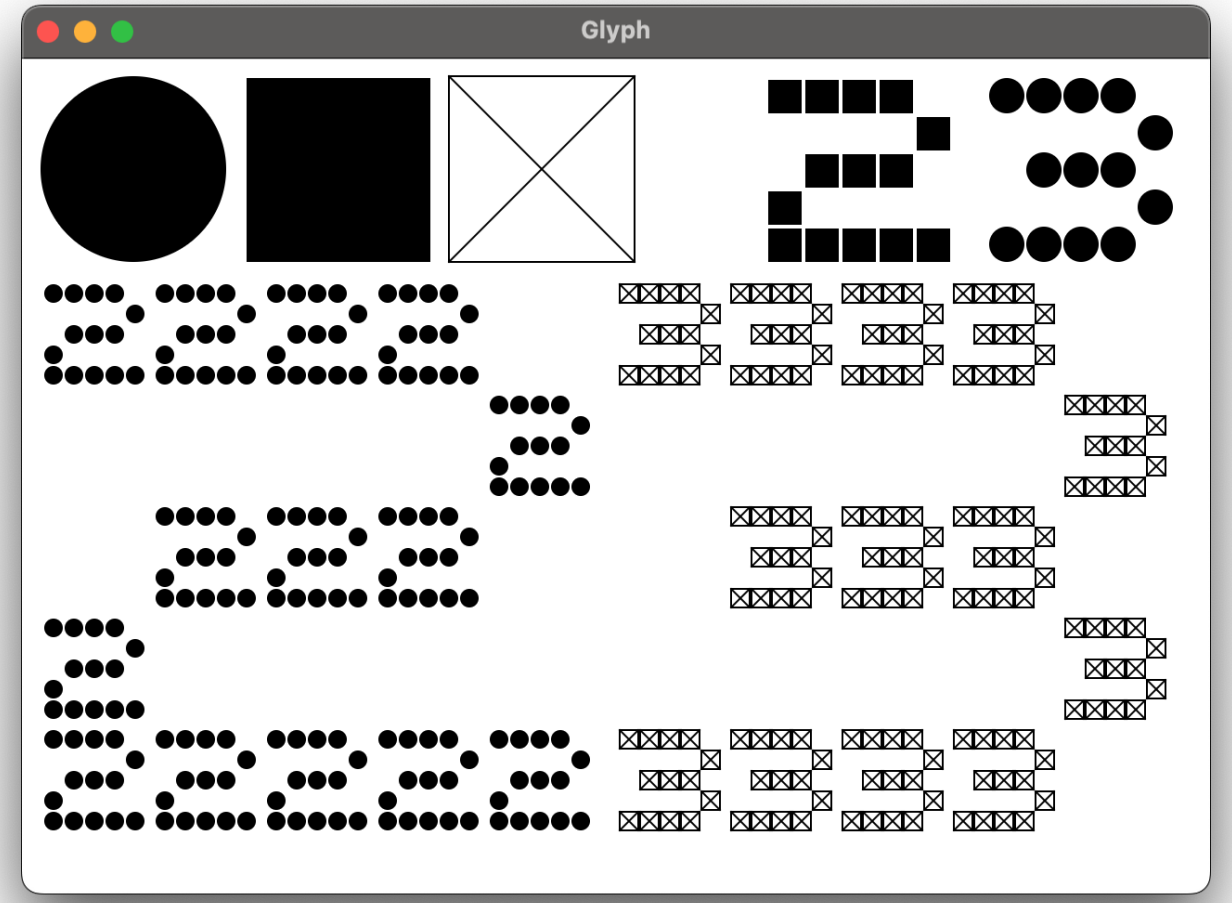
Question 5 : écrivez la classe permettant de dessiner les glyphes avec les points en forme de croix évoqués à la question 1. Vous utiliserez les deux méthodes de `GraphicsContext` ci-dessous :

```
/**
 * Strokes a line.
 *
```

```
     * @param x1 the X coordinate of the starting point of the line.
     * @param y1 the Y coordinate of the starting point of the line.
     * @param x2 the X coordinate of the ending point of the line.
     * @param y2 the Y coordinate of the ending point of the line.
     */
    public void strokeLine(double x1, double y1, double x2, double y2){/* ... */}

    /**
     * Strokes a rectangle.
     *
     * @param x the X position of the upper left corner of the rectangle.
     * @param y the Y position of the upper left corner of the rectangle.
     * @param w the width of the rectangle.
     * @param h the height of the rectangle.
     */
    public void strokeRect(double x, double y, double w, double h) {/* ... */}
```

Question 6 : On souhaite modifier le code que la méthode `paint` ci-dessous produise l'affichage ci-après. Donnez le diagramme de classes d'une nouvelle organisation du code correspondant à cette modification. Les classes ou interfaces `Drawable`, `CircleDot`, `SquareDot`, `CrossDot` et `Glyph` devront donc apparaître dans votre diagramme.

```
private void paint(GraphicsContext graphicsContext) {
    Drawable glyphC = new CircleDot();
    Drawable glyphS = new SquareDot();
    Drawable glyphX = new CrossDot();
    Drawable glyph2 = new Glyph(c2, glyphS);
    Drawable glyph3 = new Glyph(c3, glyphC);
    Drawable glyph22 = new Glyph(c2, new Glyph(c2, glyphC));
    Drawable glyph33 = new Glyph(c3, new Glyph(c3, glyphX));
    glyphC.draw(graphicsContext, 10, 10, 100);
    glyphS.draw(graphicsContext, 120, 10, 100);
    glyphX.draw(graphicsContext, 230, 10, 100);
    glyph2.draw(graphicsContext, 400, 10, 100);
    glyph3.draw(graphicsContext, 520, 10, 100);
    glyph22.draw(graphicsContext, 10, 120, 300);
    glyph33.draw(graphicsContext, 320, 120, 300);
}
```



Question 7 : Implémentez le diagramme de classes que vous avez proposé à la question 6. Les classes et les interfaces du diagramme de la question 6 devront pouvoir être utilisées par la version de la méthode `paint` ci-dessus de façon à produire l'affichage ci-dessous.