Laboratorio 1: Santiago Gualdron, Laura Sofia Gil Chaves

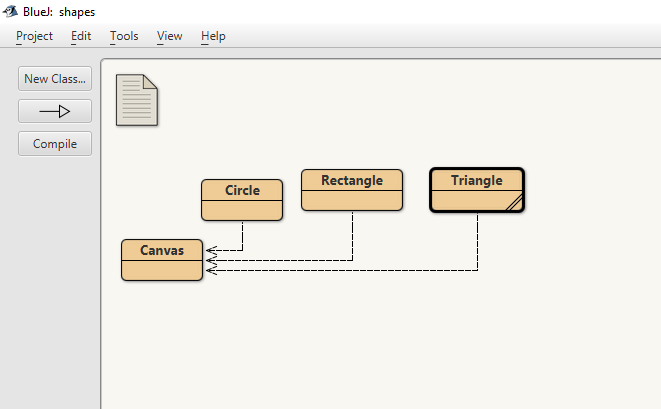
**SHAPES**

**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”

1. ¿Qué clases ofrece?

Las clases que ofrece Shapes son:

Canvas, Circle, Rectangle y Triangle

1. ¿Qué relaciones existen entre ellas?

Las relaciones que existen entre ellas que circulo, rectángulo y triangulo están relacionados a la superclase Canvas, donde cada una de ellas es una subclase.

3. La documentación presenta las clases del proyecto y, en este caso, la especificación de sus componentes públicos. De acuerdo con la documentación generada:

1. ¿Qué clases tiene el paquete shapes?

Las clases que tiene el paquete shapes son Canvas, Circle, Rectangle y Triangle

1. ¿Qué atributos tiene la clase Rectangle?

Según la documentación no tenemos ningún atributo en la clase Rectangle.

1. ¿Cuántos métodos ofrece la clase Rectangle?

La clase Rectangle ofrece 12 métodos y un constructor.

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

La clase Rectangle ofrece el método changeSize.

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

1. ¿Cuántos atributos realmente tiene?

Los atributos que tiene Rectangle en el código son 7 ( EDGES, height, width, xPosition, yPosition,color, isVisible)

1. ¿Cuáles atributos determinan su forma?

Los atributos que determinan su forma son height, width

1. ¿Cuántos métodos tiene en total?

Tienen 14 métodos.

1. ¿Quiénes usan los métodos privados?

Los 2 métodos privados son draw y erase

5. Comparando la documentación con el código

1. ¿Qué no se ve en la documentación?

Los atributos y metodos que son privados, por tanto no se encuentran en la documentacion.

1. ¿por qué debe ser así?

Para no confundir al usuario, y que no los modifique en caso de encontrarlos, también es necesario decir que solo son visibles los métodos públicos para saber que métodos se puedan usar en otras clases.

6. En el código de la clase Rectangle, revise el atributo EDGES

1. ¿Qué significa que sea public?

Es un atributo accesible y modificable para todo el ambiente de desarrollo.

1. ¿Qué significa que sea static?

Quiere decir que la variable esta creada principal y únicamente para la clase.

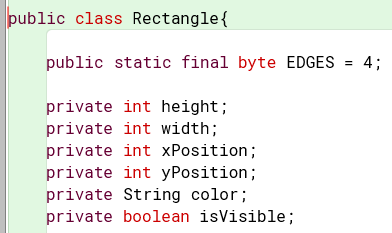
1. ¿Qué significaría que fuera final? ¿Debe serlo?

El valor que le es asignado a la variable no es modificable ni para la clase ni para ninguna otra. Puede que sea necesario que lo sea dado que la cantidad de vértices que caracteriza un rectangle siempre es 4.

1. ¿De qué tipo de dato debería ser (byte, short, int, long)? ¿Por qué?

Para que la variable ocupe poca memoria, podría considerarse el declararlo como byte.

1. Actualícenlo.



7. En el código de la clase Rectangle revisen el detalle del tipo del atributo width

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

Quiere decir que puede ingresar un numero entero de 4 bytes (de -2’147’483.647 a 2’147’483.647)

(b) Si width fuera byte, ¿cuál sería el rectangulo más grande posible?

Se podría ingresar un numero mucho más pequeño donde el triángulo más grande posible tendría una altura de 128 máx.

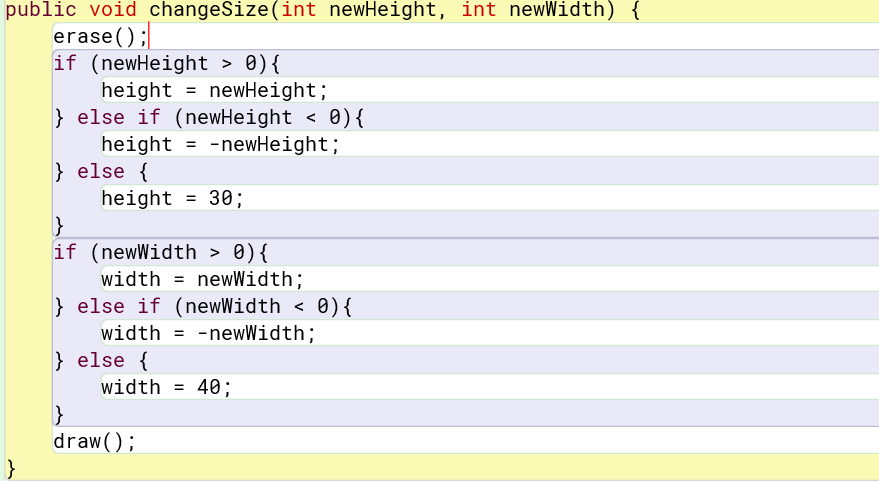
(c) y ¿si fuera long?

La altura máxima sería 9,223,372,036,854,775,807.

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

Debe ser si o si positivo.

(e) Refactoricen el código considerando (d).



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

La posibilidad de crear nuevas figuras, cambiar su tamaño y su posición, siendo todo esto visible en un plano llamado canvas.

**B. Manipulando objetos.**

Usando un objeto.

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

1. ¿Cuántas clases hay?

Hay 4 clases

1. ¿Cuántos objetos crearon?

Hay 3 objetos

1. ¿Quién se crea de forma diferente? ¿Por qué?

