Fundamentos de Ordenadores

Depurar programas usando Nemiver

Departamento de Arquitectura de Computadores

Autor: Mario Macias.

Fecha de elaboración: 16/10/2015

Manual básico Nemiver

Nemiver es un sencillo depurador que puede ser ejecutado de manera autónoma (sin necesidad de usar complejos entornos integrados de desarrollo), ofreciendo una interfaz gráfica moderna e intuitiva.

1 Ejecución

Una vez compilado un programa con `gcc` y la opción `-g` para incluir en el ejecutable la información de depurado:

```
gcc ejemplo.c -o ejemplo -g
```

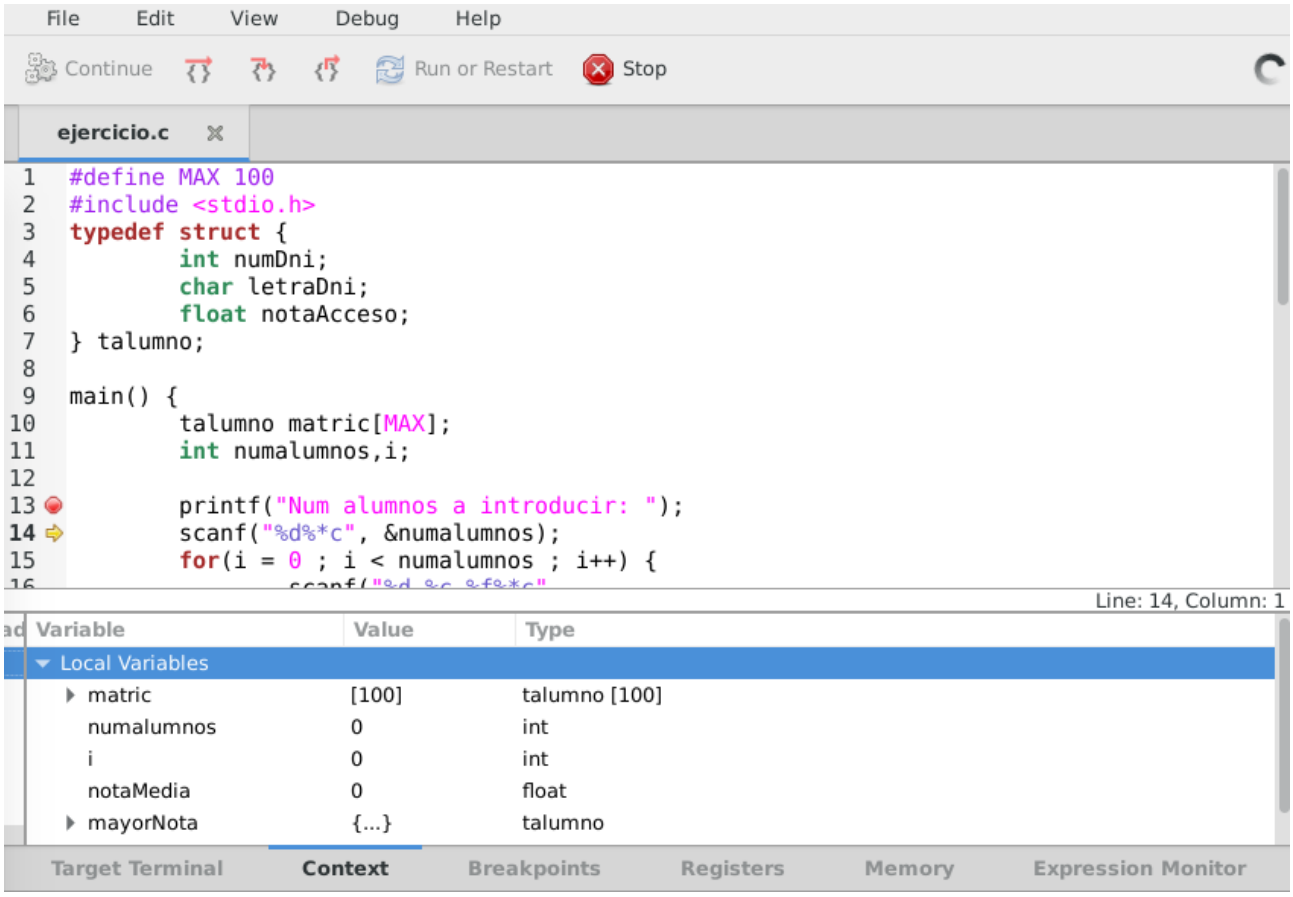
Ejecutar el comando nemiver seguido del fichero ejecutable a depurar:

```
nemiver ejemplo
```

Se abrirá una interfaz como la descrita en la siguiente sección.

