Escuela Colombiana de Ingeniería Julio Garavito

Ingeniería de Sistemas

Programación Orientada a Objetos

LABORATORIO 01
02/09/2023

Joan Steven Acevedo Aguilar
joan.acevedo@mail.escuelaing.edu.co

Santiago Sanchez Monroy
santiago.sanchez-m@mail.escuelaing.edu.co

Profesor a cargo:

Maria Irma Diaz Rozo
maria.diaz@mail.escuelaing.edu.co

Objetivos:

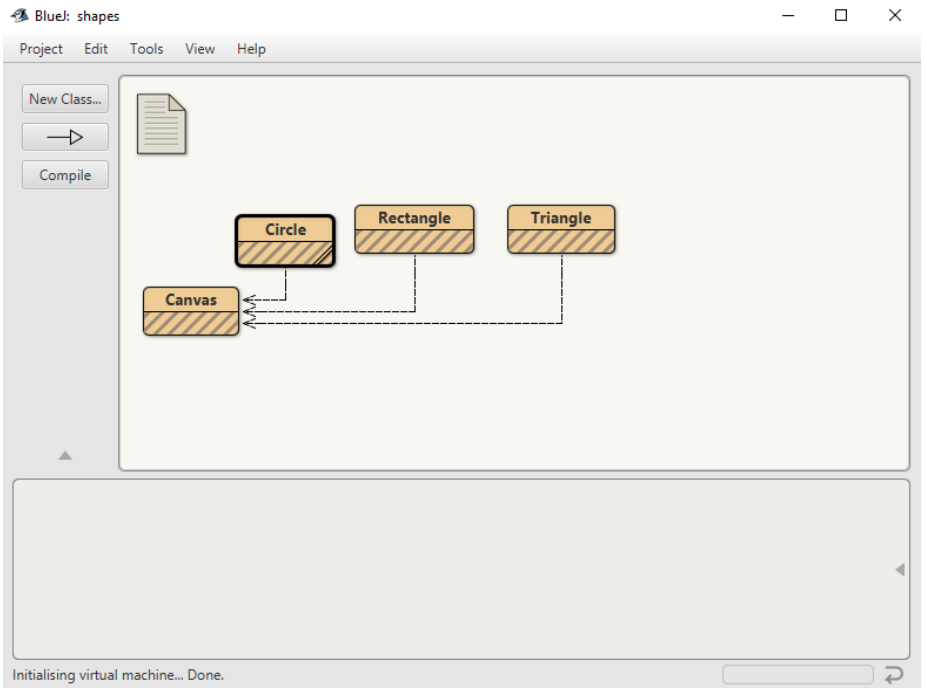
Desarrollar competencias básicas para:

1. Apropiar un paquete revisando: diagrama de clases, documentación y código.
2. Crear y manipular un objeto. Extender y crear una clase.
3. Entender el comportamiento básico de memoria en la programación OO.
4. Investigar clases y métodos en el API de java .
5. Utilizar el entorno de desarrollo de BlueJ
6. Vivenciar las prácticas XP :
    a. Planning The project is divided into iterations.
    b. Coding All production code is pair programmed

SHAPES

A. Conociendo el proyecto shapes

1. El proyecto “shapes” es una versión modificada de un recurso ofrecido por BlueJ. Para trabajar con él, bajen shapes.zip y ábranlo en BlueJ. Capturen la pantalla.



2. El diagrama de clases permite visualizar las clases de un artefacto software y las relaciones entre ellas. Considerando el diagrama de clases de “shapes”
    a. ¿Qué clases ofrece?
    Tenemos 4 clases:
        - Canvas
        - Circle
        - Rectangle
        - Triangle

b. ¿Qué relaciones existen entre ellas?
La relación que tienen 3 de las clases dadas(Circle, Rectangle, Triangle) es que son figuras geométricas y la otra relación que tienen estas mismas 3 con la cuarta(Canvas) es que se pueden plasmar en esta última, que vendría siendo el lienzo.

