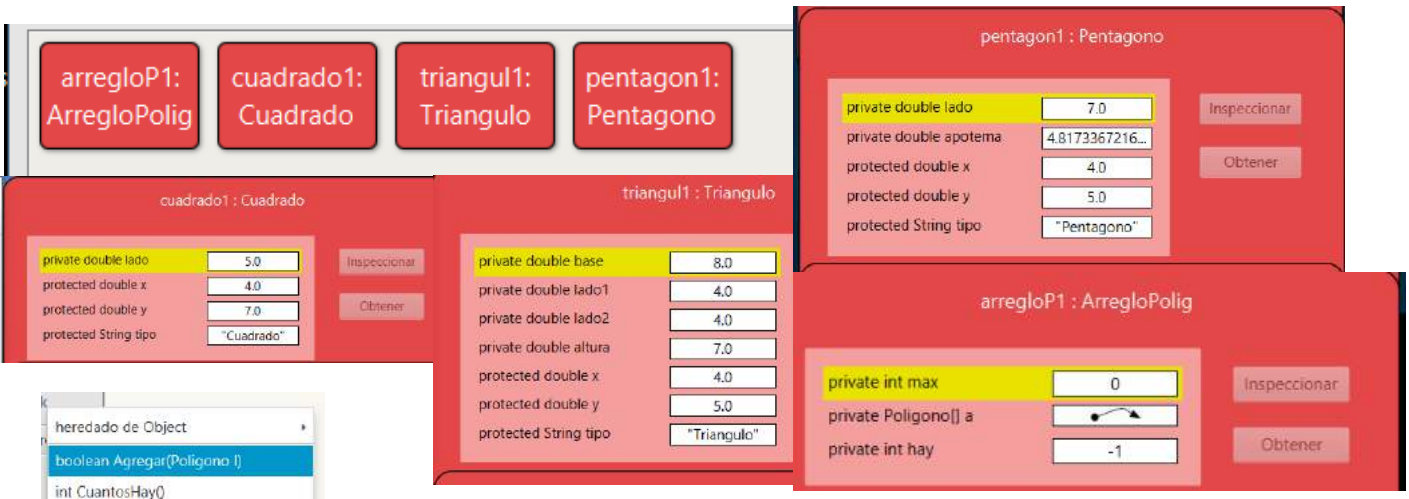
PRUEBA DE HERENCIA DE POLIGONO

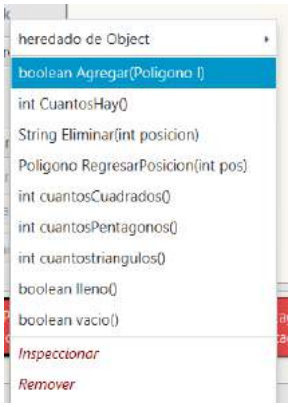
PROGRAMACIÓN ORIENTADA A OBJETOS (POO)

GARCÍA GARCÍA JOSÉ ÁNGEL
23-3-2019

PRUEBA DE HERENCIA DE POLIGONO

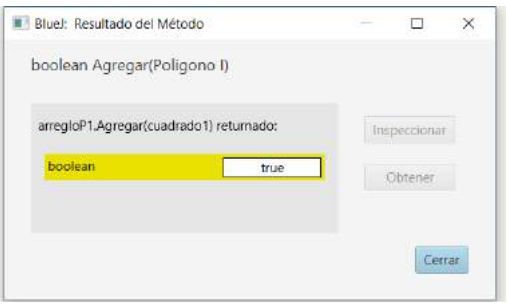
Para empezar a probar el programa, lo primero a hacer, es crear los objetos de la clase, a excepción de la clase abstracta, que no requiere que se cree dicho objeto. Vamos a centrarnos en los métodos de la clase de ArregloPolig que tiene los diferentes métodos.



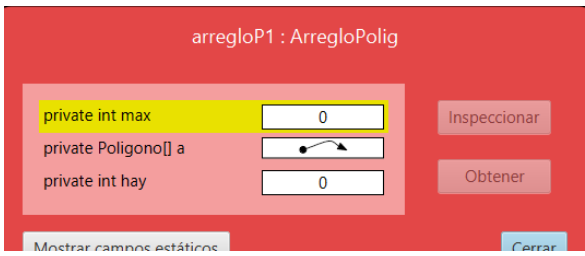


El primer método para ejecutar es **Agregar**, que como su nombre lo indica, con este método vamos a poder agregar elementos al arreglo de polígono, es decir, podemos agregar nuestros polígonos como cuadrado, triangulo y pentágono, pero en este caso solo agregaremos el cuadrado. Si se agrega un elemento regresa un true y si no, un false.

