

LABORATORIO #1

JULIAN CAMILO LOPEZ BARRERO

JUAN SEBASTIAN PUENTES JULIO

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

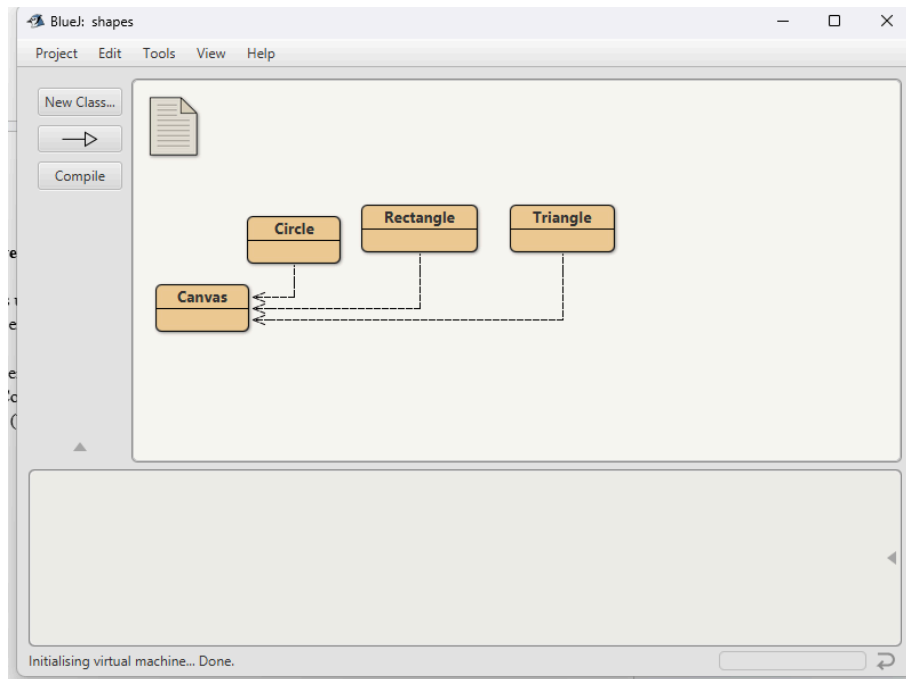
POOB

BOGOTÁ, COLOMBIA

07 DE FEBRERO DE 2025

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 ábralo en BlueJ2. 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?

Ofrece las “Circle”, “Rectangle”, “Triangle” y “Canvas”.

(b) ¿Qué relaciones existen entre ellas?

Las relaciones que existen entre ellas son que “Circle”, “Rectangle” y “Triangle” van hacia el canvas que es donde podremos visualizar las distintas clases o figuras mostradas en pantalla

3. La documentación. 3 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?

Tiene 4 clases, las cuales son:

Circle
Triangle
Rectangle
Canvas

(b) ¿Qué atributos tiene la clase Triangle?

Los atributos no se pueden determinar en la documentación.

(c) ¿Cuántos métodos ofrece la clase Triangle?

Ofrece 13 métodos. Entre ellos mover el triángulo horizontalmente, como hacer visible este y así mismo cambiar su tamaño.

(d) ¿Qué atributos determinan el tamaño de un Triangle?

Los atributos no se pueden determinar en la documentación.

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

Los métodos que ofrece la clase Triangle para cambiar su tamaño son:

1. changeSize

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?

Tiene realmente 6 atributos

```
public class Triangle{  
  
    public static int VERTICES=3;  
  
    private int height;  
    private int width;  
    private int xPosition;  
    private int yPosition;  
    private String color;  
    private boolean isVisible;
```

(b) ¿Quiénes pueden usar los atributos públicos?. Con respecto a los métodos:

(c) ¿Cuántos métodos tiene en total?

Revisando el código a detalle, este realmente tiene 14 métodos. Los restantes son Draw y Erase.

(d) ¿Quiénes usan los métodos privados?

Los métodos privados los usan draw y erase sus funciones respectivamente son dibujar el triángulo y borrar el mismo.

```
/*
 * Draw the triangle with current specifications on screen.
 */
private void draw(){
    if(isVisible) {
        Canvas canvas = Canvas.getCanvas();
        int[] xpoints = { xPosition, xPosition + (width/2), xPosition - (width/2) };
        int[] ypoints = { yPosition, yPosition + height, yPosition + height };
        canvas.draw(this, color, new Polygon(xpoints, ypoints, 3));
        canvas.wait(10);
    }
}

/*
 * Erase the triangle on screen.
 */
private void erase(){
    if(isVisible) {
        Canvas canvas = Canvas.getCanvas();
        canvas.erase(this);
    }
}
```

5. Comparando la documentación con el código

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

En la documentación no se ven los métodos privados y los atributos de los métodos.