3. La documentación presenta las clases del proyecto y, en este caso, la especificación de sus componentes públicos. De acuerdo con la documentación generada:
   a. ¿Qué clases tiene el paquete shapes?
   - Canvas
   - Rectangle
   - Circle
   - Triangle

   b. ¿Qué atributos tiene la clase Triangle?
   Los atributos que tiene esta clase Circle son los siguientes:

Fields

Modifier and Type	Field	Description
static int	VERTICES	

   c. ¿Cuántos métodos ofrece la clase Triangle?
   En la documentación la clase Triangle tiene 12 métodos que son los siguientes:

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method	Description
void	changeColor(String newColor)	Change the color.
void	changeSize(int newHeight, int newWidth)	Change the size to the new size
void	makeInvisible()	Make this triangle invisible.
void	makeVisible()	Make this triangle visible.
void	moveDown()	Move the triangle a few pixels down.
void	moveHorizontal(int distance)	Move the triangle horizontally.
void	moveLeft()	Move the triangle a few pixels to the left.
void	moveRight()	Move the triangle a few pixels to the right.
void	moveUp()	Move the triangle a few pixels up.
void	moveVertical(int distance)	Move the triangle vertically.
void	slowMoveHorizontal(int distance)	Slowly move the triangle horizontally.
void	slowMoveVertical(int distance)	Slowly move the triangle vertically.

d. ¿Cuáles métodos ofrece la clase Triangle para que la figura cambie su tamaño (incluya sólo el nombre)?

```
void                changeSize(int newHeight, int newWidth)  Change the size to the new size
```

4. En el código de cada clase está el detalle de la implementación. Revisen el código de la clase Triangle. Con respecto a los atributos:
   a. ¿Cuántos atributos realmente tiene?
      Realmente tiene 7 atributos.

   b. ¿Cuáles atributos determinan su forma?. Con respecto a los métodos:
      Hablando estrictamente de su forma, los atributos son “height” y “width”, y el método encargado de modificar estos dos atributos es “changeSize”.

   c. ¿Cuántos métodos tiene en total?
      Contando el constructor serían 15 métodos.

   d. ¿Quiénes usan los métodos privados?
      - changeSize()
      - moveVertical()
      - moveHorizontal()
      - makeInvisible()
      - makeVisible()
      - changeColor()
      - slowMoveVertical()
      - slowMoveHorizontal()

5. Comparando la documentación con el código
   a. ¿Qué no se ve en la documentación?
      Los atributos y métodos de privados no son visibles en la documentación

   b. ¿por qué debe ser así?
      Esto es debido a que hay ciertas restricciones sobre acciones que puede realizar el usuario final en el programa, ya sea por cuestiones de seguridad o por que así lo desea el desarrollador.

6. En el código de la clase Triangle, revise el atributo VERTICES
   a. ¿Qué significa que sea public?
      El estado público de un atributo significa que otras clases tienen el poder de acceder a él sin restricciones.

b. ¿Qué significa que sea static?
El estado estático de un atributo significa que se puede determinar un mismo valor para todas las clases a las que este se encuentre relacionado y puede estar sujeto a cambios, sin embargo, esto afectará a todas las clases que lo utilicen.

c. ¿Qué significa que fuera final? ¿Debe serlo?
Significa que el valor de ese atributo va a ser fijo para cualquier clase que lo utilice.
Si ya que el número de vértices debe ser el mismo para todos, además de que no debería ser modificado.

d. ¿De qué tipo de dato debería ser (byte, short, int, long)? ¿Por qué?
Debería ser “byte”, ya que es un dato con un valor pequeño.

e. Actualícenlo.

```
public static byte VERTICES=3;
```

7. En el código de la clase Triangle revisen el detalle del tipo del atributo height
   a. ¿Qué se está indicando al decir que es int?.
   Que es un valor de tipo entero.

   b. Sí height fuera byte, ¿cuál sería el triángulo más grande posible?
   Tendría como altura máxima 127.

   c. y ¿si fuera long?
   Tendría como altura máxima $2^{63} - 1$.

   d. ¿Qué restricción adicional deberían tener este atributo? Refactoricen el código considerando (d).
   Debería ser mayor a cero siempre, ya que no tendría sentido que la altura del triángulo sea menor o igual a cero.

```
public void changeSize(int newHeight, int newWidth) {
    erase();
    if(newHeight >=0 || newWidth >= 0){
        height = newHeight;
        width = newWidth;
        draw();
    }

}
```

8. ¿Cuál dirían es el propósito del proyecto “shapes”?

   Creemos que el propósito del proyecto “shapes” es realizar la graficación de figuras geométricas en un lienzo dependiendo de los requerimientos del usuario, que pueden ser la elección de la posición, el tamaño, el color, entre otras acciones que este decida o necesite.

B. Manipulando objetos. Usando un objeto.

1. Creen un objeto de cada una de las clases que lo permitan.
   a. ¿Cuántas clases hay?
      Hay 4 clases, sin embargo, solo 3 permiten crear objetos.

