**PROGRAMACIÓN ORIENTADA A OBJETOS**

**Ernesto Camacho. Daniela Ruiz**

**Laboratorio 1/6.**

**SHAPES.**

**Conociendo el proyecto shapes**

**1. El diagrama de clases permite visualizar las clases de un artefacto software y las relaciones entre ellas. Considerando el diagrama de clases de “shapes” ¿qué clases ofrece? ¿qué relaciones existen entre ellas?**

Las clases que ofrece son canvas, circulo, rectángulo y triangulo; las relaciones que existen entre las clases son todas las clases se relacionan con canvas.

**2. La documentación presenta las clases del proyecto y, en este caso, la especificación de sus componentes públicos. De acuerdo con la documentación generada: ¿qué clases tiene el paquete shapes? ¿qué atributos tiene la clase Circle? ¿cuáles métodos ofrece la clase Circle para que la figura cambie (incluya sólo el nombre)?**

Canvas, circle, rectangle, triangle

* Solo aparece PI
* métodos circle:
* changeColor
* changeSize
* makeVisisble
* makeInvisible
* draw (método privado)
* erase (método privado)

los métodos privados no son observables mediante la documentación publica.

**3. En el código de cada clase está el detalle de la implementación. Revisen el código de la clase Circle. Con respecto a los atributos: ¿cuántos atributos realmente tiene? ¿cuáles son privados y cuáles públicos? Con respecto a los métodos: ¿cuántos métodos tiene en total? ¿cuáles son privados? ¿Quienes usan los componentes privados?**

* Atributos de Circle:
* diámetro
* -xPosition
* -yPosition
* -isVisible
* PI

Donde PI es el único atributo publico.

* Tiene 14 métodos, de los cuales hay 2 privados que son:
  + draw (método privado)
  + erase (método privado)

Todos los atributos son privados.

**4. ¿Qué no se ve en la documentación? ¿por qué debe ser así?**

* Los elementos privados, para proteger la integridad del código de los usuarios, o, para limitar el acceso a la información de le los usuarios.

**5. En el código de la clase Circle revisen el detalle del atributo PI. ¿qué se está indicando?**

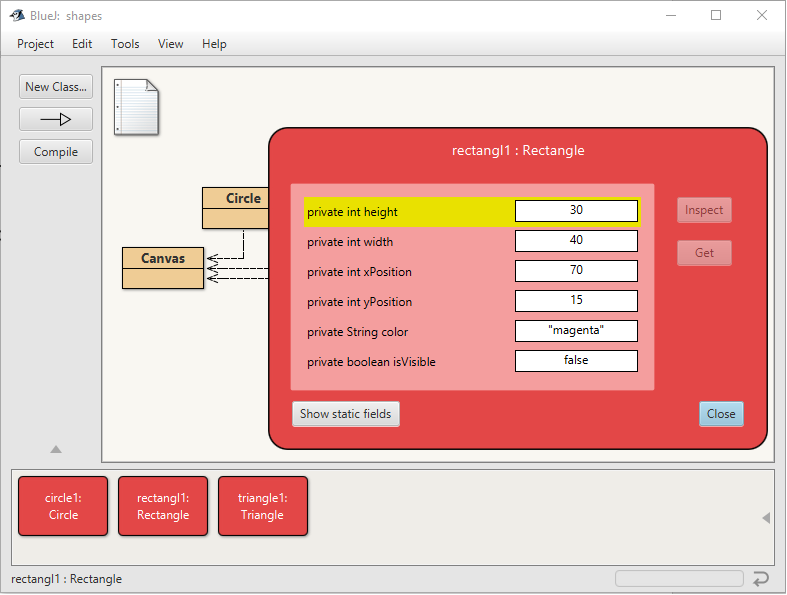
* Es un atributo necesario para un circulo, que es constante.

**Manipulando objetos. Usando opciones.**

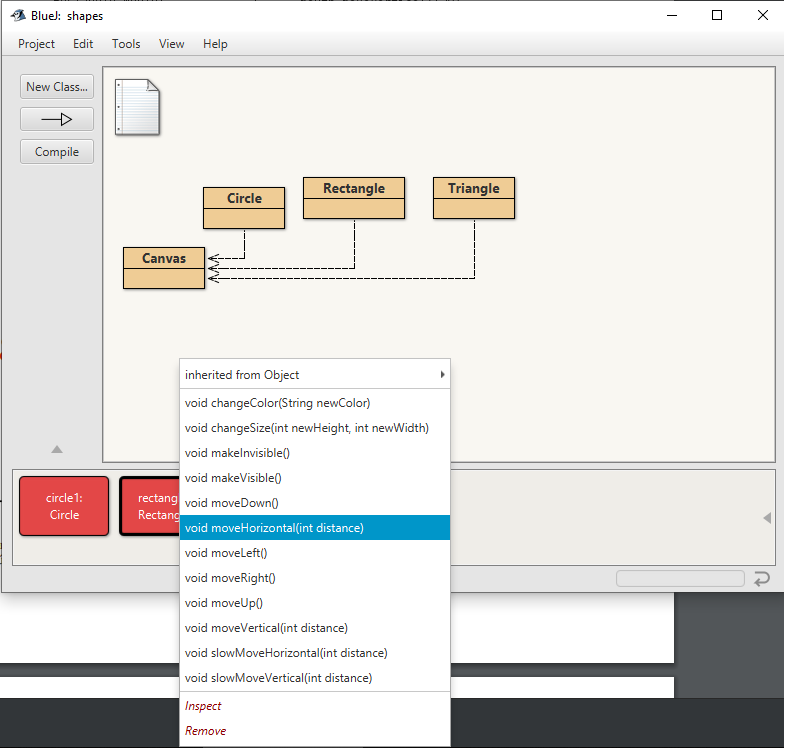
1. Creen un objeto de cada una de las clases que lo permitan. ¿cuántas clases hay? ¿cuántos objetos crearon? ¿por qué?