El que se creó de forma diferente es el Canvas ya que no hay un nombre que lo defina, sólo se obtiene y pinta lo dado por las 3 subclases

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

1. ¿Cuál es la principal diferencia entre ellos?

La diferencia es que en el constructor de canvas tiene parámetros y las otras clases no lo contienen.

1. ¿Qué se busca con la clase que tiene el creador diferente?

Se busca crear el tablero, donde pone la anchura, altura, titulo y el fondo.

3. Inspeccionen el estado del objeto :Triangle,

1. ¿Cuáles son los valores de inicio de todos sus atributos?

Los valores iniciales de todos sus atributos son:

*height = 30;*

*width = 40;*

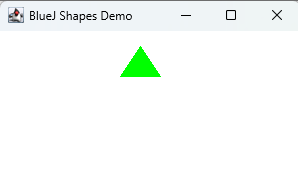
*xPosition = 140;*

*yPosition = 15;*

*color = "green";*

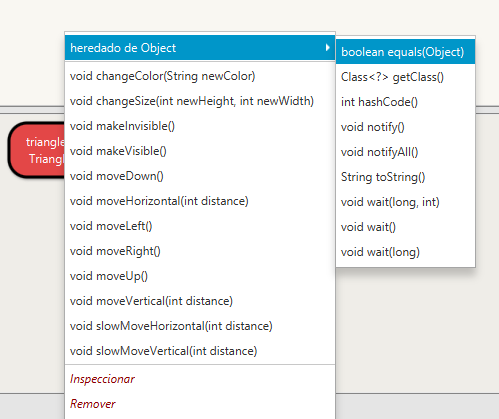
*isVisible = false;*

1. Capturen la pantalla.



4. Inspeccionen el comportamiento que ofrece el objeto :Triangle

1. Capturen la pantalla.



1. ¿Por qué no aparecen todos los que están en el código?

Debido a que hay métodos que son privados, que sólo son exclusivos en la clase

5. Construyan, con “shapes” sin escribir código, una propuesta de la imagen del logo de su marca de carro favorita.

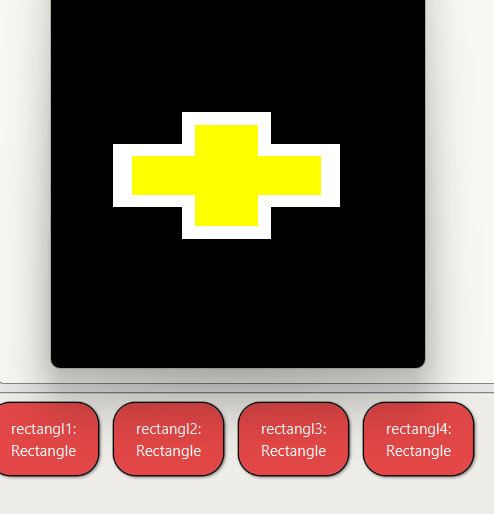
1. ¿Cuántas y cuáles clases se necesitan?

Se necesitan 2 clases: El canvas para poder dibujar el logo y Rectangle.

1. ¿Cuántos objetos se usan en total?

Se usan en total 4 objetos, todos son de la clase Rectangle

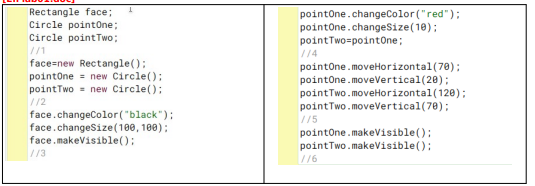
1. Capturen la pantalla.



1. Incluyan el logo original.



**C. Manipulando objetos. Analizando y escribiendo código.**



1. Lean el código anterior.

1. ¿cuál creen que es la figura resultante?

Creemos que la figura es una cara de forma rectangular.

1. Píntenla.



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

1. ¿cuántas variables existen?

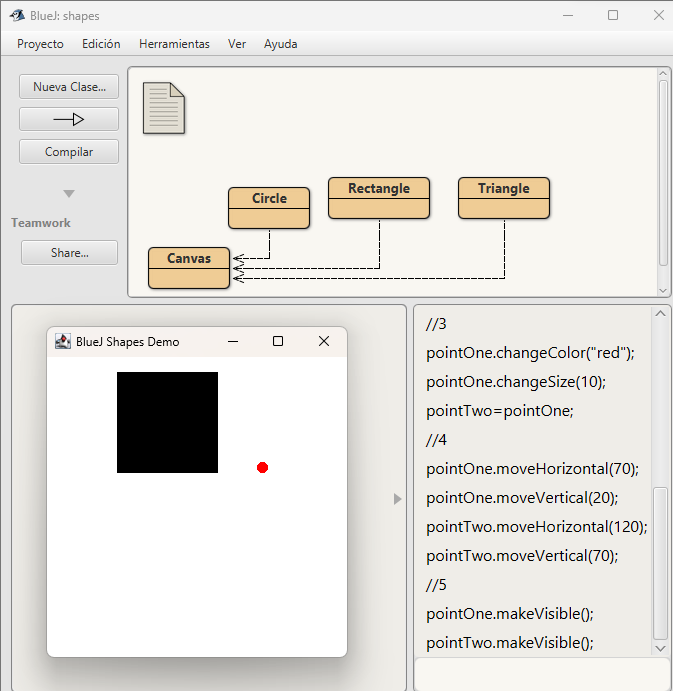
Exiten 3 variables.

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

Existen 2 objectos.

1. ¿qué color tiene cada uno de ellos?

Uno tiene el color rojo y el otro negro

1. ¿cuántos objetos se ven?
2. 2
3. Capturen la pantalla.
4. 

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

1. ¿son iguales?

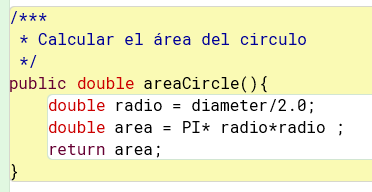
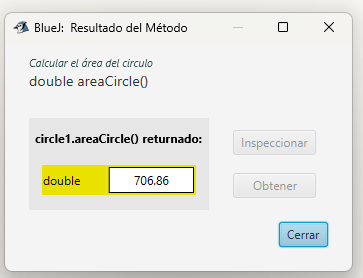
No son iguales

1. ¿por qué?