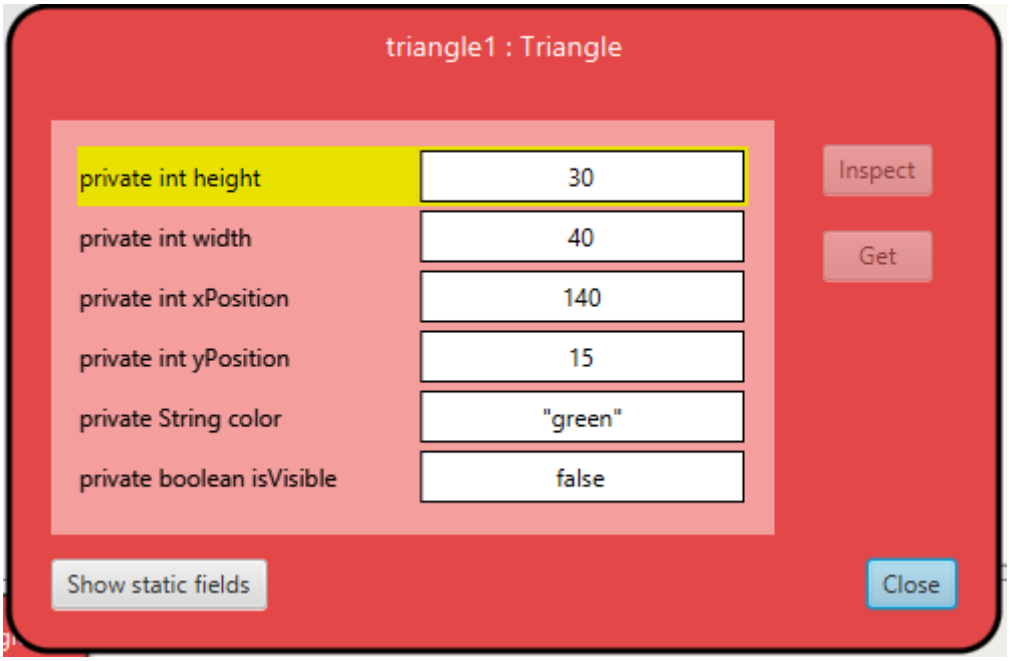
   b. ¿Cuántos objetos crearon?
      3 objetos (1 círculo, 1 triángulo, 1 rectángulo).

   c. ¿Quién se crea de forma diferente? ¿Por qué?
      El canvas, ya que este es “el lienzo” donde se mostrarán los objetos que creemos.

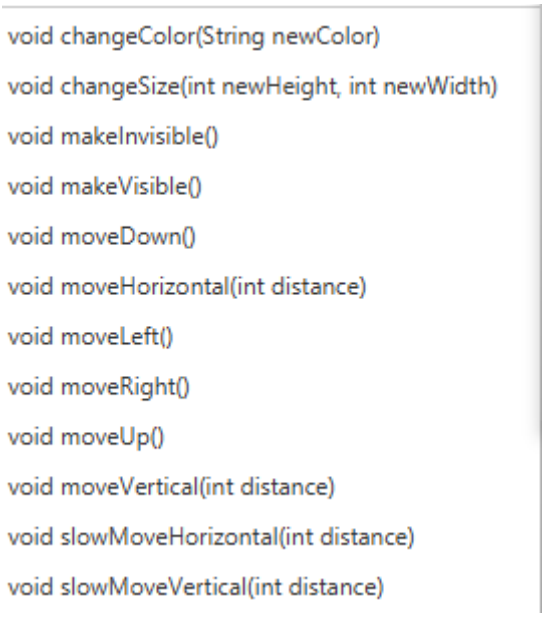
2. Inspeccionen el estado del objeto :Triangle,
   a. ¿Cuáles son los valores de inicio de todos sus atributos?
      Los valores se observan en el siguiente punto.

   b. Capturen la pantalla.

```
public static int VERTICES=3;
```



3. Inspeccionen el comportamiento que ofrece el objeto :Triangle.
    a. Capturen la pantalla.

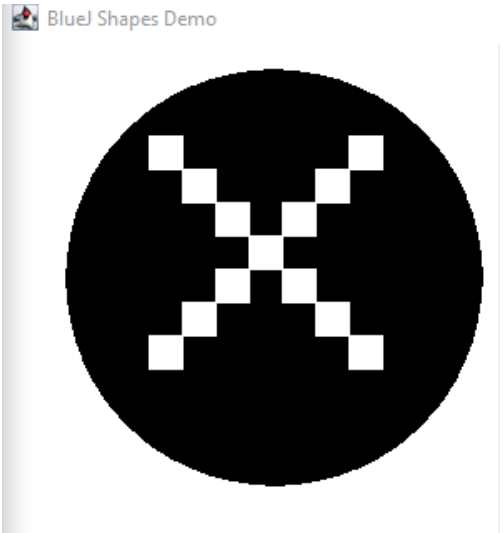


    b. ¿Por qué no aparecen todos los que están en el código?
    Porque su visibilidad no lo permite, es decir, los métodos privados no aparecen disponibles para el usuario.

