Práctica 2 – Parte I

"Proceso de depuración de un programa"

1. Introducción

Como ya hemos estudiado en clase, una vez superada la fase de compilación de nuestro código fuente, podemos pasar a probar su ejecución. En ocasiones, la ejecución del programa no produce los resultados esperados. Nos encontramos con errores en la lógica del programa: el programa no tiene errores sintácticos, pero al ejecutarlo no realiza su tarea de forma adecuada. Se producen los llamados "errores de ejecución" *(errores lógicos)*. Detectar y corregir errores de este tipo constituye uno de los principales problemas para los programadores primerizos. Debemos llevar a cabo un proceso de "DEPURACIÓN" de nuestro programa.

Para llevar a cabo esta tarea, los IDE nos brindan una herramienta especial llamada "depurador"
(debugger).

2. Objetivos

En esta práctica aprenderás las principales funcionalidades básicas del depurador del IDE Codeblocks.

Al finalizar esta práctica serás capaz de:

- Ejecutar un programa paso a paso
- Establecer y eliminar puntos de parada *(breakpoint)*, y ejecutar un programa hasta el siguiente punto de parada
- Observar el contenido que almacenan las variables declaradas en tu programa durante la ejecución del programa

3. Consideraciones

Debes crear un único proyecto de programación siguiendo el mismo procedimiento que llevamos a cabo en la práctica anterior.

Nombre del proyecto:

- Para nombrar los proyectos de programación que desarrollemos en la asignatura, utilizaremos el **nº de grupo que tenéis asignado en el listado de** grupos de laboratorio que he publicado en el campus virtual.

- Por ejemplo: **FP_P2A_GXX**, donde XX representa el nº de grupo de laboratorio, por ejemplo 03 => el nombre del proyecto sería **FP_P2A_G03** => Proyecto 2A realizado por el grupo 03 de laboratorio

2. Proyecto a realizar: Enunciado.

Proyecto 2A .- Depuración

1. Crea un proyecto de programación en lenguaje C que solicite al usuario la introducción por teclado de dos números enteros (x, y) y después, calcule el valor de la suma, resta y producto de los mismos, así como, la suma total de los x primeros números naturales. **El código fuente del proyecto se muestra a continuación**:

```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   : 15 de octubre de 2021
*****************************************************************************************/
#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
    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 + 1;


    //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. Ejecuta el programa y comprueba si funciona correctamente.
3. Debes depurar el programa para localizar el error existente.
   - Consulta el documento "Cómo depurar un programa" en el que explico cómo llevar a cabo este proceso en Codeblocks.
   - Una vez tengas en pantalla el código fuente y la ventana de observación (Watches), establece puntos de ruptura en las líneas **26, 27, 28, 31 y 33**.
   - Ejecuta el programa con ayuda del depurador. Prueba la ejecución introduciendo los números **25 y 87**
   - Cuando alcances el primer punto de ruptura, sigue ejecutando el programa paso a paso

     (Menú Debug => Next Line). Observa cómo se actualiza el contenido de las variables y completa las siguientes tablas:

Variable	Contenido
x	
y	
suma	
resta	
producto	

i	sumaTotal
1	
2	
3	

   - Una vez detectado el error, corrígelo y ejecuta el programa de nuevo para comprobar que funciona correctamente.

Es muy importante que sepas depurar correctamente un programa de forma que detectes errores lógicos rápidamente. El depurador del IDE nos ayuda en esta tarea.

Práctica 2 – Parte II

"Estructuras de Control: Secuencial, Selectiva, Iterativa"

1. Objetivos

- Comprender el funcionamiento de cada una de las tres estructuras básicas de control y decidir en qué situación resulta indicado utilizar cada una de ella.
- Realizar una correcta validación de los datos de entrada introducidos por el usuario

2. Proyectos a realizar :

En esta práctica debéis crear varios proyectos de programación. En cada caso debéis escribir en lenguaje C un programa que realice la tarea indicada *(observad la validación de datos de entrada que se debe realizar en cada caso)*:

Proyecto 2B.- Validación de rango

✓ Escribe un programa en C que pida al usuario ingresar dos números enteros positivos que representen un rango (mínimo y máximo). El programa validará que el número mínimo es menor que el máximo y que ambos son positivos. La salida en pantalla será similar a la del siguiente ejemplo:

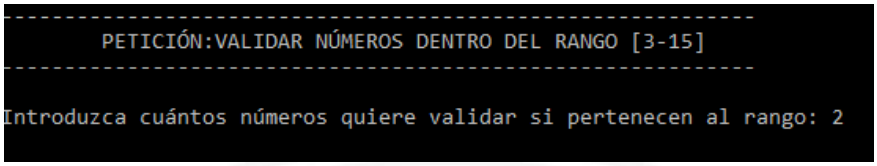
```
----------------------------------------
       PETICIÓN MÍNIMO Y MÁXIMO DEL RANGO
----------------------------------------
       Introduzca el número mínimo de rango: 45
       Introduzca el número máximo de rango: 3
       ERROR:  El número mínimo debe ser menor que el máximo

