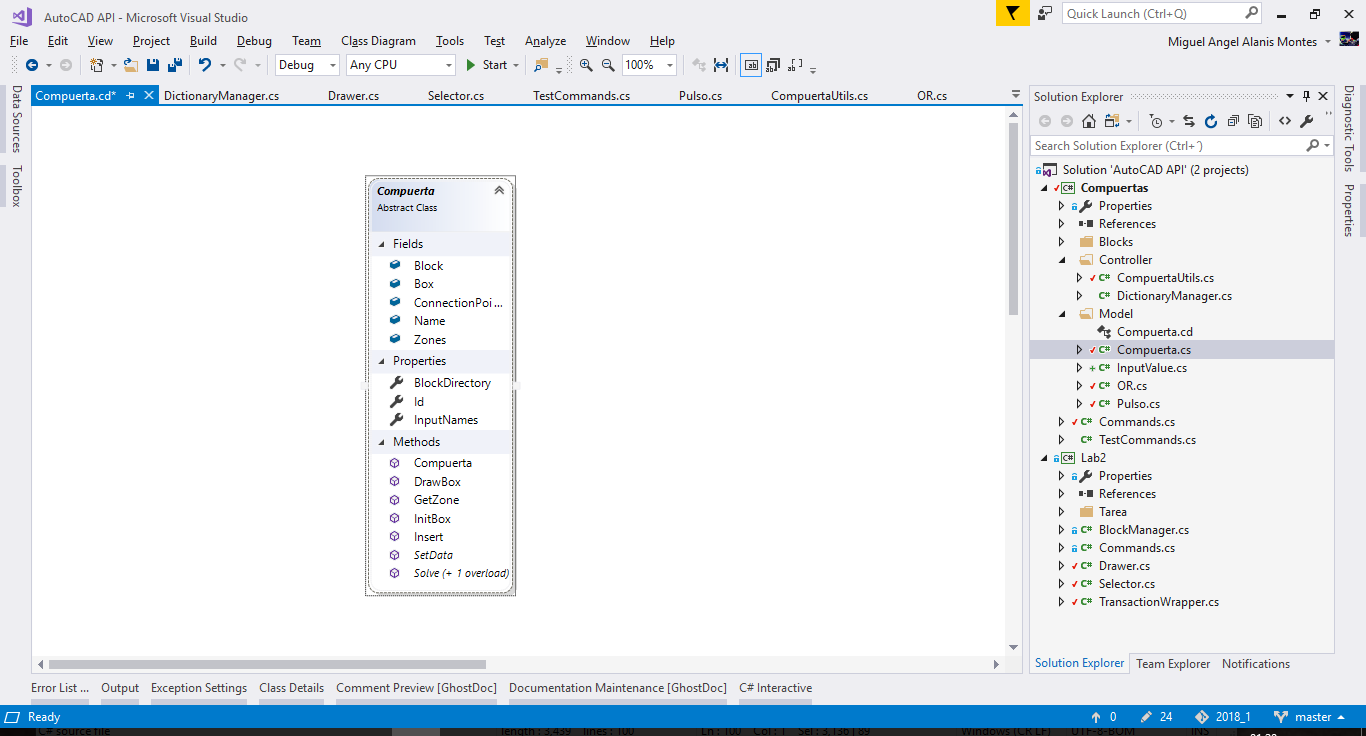
Diseño Compuerta lógica

# Diagrama de clases



# Insertar Compuerta

La compuerta para insertarse necesitara un nombre, y se insertará como una referencia de bloque.

* Name
* Block

Las compuertas creadas serán guardadas en una colección, para que puedan ser accedidas de manera posterior.

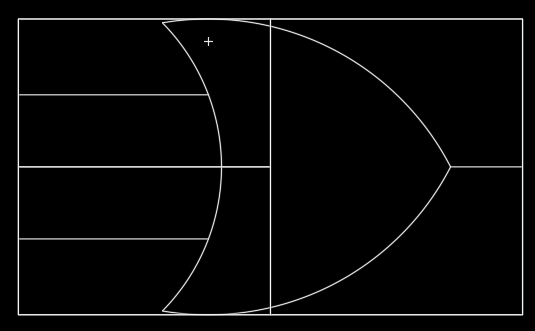
El proceso de inserción se define en el comando TestInsertCompuerta. El seudocódigo para insertar la compuerta es.

* Solicitar un punto de inserción al usuario
* Seleccionar el nombre de la compuerta a insertar
* Cargar el bloque con la herramienta de BlockManager.cs
* Dibujar el bloque con Drawery guardarlo en la variable Block
* Inicializamos los parámetros de la compuerta con una cadena vacía usando BlockManager.cs
* Se guarda la compuerta en la colección del comando Dictionary<Handle, Compuerta> Compuertas.

# Pruebas de contacto

Se definió un comando que permite tocar una identificar la zona de contacto de una compuerta, las zonas y las áreas de contacto se definen con la función InitBox.

* **Box**: Define la caja de colisión de la compuerta en el plano XY
* **Zones**: Establece las zonas de contacto de las zonas, para una compuerta de dos entradas se tienen 3 zonas, las zonas se obtienen con la función extendida CompuertaUtils GetZonesTwoInputs.



Los nombres de las zonas se definen en la propiedad InputNames, esta propiedad es virtual por la que puede cambiarse.

Para saber si un punto está dentro de una zona se utilizó el siguiente algoritmo.

* El punto *P*(*x*,*y*), y los vértices del rectángulo definidos por *A*(*x*1,*y*1),*B*(*x*2,*y*2),*C*(*x*3,*y*3),*D*(*x*4,*y*4)
* Se calculan las áreas de los triángulos △*APD*,△*DPC*,△*CPB*,△*PBA*
* Se suman las área de los triángulos
* Cuando el área es mayor a la del polígono el punto está afuera.
* Cuando el área es igual es un punto interno
* Si un triángulo tiene área igual a 0 se trata de un punto en un segmento, se considera como interior.

La solución del algoritmo se observa en la función CompuertaUtils.TestPoint, para revisar a que zona pertenece un punto en la compuerta se usa el método **GetZone**.

El método devuelve el nombre de la zona y la geometría de la zona.

El comando que prueba la funcionalidad se define TestZone

El pseudo código se definió de la siguiente manera

* Solicitar que seleccione una compuerta al usuario
* Ubicar la compuerta en la colección actual
* Dibujar las zonas disponibles
* Tocar un punto y mostrar un mensaje
  + Si es una zona especificar la zona, en otro caso notificar que es un punto exterior
* Borrar las líneas de las zonas

# Probar compuerta

La compuerta se conectará a dos pulsos insertados, con el comando dibujar pulsos y se insertará el pulso resultante.

Los puntos de conexión se guardan con la variable ConnectionPoints, y se inicializan con InitBox

Los métodos Solve, resulven la función, para un pulso o para una entrada.

SetData establece el valor de las entradas en la compuerta.

La información se ejecuta con el comando TestCompuerta, el pseudocódigo es el siguiente

* Solicitar la selección de un pulso
* Solicitar la conexión del pulso
* Solicitar la selección del segundo pulso
* Solicitar la conexión del pulso
* Realizar el cálculo con la compuerta seleccionada.
* Conectar una línea del pulso a la entrada correspondiente
* Insertar el pulso de salida
* Conectar una línea de la salida al pulso de salida
* Insertar los valores en la compuerta para el último valor del pulso.