(b) ¿Por qué debe ser así?

Para que consultores externos no puedan acceder a estos métodos y los atributos por el concepto de encapsulamiento que oculta los detalles de implementación de una clase o componente detrás de una interfaz.

6. En el código de la clase Triangle, revise el atributo VÉRTICES

(a) ¿Qué significa que sea pública?

Que otras clases van a ser capaces de usar el mismo.

(b) ¿Qué significa que sea static?

Significa que este atributo no se puede modificar..

(c) ¿Qué significa que fuera final? ¿Debe serlo?

Que no se puede modificar y asimismo restringe la modificación de variables. Debe serlo ya que por la definición matemática un triángulo tiene 3 vértices.

(d) Actualízalo.

```
public class Triangle{  
  
    public static final int VERTICES=3;  
  
    private int height;  
    private int width;  
    private int xPosition;  
    private int yPosition;  
    private String color;  
    private boolean isVisible;
```

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?

Lo que se indica al decir que es int es que es un dato de valor numérico entero.

(b) Si fuera byte, ¿cuál sería el área del Triángulo más grande posible?

El byte es un tipo de entero que abarca desde el [-128,127] así por la formula para el area del Triángulo $\frac{bxh}{2}$ reemplazamos y el area seria $\frac{127 \times 127}{2} = 8064.5$

(c) y ¿si fuera long?

El long es un tipo de entero que abarca desde el $[-2^{63}, 2^{63}-1]$ así por la fórmula para el área del Triángulo $\frac{bxh}{2}$ reemplazamos y el area seria $\frac{2^{63} \times 2^{63}}{2} = 2^{125}$

(d) ¿Qué restricción adicional deberían tener este atributo?

Que este valor sea positivo.

(e) Refactorizar el código considerando(d).

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

Creemos que el propósito de Shapes es introducir a los estudiantes a un nuevo lenguaje de programación y así mismo hacerlo intuitivo.

B. Manipulando objetos. Usando un objeto.

[En lab01.doc]

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

(a) ¿Cuántas clases hay?

Hay cuatro clases.

(b) ¿Cuántos objetos crearon?

Se crearon 4 objetos: El canvas, un círculo, un triángulo y un cuadrado.



(c) ¿Quién se crea de forma diferente? ¿Por qué?

El canvas se crea de forma diferente debido a que este es privado lo que quiere decir que se crea a sí mismo y en caso de usar el método `getCanvas` si este no existe se crea como lo decíamos anteriormente.

2. Inspeccionen los creadores de cada una de las clases.

(a) ¿Cuál es la principal diferencia entre ellos?

La principal diferencia es que El canvas se crea de forma diferente debido a que este es privado lo que quiere decir que se crea a sí mismo y las 3 figuras que tenemos son públicas.

(b) ¿Qué se busca con la clase que tiene el creador diferente?

Saber si ya existe un canvas. Si este no existe se puede crear, de lo contrario, no.

3. Inspeccionen el estado del objeto `:Triangle`

(a) ¿Cuáles son los valores de inicio de todos sus atributos?

Los valores de inicio en los atributos son:

`altura = 30`

`ancho = 40`

`xPosition = 140`

`yPosition = 12`

`color = "green"`

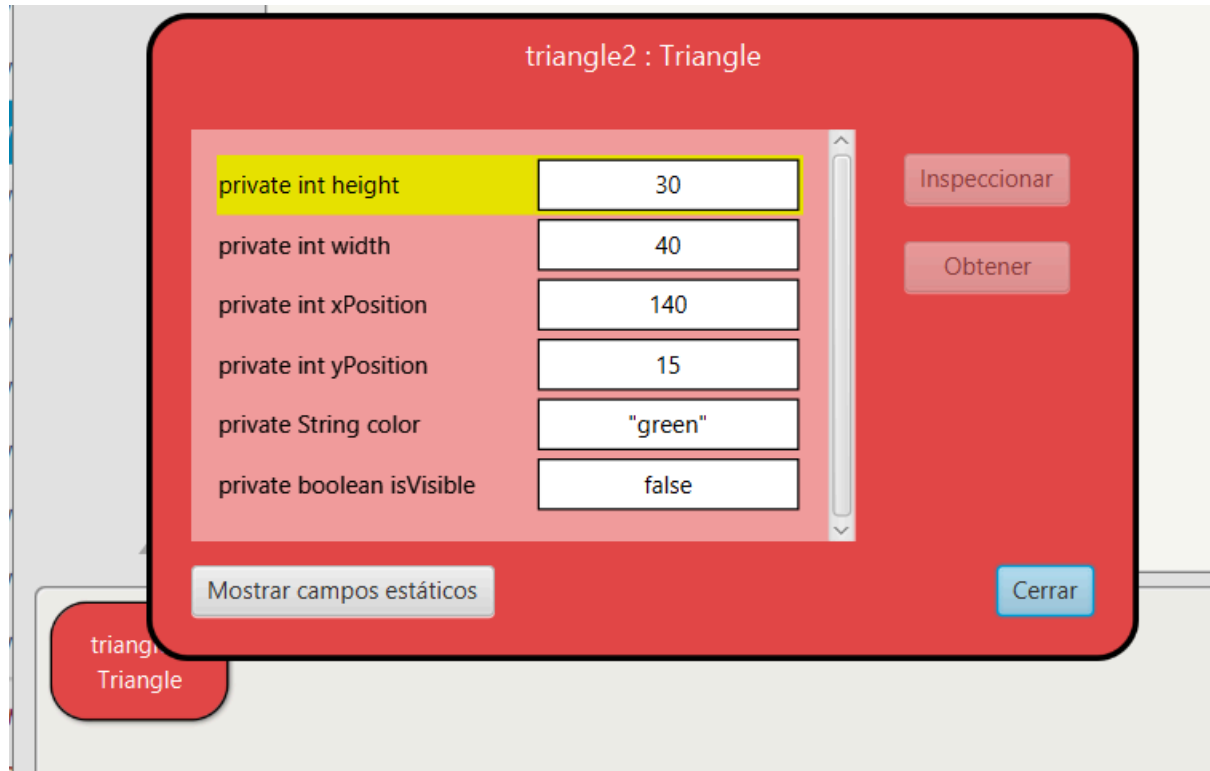
`isVisible = True`

(b) Capturen la pantalla.

```
public class Triangle{  
  
    public static int VERTICES=3;  
  
    private int height;  
    private int width;  
    private int xPosition;  
    private int yPosition;  
    private String color;  
    private boolean isVisible;
```

4. Inspeccionen el comportamiento que ofrece el objeto :Triangle.

(a) Capturen la pantalla.



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

No salen todos los que están en el código porque en este caso los vértices son de tipo static , siempre estará, esto ya que matemáticamente un triángulo tiene 3 vértices y no se puede cambiar esto.

5. Construyan, con "shapes" sin escribir código, una propuesta de la imagen del logo de de su chatbot IA favorito.

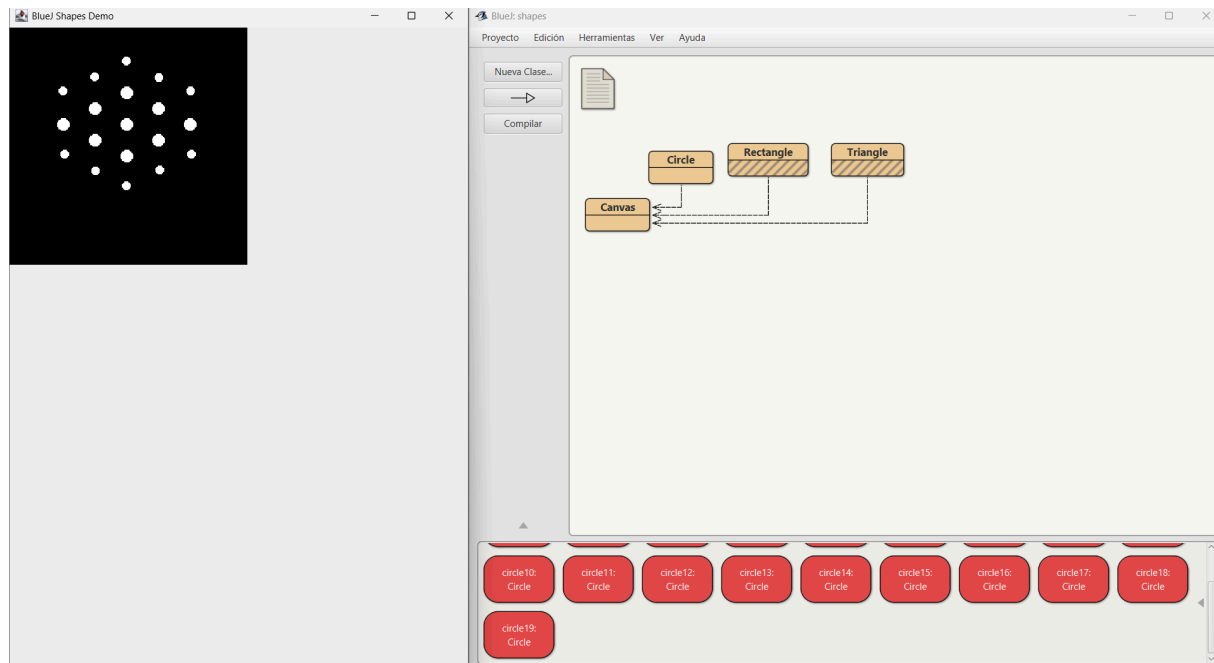
(a) ¿Cuántas y cuáles clases se necesitan?

