

PROGRAMACIÓN ORIENTADA A OBJETOS

Introducción. Clases y objetos.

2020-2

Laboratorio 1/6

OBJETIVOS

Desarrollar competencias básicas para:

1. Apropiar un paquete de clases 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 java1
5. Utilizar el entorno de desarrollo de BlueJ
6. Vivenciar las prácticas XP : Planning The project is divided into iterations, Coding All production code is pair programmed.

ENTREGA

-> Incluyan en un archivo .zip los archivos correspondientes al laboratorio. El nombre debe ser los dos apellidos de los miembros del equipo ordenados alfabéticamente.

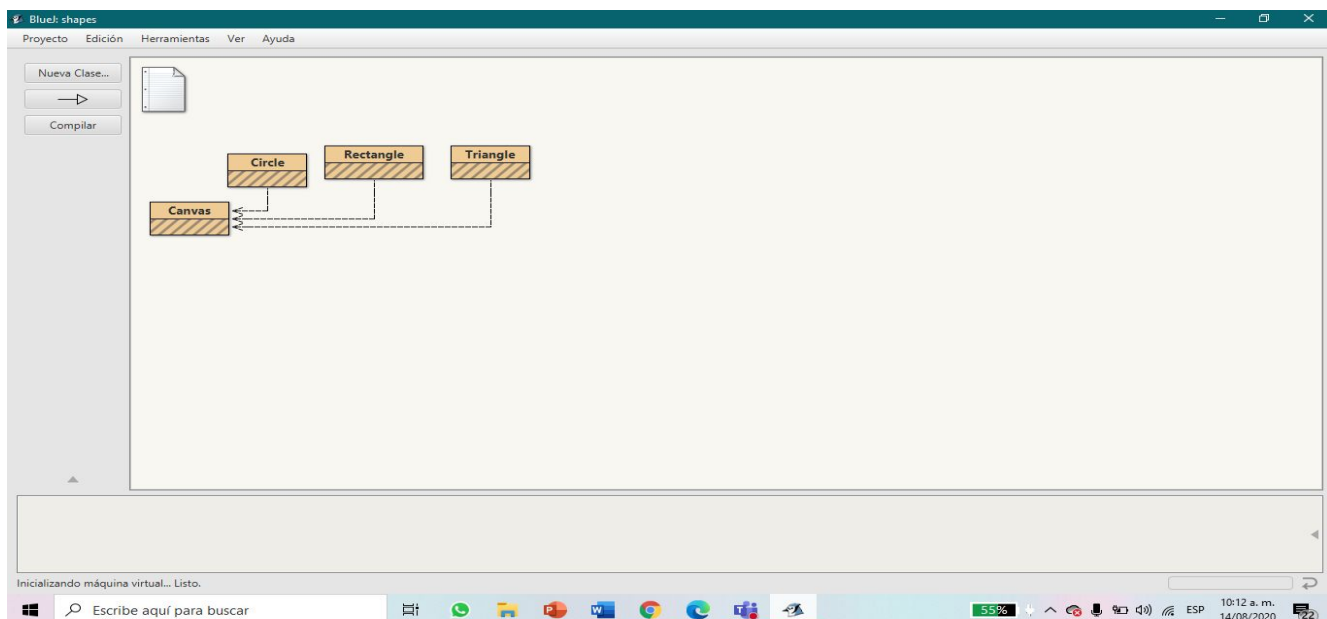
☐ Deben publicar el avance al final de la sesión y la versión definitiva en la fecha indicada en los espacios correspondientes.

SHAPES

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 ábralo en BlueJ.



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?

Circle, Rectangle, Triangle.

(b) ¿Qué relaciones existen entre ellas?

Todas las clases se relacionan en el canvas entre ellas no hay relación alguna.

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, circle, rectangle, triangle.

(b) ¿Qué atributos ofrece la clase Circle?

En la documentación no nos muestran los atributos.

(c) ¿Cuántos métodos ofrece la clase Circle?

12 métodos.

(d) ¿Cuáles métodos ofrece la clase Circle para que la figura cambie (incluya sólo el nombre)?

changeColor, changeSize.

4. En el código de cada clase está el detalle de la implementación. Revisen el código de la clase Circle. Con respecto a los atributos:

(a) ¿Cuántos atributos realmente tiene?

6 atributos.

(b) ¿Cuáles atributos describen la forma del círculo?

Diámetro.

Con respecto a los métodos:

(c) ¿Cuántos métodos tiene en total?

14 métodos.

(d) ¿Quiénes usan los métodos privados?

La clase canvas y el programador.

5. Comparando la documentación con el código

(a) ¿Qué no se ve en la documentación?

En la documentación no se ven los atributos y tampoco las condiciones que tienen los métodos.

(b) ¿por qué debe ser así?

Porque en la documentación nos dan la información de como es el objeto más no como debe programarse.

6. En el código de la clase Circle revisen el detalle del atributo PI.

(a) ¿Qué se está indicando al decir que es static?