- Hay 4 clases de las cuales 3 se dejan crear, creamos rectangle, circle, y triangle, porque la clase canvas no crea un objeto, abre un canvas.

1. Inspeccionen el estado del objeto : Rectangle, ¿cuáles son los valores de inicio de todos sus atributos? Capturen las pantallas



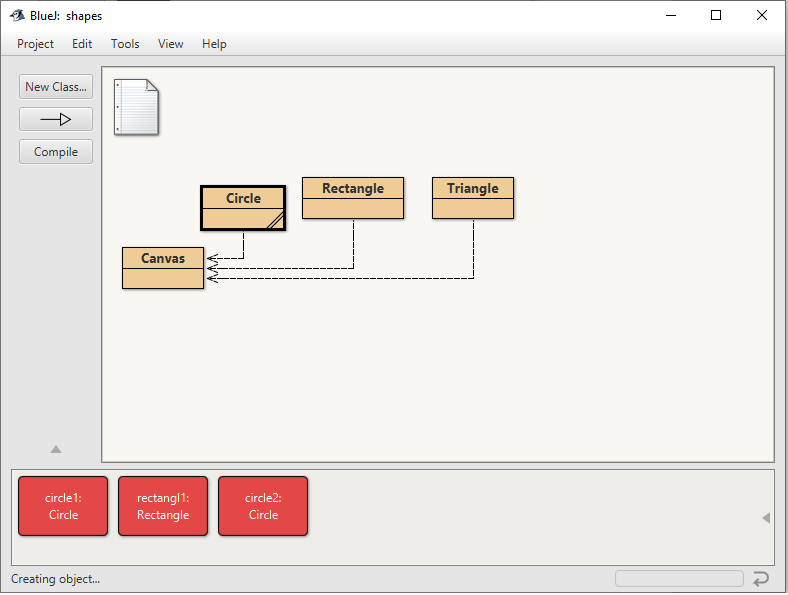
1. Inspeccionen el comportamiento que ofrece el objeto :Rectangle. Capturen la pantalla. ¿por qué no aparecen todos los que están en el código?

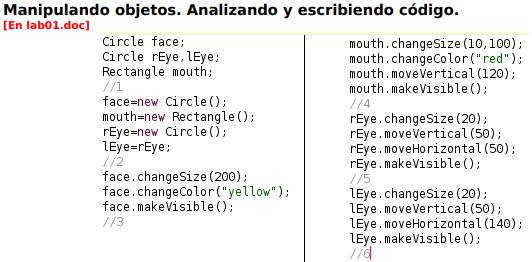


* + No se ven todos los métodos porque algunos son privados.

1. Construyan, con “shapes” sin escribir código, una propuesta de la imagen de su app favorito. ¿Cuántas y cuáles clases se necesitan? ¿Cuántos objetos se usan en total? Capturen la pantalla.

* La app elegida es Instagram, necesitamos: un cuadrado, y 2 círculos, por lo que necesitamos 3 objetos

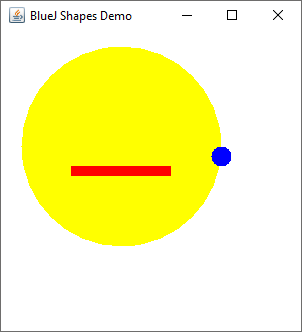




* 1. **Lean el código anterior ¿cuál es la figura resultante? Píntenla.**

****

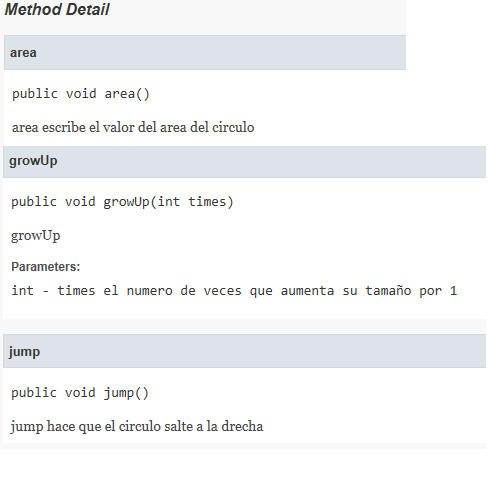
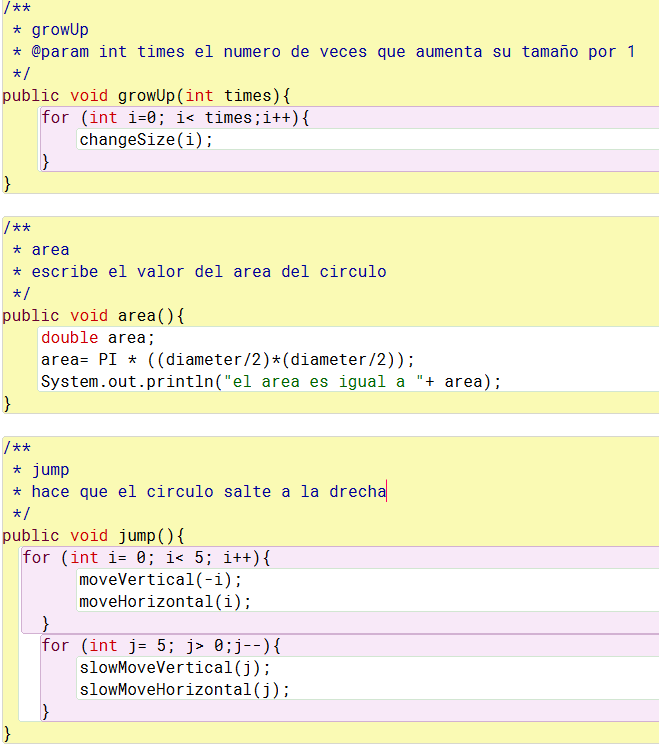
* 1. **Habiliten la ventana de código en línea, escriban el código y para cada punto señalado indiquen: ¿cuántas variable existen? ¿cuántos objetos existen? ¿qué color tiene cada uno de ellos? ¿cuántos objetos se ven? Expliquen. Capturen la pantalla.**