Necesitamos el canvas que es donde vamos a plasmar el logo e igualmente los círculos que son los protagonistas en el logo

(b) ¿Cuántos objetos se usan en total?

Necesitamos 19 objetos en este caso círculos.

(c) Capturen la pantalla.



(d) Incluyan el logo original.



[BLACKBOX.AI - Apps en Google Play](#)

C. Manipulando objetos. Analizando y escribiendo código.

1. Lean el código anterior.

(a) ¿Cuál creen que es la figura resultante?

La figura puede llegar a ser una secuencia de triángulos uno debajo del otro o uno encima de otro con los colores primarios.

(b) Píntala.

2. Habiliten la ventana de código en línea 7 , escriban el código. Para cada punto señalado indiquen:

```
yellow.changeColor("yellow");
yellow.moveHorizontal(45);
yellow.moveVertical(35);
yellow.makeVisible();
```

(a) ¿cuántas variables existen?

Existen 4 variables que son blue, yellow, red y green

(b) ¿cuántos objetos existen? (no cuentan ni los objetos String ni el objeto Canvas)

Existen 4 objetos

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

Cada uno de ellos tiene el color amarillo, rojo verde y azul

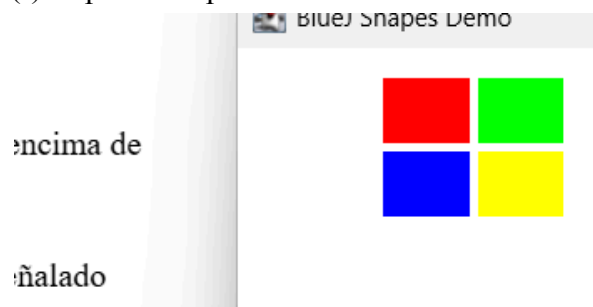
(d) ¿cuántos objetos se ven?

Contando el Canvas se ven 5 objetos.

3. Al final,

(e) Expliquen sus respuestas.

(f) Capturen la pantalla.



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

(a) ¿son iguales?

(b) ¿por qué?

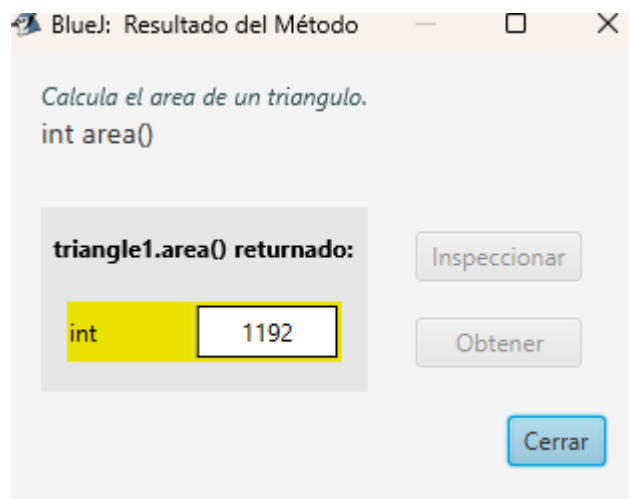
A pesar de que sabíamos de que constaba la figura, de rectángulos y los colores primarios no pudimos concretar el que era esta por lo cual no son iguales lo que se ve en pantalla después de seguir todas las instrucciones es el logo de Microsoft.