Es un valor que ocupa un solo espacio en la memoria, es decir, al llamarlo, no se genera un nuevo valor, sino que simplemente se le hace un llamado al ya creado.

(b) ¿Cómo decimos que PI es una constante?

Porque el valor no puede modificarse y ya se encuentra guardado en la memoria al definirse su tipo static.

(c) Actualícenlo.

Al actualizar el dato de PI no se genera ninguna modificación porque es una constante teórica.

7. En el código de la clase Circle revisen el detalle del tipo del atributo diameter.

(a) ¿Qué se está indicando al decir que es int?

Que el valor del diámetro siempre debe ser un entero, no puede ser un decimal o una letra.

(b) Si sabemos que todos nuestros círculos van a ser muy pequeños (diámetro menor a 100), ¿de qué tipo podría ser diameter?

De tipo int debido a que al ser tan pequeño los decimales no afectarían mucho a la forma.

Si algunos de nuestros círculos pueden ser muy grandes (diámetro mayor a 220000000),

(c) ¿de qué tipo debería ser diameter?

De tipo Double debido a que al ser muy grande la diferencia entre decimales afectaría a su tamaño.

(d) ¿qué restricción tendría? Expliquen claramente sus respuestas.

- Como el diámetro es Int, no podría ser ni un carácter ni un número decimal sino un número entero.
- Tampoco podría ser un diametro el cual haga que nuestro círculo sea mayor a la establecida en la imagen, puesto que no logramos ver dicho círculo.

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

Aprender a identificar los métodos, las clases, los atributos, y sus funcionamientos.

Manipulando objetos. Usando opciones.

1. Creen un objeto de cada una de las clases que lo permitan

(a) ¿Cuántas clases hay?

Hay 3 clases que permiten crear objetos.

¿Cuántos objetos crearon?

Un objeto de cada clase (3).

(b) ¿Por qué?

Porque queríamos visualizar cada objeto.

2. Inspeccionen el estado del objeto

¿Cuáles son los valores de inicio de todos sus atributos? Capture la pantalla.

```
diameter = 30;  
xPosition = 20;  
yPosition = 15;  
color = "blue";  
isVisible = false;
```

circle1 : Circle

private int diameter	30	Inspeccionar Obtener
private int xPosition	20	
private int yPosition	15	
private String color	"blue"	
private boolean isVisible	false	

Mostrar campos estáticos

Cerrar

3. Inspeccionen el comportamiento que ofrece el objeto:

(a) Capturen la pantalla.

heredado de Object

```
void changeColor(String newColor)  
void changeSize(int newDiameter)  
void makeInvisible()  
void makeVisible()  
void moveDown()  
void moveHorizontal(int distance)  
void moveLeft()  
void moveRight()  
void moveUp()  
void moveVertical(int distance)  
void slowMoveHorizontal(int distance)  
void slowMoveVertical(int distance)
```

(b) ¿Por qué no aparecen todos los que están en el código?

Por ser privados.