2 Interfaz gráfica

La siguiente imagen muestra la interfaz principal de Nemiver:

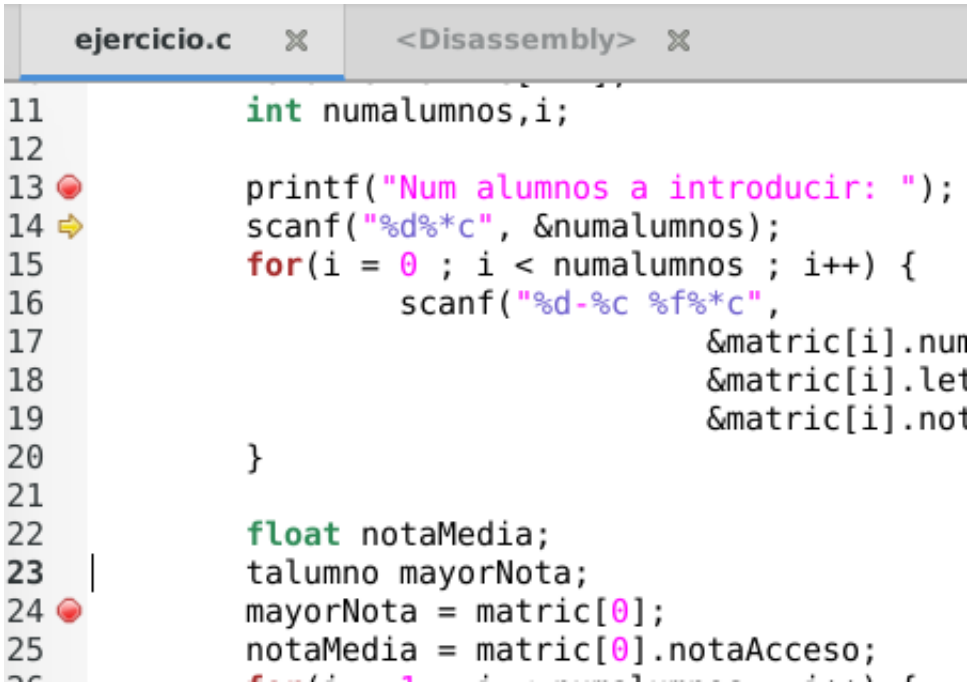


Esta interfaz tiene tres partes diferenciadas:

- **Parte superior:** barra de herramientas, que permite acceder rápidamente a las operaciones de depurado más comunes.
- **Parte media**: sección principal, que muestra los archivos de código fuente, así como los puntos de ruptura (*breakpoints*) y la siguiente instrucción a ejecutarse en modo "paso a paso".
- **Parte inferior**: pestañas auxiliares. En este manual se explicarán las siguientes:
  - *Target terminal*: terminal que mostrará la salida estándar (pantalla de texto) y capturará la entrada estándar (teclado).
  - *Context*: variables (su nombre y sus valores) visibles en el contexto de ejecución del programa, en un momento dado.
  - *Expression monitor*: permitirá evaluar expresiones matemáticas, pudiendo tomar como valores las variables visibles desde el contexto de ejecución.

