Programación II
Escuela Politécnica Superior, UAM

Práctica 0: Tareas en Makefile, Entorno NetBeans, Depuración de un Proyecto

El objetivo principal de esta práctica es mostrar las herramientas básicas necesarias para entregar, desarrollar y probar el resto de prácticas de la asignatura de la forma más cómoda y profesional. Para ello, se va a demostrar cómo se crean tareas en Makefile (para generar distintos ejecutables a partir de un mismo conjunto de ficheros fuente) y cómo se leen los argumentos de un programa mediante procesamiento de los parámetros de la función *main*.

Esta práctica, además, persigue familiarizarse con el entorno de desarrollo **NetBeans IDE 7.x** para facilitar la programación en dicho entorno, en particular en los siguientes aspectos:

- Creación de proyectos y grupo de proyectos, creación y edición de ficheros fuente y de cabecera.
- Compilación, ejecución y depuración de código.

CREACIÓN DE TAREAS EN FICHERO MAKEFILE

En esta asignatura, vamos a necesitar generar ejecutables distintos a partir de un mismo conjunto de ficheros fuente. Para conseguir esto, la forma más sencilla es haciendo uso de *tareas* en un fichero Makefile, un método alternativo es a través de NetBeans y su posibilidad de crear grupos de proyectos o configuraciones dentro de un proyecto.

Para este ejemplo, supongamos que el tipo de dato PILA (implementado en pila.c) utiliza el tipo genérico ELEPILA (definido en elepila.h), que se ha implementado de dos formas distintas: ELEPILA_A (elepila_a.c) y ELEPILA_B (elepila_b.c). Si se intentan compilar todos estos ficheros (*gcc *.c *.h*), el compilador devolverá un error indicando que algún método se ha implementado más de una vez:

> *"multiple definition of `xxxx'"*

La forma más cómoda de arreglar esto, y una buena forma de trabajar cuando se tiene proyectos con varios ficheros, es mediante la creación de tareas o reglas en ficheros makefile, de manera que se generen distintos ejecutables usando distintos ficheros fuente. En nuestro ejemplo, podríamos tener en el makefile lo siguiente (suponiendo ej_a.c y ej_b.c contienen un *main* para cada variante):

```
############################################################
CC=gcc
CFLAGS= -g -Wall -pedantic -ansi
EJS = ej_a ej_b
############################################################
OBJECTSEJA = ej_a.o pila.o elepila_a.o a.o
OBJECTSEJB = ej_b.o pila.o elepila_b.o b.o
############################################################
all: $(EJS) clear

ej_a: $(OBJECTSEJA)
	$(CC) $(CFLAGS) -o ej_a $(OBJECTSEJA)

ej_b: $(OBJECTSEJB)
	$(CC) $(CFLAGS) -o ej_b $(OBJECTSEJB)
### ... otras tareas necesarias para compilar/linkar el resto de ficheros
clear:
	rm -rf *.o
```

LECTURA DE ARGUMENTOS EN LA FUNCIÓN MAIN

A continuación se explica cómo leer argumentos mediante procesamiento de parámetros de la función *main*.

Los argumentos pasados **por línea de comandos** (o usando el menú **Properties > Run > Run Command** de NetBeans) se reciben a través de la función main en las siguientes variables:

- **argc**: es un entero que contiene el número de argumentos que se han introducido
- **argv**: es un array de punteros a caracteres (array de cadenas)

Hay que tener en cuenta que el nombre del programa se almacena en *argv[0]*, por lo que *argc* siempre será mayor que 1.

Para poder hacer uso de estos argumentos, la función *main* debe definir las variables antes mencionadas. Un ejemplo donde se muestran todos los argumentos recibidos sería:

```
#include<stdio.h>

int main(int argc, char * argv[]) {
		int i;

		printf("Nombre del programa invocado: %s\n", argv[0]);
		printf("\nArgumentos:\n");
		for(i=1; i<argc; i++)
			printf("El argumento %d es %s\n", i, argv[i]);

		return 0;
}
```