4. Construyan, con “shapes” sin escribir código, una propuesta de la imagen de su cómic favorito.

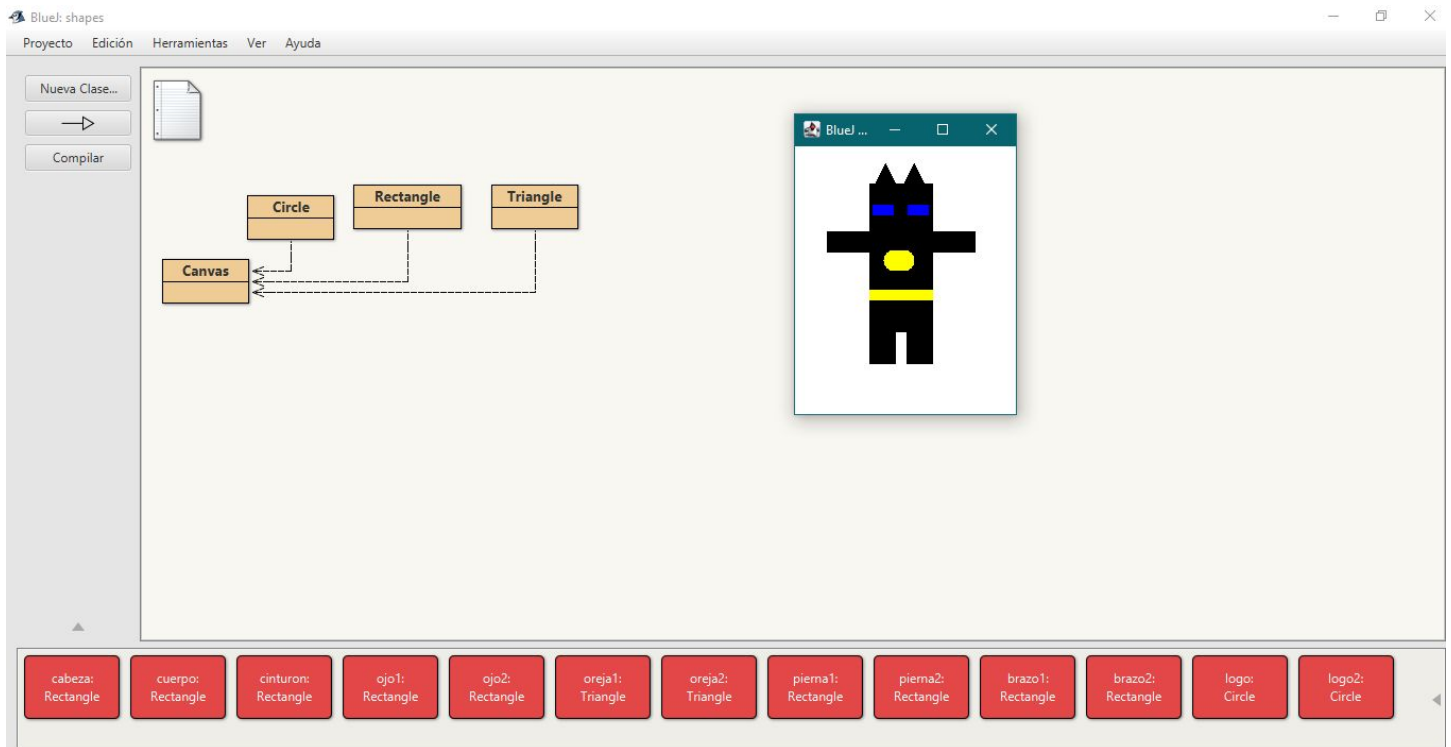


(a) ¿Cuántas y cuáles clases se necesitan?

Se necesitan 4 clases, cuadrado, círculo, rectángulo, canvas.

¿Cuántos objetos se usan en total? Capturen la pantalla.

13 objetos



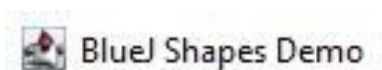
Manipulando objetos. Analizando y escribiendo código.

1. Lean el código anterior.

(a) ¿cuál es la figura resultante?

La figura resultante son dos círculos negro juntos, uno más pequeño que el otro.

(b) Píntenla.



2. Habiliten la ventana de código en línea, escriban el código. Para cada punto

señalado indiquen:

(a) ¿cuántas variables existen?

Existen tres variables

(b) ¿cuántos objetos existen?

Existen dos objetos

(c) ¿qué color tiene cada uno de ellos?

Ambos son de color negro

(d) ¿cuántos objetos se ven?

Ambos objetos se ven

Al final, (e) Expliquen sus respuestas. (f) Capturen la pantalla.

Son dos objetos porque solo manejamos face y uno más con las dos variables left y right, adicionalmente le cambiamos el color, tamaño y posición al círculo, haciendo a ambos diferentes y visibles.

```
Circle face;
```

Nota: Las variables Codepad son automáticamente iniciadas de la misma forma que campos de instancia.

```
Circle leftEar;
```

```
Circle rightEar;
```

```
face=new Circle();
```

```
face.changeSize(50);
```

```
face.moveVertical(20);
```

```
face.changeColor("black");
```

```
face.makeVisible();
```

```
leftEar=new Circle();
```

```
leftEar.changeColor("black");
```

```
leftEar.moveHorizontal(-10);
```

```
rightEar=leftEar;
```

```
rightEar.moveHorizontal(30);
```

```
rightEar.makeVisible();
```

```
leftEar.makeVisible();
```

3. Compare figura pintada en 1. con la figura capturada en 2. ,

(a) ¿son iguales?