3 Sección principal

La sección principal muestra los archivos de código abiertos en pestañas, tal y como se muestra a continuación:



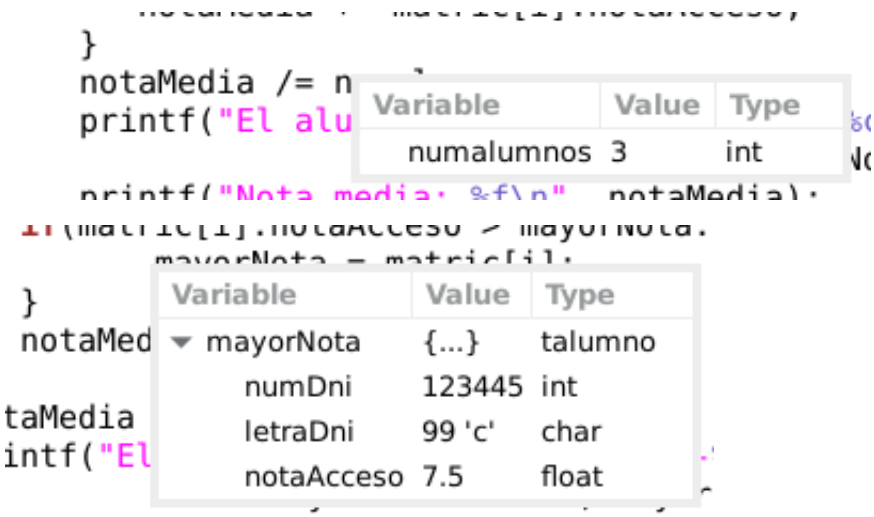
En la parte izquierda, junto a los números de línea, se muestran los puntos de ruptura como círculos rojos. Cuando la ejecución del programa pasa por una línea marcada con un punto de ruptura, la ejecución del programa se interrumpe, pudiendo pasar al modo "paso a paso", en el que las líneas se

ejecutan una a una, según las indicaciones del usuario.

La flecha amarilla muestra, en el modo “paso a paso”, la línea que se va a ejecutar en el siguiente paso.

Para añadir o quitar puntos de ruptura en el código, basta con hacer click al lado del número de línea que se quiere marcar/desmarcar como punto de ruptura.

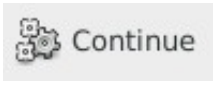
Durante la ejecución “paso a paso”, si se sitúa el puntero del ratón sobre alguna variable y se mantiene unos segundos quietos, aparecerá una ventana emergente que muestra el valor de dicha variable en ese momento dado. La siguiente figura muestra un ejemplo para la variable `numalumnos`:



4 Barra de herramientas

La barra de herramientas está compuesta por los siguientes botones, cuya pulsación inicia las siguientes acciones:

- 

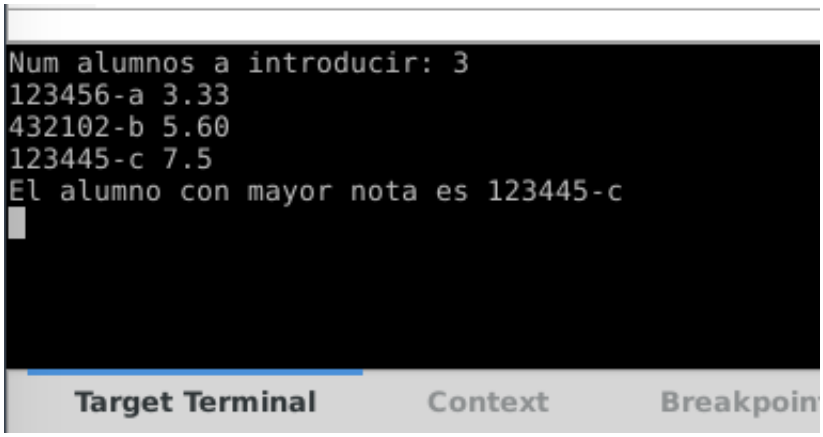
Este botón inicia la ejecución del programa, o la reinicia si ya hay una ejecución en curso. Las instrucciones del programa se ejecutarán de manera ininterrumpida hasta que la ejecución llegue a un punto de ruptura.
- 

Cuando el programa se está ejecutando en modo “paso a paso”, continua la ejecución del programa. Las instrucciones se ejecutarán de manera ininterrumpida hasta que la ejecución llegue a un punto de ruptura.
- 
Cuando la ejecución del programa se está ejecutando en modo “paso a paso”, ejecuta la siguiente sentencia. Si la sentencia a ejecutar contiene una llamada función, dicha función se ejecutará de manera ininterrumpida dentro de ese mismo paso, excepto si el código de la función contiene un punto de ruptura.