4. Construyan, con “shapes” sin escribir código, una propuesta de la imagen del logo de su red social favorita.
    a. ¿Cuántas y cuáles clases se necesitan?
    Se usan 2 clases, Circle y Rectangle.

    b. ¿Cuántos objetos se usan en total?
    Se usan 1 círculo y 13 rectángulos, 14 objetos en total.

    c. Capturen la pantalla.



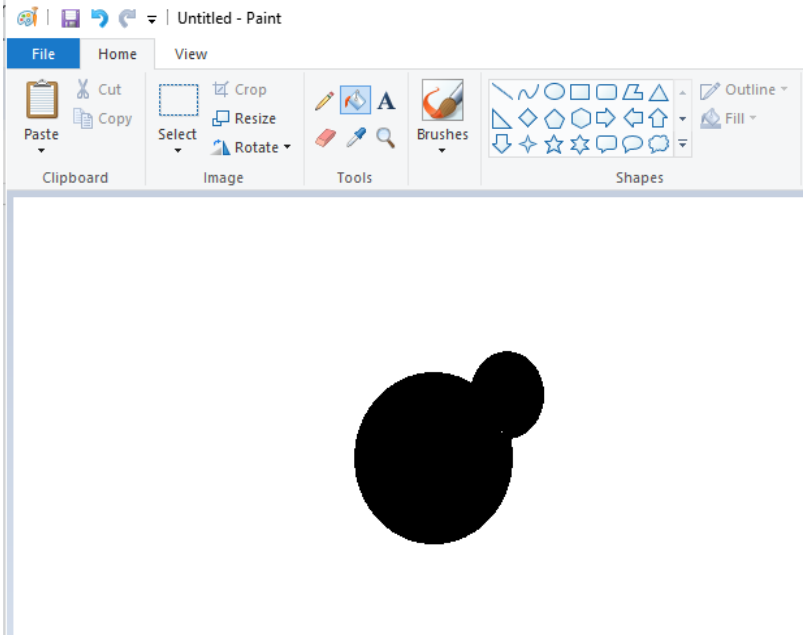
d. Incluyan el logo original.

C. Manipulando objetos. Analizando y escribiendo código.

```
Circle face;
Circle leftEar;
Circle rightEar;
//1
face=new Circle();
face.changeSize(50);
//2
face.moveVertical(20);
face.changeColor("black");
face.makeVisible();
//3
leftEar=new Circle();
leftEar.changeColor("black");
leftEar.moveHorizontal(-10);
//4
rightEar=leftEar;
rightEar.moveHorizontal(30);
rightEar.makeVisible();
//5
leftEar.makeVisible();
//6
```

1. Lean el código anterior.
   a. ¿Cuál creen que es la figura resultante?
   Una cara con solo una oreja visible, de color negro ambas cosas.

   b. Píntenla.



2. Habiliten la ventana de código en línea , escriban el código. Para cada punto señalado indiquen:
    a. ¿Cuántas variables existen?
       Hay 3 variables

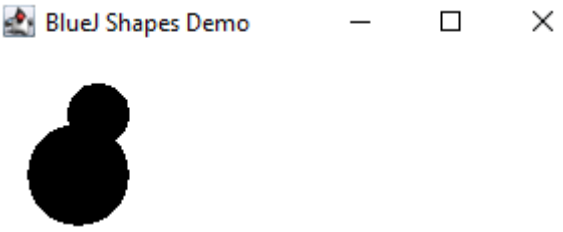
    b. ¿cuántos objetos existen? (no cuenten ni los objetos String ni el objeto Canvas)
       Hay 3 objetos.

    c. ¿qué color tiene cada uno de ellos?
       Color negro.

    d. ¿Cuántos objetos se ven? Al final
       Solo se ven 2 objetos.

    e. Expliquen sus respuestas.
       El motivo por el que solo se ven 2 objetos y no 3 es porque se programó que dos objetos (las orejas) tuvieran las mismas características (“rightEar = leftEar”)

    f. Capturen la pantalla.