No son imágenes iguales

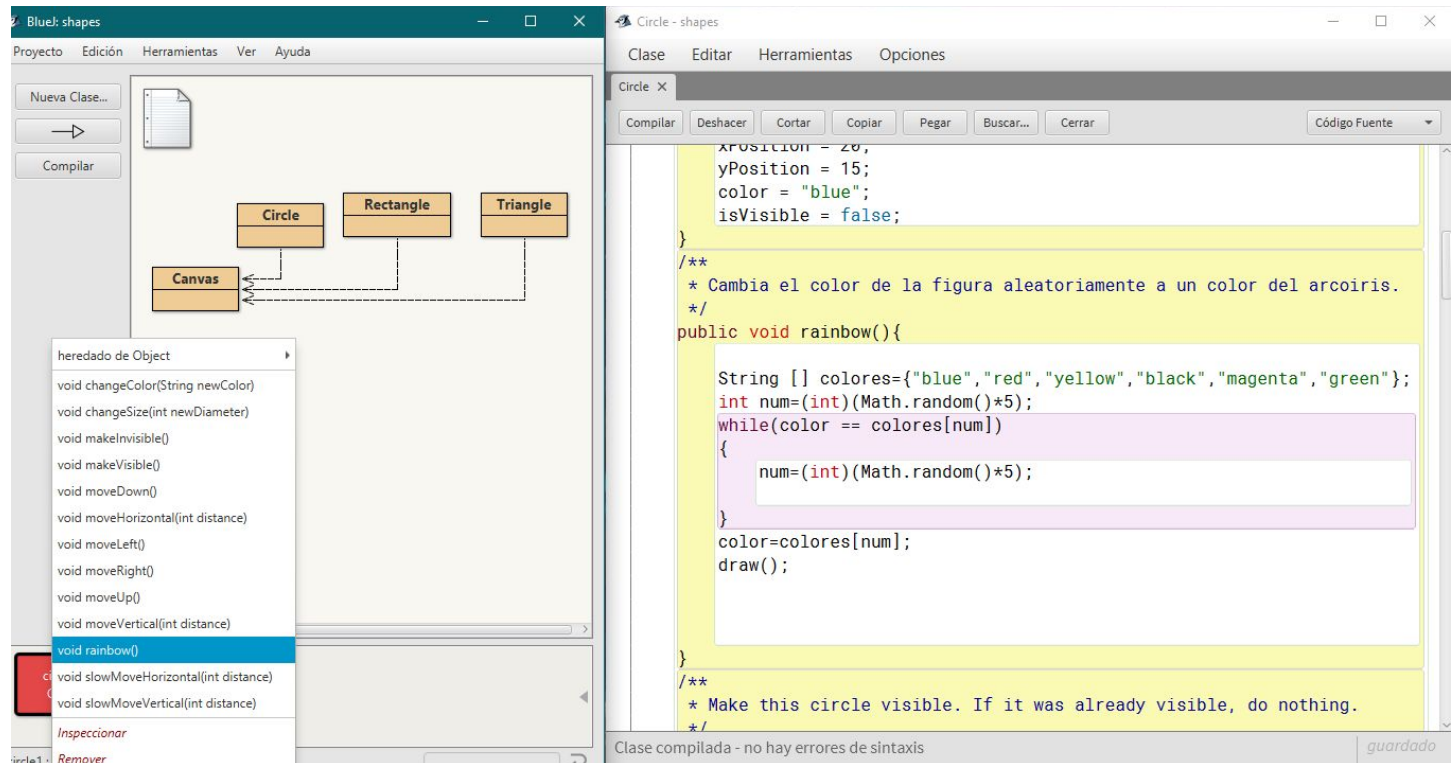
(b) ¿por qué?

Porque ambas tienen el mismo tamaño pero diferente color y posición.