-  En el modo “paso a paso”, ejecuta la siguiente sentencia del programa. Si la sentencia a ejecutar contiene una llamada a función, la ejecución del programa continuará, en modo paso a paso, en la primera sentencia de la función.
-  En modo “paso a paso”, si la ejecución está dentro de una función, se ejecutan ininterrumpidamente todas las sentencias restantes de dicha función. La ejecución continuará en modo “paso a paso” a partir de la instrucción que llamaba a la función de la que se ha salido.

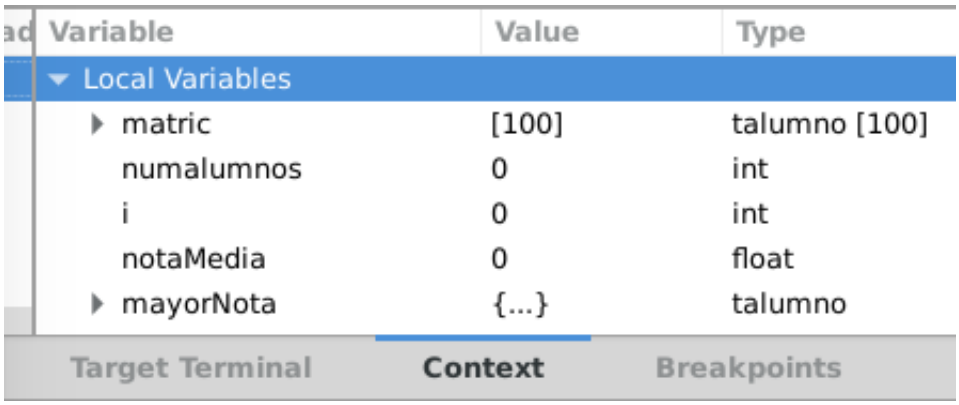
5 Pestaña 'Target terminal'

Muestra un terminal con la entrada y salida estándar del programa, tal y como se muestra a continuación:

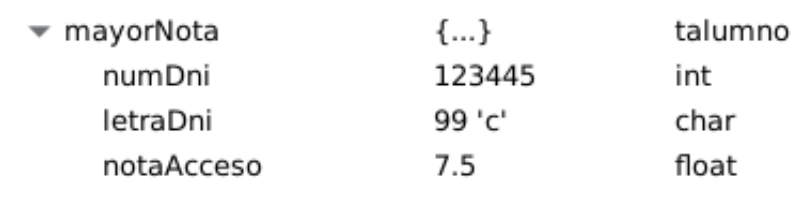


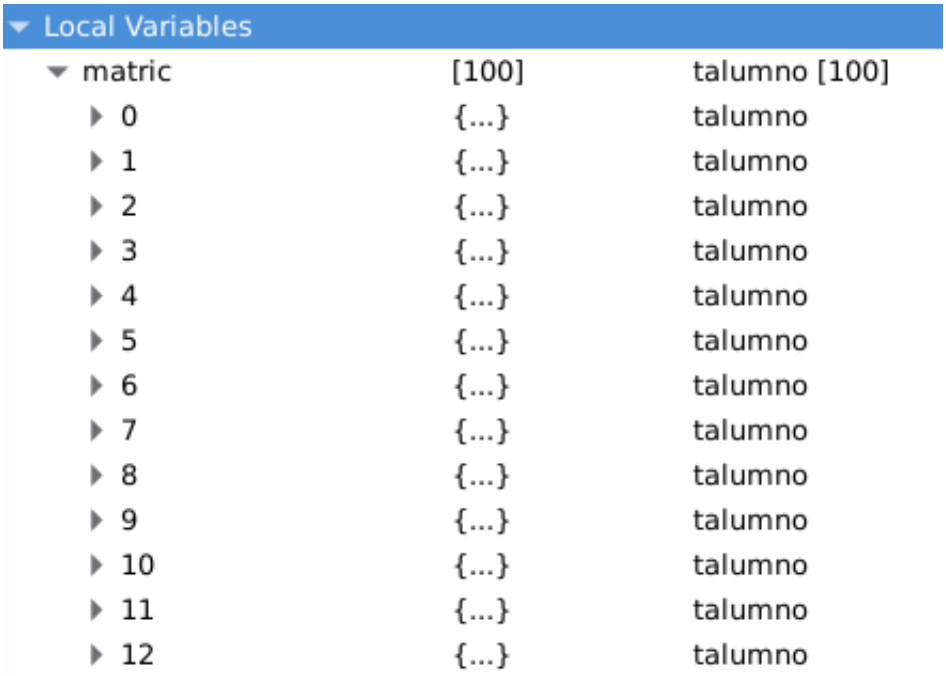
6 Pestaña 'Context'