ORGANIZACIÓN DE PROYECTOS, GRUPO DE PROYECTOS Y FICHEROS EN NETBEANS 7.x

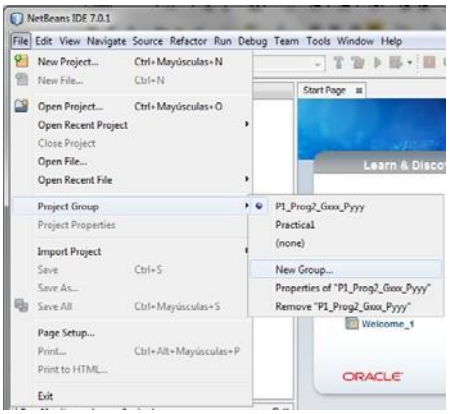
A continuación se explica el método a seguir para crear los entregables de las prácticas de esta asignatura. La idea básica es la siguiente: cada práctica constará de varios ejercicios a realizar. En la terminología de NetBeans, una práctica se podrá corresponder con un **"Grupo de Proyectos"**, mientras que cada uno de los ejercicios de la misma se corresponderá con su respectivo "**proyecto"**, o bien con un **"proyecto"** y cada ejercicio con una **"configuración"** diferente. Los pasos a seguir en cada caso son los siguientes:

Opción Grupo de Proyectos

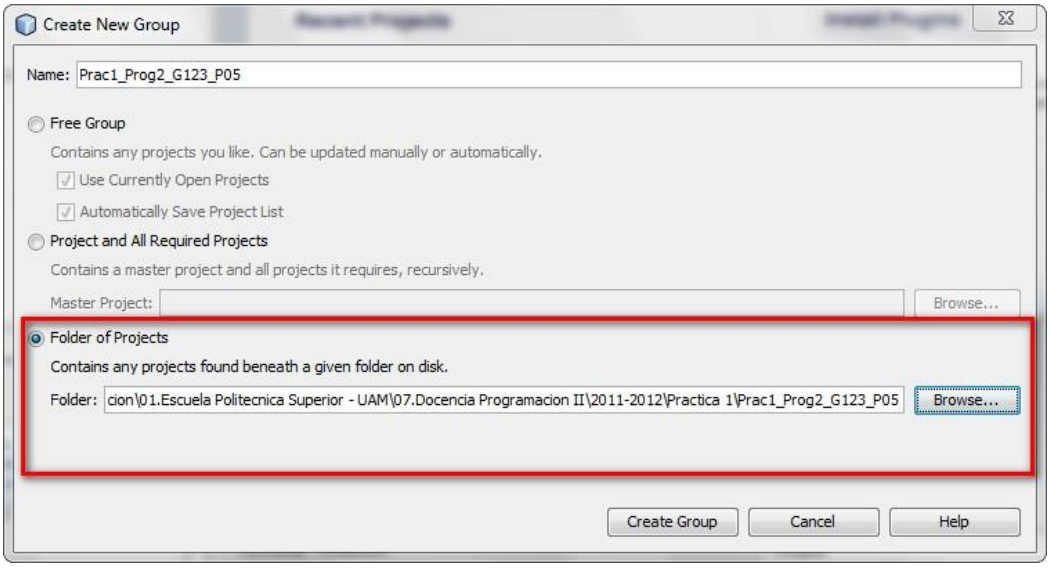
1) Creación de un Grupo de Proyectos:

a) Crear una carpeta en el entorno de ficheros (desde el explorador o desde NetBeans) para albergar todos los proyectos relacionados con la práctica en cuestión. Esta carpeta se denominará de la siguiente manera: **"Px_Prog2_Gy_Pz"**, donde '**x**' será el número de la práctica, '**y**' el grupo de prácticas y '**z**' el número de pareja. Por ejemplo, para crear la primera práctica, la pareja 5 del grupo 2123 deberá crear la carpeta: **"P1_Prog2_G2123_P05":**

