

EXAMEN BLANCA ARIAS SAEZ, GRUPO Y

Hago 12/2:

Depuración:

30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42

```
{
    divisor -= dividendo;
    cont++;
}

// Devolvemos el resultado.
return cont;
}

// Evento que se llama al pulsar el botón "Dividir".
1 referencia | jrgs, Hace 8 días | 1 autor, 1 cambio
private void btDividir_Click(object sender, EventArgs e)
{
    trv
```

100 %

✓ No se encontraron problemas.

Automático

Buscar (Ctrl+E)

Profundidad de búsqueda: 3

Nombre	Valor
dividendo	12
divisor	2
this	{ExamenED1EV2324.Form1, Text: Form1}

100 %

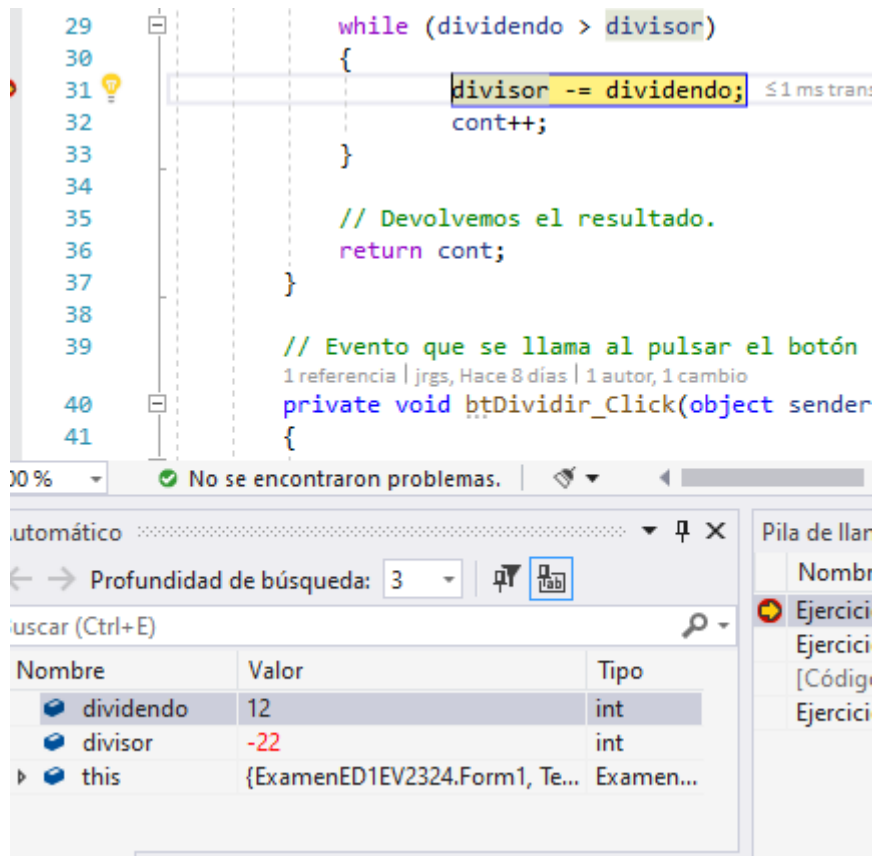
✓ No se encontraron problemas.

Automático

Buscar (Ctrl+E)

Profundidad de búsqueda: 3

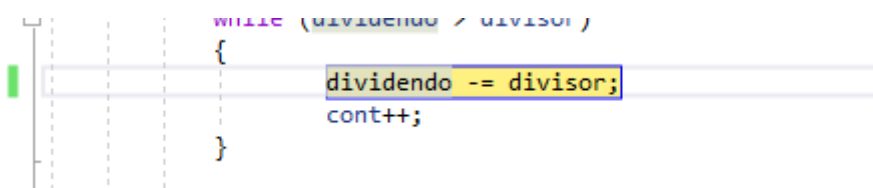
Nombre	Valor	Tipo
dividendo	12	int
divisor	-10	int
this	{ExamenED1EV2324.Form1, Text: Form1}	ExamenED1EV2



Se mete en un bucle infinito porque estoy restando al divisor el dividendo. Siempre se va a cumplir la condición while ya que el dividendo siempre será mayor al divisor.