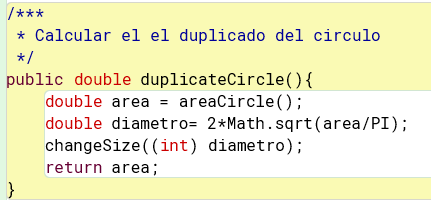
Porque la ubicación de los objectos no son iguales al código propuesto, además, que unos se instancian y otros no.

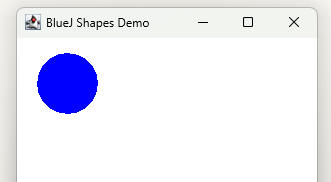
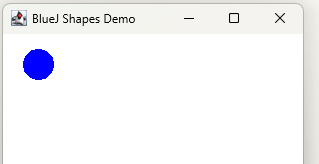
**D. Extendiendo una clase. Circle.**

1. Desarrollen en Circle el método area(). ¡Pruébenlo! Capturen una pantalla.

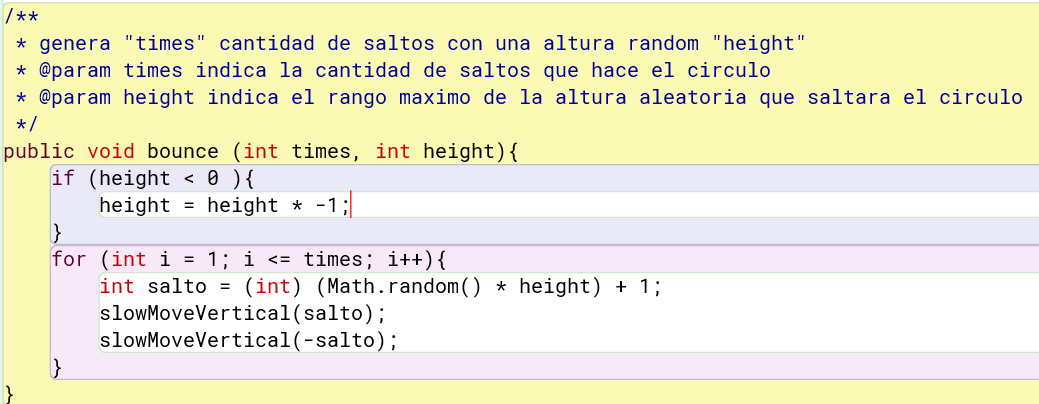
 

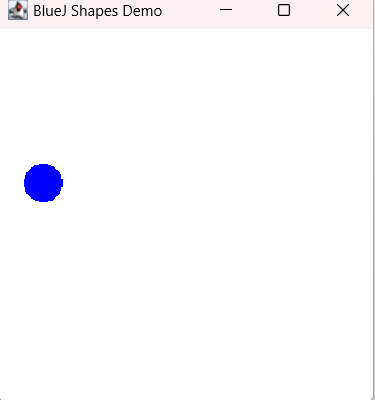
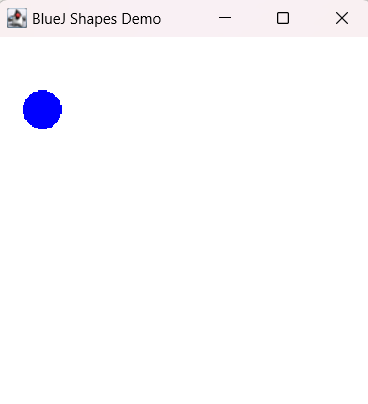
2. Desarrollen en Circle el método duplicate() (duplica el área del círculo). ¡Pruébenlo! Capturen dos pantallas.



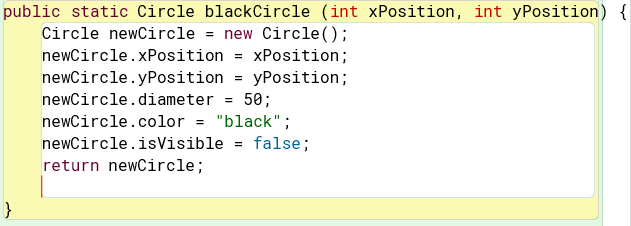


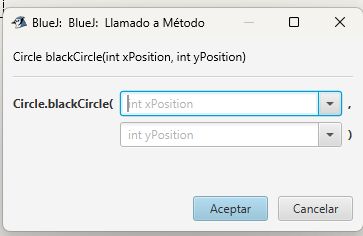
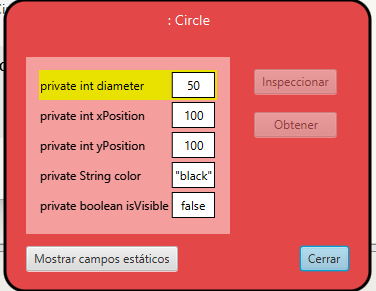
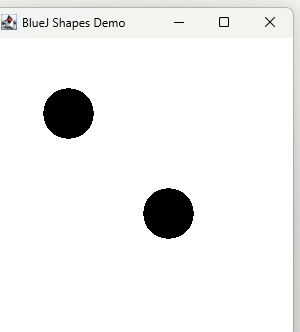
3. Desarrollen en Circle el método bounce(times:int, height:int) (salta el número de veces indicado en times. Las alturas de los saltos se seleccionan al azar con valores menores de height 7). ¡Pruébenlo! Capturen dos pantallas.