3. Compare la figura pintada en 1. con la figura capturada en 2. ,
    a. ¿son iguales?
       Son prácticamente iguales.

    b. ¿por qué?
       Porque ya sabíamos lo que iba a pasar al ver como había sido escrito el código.

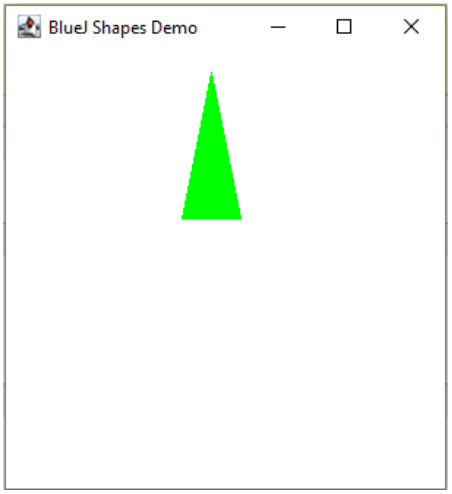
D. Extendiendo una clase. Triangle

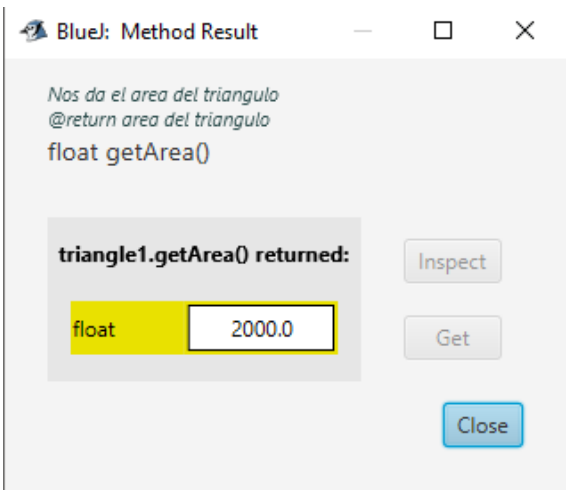
1. Desarrollen en Triangle el método área(). ¡Pruébenlo! Capturen una pantalla.

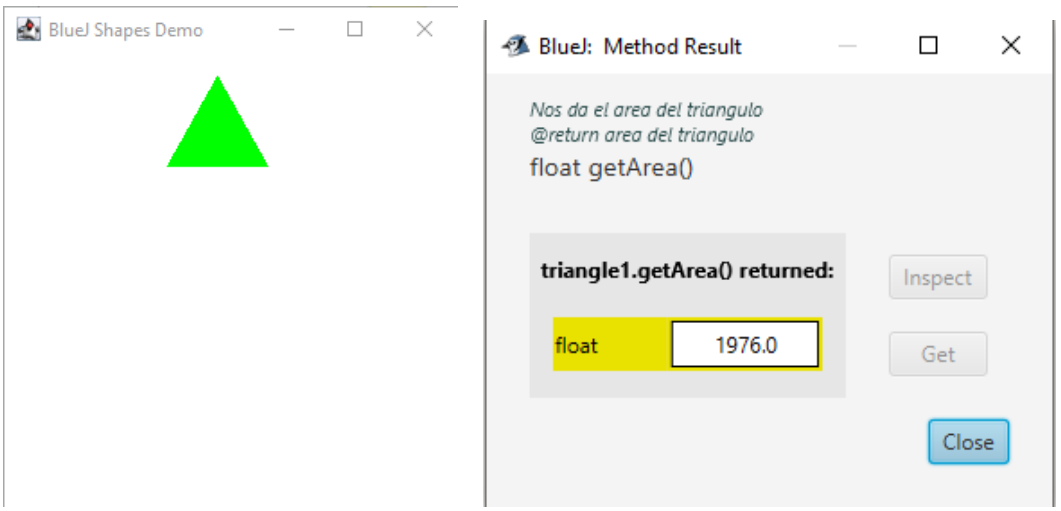
```java
/**
 * Nos da el area del triangulo
 * @return area
 */
public float area(){
    float area = 0;
    area = (this.height * this.width)/ 2;
    return area;
}
```

2. Desarrollen en Triangle el método equilateral() (convierte el triángulo en un equilátero manteniendo el área) . ¡Pruébenlo! Capturen dos pantallas.

```java
/**
 * Convierte un triangulo a uno equilatero, manteniendo su misma area(Aprox.)
 */
public void equilateral(){
    this.width = (int) Math.sqrt((4 * this.area) / Math.sqrt(3));
    this.height = (int) (2 * this.area) / this.width;
}
```