       Presione una tecla para continuar . . .
```

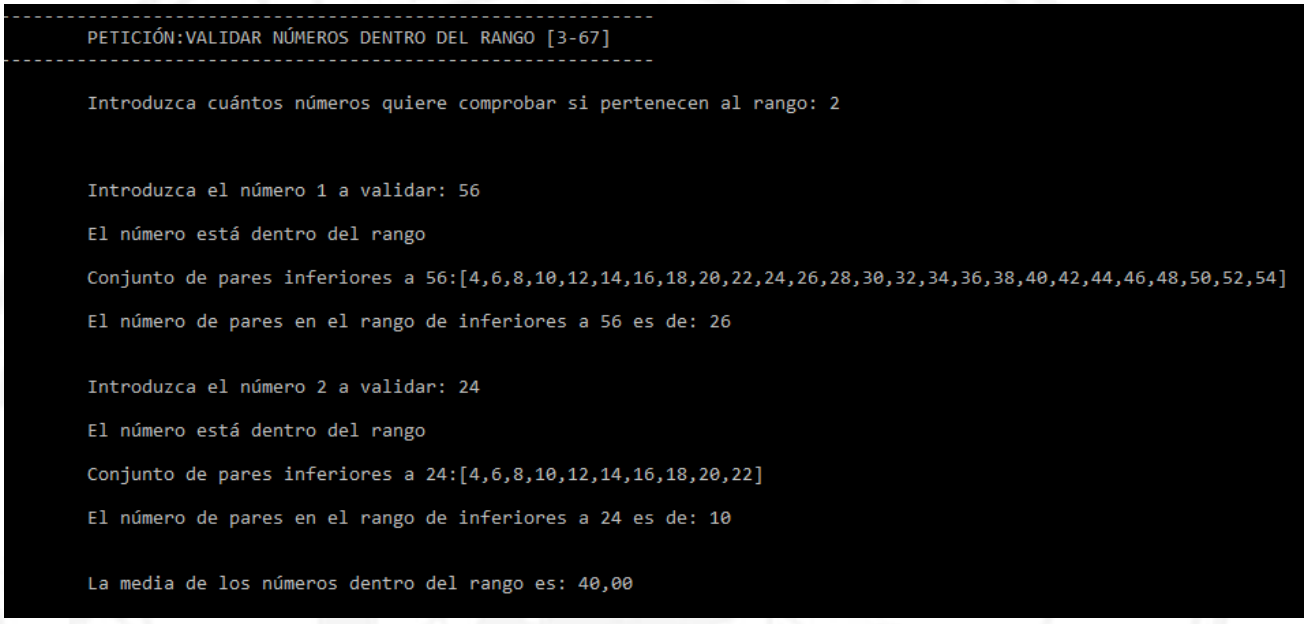
```
----------------------------------------
       PETICIÓN MÍNIMO Y MÁXIMO DEL RANGO
----------------------------------------
       Introduzca el número mínimo de rango: 3
       Introduzca el número máximo de rango: -14
       ERROR: El número máximo debe ser mayor que 0.