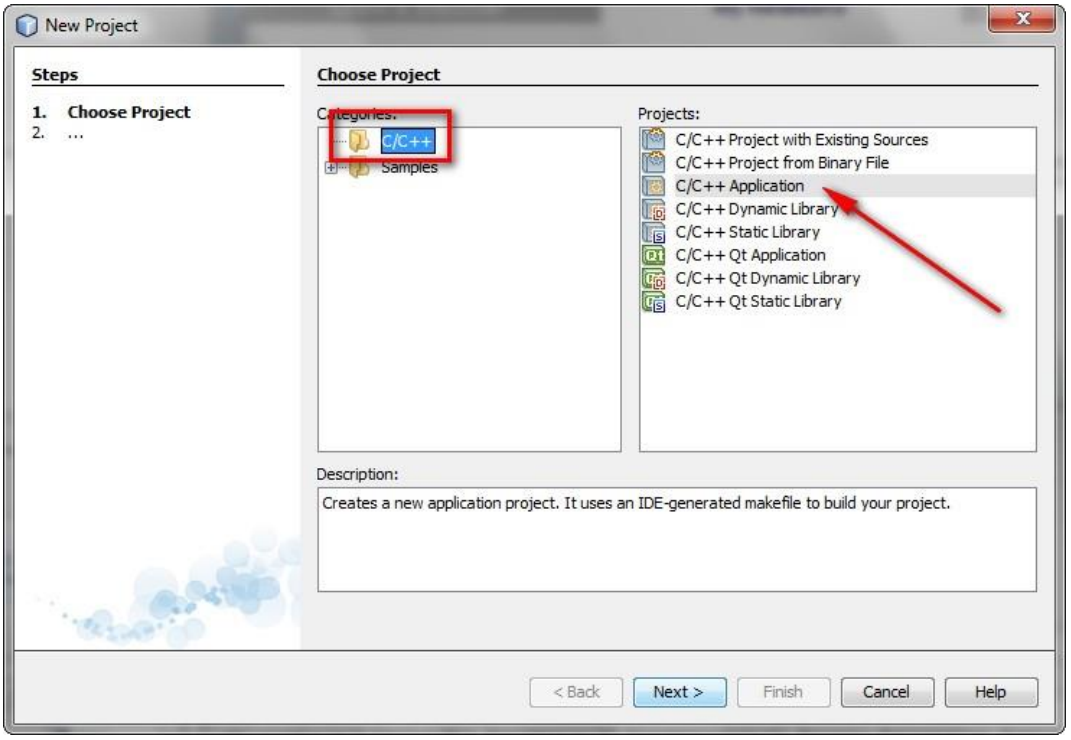
b) Abrir el entorno de desarrollo NetBeans 7.x.
c) Elegir "*File > Project Group > New Group…*".



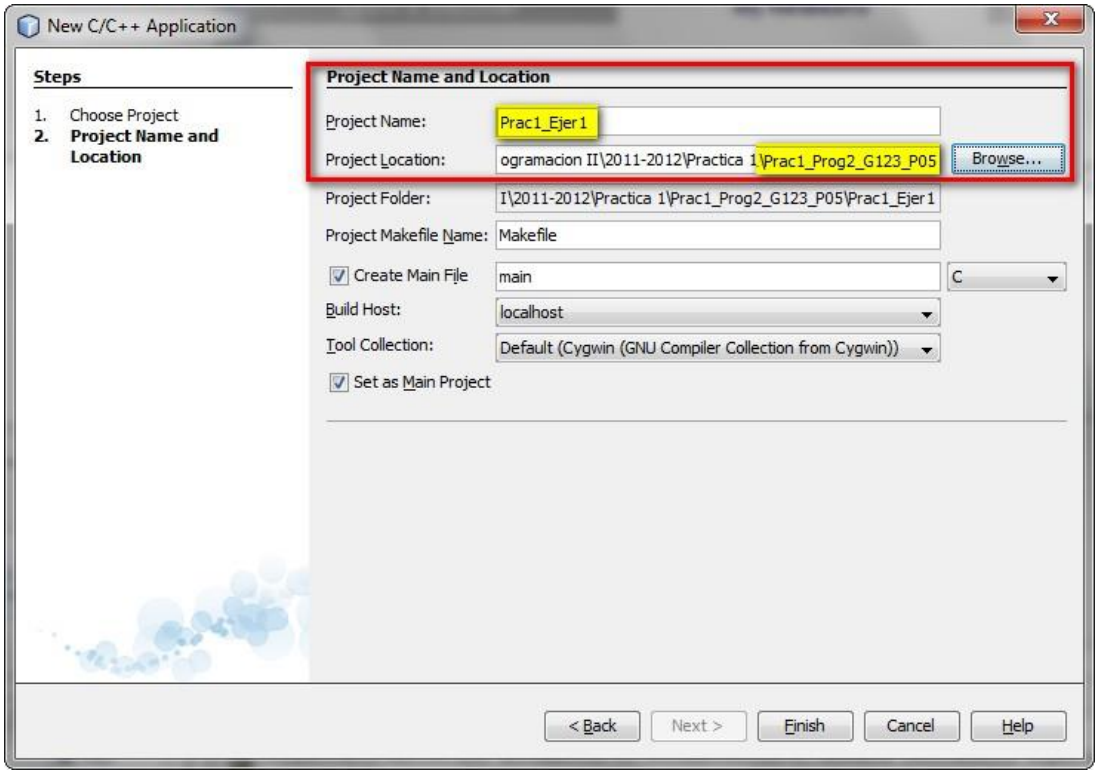
d) Seleccionar *"Folder of Project"*. Con el botón *"Browse…"*, navegar a la carpeta anteriormente creada y pulsar "Create Group":