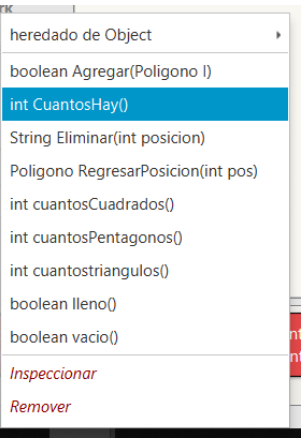
El método requiere como parámetro un polígono, que puede ser cualquier subclase de la clase polígono, es decir, los que hereden de dicha clase. Para probar el método introducimos el cuadrado como el polígono a agregar.



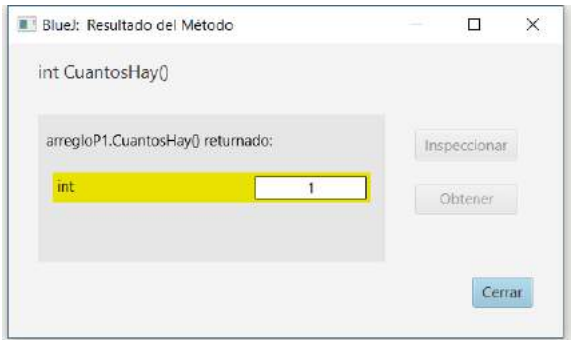
Observando la imagen de la izquierda, podemos ver que el método cumplió satisfactoriamente se función, pues ahora el arreglo tendrá un elemento. Ahora la variable hay pasa a ser de -1 a 0, pues incrementa conforme se agregue un polígono.



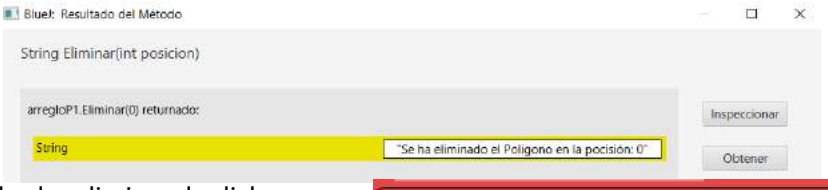
Ahora ejecutaremos el siguiente método, que es **Cuantoshay**, que nos regresa el valor de los elementos que hay en dicho arreglo; es decir, nos regresa el numero de polígonos que hay en el arreglo.



En la imagen de la derecha se observa que el método se ejecuta de buena forma, ya que anteriormente habíamos agregado 1 cuadrado por lo que el arreglo si tenía un polígono.

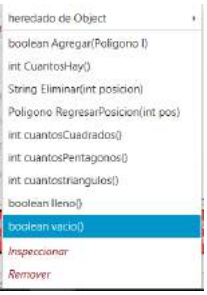


El otro método es **Eliminar**, que su función es eliminar el elemento del arreglo de Polígonos en la posición que se le indica.



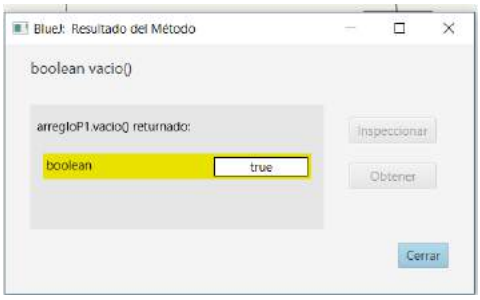
Como se puede observar, el método funciona muy bien, pues le introducimos como parámetro la posición 0, que es donde estaba el cuadrado, eliminando dicho elemento. Además, restando 1 a la variable hay, que pasa de 0 a -1.

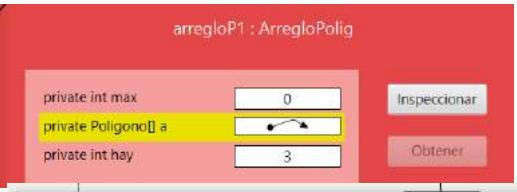
Partiendo de que ahora el arreglo no tiene elementos podemos comprobar otro método.