3. Desarrollen en Triangle el método blink(times:int) (si times>0, cambia de color el número de veces indicado (Por ejemplo, original – nuevo – original serían dos veces. El color nuevo se escoge al azar entre los disponibles 8) . ¡Pruébenlo! Capturen dos pantallas.

```
public void blink(int times){
    if(times > 0){
        for(int i = 0; i < times; i++){
            int aleatorio = random.nextInt(6);
            this.color = colorList[aleatorio];
            if(this.color == colorList[aleatorio]){
                aleatorio = random.nextInt(6);
                this.color = colorList[aleatorio];
                draw();

                if(i < times-1){
                    this.color = "white";
                    draw();
                    draw();
                }

            }
            draw();
        }
    }
}
```

4. Desarrollen en Triangle un nuevo creador que dada el área crea un triángulo al azar con dicha área.

```java
/**
 * Crea un triangulo al azar a partir del area dada
 * @param area, el area del triangulo deseado
 */
public Triangle(int area){
    this.area = area;
    this.width = random.nextInt(150);
    this.height = (2 * area) / this.width;
    xPosition = 140;
    yPosition = 150;
    color = "green";
    isVisible = false;
}
```

5. Propongan un nuevo método para esta clase. Desarrollen y prueben el método.

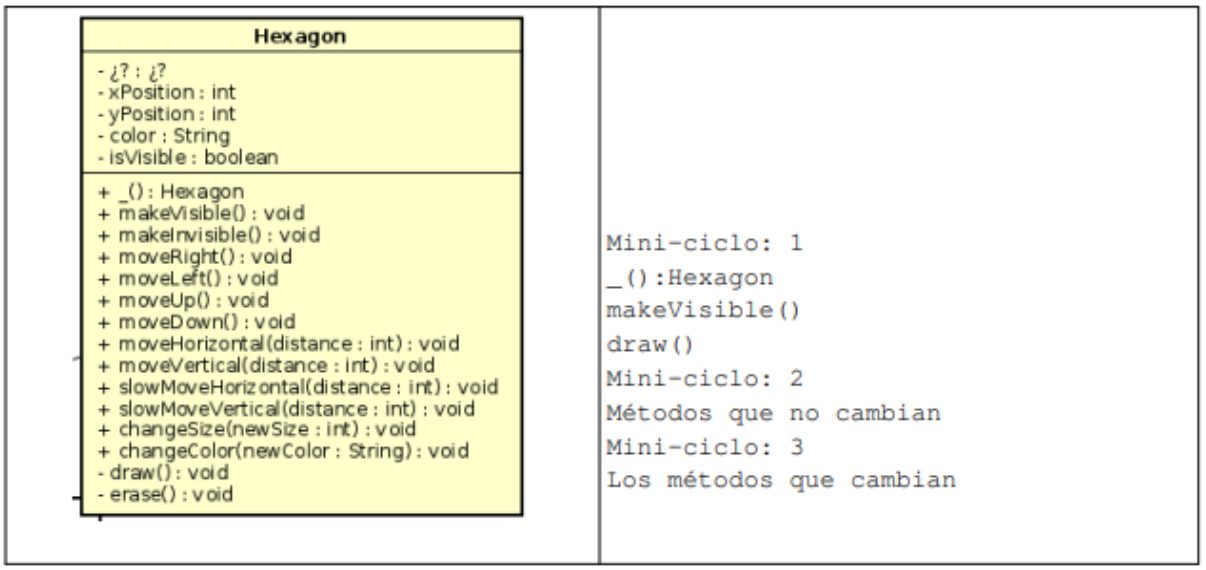
```java
/**
 * El triangulo hara espirales dejando un rastro de puntos
 * en la esquina mas alejada de su punto de inicio
 * @param cantidad, numero de espirales que hara
 */
public void espiral(int cantidad){
    int distance = 100;
    for(int i = 0; i < cantidad; i++){
        slowMoveVertical(-distance);
        slowMoveHorizontal(-distance);
        punto = new Circle(this.xPosition, this.yPosition);
        punto.changeSize(15);
        punto.changeColor(this.color);
        punto.makeVisible();
        slowMoveVertical(distance);
        slowMoveHorizontal(distance);
        randomChangeColor();
        distance -= 25;
    }
}
```

6. Generen nuevamente la documentación.

HECHO!

E. Extendiendo un paquete. shapes + Hexagon

En este punto vamos a adicionar una nueva forma al paquete shapes: un hexágono regular. Los hexágono regulares tienen todos sus lados iguales.