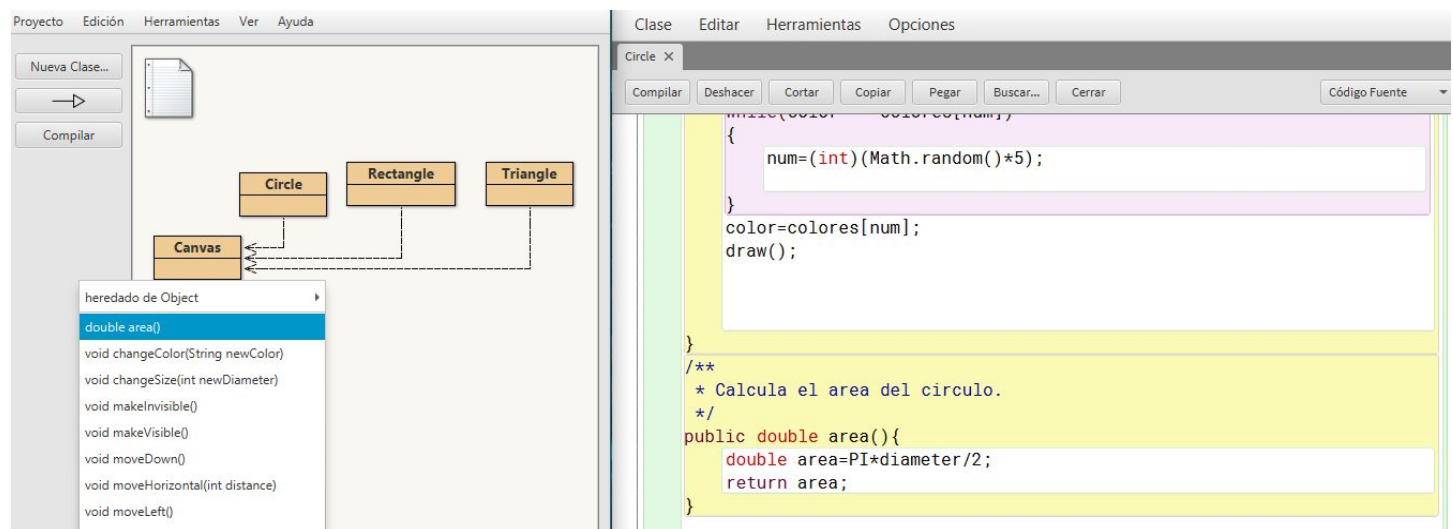
Extendiendo clases

1. Desarrollen en Circle el método rainbow() (pasa por los colores del arco iris) .

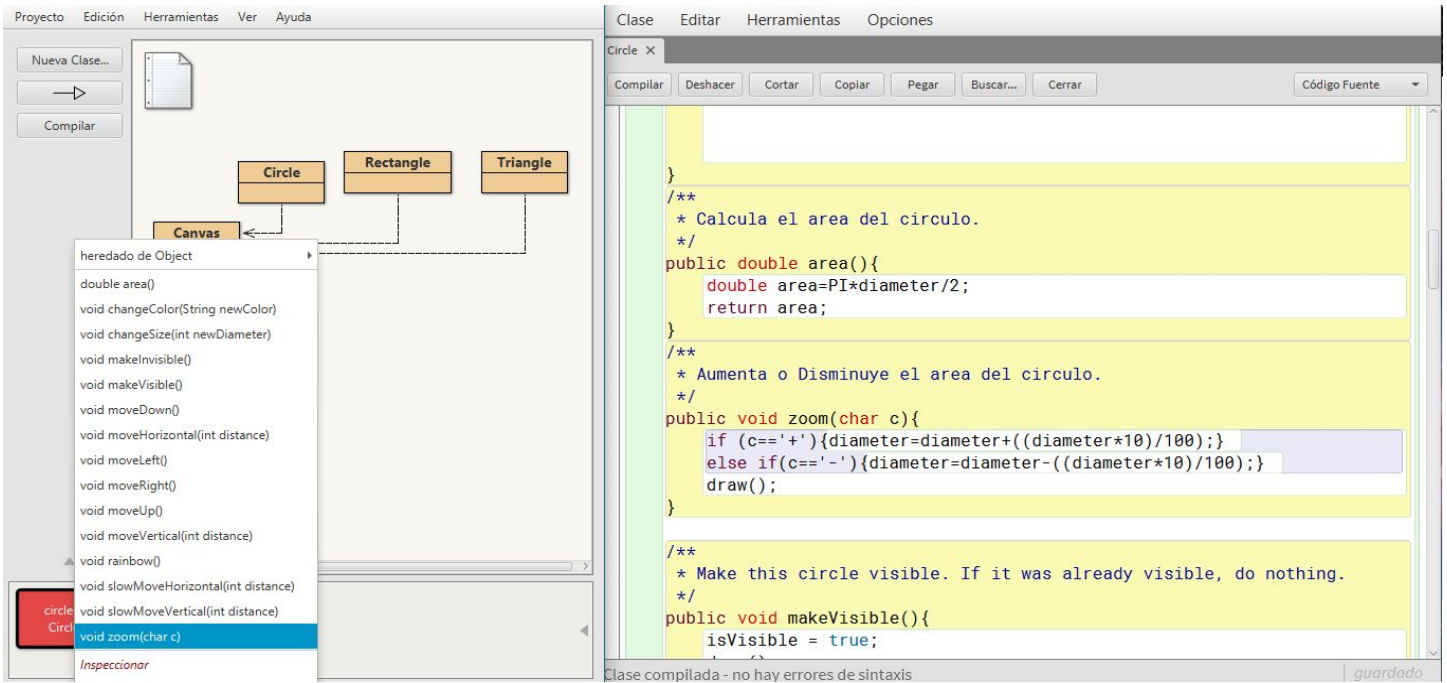
¡Pruébenlo! Capturen dos pantallas.