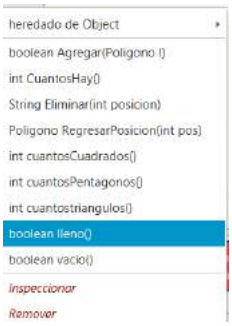
El método para probar ahora es el de **vacio**, que retorna true si el arreglo de polígonos no tiene elementos; es decir, se encuentra vacío o false en caso contrario; es decir, cuando tenga al menos un elemento.

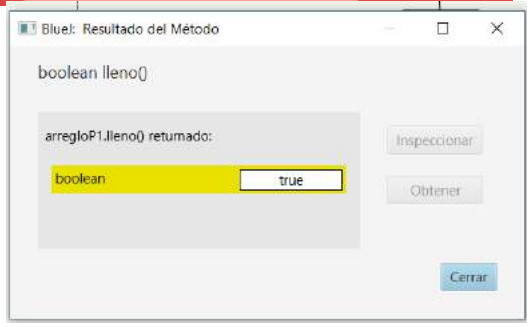
Observando la imagen de la derecha, podemos concluir que el método se ejecuto de manera efectiva, ya que como habíamos dicho antes, el arreglo de polígonos ya no tenia elementos.





Para probar el siguiente método, hay que partir de que el arreglo de polígonos tiene 4 elementos y su capacidad de igual forma es 4.



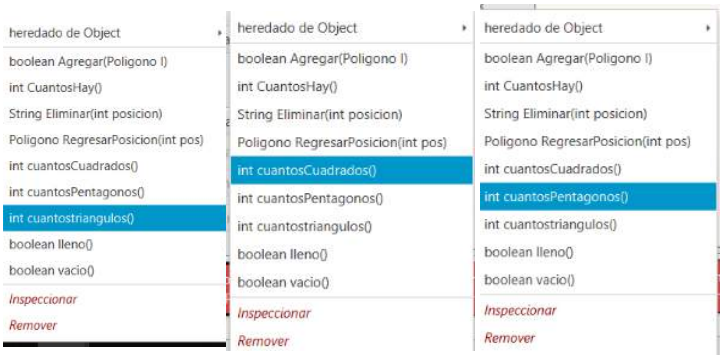


El método es **lleno**, que regresa true si el arreglo se encuentra lleno y un false si tiene un numero de elementos menor a la capacidad.

En la imagen de la izquierda se observa que después de ejecutar el método, arrojo un true, ya que como habíamos dicho, nuestro arreglo se encontraba lleno, por lo tanto, el método cumple con su función satisfactoriamente.

Ahora partiendo que el arreglo de polígonos está lleno de 2 cuadrados, 1 triangulo y 1 pentágono, procedemos a ejecutar los métodos posteriores.

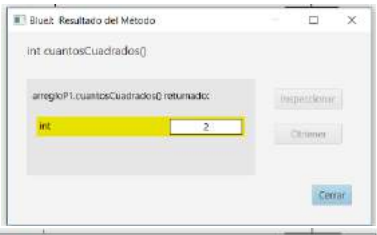
Los 3 métodos siguientes son similares, lo que cambia es que comprueban diferentes polígonos, los 3 retornan el numero de la cantidad del tipo de polígonos que hay en el arreglo.



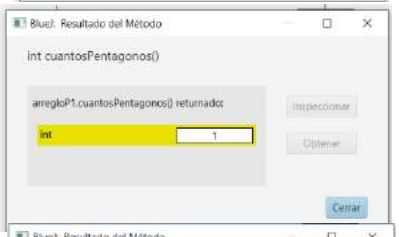
El de **cuantostriangulos**, retorna la cantidad de triángulos

El de **cuantosCuadrados**, retorna la cantidad de cuadrados

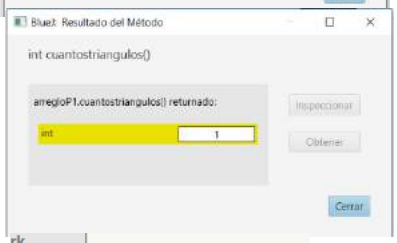
El de **cuantosPentagonos**, retorna la cantidad de pentágonos



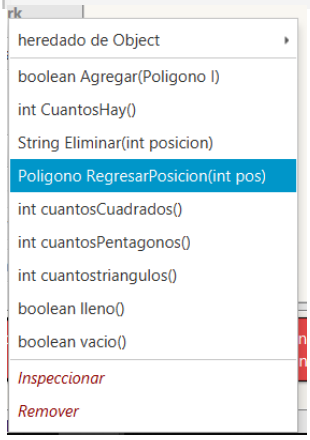
El método que retorna la cantidad de cuadrados es correcto, ya que habíamos planteado que nuestro arreglo tenía 2 cuadrados almacenados.