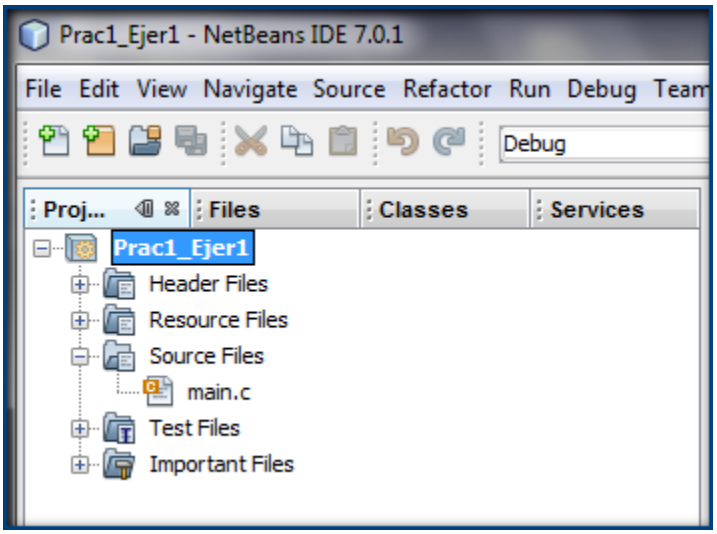
e) Ya se está en disposición de crear un nuevo Proyecto. Pulsar menú *"File > New Project…"*. Aparecerá la siguiente ventana. Seleccionar *"C/C++ Application"*.



f) Pulsar el botón *"Next >"*. El campo *"Project Name"* se completará usando la siguiente nomenclatura: **"Px_Ey"**, donde 'x' es el número de la práctica e 'y' es el número de ejercicio de la práctica. Para el ejercicio 2 de la práctica 1, el nombre del proyecto será: "P1_E2". En el campo *"Project Location"*, seleccionar la carpeta que contiene el Grupo de Proyectos de la Práctica:

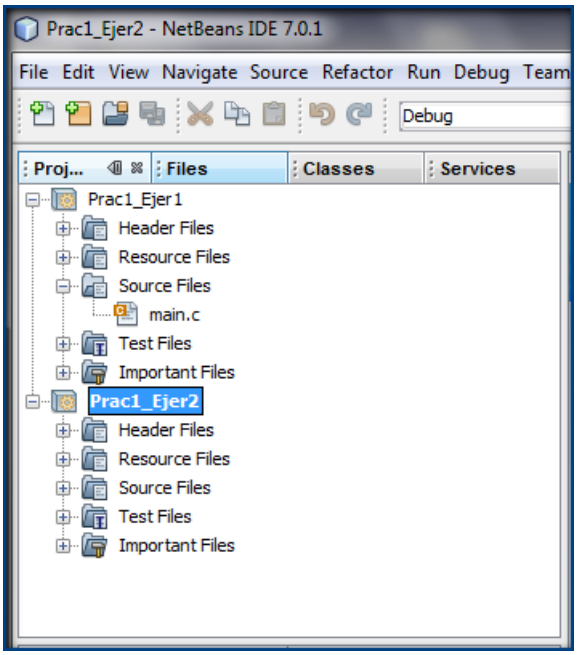


g) Una vez pulsado el botón *"Finish"*, se creará el proyecto en el entorno de desarrollo. Cada proyecto se representa como una carpeta con ficheros dentro (.c, .h, etc.):