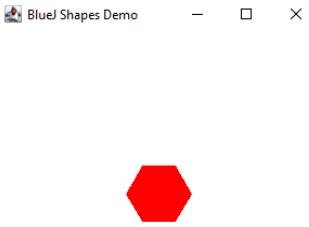
1. Para la construcción, tomen como base la clase Triangle inicial.
    a. ¿Qué métodos podrían quedar iguales?
        - makeVisible();
        - makeInvisible();
        - moveRight();
        - moveLeft();
        - moveUp();
        - moveDown();
        - moveHorizontal();
        - moveVertical();
        - slowMoveHorizontal();
        - slowMoveVertical();
        - changeColor();
        - erase();

    b. ¿Qué atributos podrían quedar iguales?
        - xPosition
        - yPosition
        - color
        - isVisible
    c. Justifiquen su respuesta.
        Esto es debido a que sus atributos y métodos son generales y no específicos de Hexagon, los cuales comparte con Triangle.

2. Inicie la construcción de la clase Hexagon únicamente con los atributos.
    a. Adicione pantallazo con los atributos.

```
private int sides = 6;
private int radio = 30;
private int xPosition;
private int yPosition;
private String color;
private boolean isVisible;
```

3. Desarrollen la clase Hexagon considerando los 3 mini-ciclos. Al final de cada miniciclo realicen una prueba indicando su propósito. Capturen las pantallas relevantes
    - Mini - ciclo: 1



    - Mini - ciclo: 2
      Ya se nombraron anteriormente los métodos que quedan iguales.

    - Mini - ciclo: 3

```
/**
 * Change the size to the new size
 * @param newSize the new radio in pixels. newSize must be >=0.
 */
public void changeSize(int newSize) {
    erase();
    if(newSize >=0){
        this.radio = newSize;
        draw();
    }
}
```

```
/*
 * Draw the hexagon with current specifications on screen.
 */
private void draw(){
    if(isVisible) {
        int[] xpoints = new int[sides];
        int[] ypoints = new int[sides];
        for(int i = 0; i < sides; i++){
            double theta = 2 * Math.PI * i / sides;
            xpoints[i] = (int) (xPosition + radio * Math.cos(theta));
            ypoints[i] = (int) (yPosition + radio * Math.sin(theta));
        }
        Canvas canvas = Canvas.getCanvas();
        canvas.draw(this, color, new Polygon(xpoints, ypoints, 6));
        canvas.wait(10);
    }
}
```

F. Definiendo y creando una nueva clase. Replicate

El objetivo de este trabajo es programar una mini-aplicación para Replicate.

Requisitos funcionales

- Crear un replicate indicando su dimensión.
- Llenar una celda. Deben ofrecer dos formas de crear la pieza:
    a. dada su posición
    b. al azar Replicar. Debe seguir las reglas.
- Reflejar vertical u horizontalmente el replicate
-