D. Extendiendo una clase. Triangle.

[En lab01.doc y *.java]

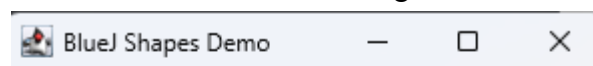
1. Desarrollen en Triangle el método `area()` (retorna el área del triángulo). ¡Pruébenlo! Capturen una pantalla.

```
/**
 * Calcula el area de un triangulo.
 */
public int area(){
    int areaTriangle;
    areaTriangle = (height*width)/2;
    return areaTriangle;
}
```

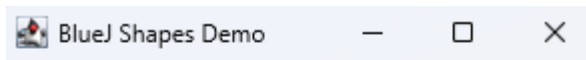


2. Desarrollen en Triangle el método `equilateral()` (transforma el triángulo en un triángulo equilátero de área equivalente). ¡Pruébenlo! Capturan dos pantallas.

Antes de Transformar el Triángulo:



Luego De Transformar el Triángulo a uno Equilátero:



Código:

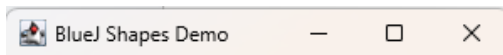
```
/**
 * Transform the triangle into an equilateral triangle of equivalent area.
 */
public void equilateral(){
    int areaTriangle = area();
    int newWidth = (int)(Math.sqrt((4*areaTriangle)/Math.sqrt(3)));
    int newHeight = (int)(Math.sqrt(3)* newWidth)/2;
    changeSize(newHeight,newWidth);
}
```

3. Desarrollen en Triangle el método shrink(times:int, height: int)
(disminuye su tamaño times veces. Hasta llegar a una altura de height.) ¡Pruébenlo! Capturan tres pantallas.

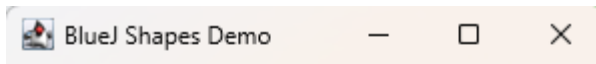
Código:

```
/**
 * It decreases its size times times. Until reaching a height of height.
 */
public void shrink(int times,int heightReduction){
    double heightR = (height - heightReduction)/times;
    for (int i = 0; i < times; i++){
        height = height - (int)heightR;
        changeSize(height,width);
    }
}
```

Antes de la reducción de su tamaño:

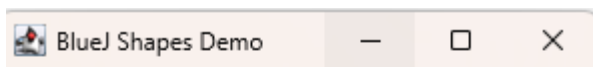


Luego de usar el método shrink para su reducción:



4. Desarrollen en Triangle un nuevo creador que permita crear un triángulo en una posición específica. ¡Pruébenlo! Capturen una pantalla.

```
/**
 * Allows you to create a triangle in a specific position.
 */
public Triangle(int xPositon,int yPositon){
    this.xPosition = xPositon;
    this.yPosition = yPositon;
    color = "red";
    isVisible = true;
    height = 30;
    width = 40;
}
```



Minimizar



5. Propongan un nuevo método para esta clase. Desarrollan y prueban el método.

Se propone un método para calcular el perímetro de un triángulo dados sus lados. Se adjunta una captura del método:

```
/**
 * Calculates the perimeter of a Triangle.
 */
public int perimeter(int side1,int side2,int side3){
    int trianglePerimeter = side1 + side2 + side3;
    return trianglePerimeter;
}
```

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

Modifier and Type	Method	Description
int	<code>area()</code>	Calculate the area of a triangle.
void	<code>changeColor(String¹² newColor)</code>	Change the color.
void	<code>changeSize(int newHeight, int newWidth)</code>	Change the size to the new size
void	<code>equilateral()</code>	Transform the triangle into an equilateral triangle of equivalent area.
void	<code>makeInvisible()</code>	Make this triangle invisible.
void	<code>makeVisible()</code>	Make this triangle visible.
void	<code>moveDown()</code>	Move the triangle a few pixels down.
void	<code>moveHorizontal(int distance)</code>	Move the triangle horizontally.
void	<code>moveLeft()</code>	Move the triangle a few pixels to the left.
void	<code>moveRight()</code>	Move the triangle a few pixels to the right.
void	<code>moveUp()</code>	Move the triangle a few pixels up.
void	<code>moveVertical(int distance)</code>	Move the triangle vertically.
int	<code>perimeter(int side1, int side2, int side3)</code>	Calculates the perimeter of a Triangle.
void	<code>shrink(int times, int heightReduction)</code>	It decreases its size times times.
void	<code>slowMoveHorizontal(int distance)</code>	Slowly move the triangle horizontally.
void	<code>slowMoveVertical(int distance)</code>	Slowly move the triangle vertically.

E. Creando una nueva clase. Usando un paquete. shapes

[En lab01.doc y *.java]

En este punto vamos a crear huecos con fondos cuadrados para guardar semillas. El diseño gráfico lo definen ustedes. Estos son algunos ejemplos.