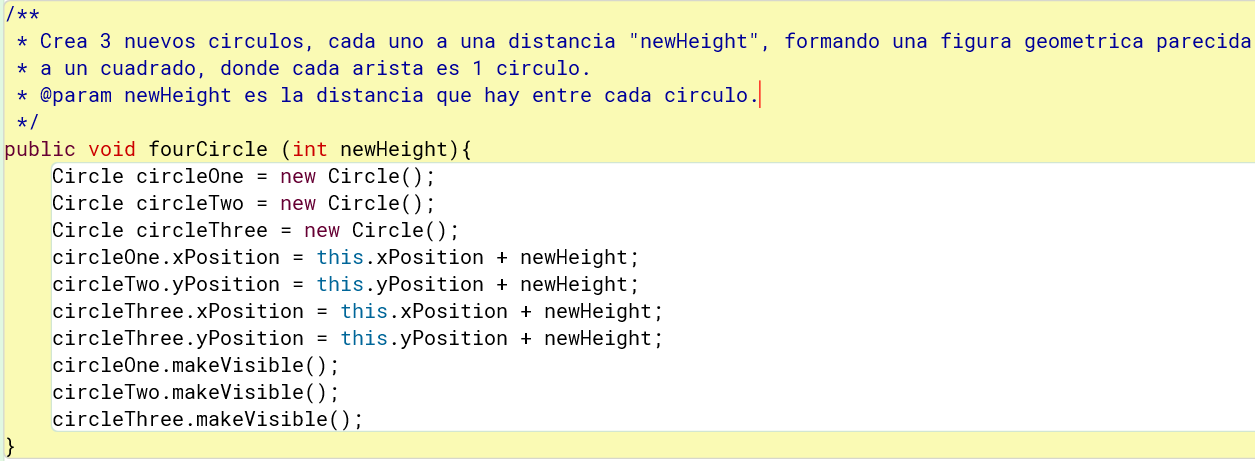
4. Desarrollen en Circle un nuevo creador que, dada una posición, crea círculos negros de radio 50.¡Pruébenlo! Capturen una pantalla.

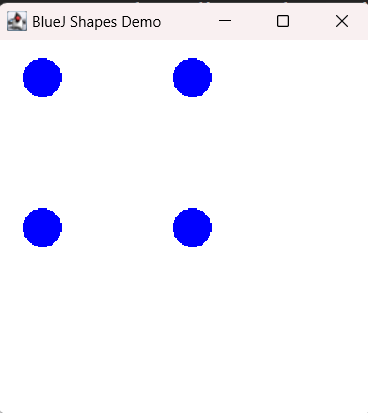
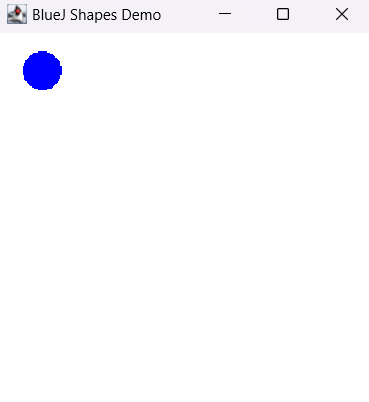


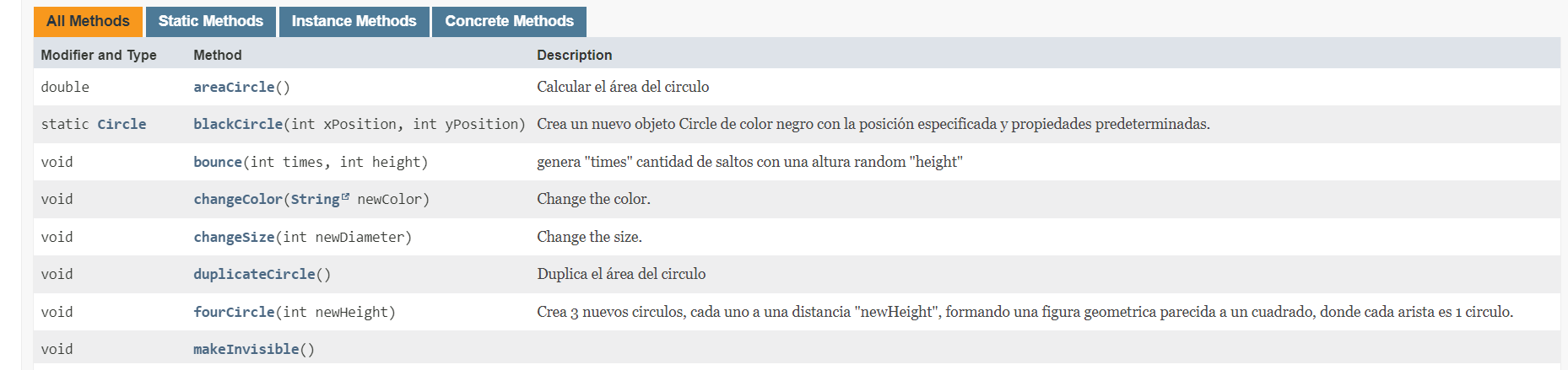
5. Propongan un nuevo método para esta clase. Desarrollen y prueban el método.

Para este método, decidimos crear una figura geométrica parecida a un cuadrado hecho con 4 círculos, donde cada uno es una arista del cuadrado; le pedimos al usuario cuál es la distancia que quiere dejar entre cada círculo y con eso los nuevos círculos.