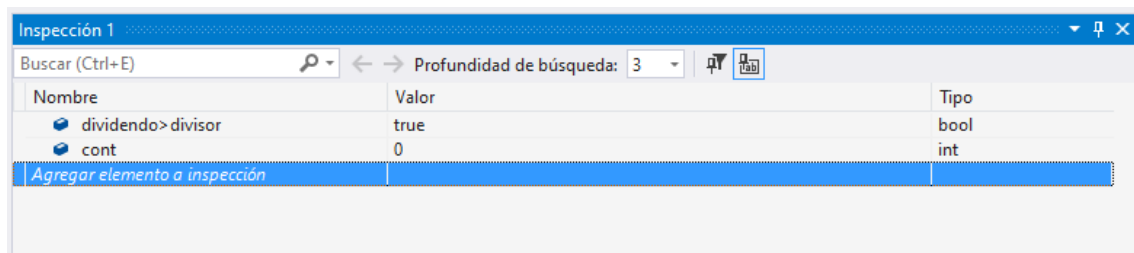
Modifico el código:

Hay que restar al dividendo el divisor:



Depuro:

Agrego variable para inspeccionar: **cont**



Para ver hasta dónde llega el contador.

Buscar (Ctrl+E)			← → Profundidad de búsqueda: 3	🔍
Nombre	Valor	Tipo		
dividendo > divisor	false	bool		
cont	5	int		
Agregar elemento a inspección				

llega hasta 5

Sigue estando mal.

Modifico la condición while: Por ejemplo, si llega a 2-2=0 (dividendo=divisor), no entraría ya que (dividendo > divisor)

**while (dividendo >= divisor)**

Inicio la depuración:

```

24 // Declaramos una variable para almacenar el resultado.
25 int cont = 0;
26
27 // Restamos el divisor al dividendo
28 //
29 while (dividendo >= divisor)
30 {
31     dividendo -= divisor;
32     cont++;
33 }
34
35 // Devolvemos el resultado.
36 return cont;
37
38
39 // Evento que se llama al pulsar el botón "Dividir".
40 private void btDividir_Click(object sender, EventArgs e)
41 {
42     trv

```

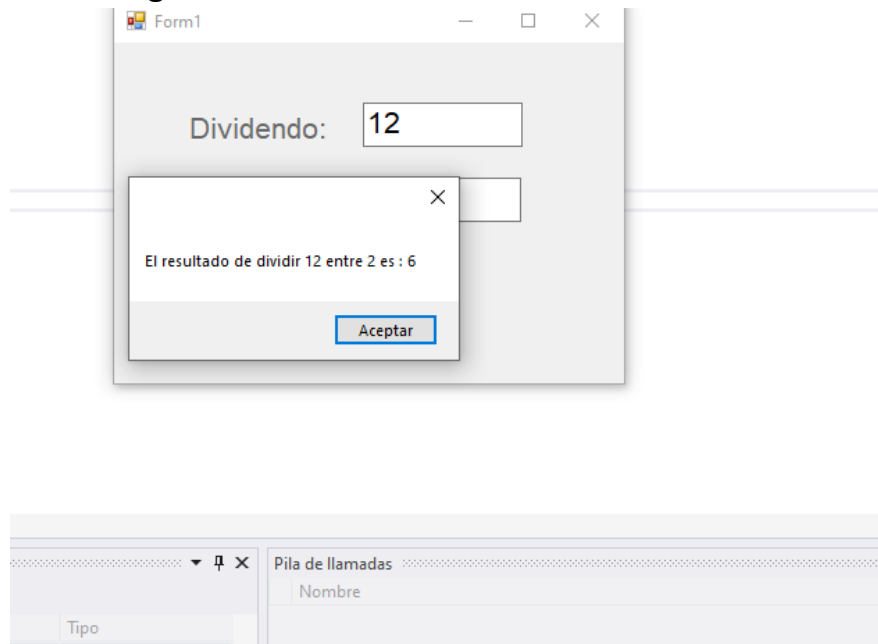
Nombre	Valor	Tipo
dividendo > divisor	false	bool
cont	6	int