El método que retorna la cantidad de pentágonos funciona correcto, pues retorno 1, y era la cantidad de pentágonos que si se había planteado.

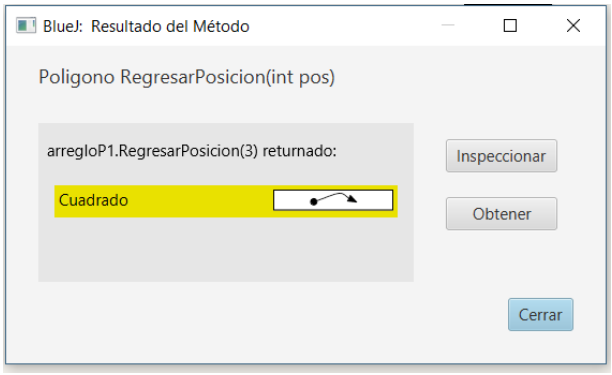


El método que retorna la cantidad de triángulos funciona de igual forma que los restantes, ya que retorno 1 y ese es el numero que hay de triángulos; es decir, se tenia 1 triangulo en el arreglo.

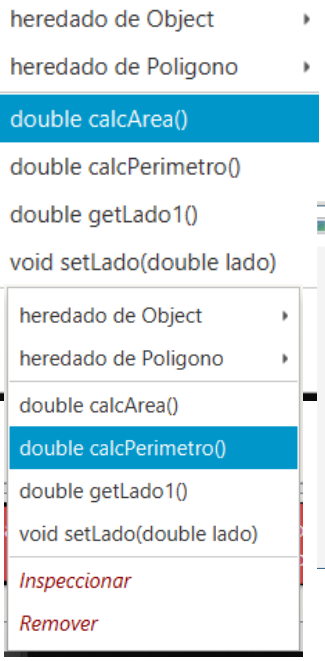


El otro método es **RregresarPosicion**, que regresa el elemento en la posición que le indiquemos, en este caso regresaría un polígono del arreglo de polígonos.

Ejecutando el método y poniendo de parámetro la posición 3, nos regresaría el último elemento del arreglo, que corresponde a un cuadrado, por lo que el método es correcto, como se observa en la imagen de la derecha.

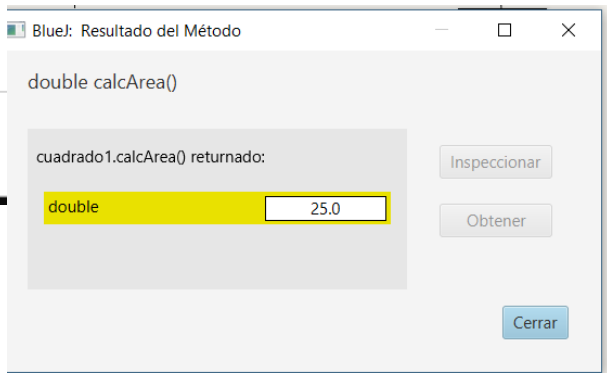


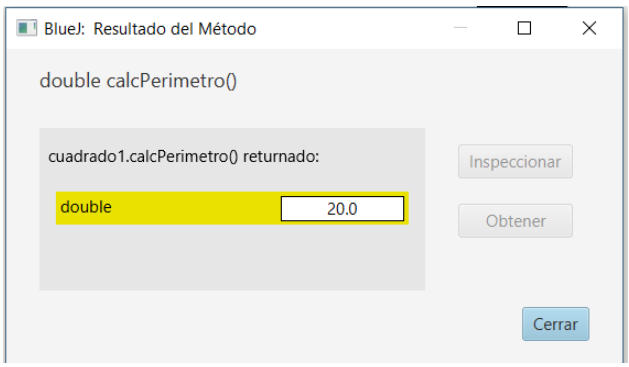
Estos son todos los métodos de la clase ArrelgoPolig, pero el programa tiene mas métodos en las otras clases, es decir, el cuadrado, el triángulo y el pentágono, tienen el método de calcular área y perímetro, en realidad son metodos heredado de la clase padre Polígonos, pero estos metodos son abstractos, ya que cada subclase ejecuta el método de forma diferente.