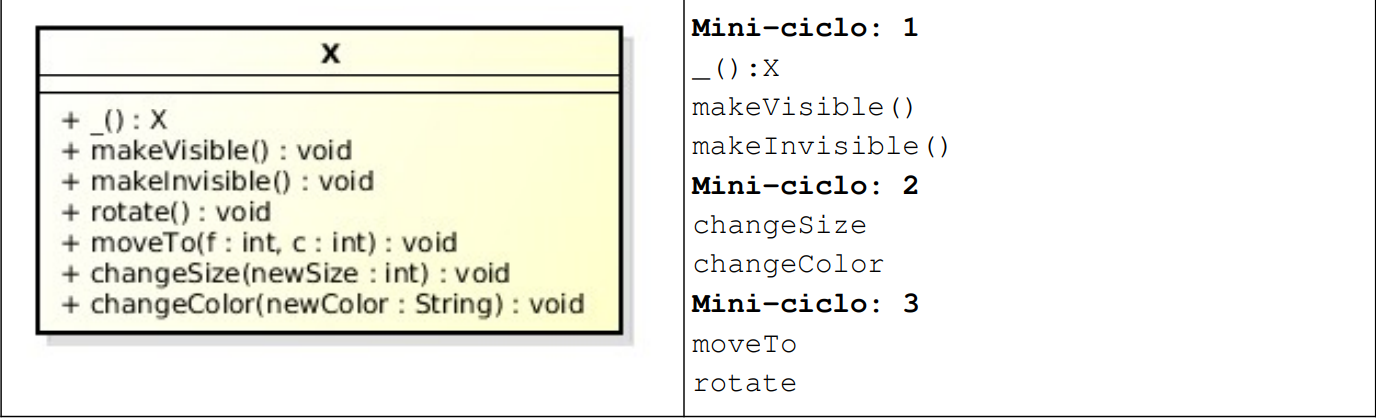




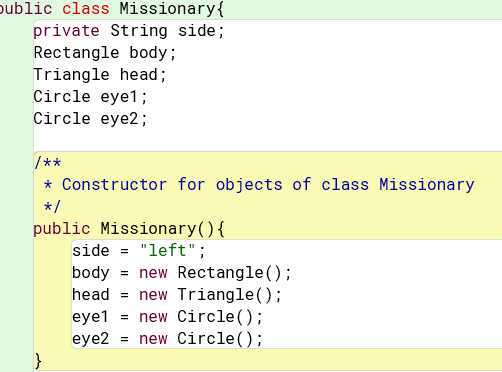
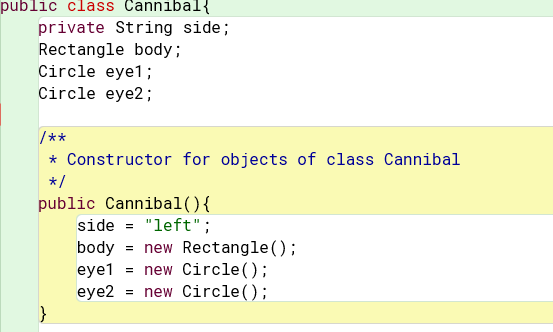
6. Generen nuevamente la documentación y revise la información de estos nuevos métodos. Capturen la pantalla.

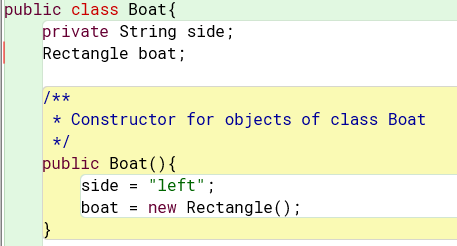


**E. Usando un paquete. Shapes**



1. Inicie la construcción únicamente con los atributos. Adicione pantallazo con los atributos.



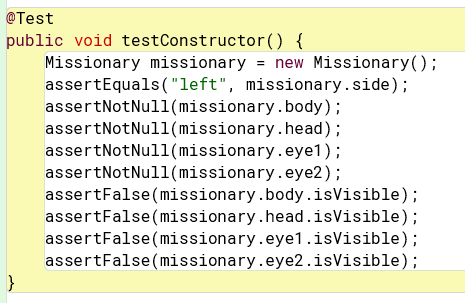


2. Desarrollen la clase considerando los 3 mini-ciclos. Al final de cada mini ciclo realicen dos pruebas indicando su propósito. Capturen las pantallas relevantes

Clase Missionary:

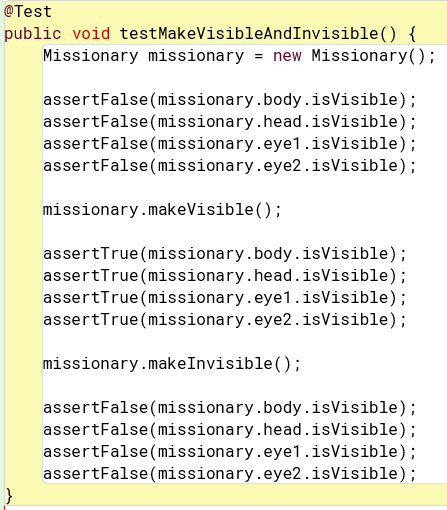
-testconstructor: El propósito de esta prueba es verificar si el objecto Missionary se inicializa correctamente.





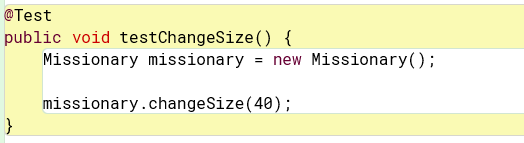
- testMakeVisibleAndInvisible: Verifica si los métodos makeVisible y makeInvisible funcionan como se espera.





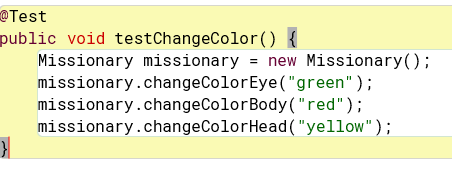
-TestChangeSize: prueba el metodoChangeSize con un valor de tamaño de 40





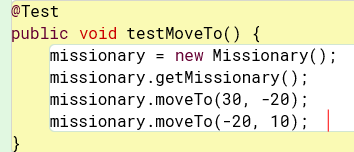
-TestChangeColor: prueba el método changeColor llamando a los métodos de cambio de color con colores específicos para ojos, cuerpo y cabeza.





-TestMoveTo: Probar el método moveto y verifica que el lado cambie correctamente. Donde se coloca coordenadas x o y para mirar cómo se mueve la figura, este método acepta números positivos y negativos.





-testrotate: Verifica como el método se comparta en la clase Missionary, primero asumiendo la posición inicial que la izquierda, evalúa la rotación después del lado derecho y luego del lado izquierdo.

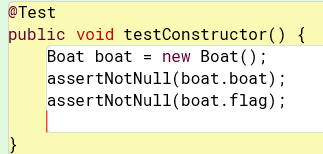




Clase Boat:

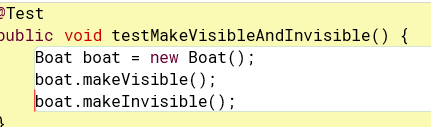
-testConstructor: la prueba verifica que el constructor de la clase Boat establezca de manera adecuada las instancias boat y flag, garantizando que no sean nulas después de la creación del objeto Boat.





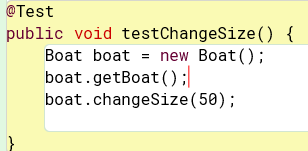
-testMakeVisibleAndInvisible: Verifica si los métodos makeVisible y makeInvisible funcionan como se espera.





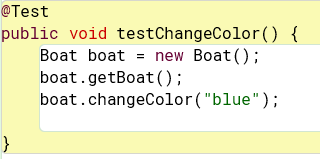
-testChangeSize: Llama al método y cambia el tamaño del bote.





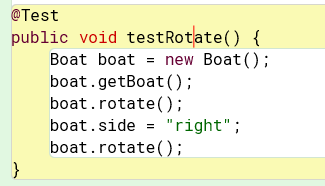
-testChangeColor: Llama al método y luego cambia el color del bote a azul.





