Depuración en Codeblocks

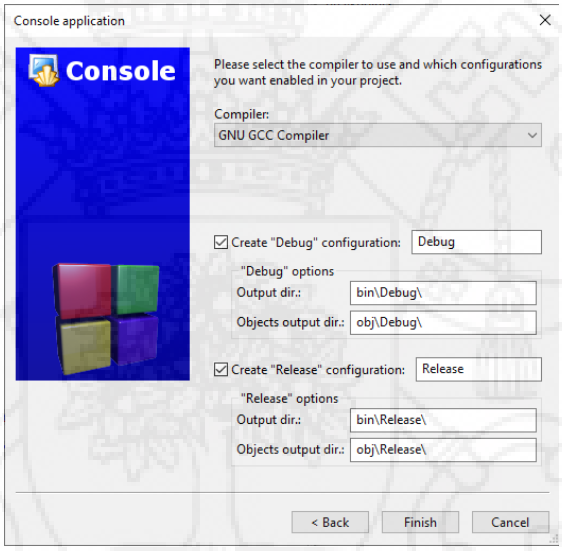
El proceso de ***"Depuración de un programa"*** consiste en detectar y corregir los errores que tenga el mismo.

El **debugger** *(depurador)* nos permite analizar con detalle cómo se comporta el código fuente que hemos implementado a medida que lo vamos ejecutando. Esto resulta de gran utilidad a la hora de detectar y corregir errores en nuestros programas.

El proceso a seguir se detalla a continuación:

1.- Crear un nuevo proyecto de programación.

Menú File -> New -> Project -> Console Application



Es muy importante que estén marcadas las opciones *"Create "Debug" configuration"* y *"Create "Release" configuration"* , tal y como aparecen en la imagen anterior. De este modo, conseguiremos que el compilador genere información de depuración en la salida de la compilación *("Debug configuration"),* y también, que se genere el código objeto y el ejecutable de la aplicación *("Release" configuration).*

Escribir el programa.

```c
/*****************************************************************************************
    Programa: ejemploDepuracion
    Tarea   : Solicita al usuario la introducción por teclado de dos números enteros (x, y)
              y después, calcula el valor de la suma, resta y producto de los mismos. También
              calcula la suma total de los x primeros números naturales.
    Realizado por: Pilar Grande González
    Fecha        : 9 noviembre de 2020
*****************************************************************************************/
#include <stdio.h>
#include <locale.h>

int main(){
    int x, y;
    int suma, resta, producto; //almacenan los resultados de los cálculos
    int i, sumaTotal;

    setlocale(LC_ALL, "spanish");

    //Pedimos los números al usuario
    printf("\n\t\t PETICIÓN DE DATOS ");
    printf("\n\t\t ================= \n\n");
    printf("\n\t\t Introduzca dos números enteros: ");
    scanf("%d %d", &x, &y);

    //Realizamos cálculos 4
    suma = x + y;
    resta = x / y;
    producto = x * y;

    //Suma los primeros x números enteros
    sumaTotal = 0;
    for (i = 1; i <= x; i++)
        sumaTotal = sumaTotal + i;


    //Mostramos resultados en pantalla
    printf("\n\t\t RESULTADOS ");
    printf("\n\t\t ========== \n\n");

    printf("\n\t\t La suma es %d", suma);
    printf("\n\t\t La resta es %d", resta);
    printf("\n\t\t El producto es %d", producto);
    printf("\n\t\t La suma de los %d primeros números naturales es: %d", x, sumaTotal);

    printf("\n\n\t\t Pulse <Intro> para finalizar...\n\n");
    getchar();
}
```

2.- Poner **breakpoints** *(puntos de ruptura)* en aquellos puntos donde queramos detener la ejecución del programa.

Colocar el cursor del ratón junto al número de la línea donde queramos establecer un breakpoint.

Click en el botón derecho del ratón -> Add Breakpoint

O bien, Click con el botón izquierdo del ratón

Cada breakpoint que establezcamos quedará marcado con un punto rojo. Esto indica que cuando se ejecute el depurador, la ejecución del programa se detendrá en ese punto, justo antes de ejecutar esa instrucción del programa.

```
11      //Pedimos los números al usuario
12      printf("\n\t\t PETICIÓN DE DATOS ");
13      printf("\n\t\t ================= \n\n");
14      printf("\n\t\t Introduzca dos números enteros: ");
15      scanf("%d %d", &x, &y);
16
```

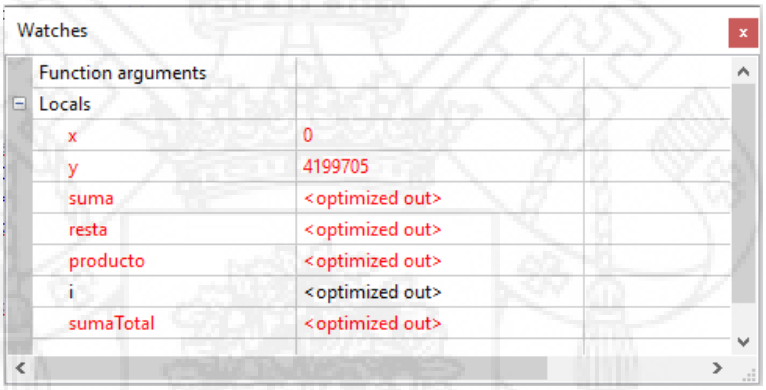
[Cuando queramos quitar todos los puntos de ruptura que hemos establecido, haremos lo siguiente:

Menú Debug -> Remove all breakpoints]

3.- Abrir **Watches** *(ventana de observación)*:

Menú Debug -> Debugging windows -> Watches

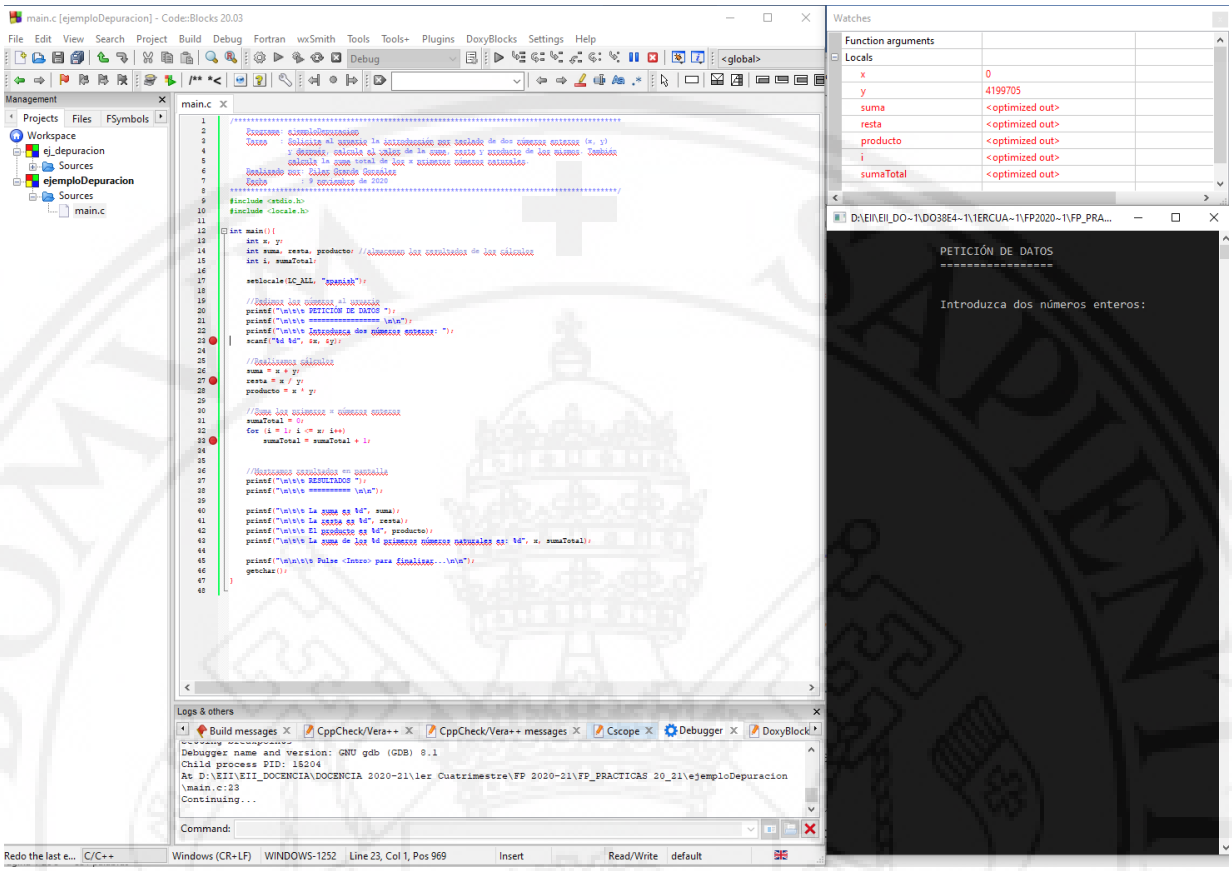
Se muestra una tabla. Cuando arranque la ejecución del programa se colocarán en ella las variables que han sido declaradas en el programa.



Podremos observar la evolución de estas variables a medida que avance la ejecución del programa *(observaremos cómo cambia su valor)*.

4.- Ejecutar el programa PERO desde el menú Debug:

Menú Debug -> Start/Continue (arranca el depurador y ejecuta nuestro programa)



Colocad bien las ventanas para poder trabajar de forma cómoda (observad la figura anterior).

Tened en cuenta también que disponeis de una barra con botones asociados a distintas opciones del menú Debug:

El significado de los principales botones que vamos a utilizar es el siguiente:

	Start/Continue	Arranca el depurador / Continúa la ejecución del mismo
	Run to cursor	Ejecuta el programa hasta llegar a la instrucción donde tenemos situado el cursor y ahí se detiene
	Next Line	Ejecuta el programa paso a paso (instrucción a instrucción). Cada vez que presionamos este botón, ejecuta la siguiente instrucción del programa
	Step Into	Ejecuta el programa paso a paso (instrucción a instrucción) y si una sentencia es una llamada a un subprograma, entra en el subprograma y también continúa ejecutando instrucción a instrucción dentro del mismo
	Stop Debugger	Detiene el depurador

Iremos ejecutando las distintas instrucciones de nuestro programa pulsando

 ,  , o bien,  según necesitemos en cada momento.

Al ejecutar una instrucción que modifica el valor que almacena una de las variables observadas, ese cambio quedará registrado en la ventana de observación (Watches). Esto nos servirá para comprobar el comportamiento del programa, la evolución de sus variables y detectar errores en el mismo.