Inspección 1

Buscar (Ctrl+E) ← → Profundidad de búsqueda: 3 🔍

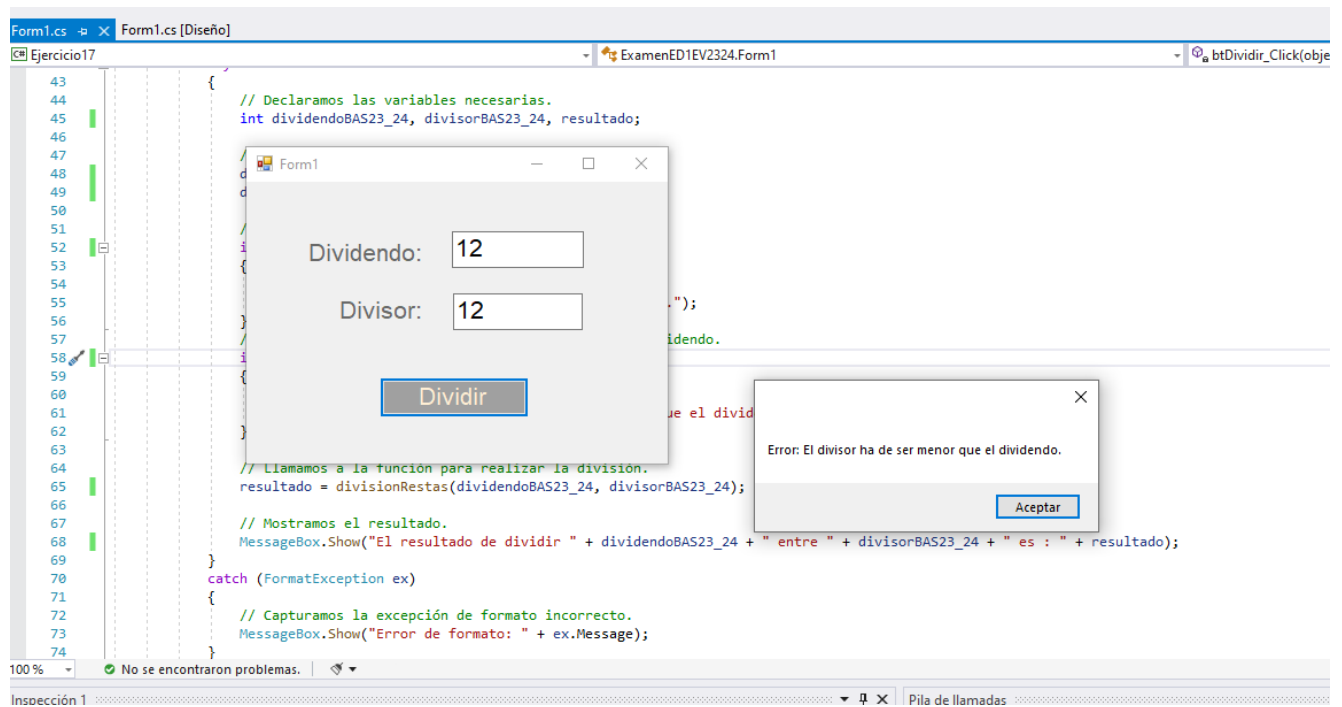
Automático Variables locales Inspección 1