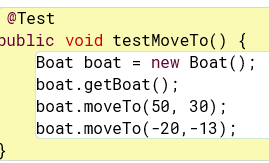
-test rotate: Verifica como el método se comparta en la clase Boat primero asumiendo la posición inicial que la izquierda, evalúa la rotación después del lado derecho





-test moveTo:

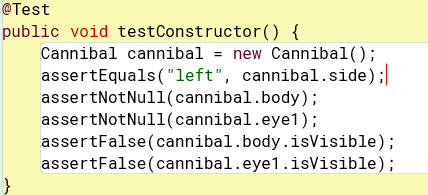




Clase Caníbal:

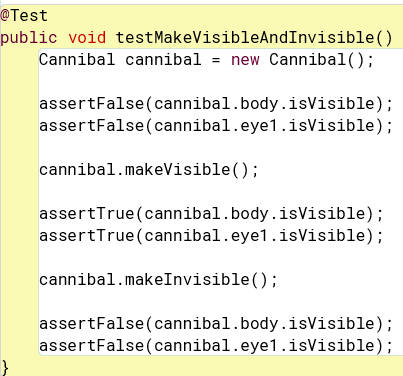
TestConstructor: El propósito de esta prueba es verificar si el objecto Cannibal se inicializa correctamente.





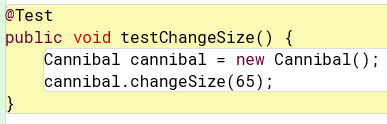
TestMakeVisibleandInvisible: Verifica si los métodos makeVisible y makeInvisible funcionan como se espera.





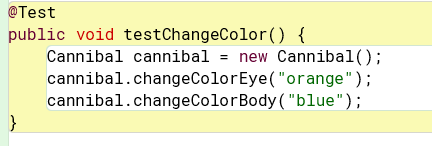
TestChangeSize: prueba el metodoChangeSize con un valor de tamaño de 65





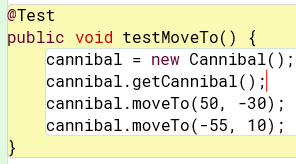
TestChnageColor: prueba el método changeColor llamando a los métodos de cambio de color con colores específicos para ojos y cuerpo.





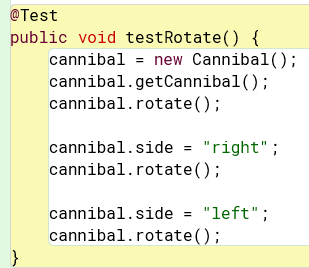
TestMoveTo: Probar el método moveto y verifica que el lado cambie correctamente. Donde se coloca coordenadas x o y para mirar cómo se mueve la figura, este método acepta números positivos y negativos.





TestRotate: Verifica como el método se comparta en la clase Cannibal, primero asumiendo la posición inicial que la izquierda, evalúa la rotación después del lado derecho y luego del lado izquierdo.





**F. Definiendo y creando una nueva clase. MissionariesCannibals.**

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

Para el juego de misioneros y caníbales, se necesitan esencialmente 3 métodos, además del constructor: subir y bajar a los misioneros y caníbales, además de mover el bote de un lado a otro con los personajes que estén arriba; además de unas restricciones cuando quedan en algún lado 2 caníbales con un misionero y una felicitación cuando logra pasar los 6 personajes al otro lado.

2. Planifiquen la construcción considerando algunos mini-ciclos.

**Mini - Ciclo 1:**

\_(): X

estadoInicial()

**Mini – Ciclo 2:**

subir():

bajar():

moverBote():

**Mini – Ciclo 3**:

perder();

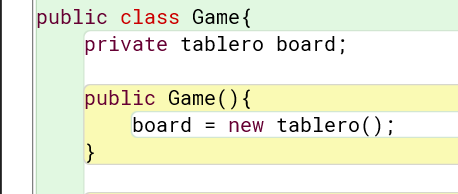
ganar();

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

Mini-Ciclo 1:

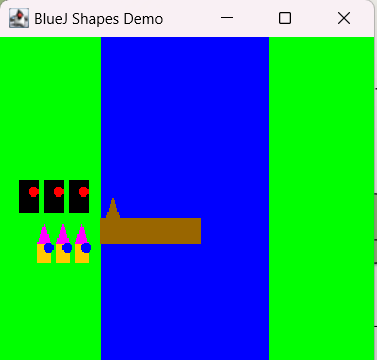
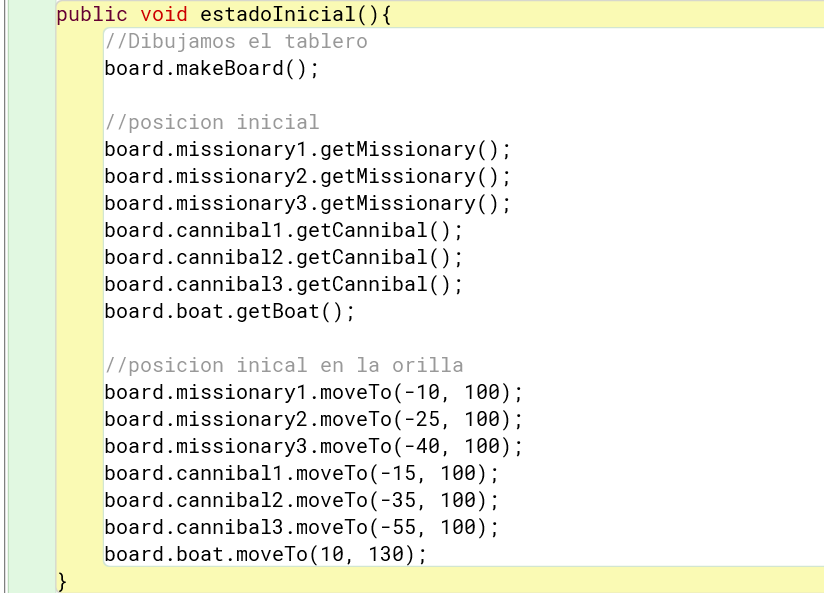
* Constructor:

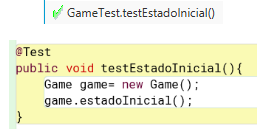
Estado inicial donde llamamos el tablero “board”



* EstadoInicial:

La creación de todo el juego, donde se crea el escenario de fondo, los 3 misioneros, los 3 canibales y el barco en las posiciones preestablecidas. Para la prueba, se busca que el juego se creara correctamente al estado inicial del reto.