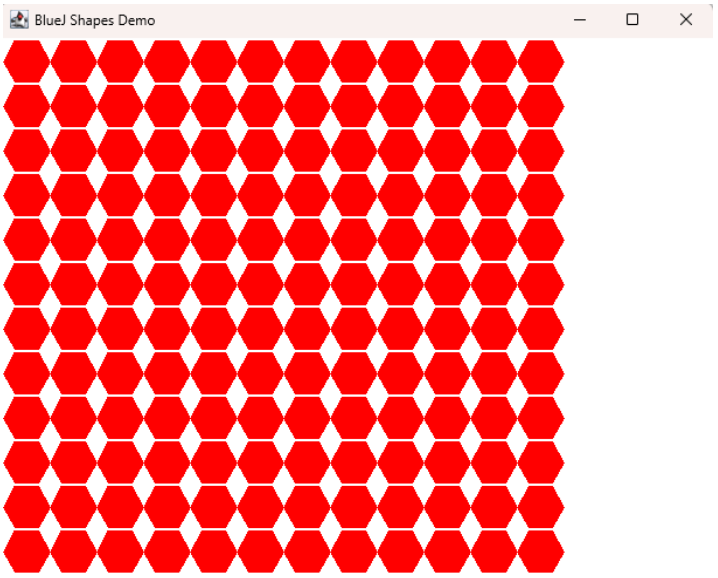
Requisitos de interfaz

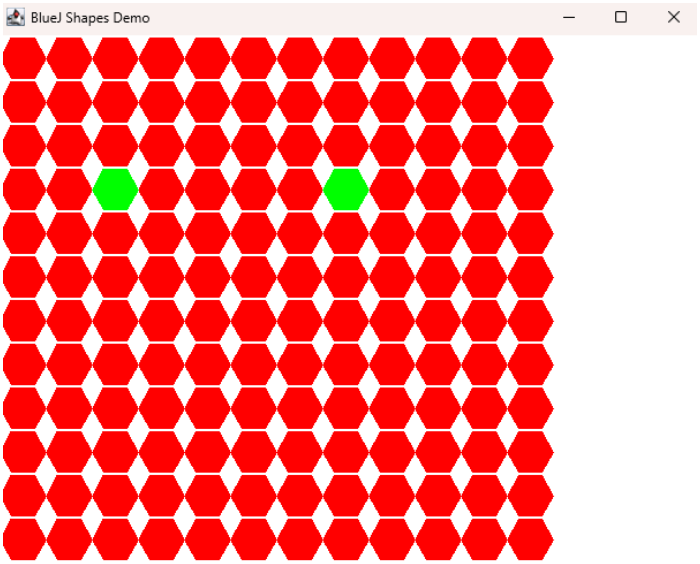
- Las celdas se identifican por su fila y su columna.
- En caso que no sea posible realizar una de las acciones, se debe generar un mensaje de error. Use JoptionPane.

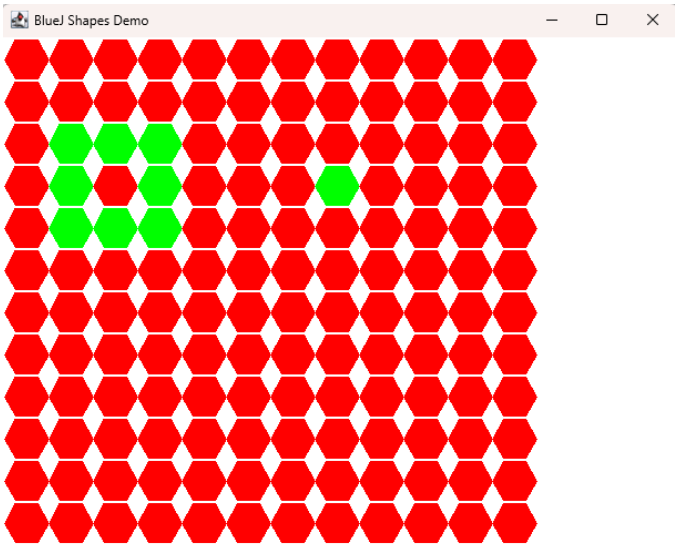
1. Diseñen la clase, es decir, definan los métodos que debe ofrecer.

```
void makeInvisible()
void makeVisible()
void pieza()
void pieza(int iPosition, int jPosition)
void replicar()
```

2. Planifiquen la construcción considerando algunos mini-ciclos.
3. Implemente la clase. Al final de cada mini-ciclo realicen una prueba indicando su propósito. Capturen las pantallas relevantes.







4. Indiquen las extensiones necesarias para reutilizar el paquete shapes y la clase Hexagon. Explique.

   Las extensiones que se usaron fue como tal creación de objetos “Hexagon” para poder crear un arreglo de Hexágonos el cual sería el tablero principal.

Retrospectiva

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes? (Horas/ Hombre)
   Joan Acevedo = 11 horas
   Santiago Sanchez = 7 horas

2. ¿Cuál es el estado actual del laboratorio? ¿Por qué?
   Se cumplió con la gran mayoría de requisitos, se podría decir que su estado es aproximadamente de 94%. Esto es debido a que la función de “Replicar” no funciona al 100% de la forma deseada.

3. Considerando las prácticas XP del laboratorio. ¿Cuál fue la más útil? ¿por qué?
   Hasta el momento solo hemos visto una práctica XP (trabajo en parejas), considero que es la más importante ya que si se realiza una correcta aplicación de esta se puede llegar muy lejos.

4. ¿Cuál consideran fue el mayor logro? ¿Por qué?
   Realizar la matriz de hexágonos y poder generar piezas dentro de ella, esto es debido a que fue uno de los requisitos que más nos exige.

5. ¿Cuál consideran que fue el mayor problema técnico? ¿Qué hicieron para resolverlo?
   Que la función “Replicar” no funcionó como se deseaba en su totalidad. Lastimosamente no se logró solucionar.

6. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los resultados?
   Se hizo lo posible por realizar una correcta práctica XP (un buen trabajo en pareja), sin embargo, aún se debe mejorar ya que no se logró tener el rendimiento deseado como compañeros de trabajo.