Durante la ejecución “paso a paso”, muestra el nombre, el valor y el tipo de las variables locales para el contexto de ejecución. A continuación mostramos un ejemplo:

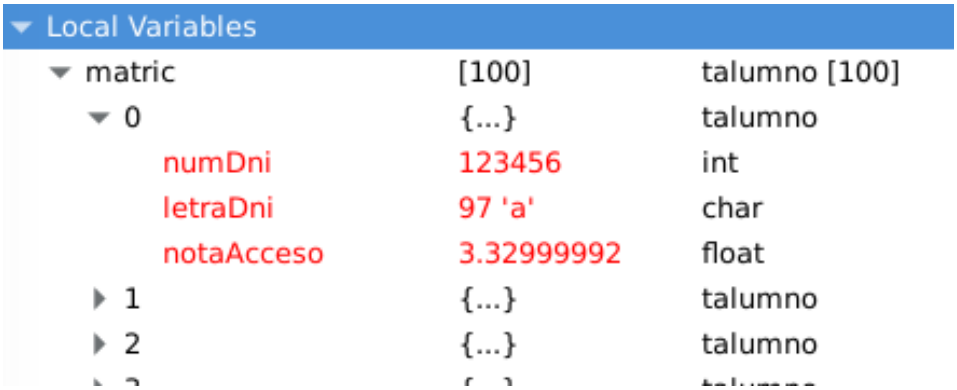


Las variables del tipo *struct*, o del tipo vector o *array* pueden desplegarse haciendo click sobre ellas con el ratón:





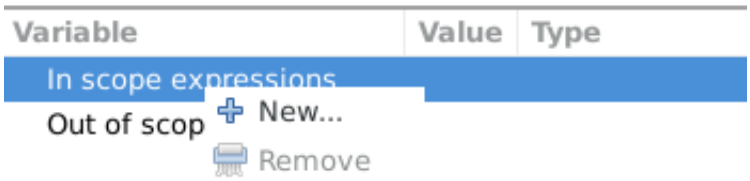
En la imagen de ejemplo anterior, como el vector contiene variables de tipo *struct*, el valor de cada una de éstas puede desplegarse haciendo click sobre los elementos individuales:



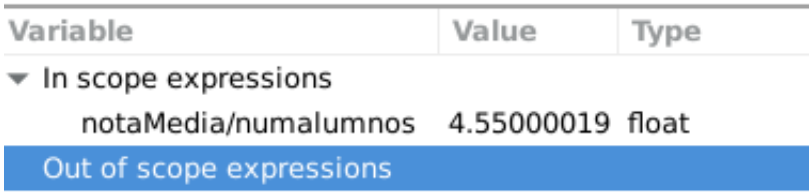
7 Pestaña 'Expression Monitor'

Esta pestaña nos permitirá evaluar variables y expresiones simples, y ver cómo el resultado de éstas pueden cambiar durante la ejecución del programa.

Para añadir una nueva expresión, hacer click izquierdo sobre el texto “In scope expressions” y seleccionar la opción “New...” (ver la siguiente figura).



En la ventana emergente, escribir el nombre de la expresión, y hacer click sobre “Add to monitor”. Esta expresión será visible y se actualizará durante la ejecución del programa, tal y como se muestra a continuación para `numalumnos`:



Para eliminar una expresión, hay que hacer click con el botón izquierdo del ratón sobre la expresión y seleccionar “Remove”.