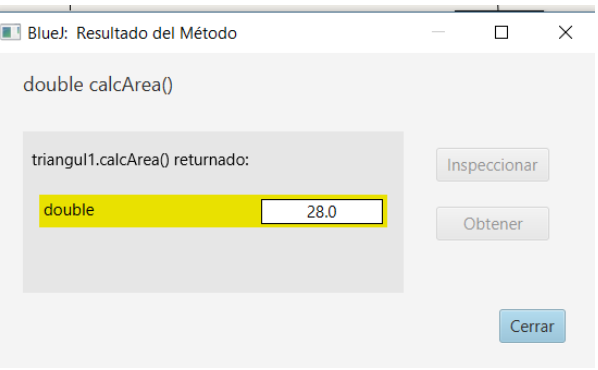
Los metodos **calcArea** y **calcPerimetro**, corresponde a la subclase Cuadrado. El primer método calcula el área del cuadrado y el segundo su perímetro.

Sabiendo que el lado mide 5, podemos calcular el área haciendo la operación 5x5=25 y el perímetro a 5x4=20. Por lo tanto, los metodos son correctos ambos.





De igual forma se ejecutan ambos metodos en la subclase triangulo.

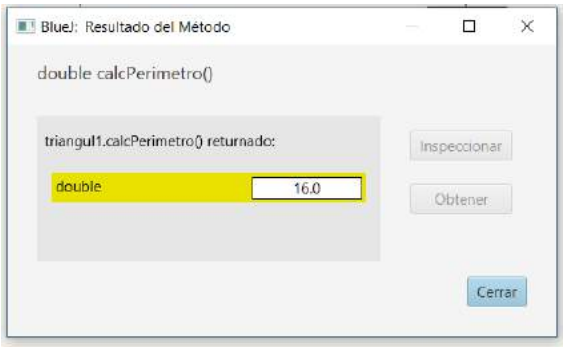


Partiendo de los datos:

b=8 y h=7 su área es igual a $\frac{bxh}{2}$

por lo tanto $\frac{8x7}{2} = \frac{56}{2} = 28.$

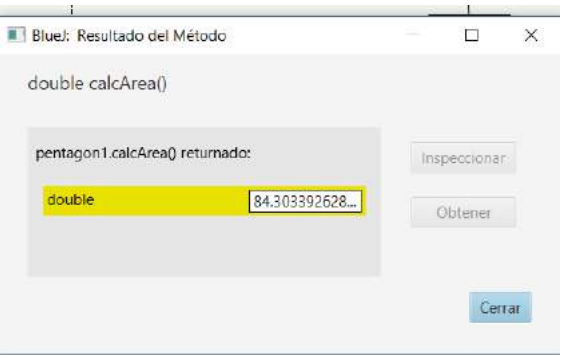
Y su perímetro es 8+4+4=16. Por lo tanto, ambos metodos funcionan correctamente.



Así también se ejecutan ambos metodos, pero ahora en la subclase pentágono.

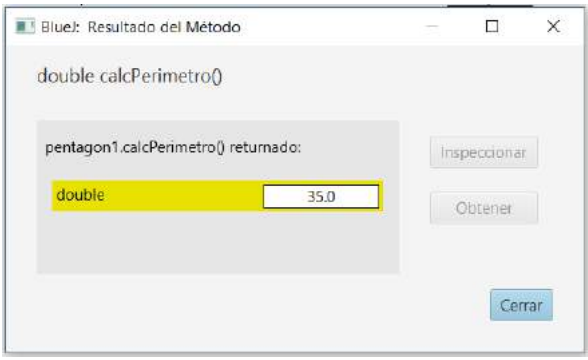
Sabiendo que el L=7 y $\theta = 72/2$ y el perímetro=7x5=35

apotema=$\frac{L}{2\tan(\theta)}$=$\frac{7}{2\tan(36)}$=4.817336722



Área=$\frac{perimetro\ x\ apotema}{2}$ = $\frac{35\ x\ 4.817336722}{2} = 84.30339263$

Concluimos que ambos metodos si funcionan como se esperaba, haciendo los cálculos de la forma correcta.



En las 3 subclases de polígonos podemos ejecutar el método que tiene la clase abstracta, que corresponde al **centro**, que tiene cada polígono.