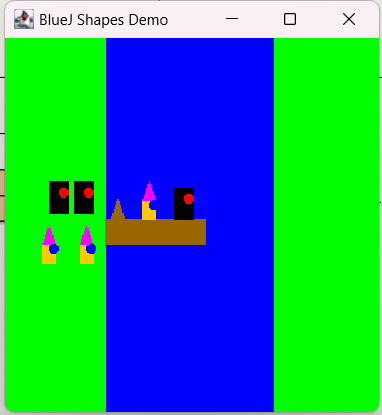


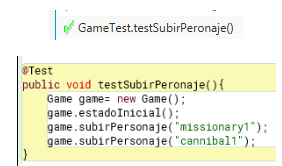
Mini-Ciclo 2:

* Subir:

Metodo que ayuda a subir a cualquiera de los 3 caníbales o los 3 misioneros al barco.

La prueba en subir se inicializa el tablero y se sube tanto 1 canibal como 1 misionero al barco.

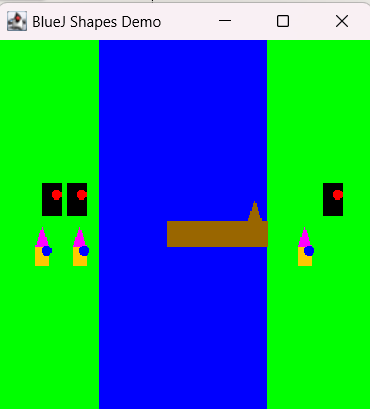
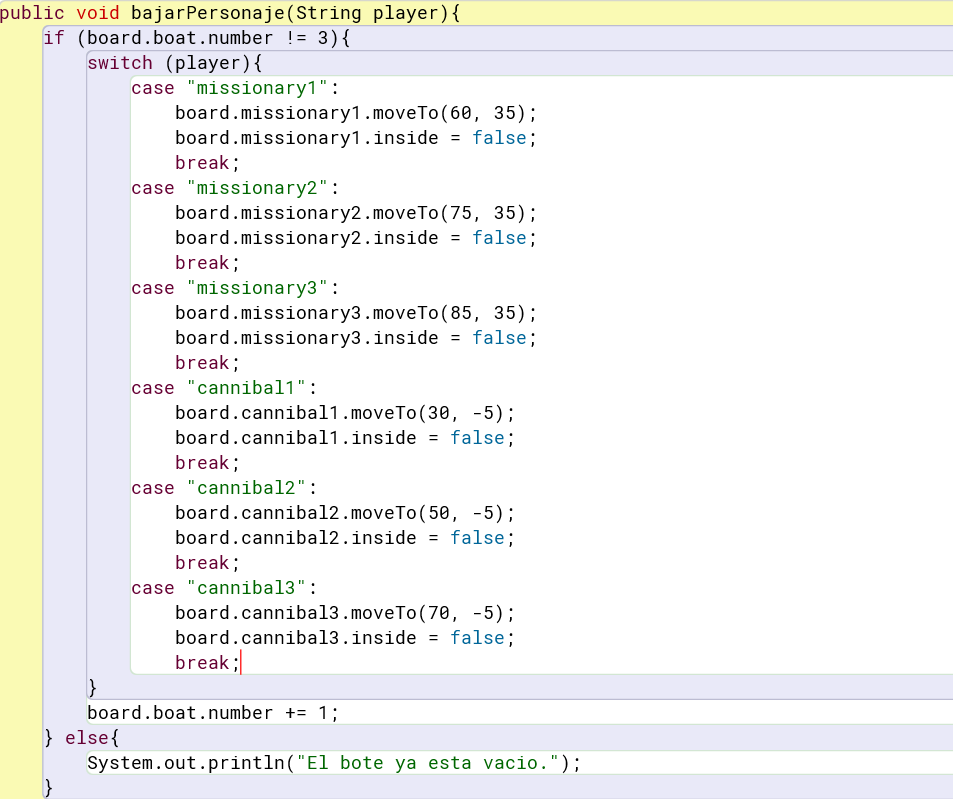


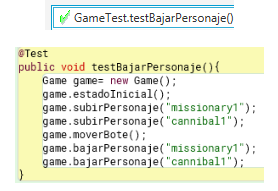


* Bajar:

Metodo que hace bajar a los cannibales o misioneros del barco.

La prueba en bajar se continua el tablero anterior y se bajan los 2 personajes que estaban en el barco.