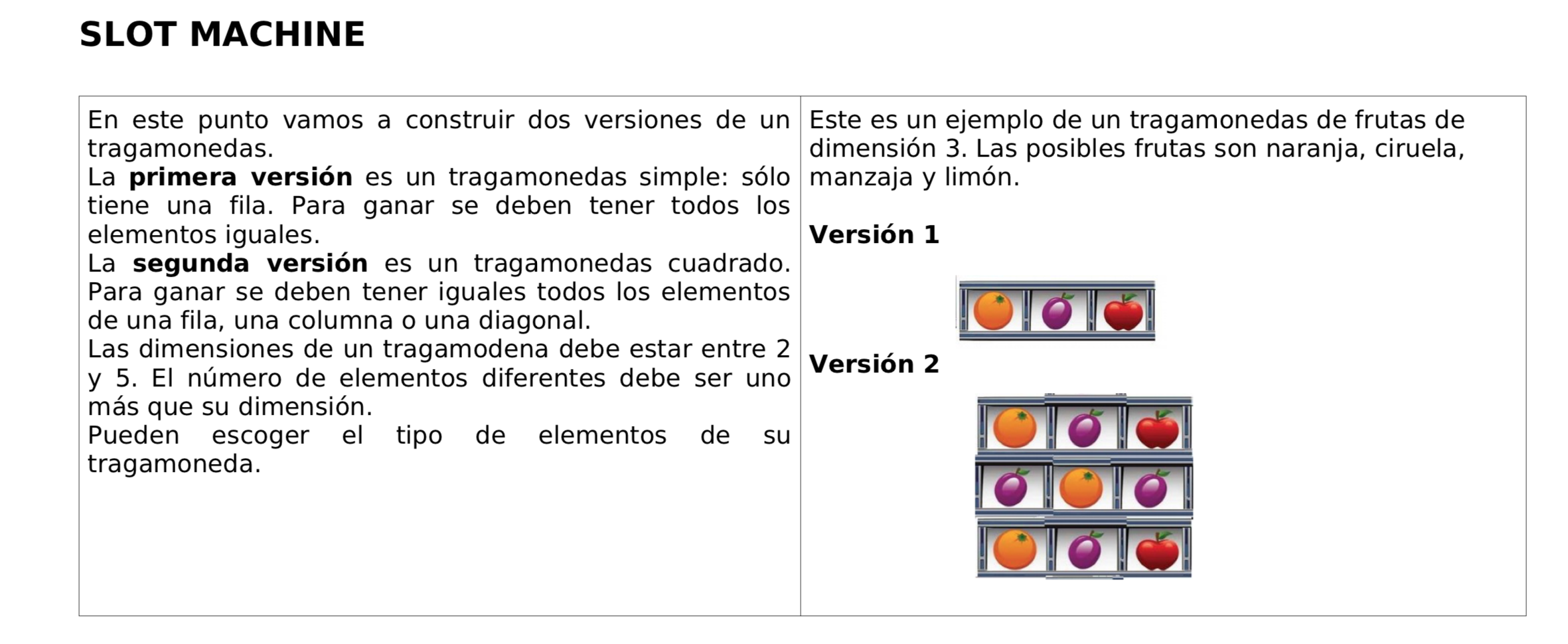
****

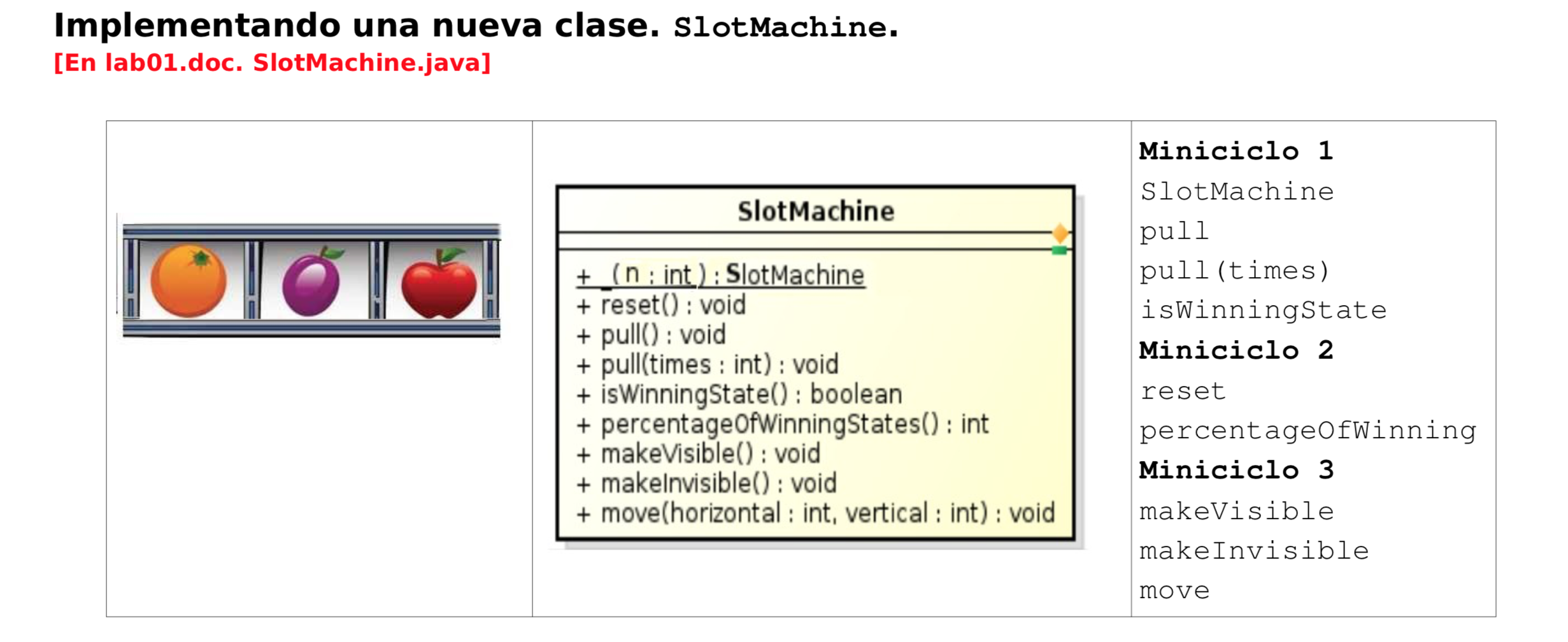
* + Existen 4 variables, 3 objetos, 2 círculos uno azul y otro amarillo, y 1 rectángulo rojo. Se ven 3 objetos, y la razón es que la variable “lEye” es una referencia a “rEye” no un objeto como tal, por lo que los cambios a “lEye” afectaron a “rEye”.
  1. **Es igual la figura pintada en 1. igual a la figura capturada en 2.¿por qué?**
  + No.