1. Inicie la construcción únicamente con los atributos. Justifique su selección. Adicione pantallazo con los atributos.

```
public class Pit{

    private Rectangle rectangle1;
    private Rectangle rectangle2;
    private int posX;
    private int posY;
    private ArrayList<Rectangle> seeds = new ArrayList<>();
```

Los atributos elegidos como los podemos observar en la imagen adjuntada son porque las posiciones X y Y va a ser la localización de los rectángulos en el canvas dado , el ArrayList usado para las semillas va a ser donde serán guardadas las semillas.

2. Desarrollen la clase considerando los 3 mini-ciclos. Al final de cada mini-ciclo realicen dos pruebas indicando su propósito. Capturen las pantallas relevantes.

private Rectangle rectangle1	
private Rectangle rectangle2	
private Rectangle rectangleSeeds	null
private int posX	110
private int posY	130
private ArrayList<Rectangle> Seeds	
private int limitSeeds	4
private int sizePit	16

heredado de Object ▶

- void changeColors(String background, String seeds)
- void makeInvisible()
- void makeVisible()
- void moveTo(int x, int y)**
- void putSeeds(int seeds)
- void removeSeeds(int seeds)
- int seeds()

Inspeccionar

Remover

BlueJ: BlueJ: Crear Objeto

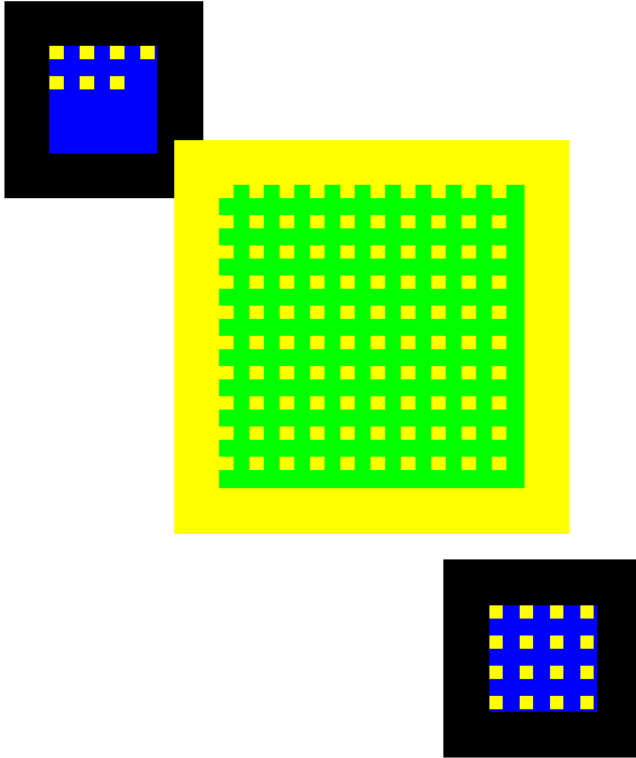
Constructor

Pit(boolean big)

Nombre de Instancia:

new Pit()

Aceptar Cancelar



G. De python a java

En este punto vamos a usar y evaluar dos recursos de apoyo para la transición de Python a Java. Realicen la evaluación en las encuestas preparadas con ese objetivo.

1. El video.

☒ EVALUACIÓN DEL VIDEO DE PYTHON A JAVA

Marcar como hecha

Intentos permitidos: 1

RESUMEN DE SUS INTENTOS PREVIOS

Estado	Revisión
Finalizado	
Enviado: sábado, 8 de febrero de 2025, 13:30	

?

2. Los prompts.

☒ EVALUACIÓN DE LOS PROMPTS DE PYTHON A JAVA

Marcar como hecha

Intentos permitidos: 1

RESUMEN DE SUS INTENTOS PREVIOS

Estado	Revisión
Finalizado	
Enviado: sábado, 8 de febrero de 2025, 13:35	

No se permiten más intentos

Volver al curso

?

KALAH

F. Definiendo y creando una nueva clase. Kalah.

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

Requisitos funcionales:

Crear el estado inicial

Realizar los movimientos

Reiniciar el juego

Consultar el estado del juego. (Un mensaje con el número de semillas en cada almacén)

Informar cuando alguien gana el juego. (Un mensaje de felicitación)

Requisitos de interfaz

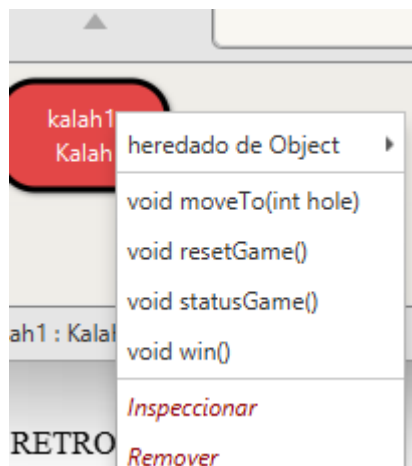
Las casas se identifican por el jugador ('North', 'South') y el número de la casa (1 a 6).