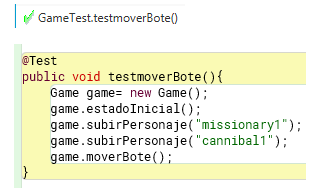
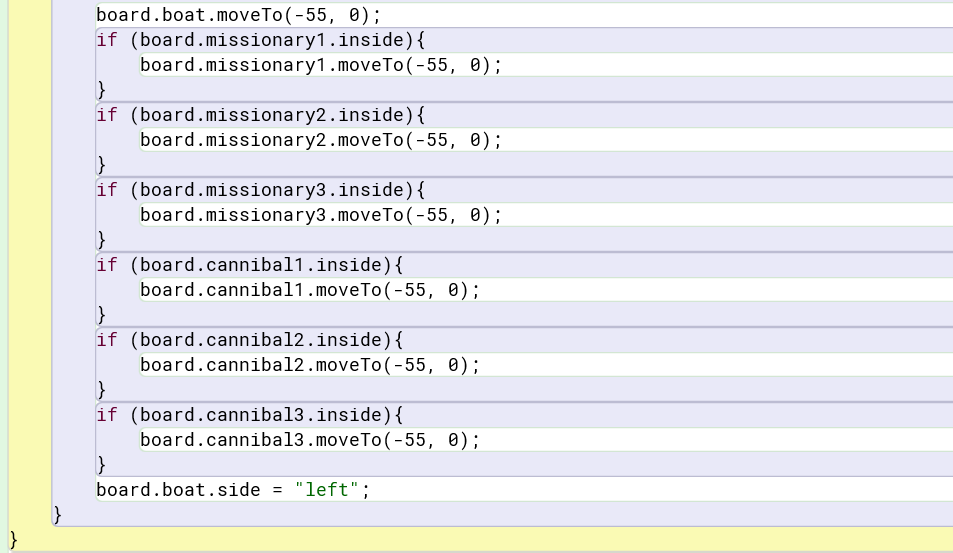
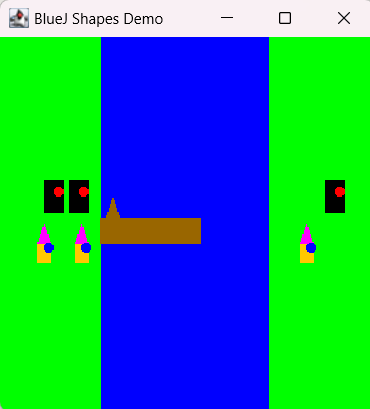
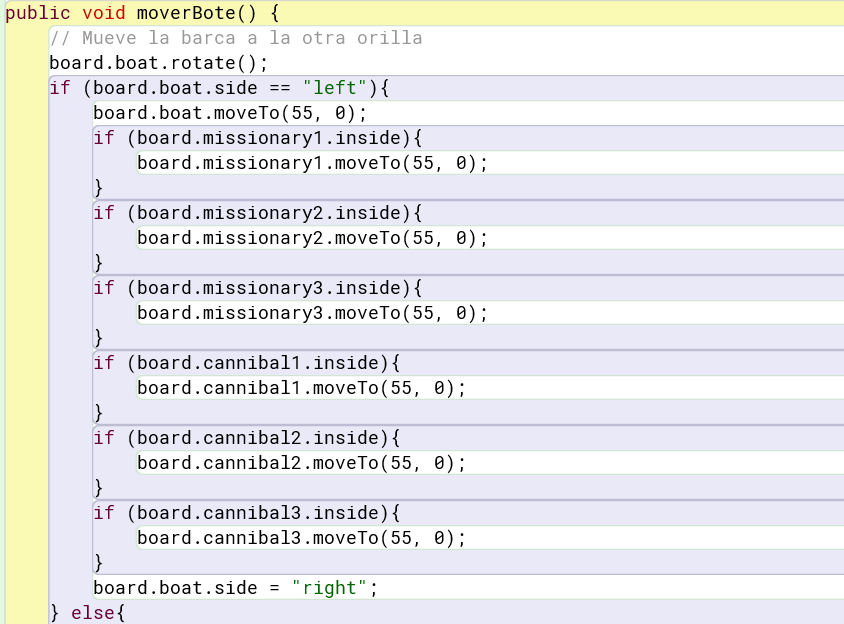




* moverBote:

Metodo que mueve el bote de un extremo del rio al otro, moviendo a su vez el caníbal o misionero que tiene adentro.

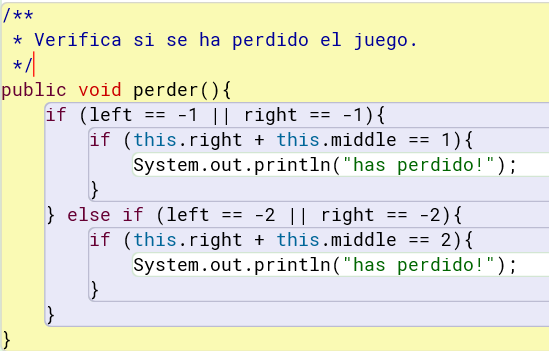
La prueba en mover el bote se continua el tablero anterior y se mueven el bote de izquierda a la derecha y viceversa.



Mini-Ciclo 3:

* Perder:

Para el método perder, manda un aviso (print) donde dice que ha perdido el usuario si han quedado más caníbales que misioneros en algún lado del rio.

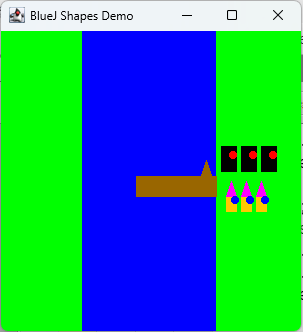


La prueba de perder se empieza subiendo a un misionero al bote y llevándolo al otro lado, de esta forma quedan 2 misioneros y 3 caníbales en la izquierda por lo que se ha perdido.



* Ganar:

Para este caso, es la prueba completa de pasar a todos los misioneros y caníbales al otro lado sin que aparezca el mensaje de “perder” en ningún momento y de la forma más optima.



4. Indiquen las extensiones necesarias para reutilizar las clases de las figuras y el paquete shapes. Expliquen.

Las extensiones necesarias para reutilizar las clases de las figuras y el paquete shapes fueron canvas, circulo, rectángulo y triangulo, debido que ayudaron a realizar el tablero del juego, los diferentes personajes como Cannibal y Misionero, y el transporte que es el bote.

**G. De python a java**

En este punto vamos a evaluar el video DE PYTHON A JAVA en la encuesta preparada con ese objetivo. ¿En cuáles puntos estuvieron en desacuerdo? Expliquen.

Después de haber visto el video, al momento de desarrollar el cuestionario “De Python a Java”, pudimos quedan en acuerdo respecto a todas las preguntas (siendo algunas diferencias entre “de acuerdo” y “totalmente de acuerdo”), más en ningún caso estuvimos contrariados respecto a la opinión de cada pregunta.

**RETROSPECTIVA**

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

Tiempo invertido 14 horas cada uno

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

El estado del laboratorio esta completo, a excepción del bono que era opcional.

3. Considerando las prácticas XP del laboratorio. ¿cuál fue la más útil? ¿por qué? ¿Cuál consideran fue el mayor logro? ¿Por qué?

Las practica XP más útil fue la de coding debido a que pudimos trabajar en pares con el tema del entendimiento y agregando la planeación del laboratorio. Nuestro mayor logro fue que terminamos completo el laboratorio, gracias a la comunicación y manejo de tiempo de nosotros.

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

El mayor problema técnico fue hacer el juego, y lo resolvimos mirando las estrategias de este y usando los comportamientos básicos que se necesitaban.

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

Hicimos bien como equipo establecer nuestros tiempos y la comunicación del trabajo, nos comprometemos a mejorar una plataforma para compartir los trabajos.