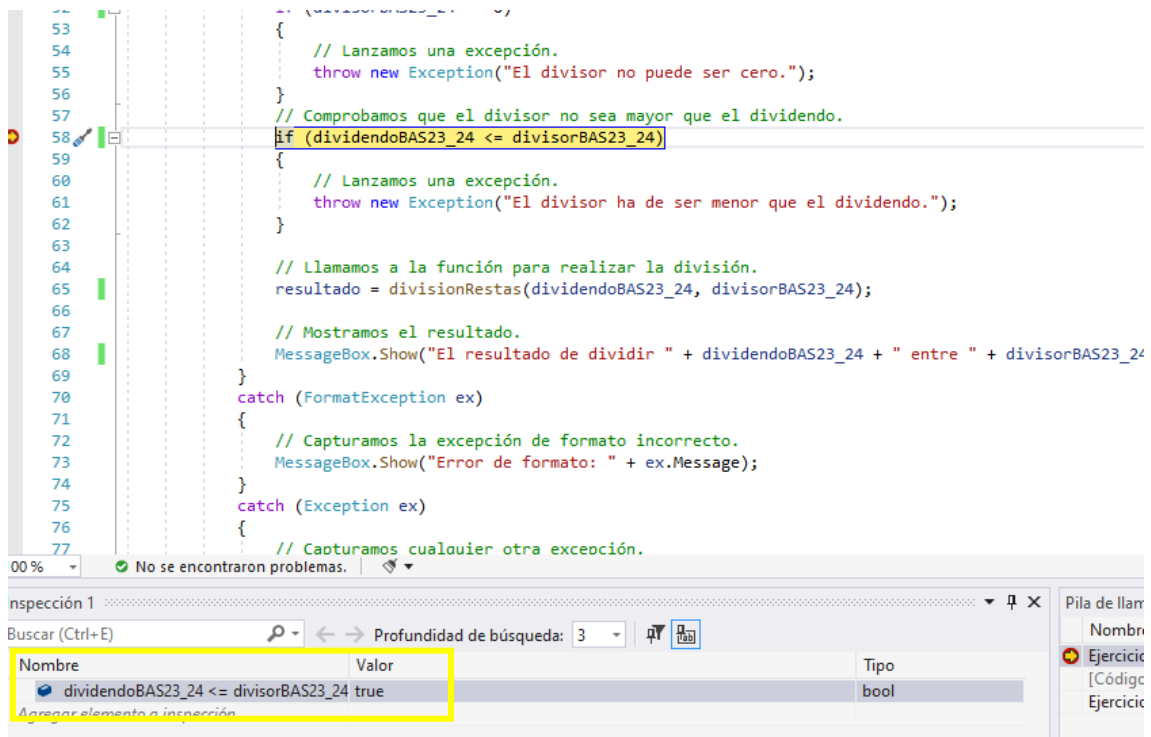
Ahora llega hasta 6.



```
}  
// Comprobamos que el divisor no sea mayor que el dividendo.  
if (dividendo <= divisor)  
{
```

Modifico esa línea de código, ya que si hago 12/12 me salta la excepción.