       Presione una tecla para continuar . . .
```

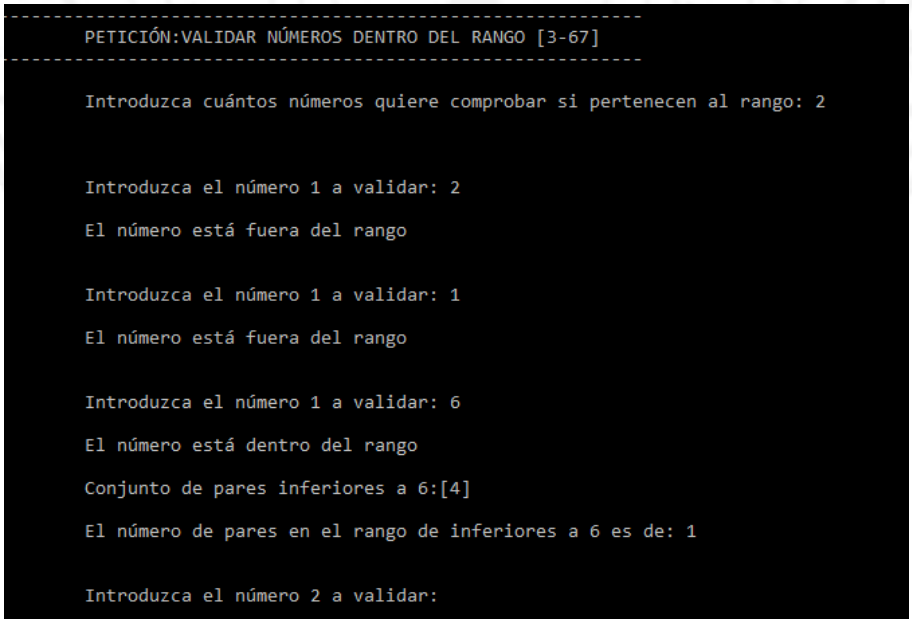
- ✓ Después, pedir al usuario que introduzca cuántos números quiere validar, tal y como se muestra en la imagen que viene a continuación (fijaos que en la cabecera aparece el rango seleccionado por el usuario):



- ✓ Y posteriormente ya valide los números que indique el usuario tantas veces como nos ha indicado, si el número a validar es correcto aparecerá:

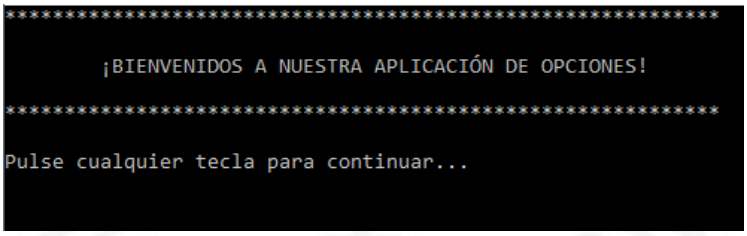


- ✓ Si hay errores, hay que informar al usuario:

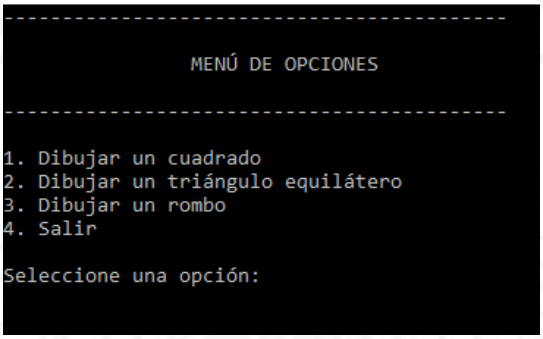


Proyecto 2C.- Menú de opciones

Implementar un programa en lenguaje C que muestre en pantalla un mensaje de bienvenida



y, a continuación, limpie pantalla y, de forma repetitiva, muestre el siguiente menú de opciones:

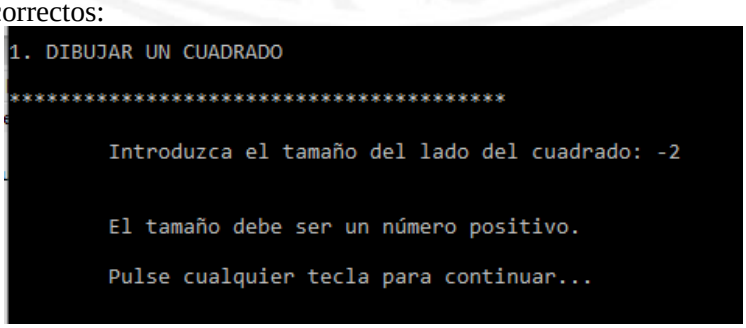


Cada vez que el usuario elija una de las opciones 1 a 4, se limpiará pantalla y, a continuación, se ejecutará la tarea correspondiente. Después se volverá a limpiar pantalla y se mostrará de nuevo el menú anterior.

- IMPORTANTE: Cuidad especialmente la interfaz con el usuario *(limpiar pantalla antes de ejecutar una opción del menú, hacer una pausa ("Pulse cualquier tecla para continuar…"), etc.)*
- IMPORTANTE:
  - No olvidéis validar los datos de entrada introducidos por el usuario en cada caso
  - Es muy importante que la salida en pantalla siga el mismo formato que el indicado en los ejemplos *(recuadros en negro)*. Prestad especial atención al mensaje de cabecera que se muestra en cada caso.

La tarea a realizar al ejecutar cada una de las opciones del menú se indica a continuación *(se muestra la salida en pantalla que se obtendrá al ejecutar cada opción del menú, una vez ejecutada cada opción debe presentar de nuevo el menú para elegir nueva opción):*

OPCIÓN 1:
Si introduce valores incorrectos:



Al presionar cualquier tecla, limpie pantalla y vuelva a pedir de nuevo el tamaño hasta que sea correcto:

```
1. DIBUJAR UN CUADRADO
****************************************
       Introduzca el tamaño del lado del cuadrado:
```

Ejecución con valores correctos:

```
1. DIBUJAR UN CUADRADO
****************************************
       Introduzca el tamaño del lado del cuadrado: 5
              *****
              *****
              *****
              *****
              *****
       Pulse cualquier tecla para continuar...
```

OPCIÓN 2:

Si introduce valores incorrectos:

```
2. DIBUJAR UN TRIÁNGULO
****************************************

       Introduzca la altura del triángulo equilátero: -4

       La altura debe ser un número positivo.
       Pulse cualquier tecla para continuar...
```

Al presionar cualquier tecla, limpie pantalla y vuelva a pedir la nueva altura hasta que sea correcto:

```
2. DIBUJAR UN TRIÁNGULO
****************************************

       Introduzca la altura del triángulo equilátero:
```

Ejecución con valores correctos:

```
2. DIBUJAR UN TRIÁNGULO
****************************************

       Introduzca la altura del triángulo equilátero: 4
                 *
                ***
               *****
              *******

       Pulse cualquier tecla para continuar...
```

OPCIÓN 3:

Si introduce valores incorrectos:

```
3. DIBUJAR UN ROMBO
****************************************

       Introduzca un número impar y positivo: -4

       El número debe ser impar y positivo.

       Pulse cualquier tecla para continuar...
```

Al presionar cualquier tecla, limpie pantalla y vuelva a pedir la nueva altura hasta que sea correcto:

```
3. DIBUJAR UN ROMBO
****************************************

       Introduzca un número impar y positivo:
```

Ejecución con valores correctos:

```
3. DIBUJAR UN ROMBO
****************************************

       Introduzca un número impar y positivo: 7
                 *
                ***
               *****
              *******
               *****
                ***
                 *

       Pulse cualquier tecla para continuar...
```

3. Normas de realización de la práctica.

- El código fuente del proyecto de programación 2C debe estar organizado con subprogramas

- La entrega de las prácticas que desarrollaremos en la asignatura se realizará a través del Campus virtual. Dicha entrega debe constar de:

  1) Proyectos de programación desarrollados *(IMPORTANTE: deben compilar y ejecutar correctamente en el IDE Codeblocks)*
  2) Memoria completa de la práctica *(.pdf) (ver apartado 4)*

- *Entregad un único archivo comprimido .zip o .rar que contenga 1) y 2)*

4. Memoria a entregar con la práctica 2

La **MEMORIA DE LA PRÁCTICA** debe estar organizada de la siguiente forma:

- ✓ **Portada:** donde se especifique el título de la práctica, la fecha de realización, el/los autor/es de la misma, número de grupo de laboratorio, turno *(Lunes/Martes/Miércoles/Viernes)* y profesor de laboratorio *(Pilar, Francisco o Berta)*. *(¡Imprescindible!)*
- ✓ **Índice** *(apartados de la memoria y números de página)*

- **Proyectos de programación:**

- ✓ **Práctica 2A** => Solo es preciso entregar la tabla que aparece en la página 3 de este enunciado indicando cuál ha sido el error detectado y cómo lo has corregido.

- ✓ **Práctica 2B y 2C:**

  Para cada proyecto debéis incluir en la memoria de la práctica:

  - Análisis del problema *(Entrada, Salida y Proceso a seguir).*
    - Práctica 2B: entregar el análisis del problema completo
    - Práctica 2C: solo es preciso entregar el análisis del problema de las tareas asociadas a cada una de las opciones del menú.
  - Diseño del Algoritmo:
    - Solo es preciso entregar el diseño del algoritmo del proyecto de programación 2B realizado mediante un Diagrama de flujo y también mediante Pseudocódigo.
  - Pruebas de ejecución del programa (*"capturas de pantalla de la ejecución del programa).*

- ✓ No es preciso incluir el código fuente en la memoria de la práctica.

5. Fechas a considerar.

- Publicación de la práctica: Jueves, 17 de octubre de 2024
- Límite para la entrega en el campus virtual: **10 de noviembre de 2024, 22 h.**

** Una vez pasada la fecha de entrega, NO se recogerán más prácticas.
** No se admiten entregas por correo electrónico ni por cualquier otra vía diferente al campus virtual
** La entrega en el campus virtual se cierra automáticamente a las 22 h. del día indicado.

¡No esperéis al último momento para entregar vuestro trabajo!!