**Extendiendo clases**

1. Desarrollen en Circle el método growUp(times) (que hace que crezca lentamente el número dado de veces) . ¡Pruébenlo! grow up
2. Desarrollen en Circle el método area(). ¡Pruébenlo!
3. Desarrollen en Circle el método jump() (que hace que salte a la derecha) . ¡Pruébenlo!
4. Generen nuevamente la documentación y revise la información de estos nuevos métodos. Capture la pantalla.

****



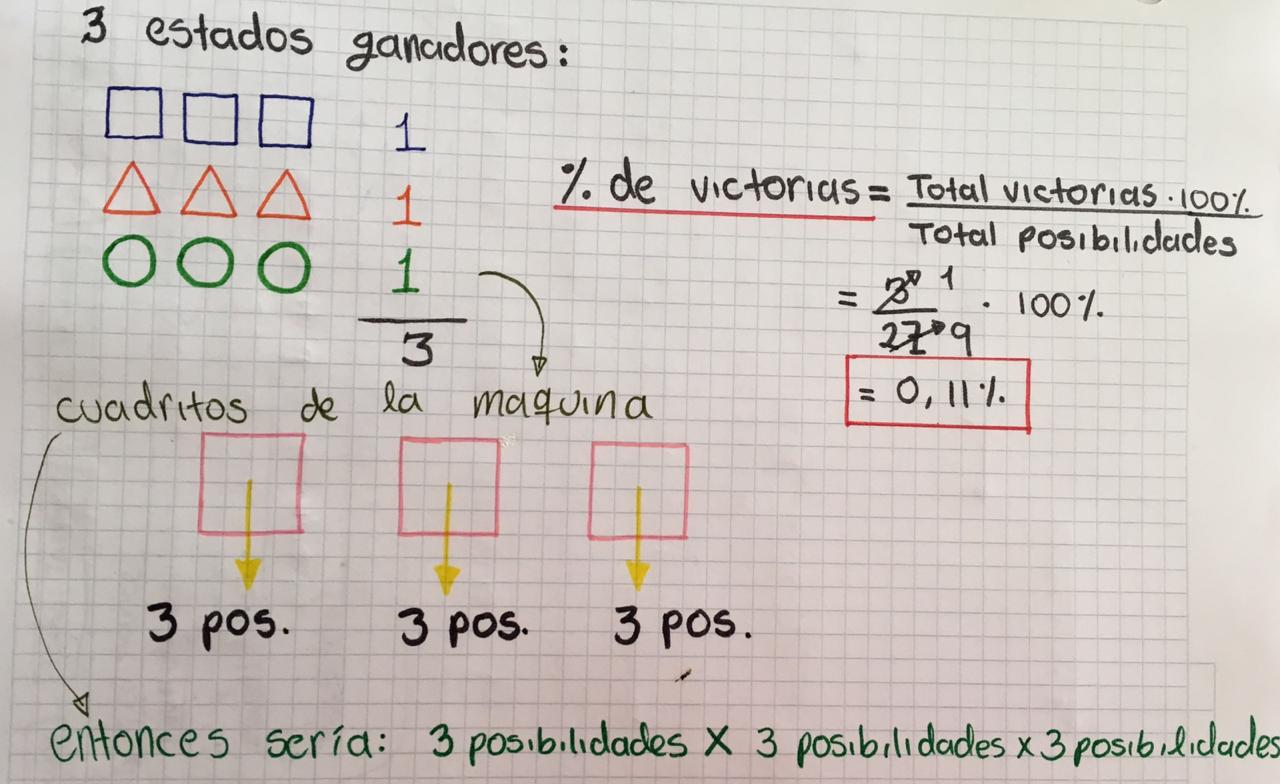


1. ¿De qué tipo de elementos es el tragamonedas? ¿Cuáles son los diferentes elementos?

