
Sesión 9

Clases

► Actividades a realizar en casa

Actividad: Resolución de problemas.

Resolved los siguientes ejercicios de la relación de problemas III:

- *Obligatorios:*
 - 7 (Recta)
 - 8 (Progresión geométrica)
 - 10 (Interés)
 - 15 (Empresa. Sólo el diseño)
- *Opcionales:*
 - 9 (Gaussiana)
 - 16 (Empresa. Diseño e implementación)

► Actividades a realizar en las aulas de ordenadores

Inspección de objetos

En esta sesión de prácticas vamos a trabajar sobre la manera en la que Dev C++ presenta la información sobre la estructura de las clases definidas en el programa sobre el que se está trabajando y las facilidades que ofrece para monitorizar la ejecución de programas que incluyen clases.

Usaremos como ejemplo una versión simple de la clase `SegmentoDirigido` que puede encontrar en los apuntes de teoría. En primer lugar, crearemos la carpeta `III_SegmentoDirigido` en `U:\FP` y copiaremos en ella el fichero fuente `III_SegmentoDirigido.cpp` (disponible en decsai).

Emplearemos de nuevo el *explorador de clases*: seleccionar `Ver | Ir al Explorador de Clases` (ó simplemente `Ctrl+F3`). En la figura 11

puede observar que el explorador de clases muestra, para este programa, información acerca de los componentes de la clase `SegmentoDirigido`, definida en el fichero fuente abierto en el editor. También muestra las cabeceras de las funciones definidas en el fichero, como ya conocemos (en este caso, únicamente la función `main`).

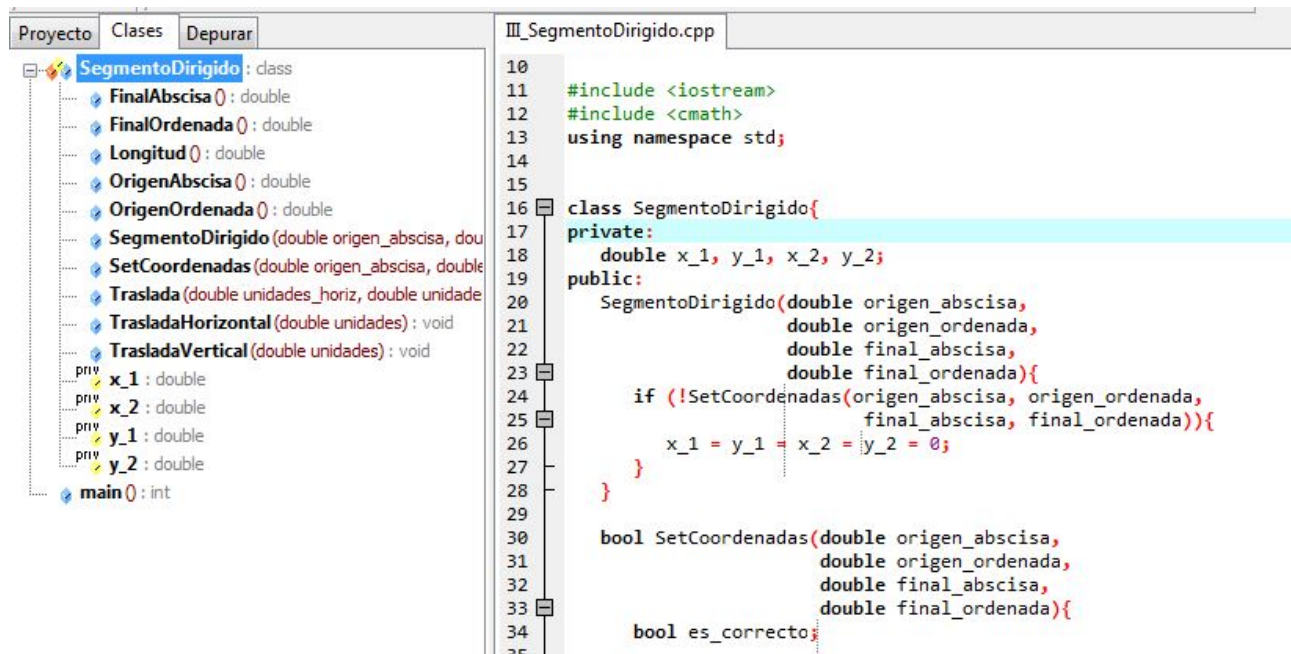


Figura 11: Explorador de clases

En primer lugar observe que el explorador de clases organiza los elementos que muestra de manera jerárquica, por niveles. En este ejemplo encontramos dos elementos en el nivel principal (por orden de aparición en el explorador de clases):

- La clase `SegmentoDirigido`
- La función `main`.

Se etiqueta al elemento `SegmentoDirigido` con la palabra `class`. Observe que se enumeran una serie de elementos dentro de esa clase que gráficamente se muestran asociados a `SegmentoDirigido` mediante líneas.

Dev C++ organiza la presentación de los elementos de la clase enumerando en primer lugar los *métodos* (por orden alfabético) y a continuación los *datos* (también por orden alfabético). Haciendo click sobre el nombre de cualquier componente de la clase el editor muestra el código de la función seleccionada (su implementación o definición), o se accede a la declaración del dato seleccionado. Es una manera rápida de acceder al código de cualquier elemento de una clase.