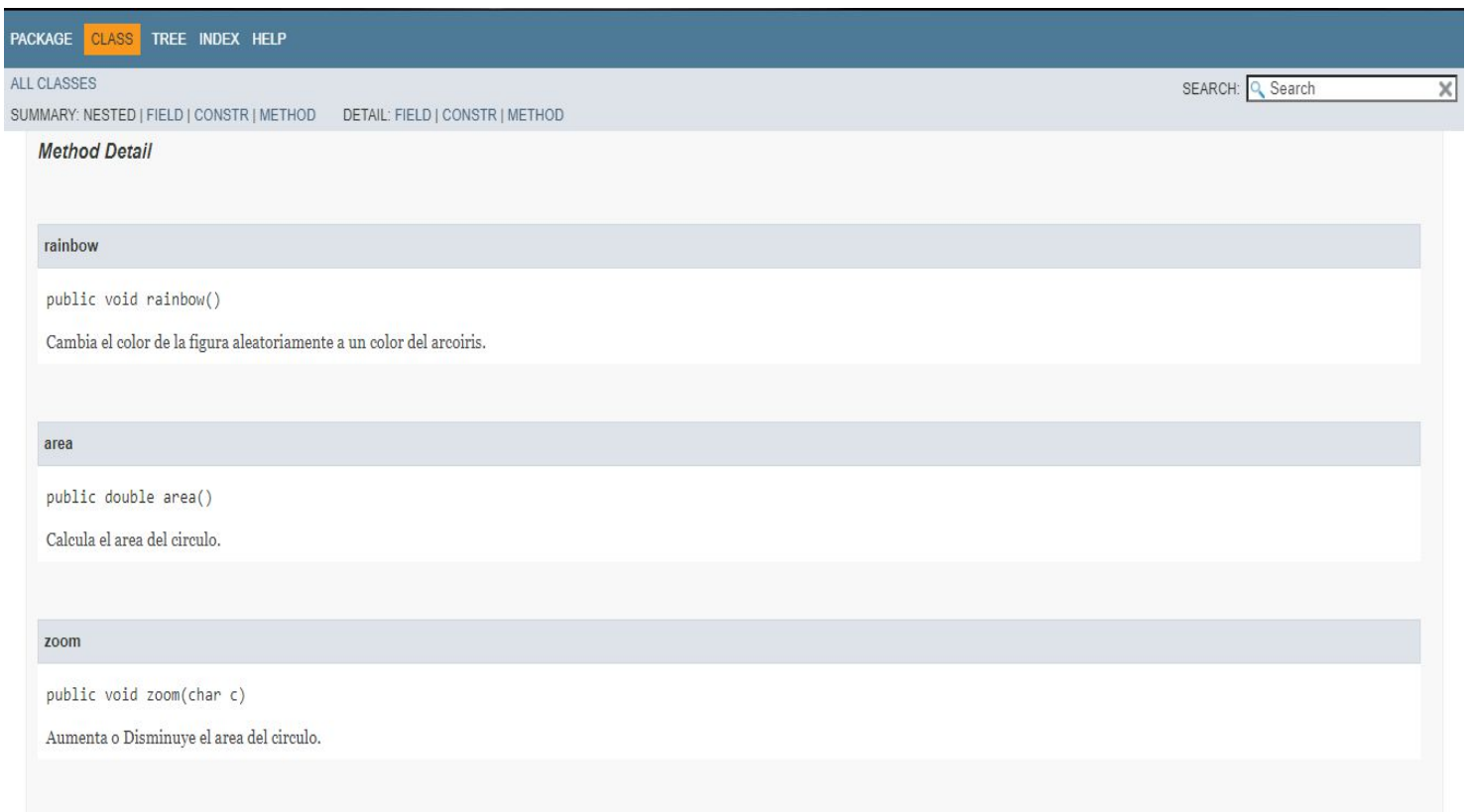
2. Desarrollen en Circle el método área(). ¡Pruébenlo! Capturen una pantalla.



3. Desarrollen en Circle el método zoom(char c) (aumenta (+) y disminuye (-) 10% de su area) . ¡Pruébenlo! Capturen dos pantallas.



4. Generen nuevamente la documentación y revise la información de estos nuevos métodos. Capturen la pantalla.



SELF-ASSEMBLY. Pieces.

Durante las Finales Mundiales ACM ICPC del 2017 en Marrakesh, uno de los jueces compró un bonito rompecabezas de madera. A diferencia de los tradicionales, que son creados cortando una imagen rectangular existente, todas las piezas de este rompecabezas han sido cortadas y pintadas por separado. Dadas estas propiedades, la forma de cada pieza individual es la única manera de saber dónde se debe colocar cada pieza. El juez quiere saber si es posible escribir un programa para resolver este rompecabezas

[De 2016 WorldFinals Problem H Polygonal Puzzle]

NO DEBEN RESOLVER EL PROBLEMA DE LA MARATÓN



LAS PIEZAS VAN A SER ESPECIALES

Para crearlas se darán las esquinas superior derecha de los rectángulos que la conforman de izquierda a derecha

figura = {rectangulo1, rectangulo2, ... ,rectangulon}
rectangulo={x, y}



Implementando una nueva clase. Piece.

Piece
<pre> + _(color : String, points : int[][]) : Piece + take() : void + put() : void + reflect(d : char) : void + rotate() : void + resize(percentage : int) : void + translate(dx : int, dy : int) : void </pre>

Una pieza recién creada queda en la mano en la posición (0,0).
Las pieza que está en la mano debe tener una señal que la distinga.
La pieza se refleja sobre sus cuatro lados: N, S, E, W. (Norte, Sur, Este, Oeste)
La rotación siempre es de 90 grados en sentido de las agujas del reloj.

MINICICLOS

7. _(color:String, points: int[][])
8. take()
put()
9. reflect(d: char)
rotate()
10. resize(percentage:int)
translate(dx:int,dy:int)

1. Revisen el diseño y clasifiquen los métodos en: constructores, analizadores y modificadores.

Constructor:_(color: String,points: int[][])

Analizadores:take(),put()

Modificadores:reflect(d:char),rotate(),resize(percentage:int),translate(dc:int,dy:int)