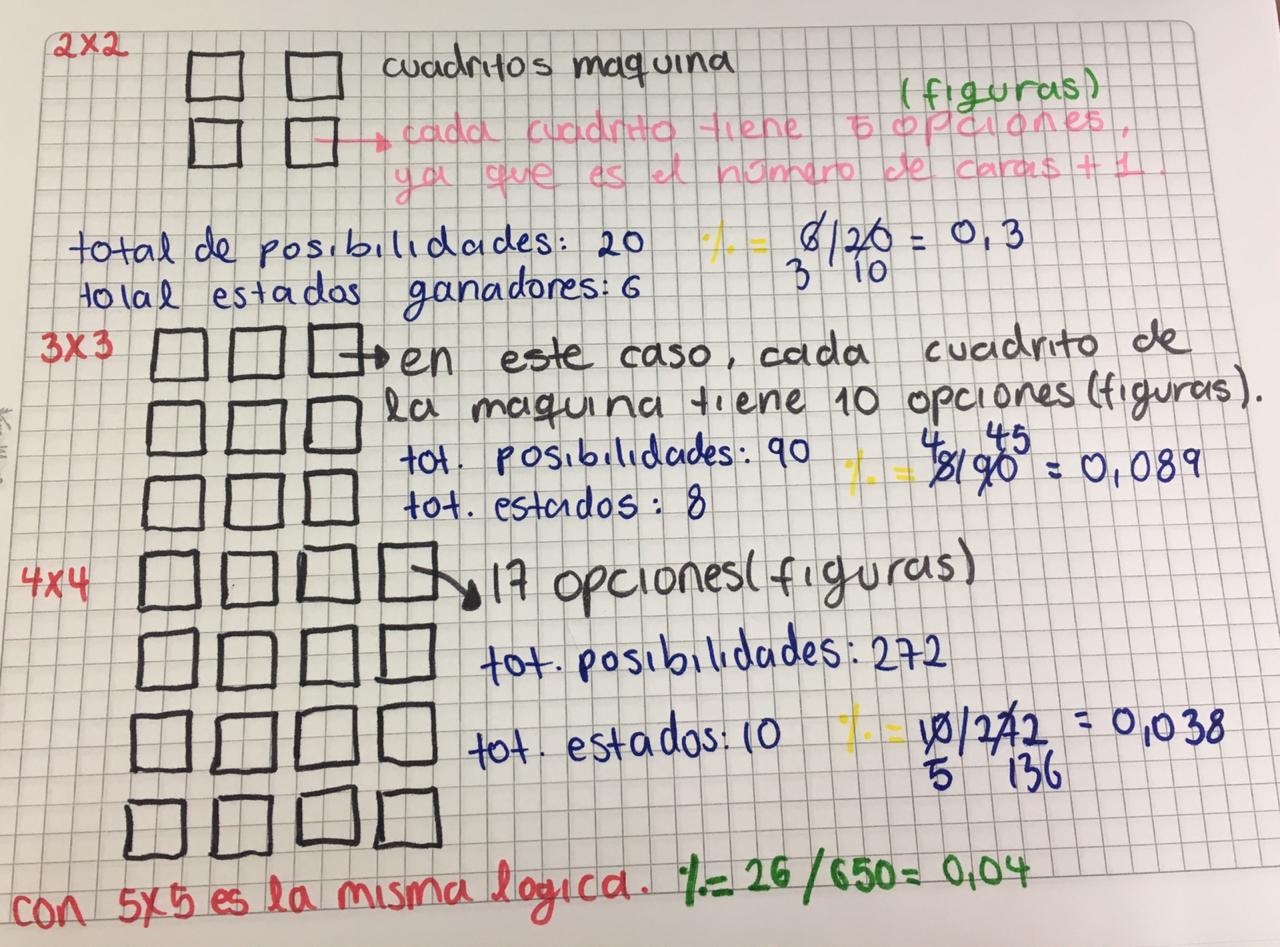
* Rectángulos, círculos, triángulos
* 7 rectángulos (4 que hacen la figura de la maquina (vacío), 3 que son las caras de la maquina), 3 círculos(caras de la maquina), 3 triángulos (caras de la maquina)

1. ¿Cuál es la probabilidad de ganar en esta máquina? Explique su respuesta.

Con el primer tipo de maquina:



Teniendo en cuenta lo indicado en la guía:



1. Clasifiquen los métodos en: constructores, analizadores y modificadores.

Constructores: \_ SlotMachine

Analizadores: \_ isWinningState

\_ percentageOfWinningStates

Modificadores: \_ makeInvisible

\_ makeVisible

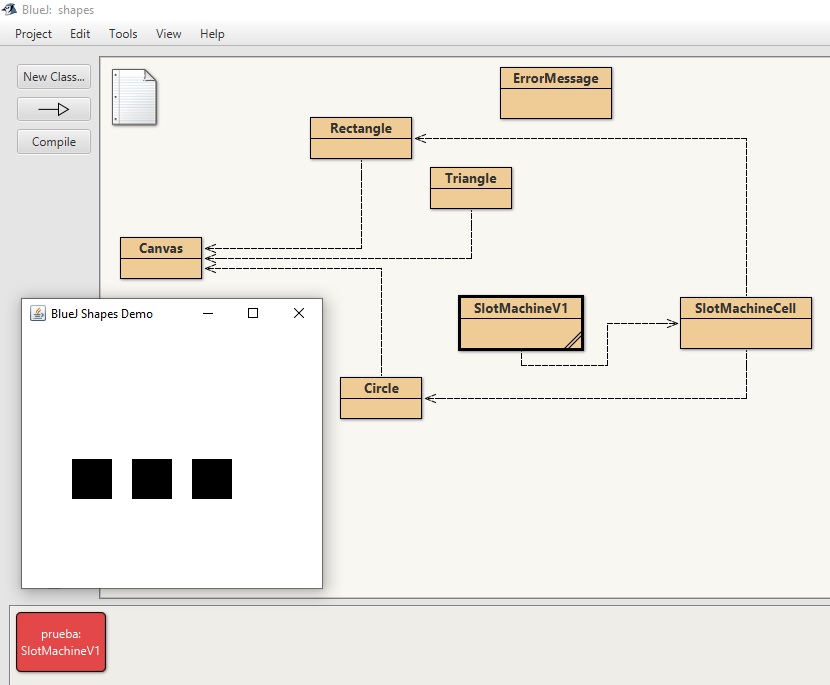
\_ move

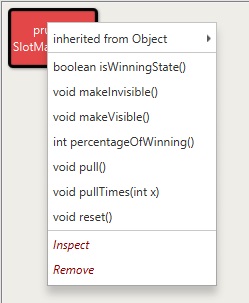
\_ reset

\_ pull

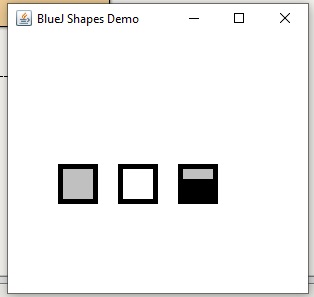
1. Desarrollen la clase SlotMachine considerando los miniciclos. Al final de cada miniciclo realicen una prueba. Capturen las pantallas relevantes.