Dev C++ usa un conjunto de iconos para mostrar los diferentes elementos del programa:



Los objetos, como cualquier otro dato, pueden ser monitorizados durante la depuración de un programa. Cuando se añade un objeto para su inspección se muestran sus campos y sus valores. Por ejemplo, en la figura 12 se muestra el contenido del objeto `un_segmento` después de haber asignado los valores a sus cuatro campos.

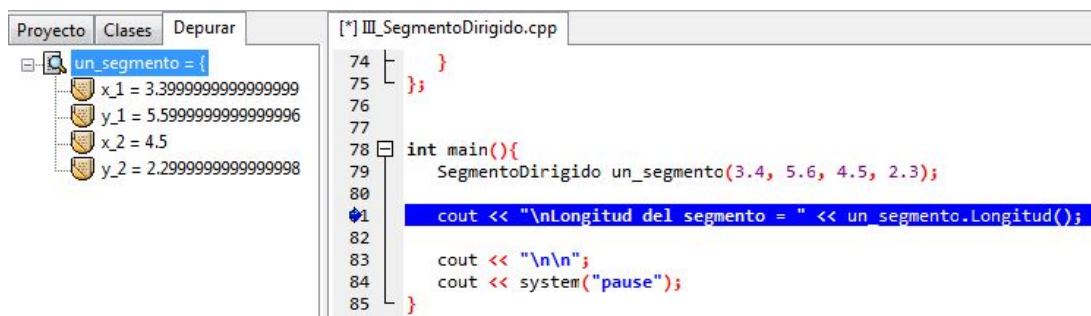


Figura 12: Explorador de clases mostrando el contenido de un objeto `SegmentoDirigido`

Para llegar al estado mostrado en la figura 12:

1. se estableció un punto de ruptura después de crear el objeto,
2. se añadió la variable `un_segmento` a la lista de datos monitorizados (botón derecho sobre `un_segmento` y Añadir watch)
3. se inició la depuración del programa (Ejectuar | Depurar).

Una vez que aparece el objeto `un_segmento` entre los datos monitorizados puede trabajar con él (y con cualquiera de los campos que lo forman) de la misma manera que con cualquier otro dato monitorizado.

Durante la depuración puede *avanzar paso a paso* (F8), y entrar a ejecutar los métodos de la clase línea a línea, tal y como se vio en la sesión de prácticas con funciones. Por ejemplo, establezca un punto de ruptura en la línea

```
un_segmento.TrasladaHorizontal (10);
```

continúe la depuración hasta llegar a esa línea y pulse F8. Cuando se está ejecutando el método `TrasladaHorizontal`, los cambios realizados por éste en los datos miembro no aparecen reflejados en la ventana de inspección del objeto `un_segmento` (para comprobarlo es posible que tenga que mover el ratón sobre el código para que se refresque la

ventana de inspección) Si estamos ejecutando un método y queremos ver cómo cambian los datos miembro, debemos añadir explícitamente una inspección para los datos miembros (ver figura 13).

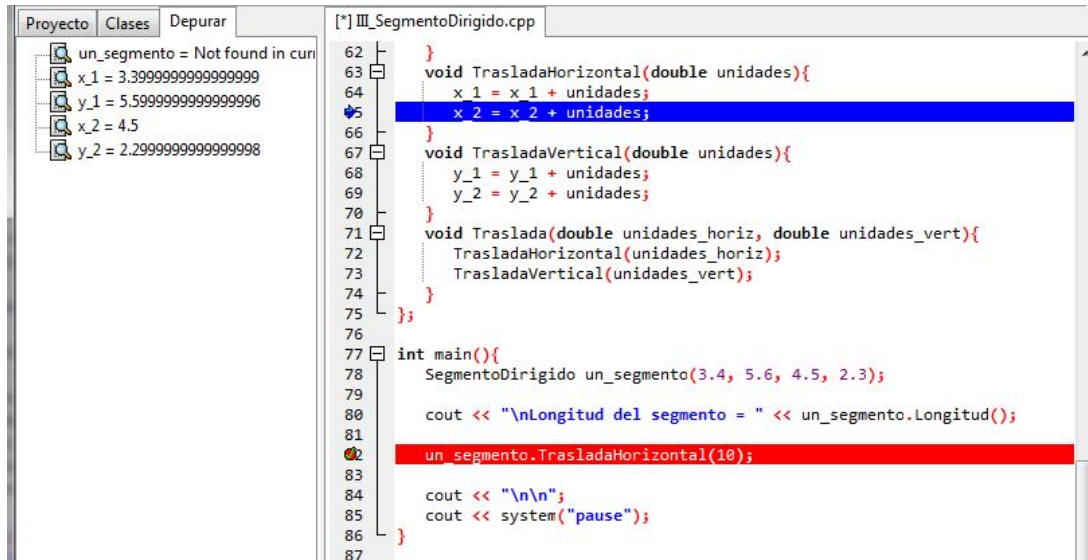


Figura 13: Explorador de clases mostrando los datos miembro durante la depuración

Al salir del método y volver a la función `main` comprobaremos que el objeto `un_segmento` se actualiza y se muestran los valores correctos.