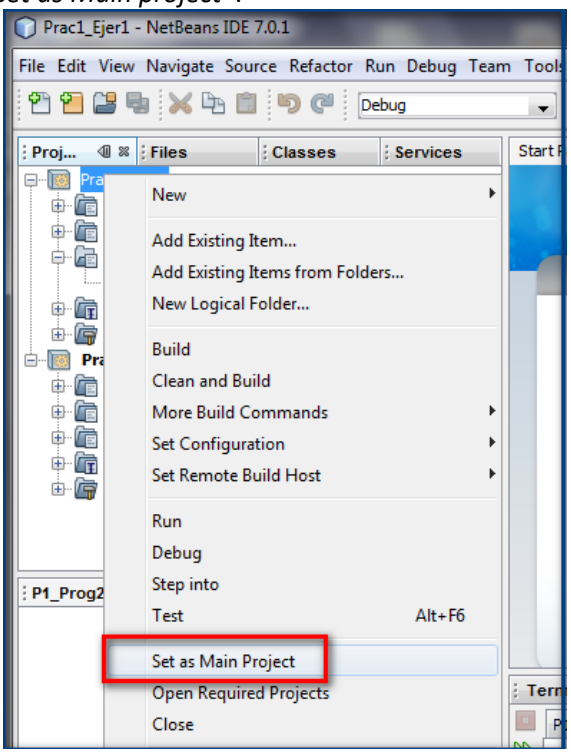


2) Para crear proyectos adicionales (cada nuevo ejercicio de la práctica será un nuevo proyecto), se deberán seguir los siguientes pasos:

a) Elegir con el ratón *"File > New Project…"*
b) Volver a elegir *"C/C++ Application"*.
c) Si ahora queremos implementar el ejercicio 3 de la practica1, crearemos el siguiente nuevo proyecto: "P1_E3"

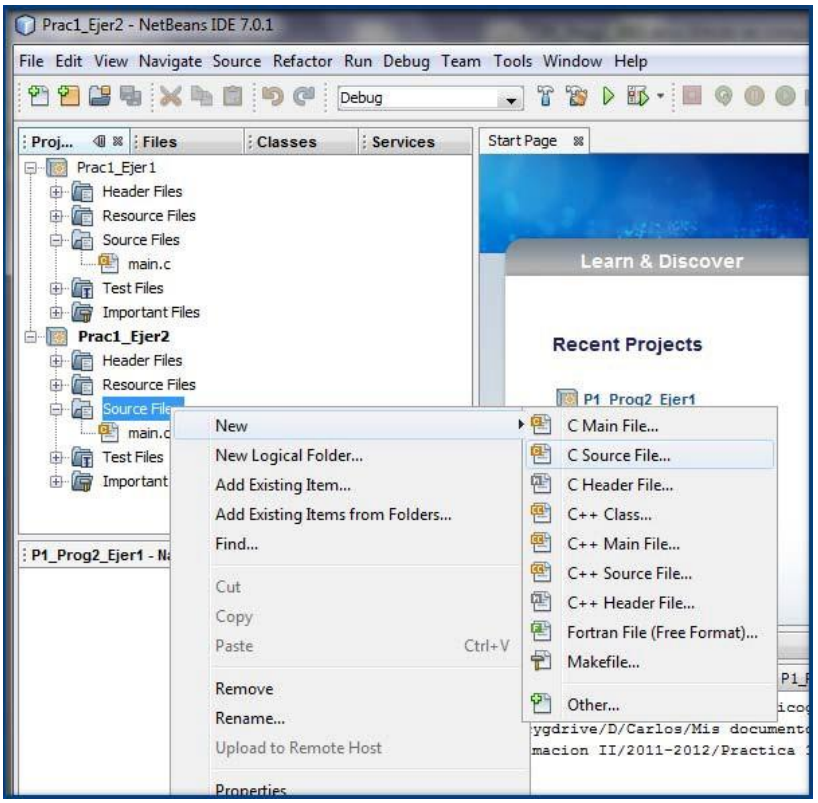


d) Si ahora se deseara volver al Ejercicio 1 para compilar, ejecutar, etc. (es decir, como "Proyecto activo"), colocar el ratón sobre el proyecto "P1_E1" y con el botón derecho seleccionar *"Set as Main project"*:



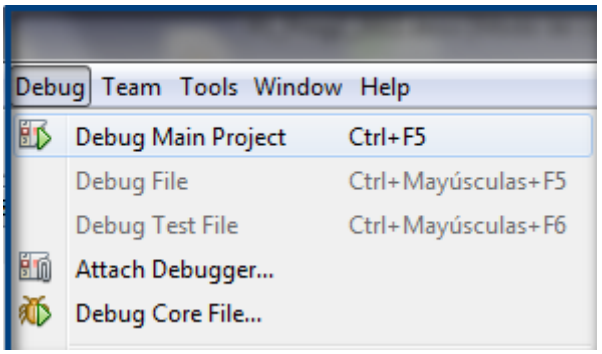
3) Creación de archivos fuente (*.c*) y de encabezado (*.h*):

a) Elegir con el ratón el ejercicio deseado en el "*Explorador de Proyectos*".
b) Hacer click con el botón derecho del ratón sobre el proyecto activo y elegir "*New > Source File…*" o bien *"New > Header File…"*, tal y como se puede observar en la siguiente figura:



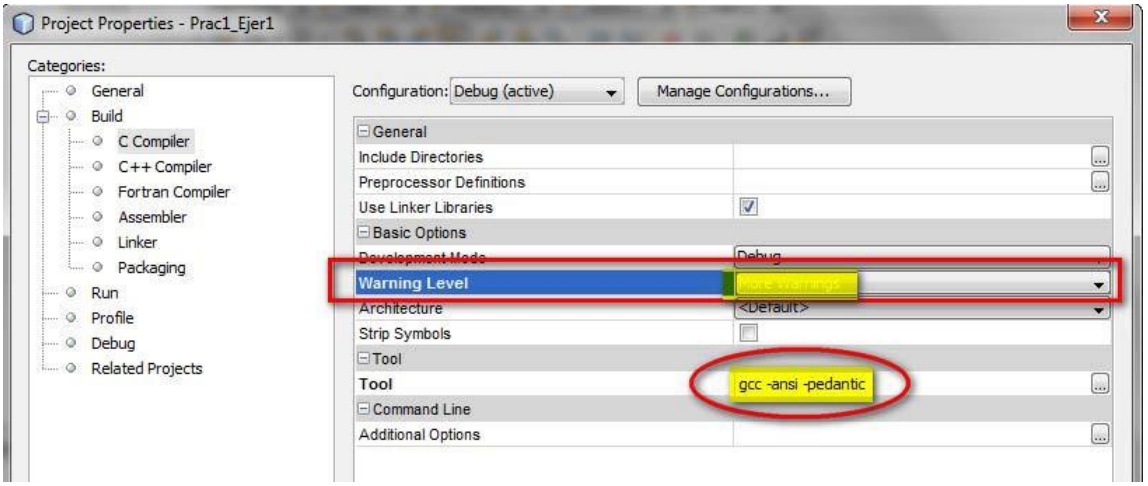
4) Compilación y ejecución de los programas:

a) En todo momento sólo hay un proyecto activo para un determinado "Grupo de Proyectos". En el "*Explorador de proyectos*" se puede distinguir porque su nombre estará escrito en negrita.
b) Si se elige en el menú superior la opción "*Debug*", se puede compilar, o bien seleccionando la opción *"Debug Main Project"* o bien con la combinación de teclas "Ctrl+F5".



c) Para ejecutar el programa desde el menú superior, se elegirá la opción *"Run > Run Main Project"* o bien pulsamos la tecla "F6".