SlotMachine:

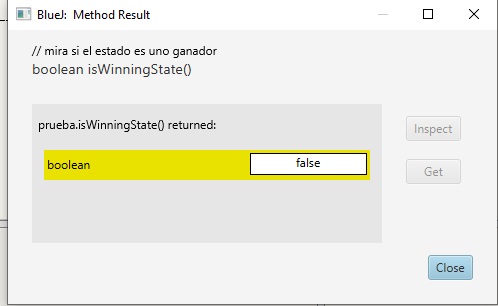




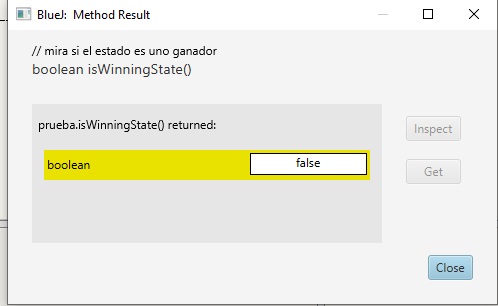
pull:



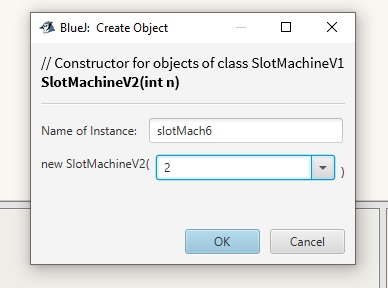
isWinningState:

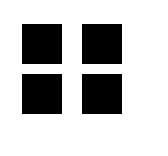


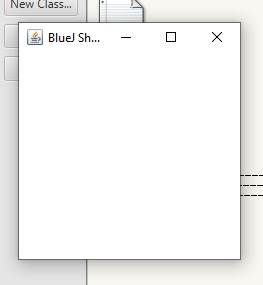
percentageOfWinningStates:



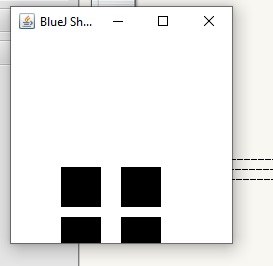
makeInvisible:



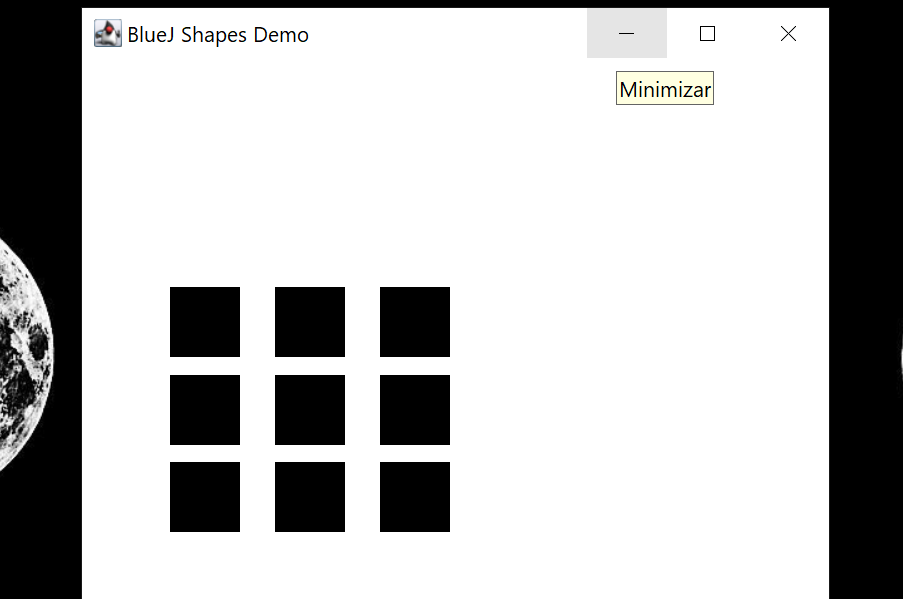


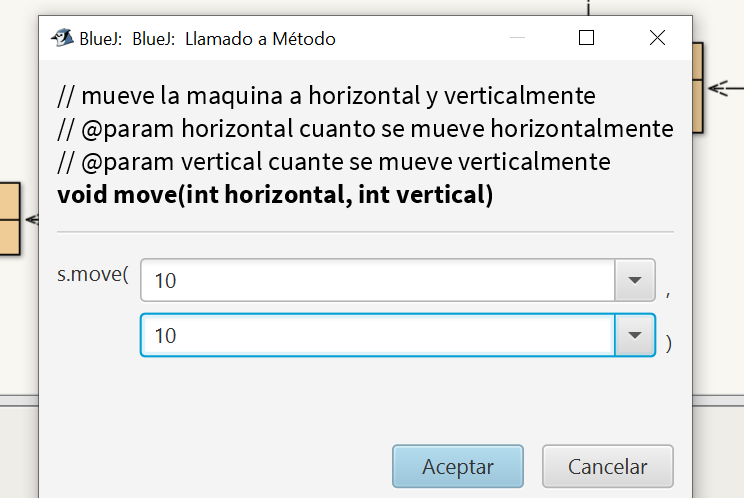


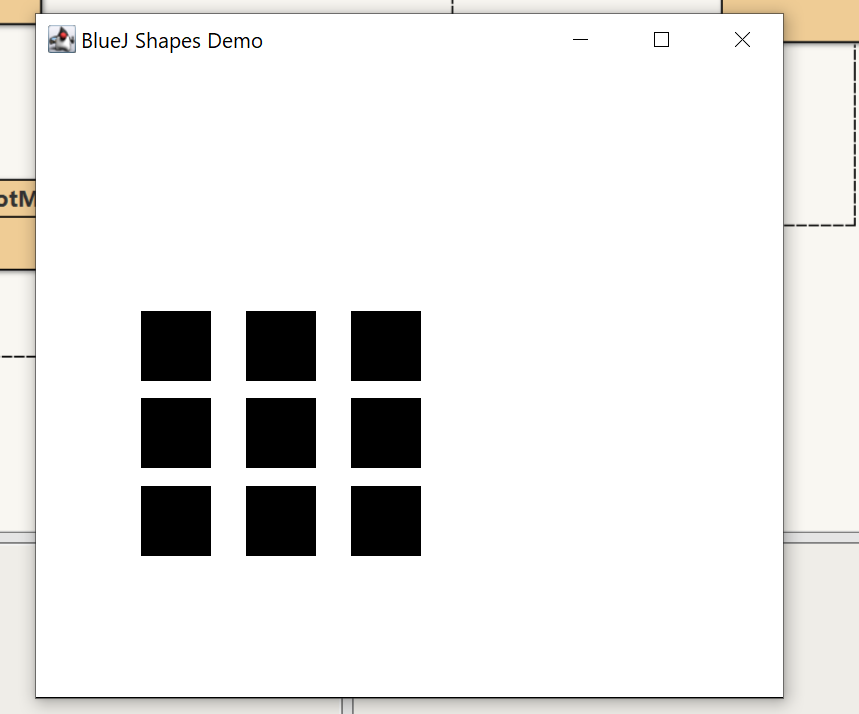
makeVisible:



mover:







1. ¿Cuáles el porcentaje de estados ganadores después de hacer 1, 10, 100 y 1000 jugadas? Presente un análisis de los datos considerando la respuesta dada en 2. Se multiplica el número de jugadas hechas por el porcentaje de victorias del tamaño de la maquina con la que se juega, porque estos porcentajes de estados ganadores depende del tamaño de la máquina.

**Definiendo y creando una nueva clase. SquareSlotMachine:** El objetivo es implementar diseñar e implementar un tragamonedas cuadrado.

**Requisitos funcionales:**

1. Permitir crear un tragamonedas, indicando el tamaño.
2. Permitir reiniciar el tragamonedas
3. Permitir jugar (todas las filas giran)
4. Permitir jugar indicando la fila con el que se desea jugar (1 ...n de arriba a abajo)
5. Permitir consultar si se ha ganado
6. Permitir consultar el porcentaje de juegos ganadores desde el último reinicio

**Requisitos de interfaz:**

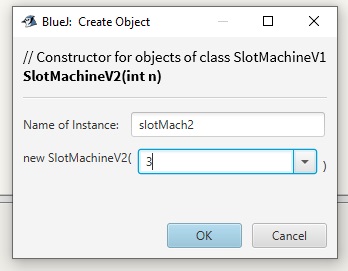
1. Las operaciones se deben ofrecer como métodos públicos de la clase SquareSlotMachine
2. El tragamodedas compuesto debe “sonar” cada vez que llega a un estado ganador
3. Se debe presentar un mensaje amable al usuario si hay algún problema. Consulte y use el método showMessageDialog de la clase JOptionPane.

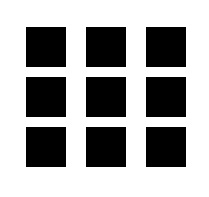
1. Diseñen la clase SquareSlotMachine, es decir, definan los métodos que debe ofrecer.

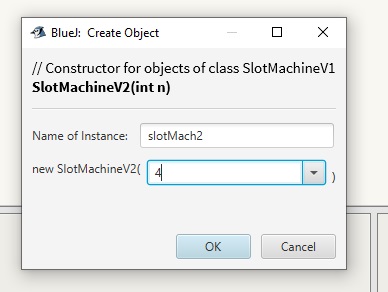
2. Planifiquen la construcción considerando algunos miniciclos.

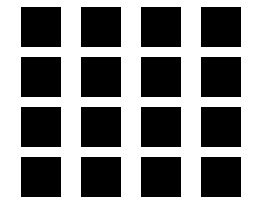
3. Implementen la clase. Al final de cada miniciclo realicen una prueba de aceptación. Capturen las pantallas relevantes.

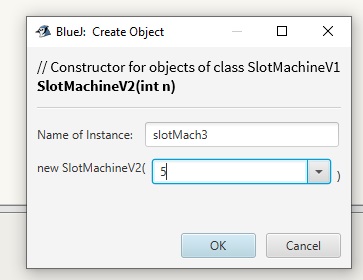
* Permitir crear un tragamonedas, indicando el tamaño:

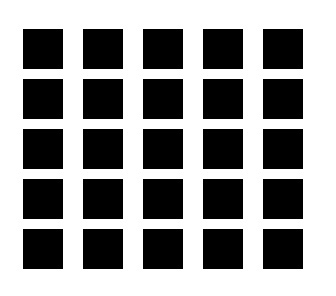






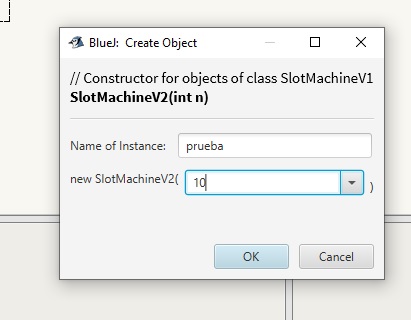


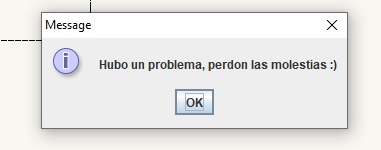




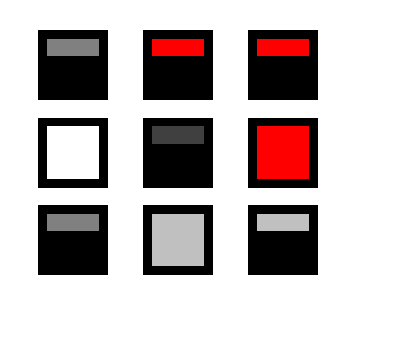
* Se debe presentar un mensaje amable al usuario si hay algún problema:.