En caso que no sea posible realizar una de las acciones, se debe generar un mensaje de error.

Para los mensajes use JOptionPane.

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

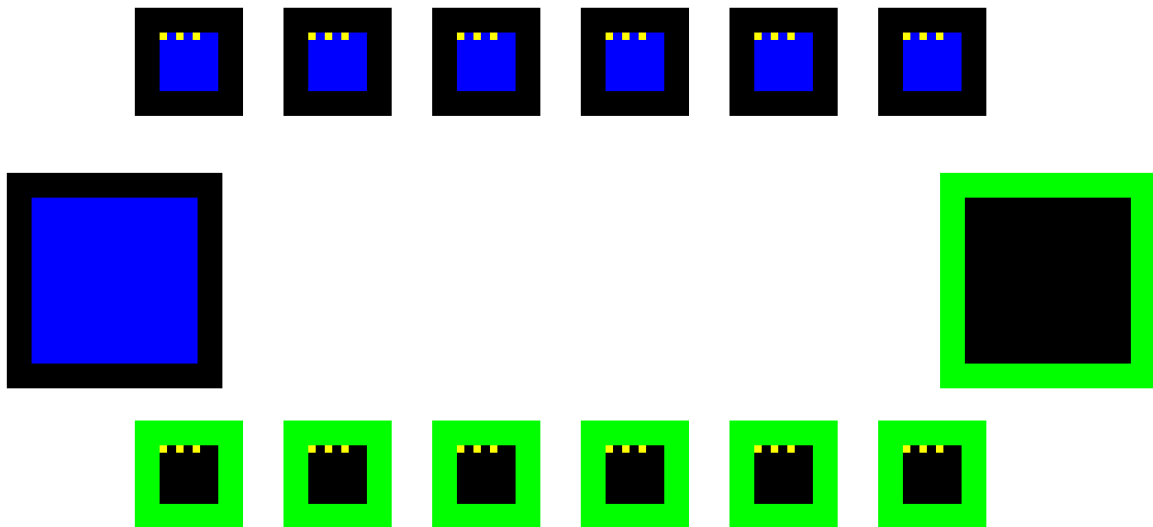
Debe ofrecer cuatro métodos.



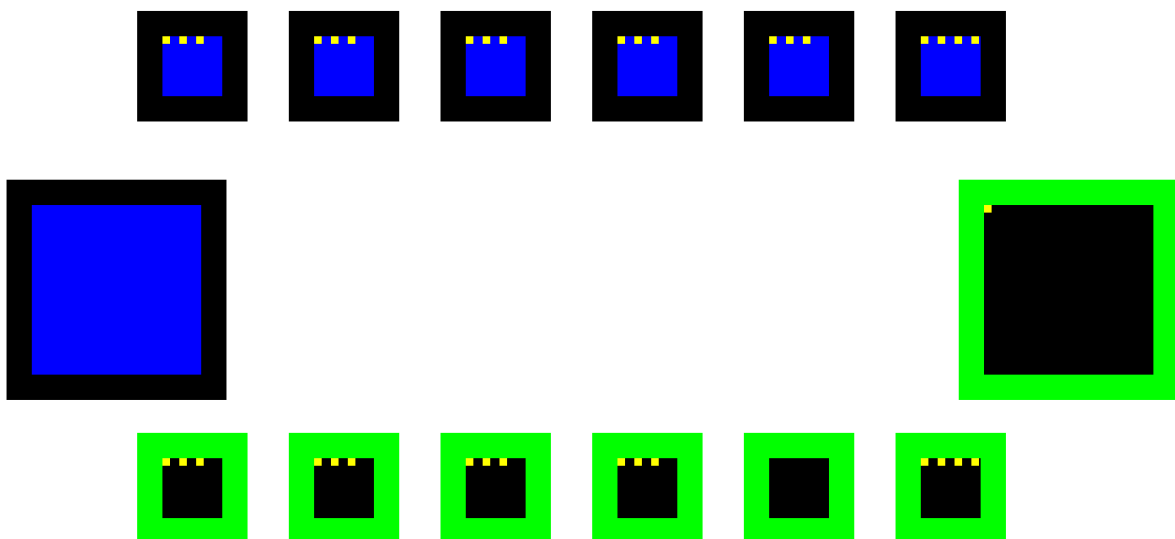
2. Planifiquen la construcción considerando algunos mini-ciclos.