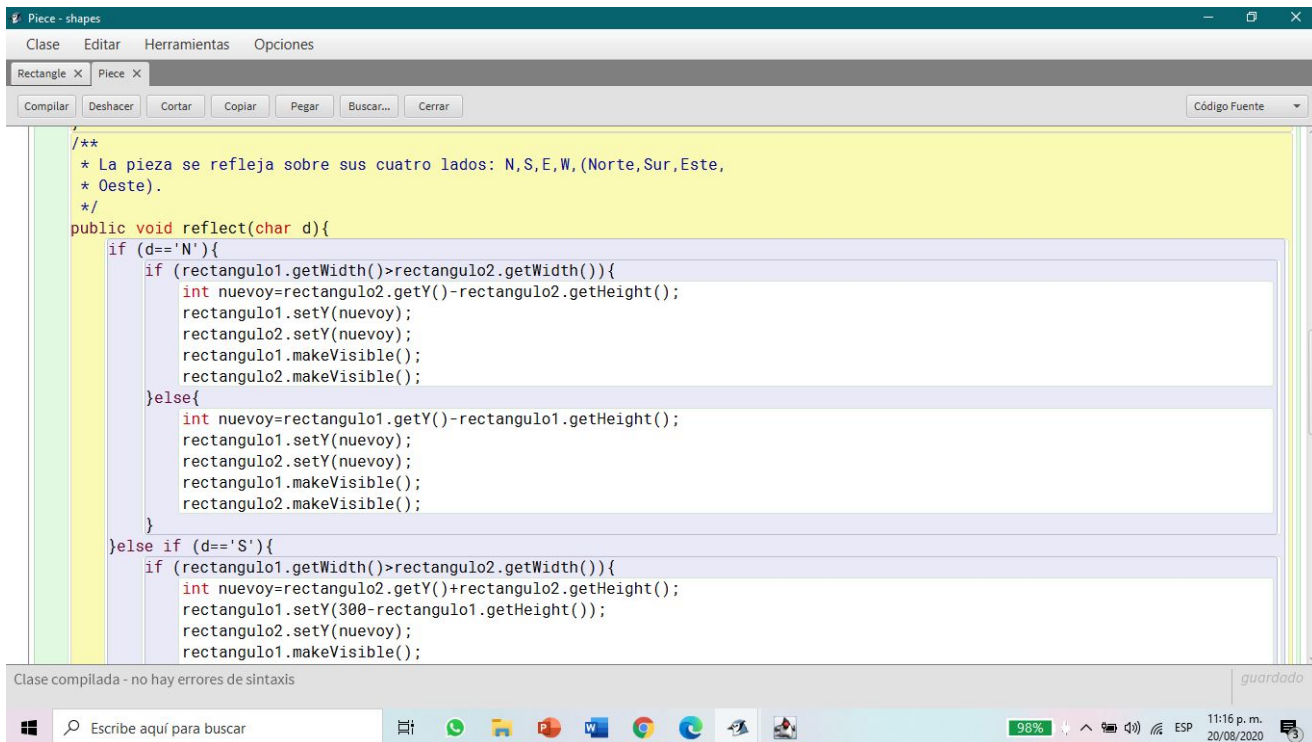
2. Desarrollen la clase **Pieza** considerando los mini-ciclos. Al final de cada mini-ciclo realicen una prueba indicando su propósito. Capturen las pantallas relevantes.