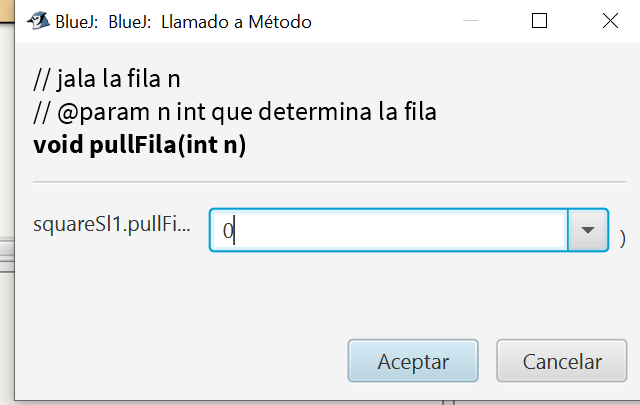
Por ejemplo:

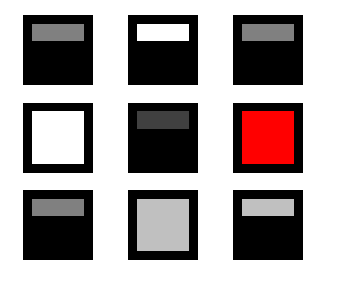


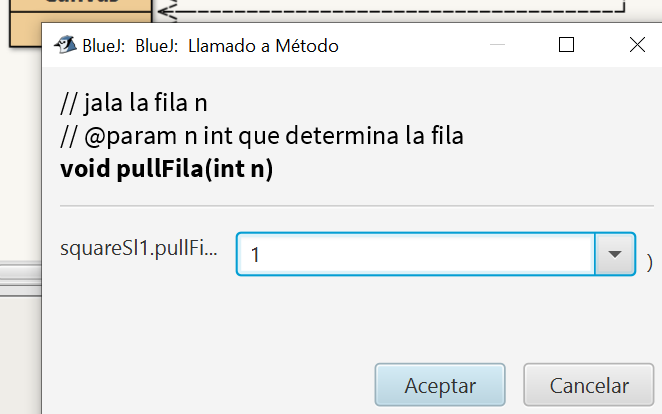


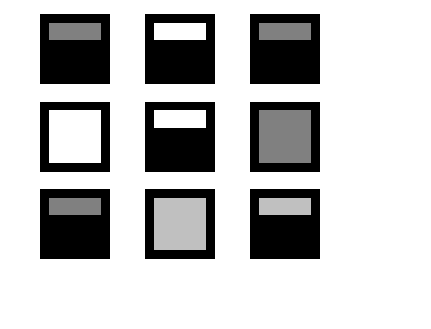
* Permitir jugar indicando la fila con el que se desea jugar (1 ...n de arriba a abajo):

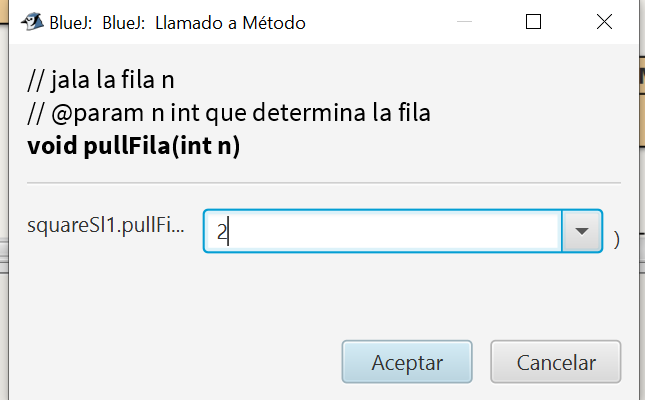


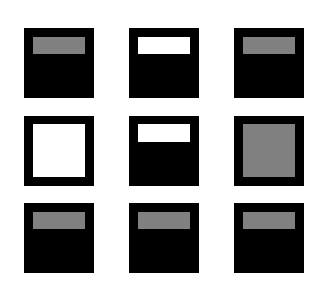












4. Indiquen las extensiones necesarias para reutilizar la clase SlotMachine. Explique.

Creamos la clase del primer cuadrado como una clase base que luego reutilizamos para crear la maquina tragamonedas; además usando lo que nos proporcionaba shapes.

5. Propongan un nuevo método para enriquecer el juego.

Proponemos que el usuario gane el doble del premio si logra formar el perímetro de un cuadrado de celdas iguales.

**Extendiendo una clase. SquareSlotMachine:** El objetivo es crear dos nuevas funcionalidades para jugadas “arregladas: una que encuentre el mínimo número de movimientos para ganar y la otra que permita hacer uno de los movimientos planeados. Nuevos requisitos funcionales

1. Mínimos números de movimientos

2. Mueva siguiendo la estrategia

**RETROSPECTIVA**

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes? (Horas/ Hombre) El tiempo de trabajo fue bastante equitativo entre los dos, trabajamos aproximadamente 12 horas.

2. ¿Cuál es el estado actual del laboratorio? ¿Por qué? Fue un laboratorio muy explicativo, entretenido y didáctico. Nos complementó lo visto en la teoría y nos ayudó a entender más en cuestión de la práctica.

3. Considerando las prácticas XP del laboratorio. ¿cuál fue la más útil? ¿por qué? Trabajar a pares hace que el trabajo no se recargue solo sobre una persona, y que al turnarse de actividades el trabajo rinda más. Tener un compañero también ayuda a tener más caminos a seguir que muchas veces solos no vemos.

4. ¿Cuál consideran fue el mayor logro? ¿Por qué? Haber podido realizar la parte de SlotMachine, debido a que cuando vimos esa parte del laboratorio se nos hizo muy complicado.

5. ¿Cuál consideran que fue el mayor problema técnico? ¿Qué hicieron para resolverlo? Trabajar a distancia. Nos reunimos en tiempos libres que tuvimos en común e hicimos llamadas para explicarnos las ideas mutuamente cuando estábamos cada uno en su casa.

6. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los resultados? Escuchar al otro. Seguir dedicándole el tiempo suficiente a los laboratorios y llevar una buena relación de aprendizaje con el otro.