3. Implementan la clase . Al final de cada mini-ciclo realicen una prueba indicando su propósito. Capturen las pantallas relevantes.

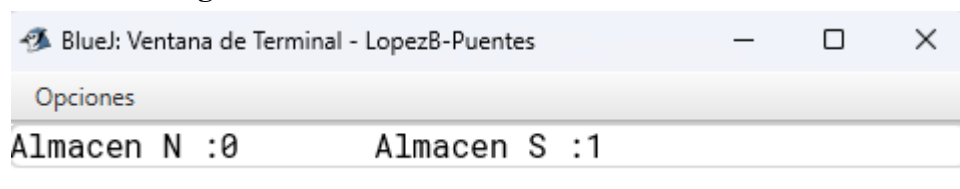
Creación del Pit:



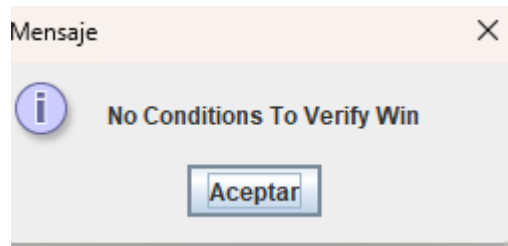
Movimientos:



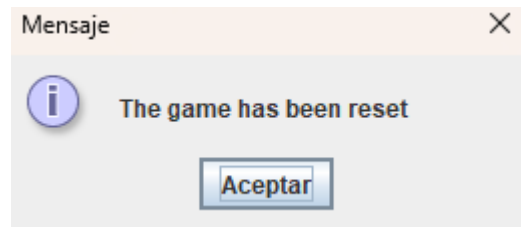
Status del Juego:



Si no hay condiciones para verificar si gana alguno :



Reset del Juego:



4. Indiquen las extensiones necesarias para reutilizar la clase Pit y el paquete shapes. Expliquen.

Para facilitar el juego, decidimos usar solo cuadrados y así mismo arreglos para realizar el juego. Reutilizamos la clase pit para hacer los hoyos del juego y los métodos para poner las semillas.

RETROSPECTIVA

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes? (Horas/Hombre)

El tiempo invertido por persona fue de alrededor de 12 Horas

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

El estado actual del laboratorio está terminado pero nos gustaría mejorar más en cuanto a la implementación de los métodos y nos gustaría hacer otro método para crear el kalah más rápido.

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

La práctica XP más útil para desarrollar el laboratorio fue la programación en pareja ya que nos ayudó a complementarnos con ideas y llegar a conclusiones que nos podían servir para desarrollar este.

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

El mayor logro que tuvimos fue el implementar el sistema de turnos y el sistema de mover las semillas en cada pit para el juego del kalah porque no subíamos como acomodarlas de tal forma que fueran equitativos pues siguiendo las reglas del kalah.

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

El mayor problema técnico fue el como saber como mover las semillas en el kalah y lo que hicimos fue mejorar la comunicación y llegar a un acuerdo con las ideas que teníamos.

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

Como equipo resaltamos nuestra comunicación y la forma en cómo abordamos los problemas que iban surgiendo como el tiempo invertido en este laboratorio que fue distribuido de una forma equitativa y eficaz.

7. ¿Qué referencias usaron? ¿Cuál fue la más útil? Incluyan citas con estándares adecuados.

- GeeksforGeeks. (2024, December 12). *Java Strings*. GeeksforGeeks.
<https://www.geeksforgeeks.org/strings-in-java/>
- De Roer, D. D., & De Roer, D. D. (2022, June 12). Métodos get y set en Java. *Disco Duro de Roer* -. <https://www.discoduroderoer.es/metodos-get-y-set-en-java/>
- Tiempo de pensar. (2020, January 30). ¿CÓMO JUGAR MANCALA (KALAH)?
INTRODUCCIÓN JUEGO DE TABLERO - Nayeli [Video]. YouTube.
<https://www.youtube.com/watch?v=NvGFnr7aTUQ>
- *Java Documentation - Get started*. (2023, January 31). Oracle Help Center.
<https://docs.oracle.com/en/java/>
- colaboradores de Wikipedia. (2024, July 11). *Kalah*. Wikipedia, La Enciclopedia Libre. <https://es.wikipedia.org/wiki/Kalah>