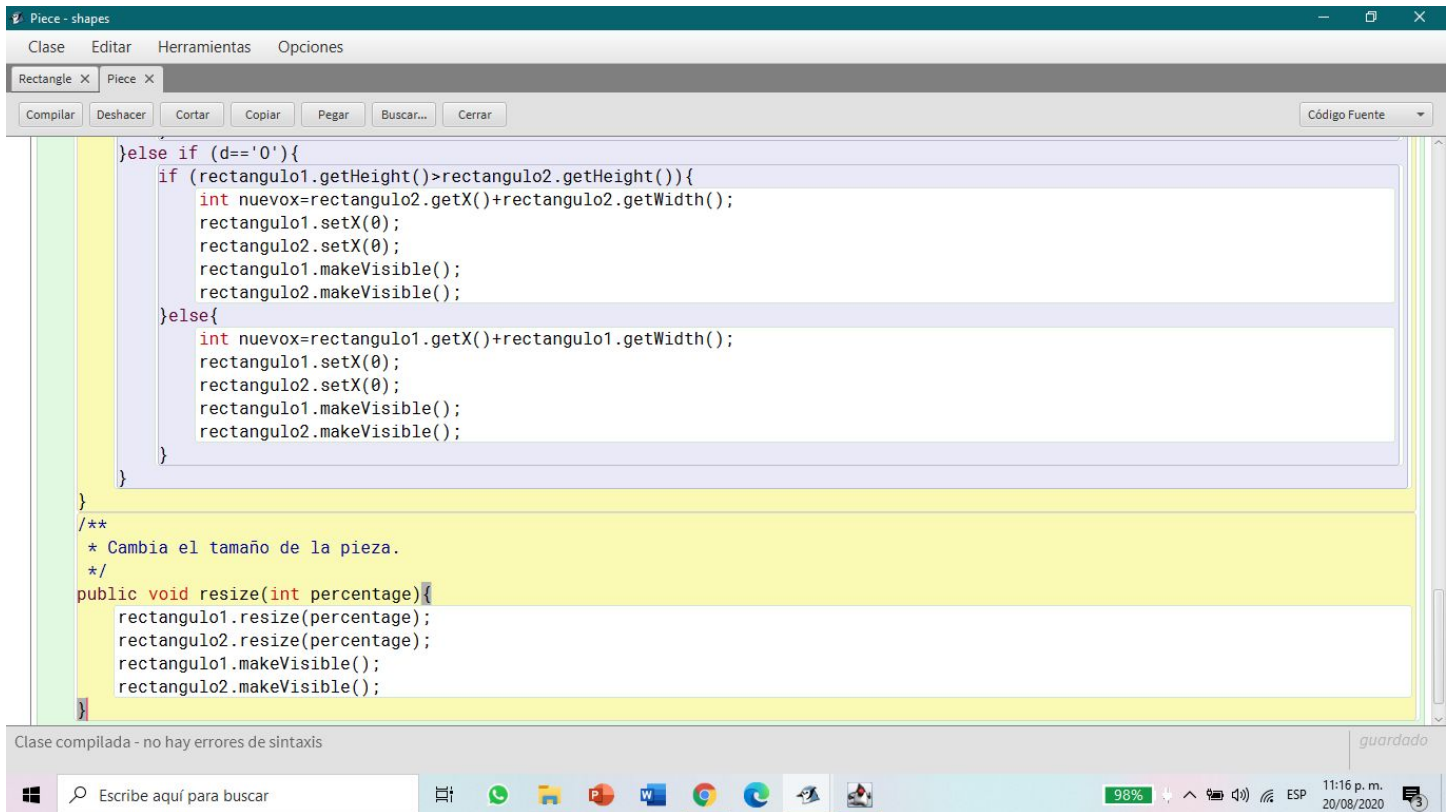
```
* Las piezas seran especiales, se daran las esquinas superior derecha de
* los rectangulos que la conforman de izquierda a derecha.
*
* @author (Juan Esteban Aaron-Natalia Orjuela)
* @version (19 de agosto 2020)
*/
public class Pieza
{
    // instance variables - replace the example below with your own
    private Rectangle rectangulo1;
    private Rectangle rectangulo2;

    /**
     * Construye la pieza utilizando la clase Rectangulo, el color y las 7
     * coordenadas las ingresa el usuario, y estas son enviadas a la clase.
     */
    public Pieza(String color, int[][] points)
    {
        // initialise instance variables
        rectangulo1 = new Rectangle(color, points[0][0], points[0][1]);
        rectangulo2 = new Rectangle(color, points[1][0], points[1][1]);
        rectangulo1.makeVisible();
        rectangulo2.makeVisible();
    }
}
```

```
/**
 * Quita la ficha del tablero.
 */
public void take()
{
    rectangulo1.makeInvisible();
    rectangulo2.makeInvisible();
}

/**
 * Pone la ficha en el tablero.
 */
public void put()
{
    rectangulo1.makeVisible();
    rectangulo2.makeVisible();
}
```





```

}else if (d=='0'){
    if (rectangulo1.getHeight()>rectangulo2.getHeight()){
        int nuevox=rectangulo2.getX()+rectangulo2.getWidth();
        rectangulo1.setX(0);
        rectangulo2.setX(0);
        rectangulo1.makeVisible();
        rectangulo2.makeVisible();
    }else{
        int nuevoy=rectangulo1.getX()+rectangulo1.getWidth();
        rectangulo1.setX(0);
        rectangulo2.setX(0);
        rectangulo1.makeVisible();
        rectangulo2.makeVisible();
    }
}

/**
 * Cambia el tamaño de la pieza.
 */
public void resize(int percentage){
    rectangulo1.resize(percentage);
    rectangulo2.resize(percentage);
    rectangulo1.makeVisible();
    rectangulo2.makeVisible();
}

```

RETROSPECTIVA

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes?

Cada uno invirtió 10 horas.

2. ¿Cuál es el estado actual del laboratorio? ¿Por qué?

Está incompleto debido a que el laboratorio estaba muy extenso y estábamos aprendiendo del lenguaje ya que era la primera vez que estábamos programando en este lenguaje y aprendiendo a manejar bien el ambiente de desarrollo.

3. Considerando las prácticas XP del laboratorio. ¿cuál fue la más útil? ¿por qué?

La programación por pares, porque el progreso es mayor y la detección de errores se facilita mucho.

4. ¿Cuál consideran fue el mayor logro? ¿Por qué?

La creación de la clase pieza ya que nos permitió diferenciar las clases, los métodos, utilizar objetos.

5. ¿Cuál consideran que fue el mayor problema técnico? ¿Qué hicieron para resolverlo?

La no presencialidad, y la solución fue el uso de herramientas como teams, zoom.

6. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los resultados? }

Como equipo nos ayudamos mucho el uno al otro para solucionar los inconvenientes que se nos presentaban, mantuvimos contacto en toda la elaboración del proyecto.

Nos comprometemos a trabajar el laboratorio con más tiempo para lograr terminar todos los puntos.