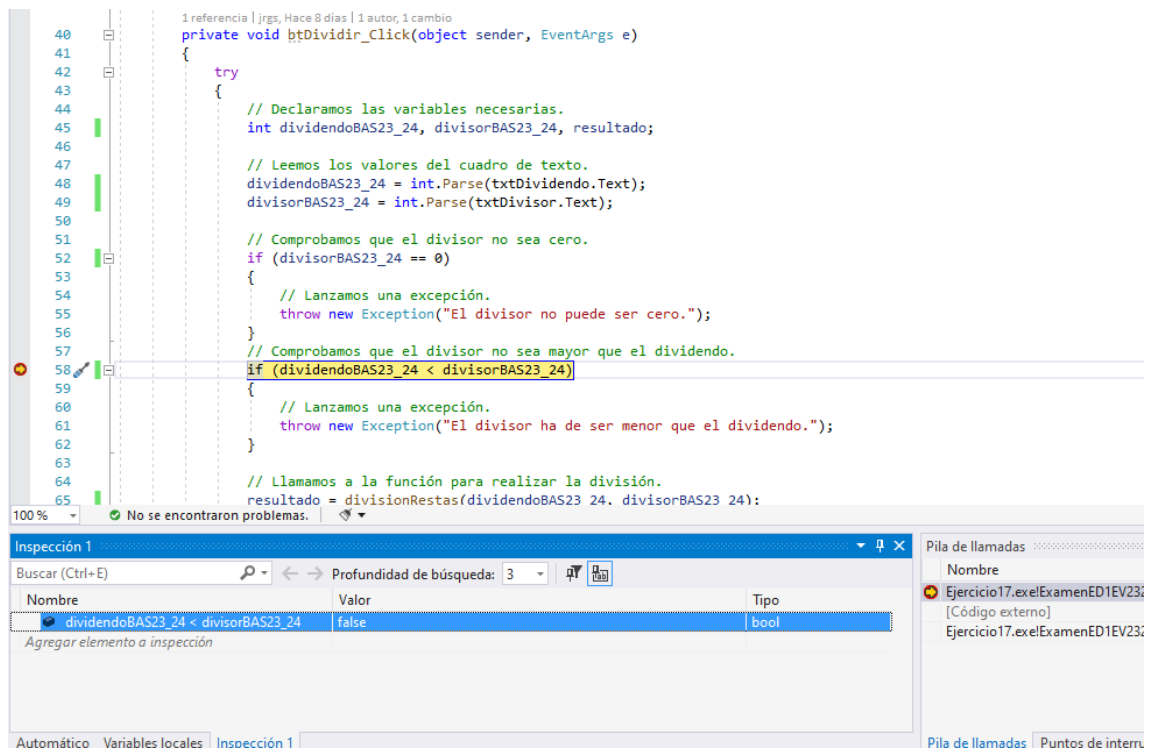


Como hay un menor igual <= entra en ese if.

Modifico condición:

**if (dividendoBAS23\_24 < divisorBAS23\_24)**

Vuelvo a hacer: 12/12



Ya no entra en ese if:

Form1

Dividendo: 12

Divisor: 12

Dividir

Continuar

Subproceso:

Marco de pila:

ExamenED1EV2324.Form1

btDividir\_Click

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

// Declaramos las variables necesarias.

int dividendoBAS23\_24, divisorBAS23\_24, resultado;

// Leemos los valores del cuadro de texto.

dividendoBAS23\_24 = int.Parse(txtDividendo.Text);

divisorBAS23\_24 = int.Parse(txtDivisor.Text);

// Comprobamos que el divisor no sea cero.

if (divisorBAS23\_24 == 0)

{

// Lanzamos una excepción.

throw new Exception("El divisor no puede ser cero.");

}

// Comprobamos que el divisor no sea mayor que el dividendo.

if (dividendoBAS23\_24 < divisorBAS23\_24)

{

// Lanzamos una excepción.

throw new Exception("El divisor ha de ser menor que el dividendo.");

}

// Llamamos a la función para realizar la división.

resultado = divisionRestas(dividendoBAS23\_24, divisorBAS23\_24);

100%

No se encontraron problemas.

Inspección 1

Buscar (Ctrl+E)

Profundidad de búsqueda: 3

Nombre	Valor	Tipo
dividendoBAS23_24 < divisorBAS23_24	false	bool

Agregar elemento a inspección

El resultado de dividir 12 entre 12 es : 1

Aceptar

Pila de llamadas

Nombre

Ejercicio17.exeExamenED1EV2324.Form1.btDividir\_Click(obj)

[Código externo]

Ejercicio17.exeExamenED1EV2324.Program.Main() Línea 19

Automático

Variables locales

Inspección 1

Pila de llamadas

Puntos de interrupción

Configuración de