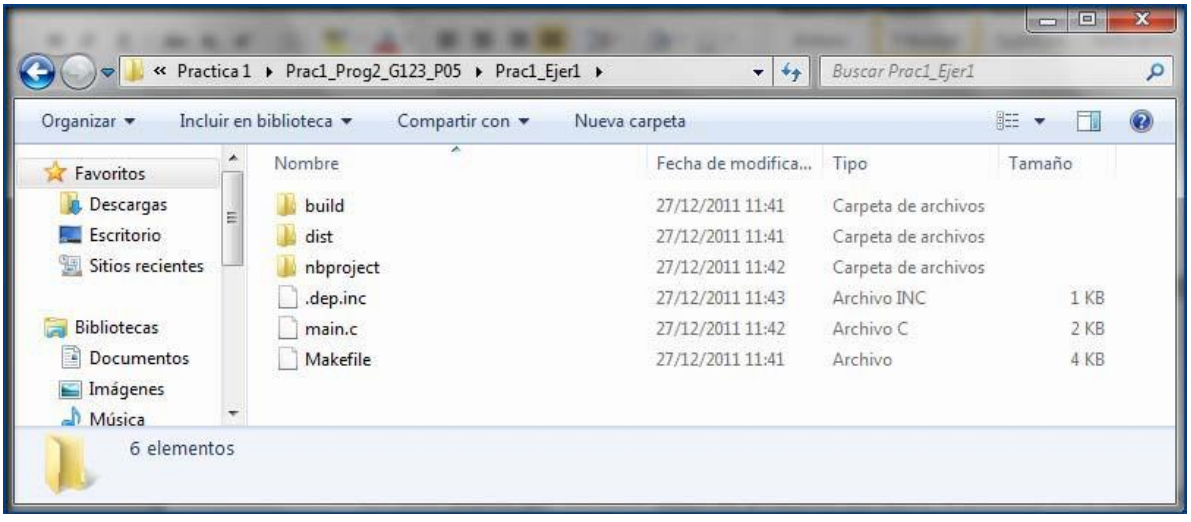
IMPORTANTE: Para que el compilador avise de todos los errores y *warnings*, se establecerán los valores "**More Warnings**" y en el campo *"Tool"* los siguientes parámetros de compilación: "**gcc -ansi –pedantic**".



Si se han seguido todos estos pasos, al final la organización de ficheros es la siguiente:

Una carpeta que contiene todos los proyectos (es decir, la práctica) en la ubicación elegida en el paso 1a y con el nombre "Px_Prog2_Gy_Pz", que contendrá lo siguiente:

- Una carpeta por cada proyecto (es decir, por cada ejercicio), con el nombre "*Px_Ey*" (ver paso 2c). Esta carpeta contendrá, entre otros elementos, los ficheros fuente (*.c*) y de encabezado (*.h*) que se hayan añadido según el paso 3.

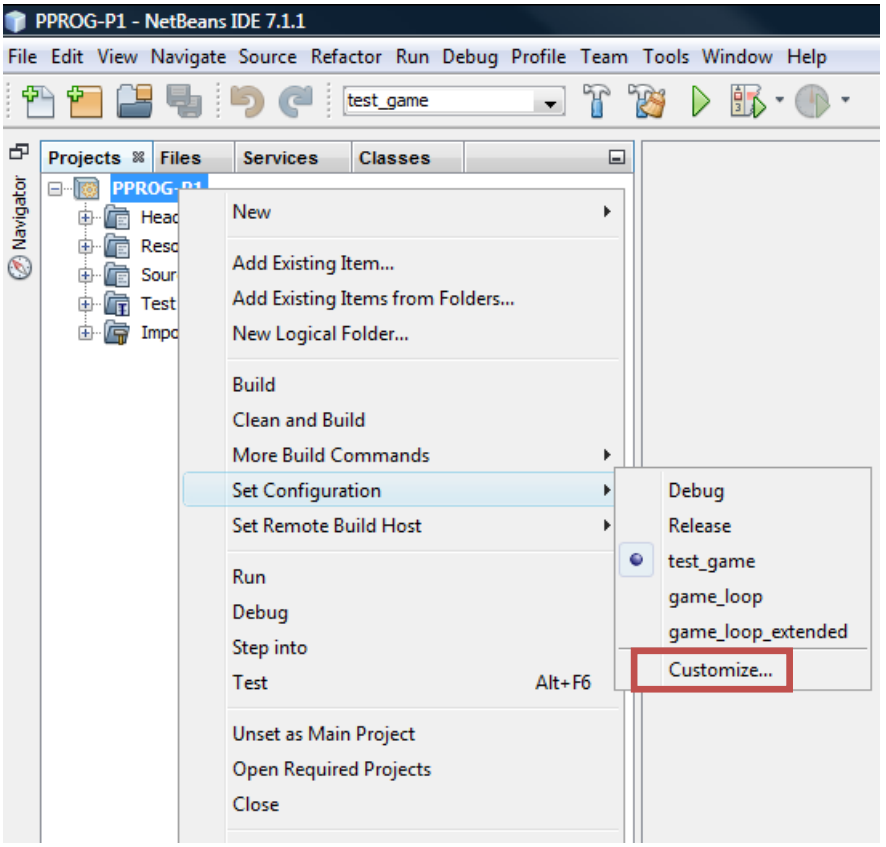


Opción Proyectos con varias configuraciones

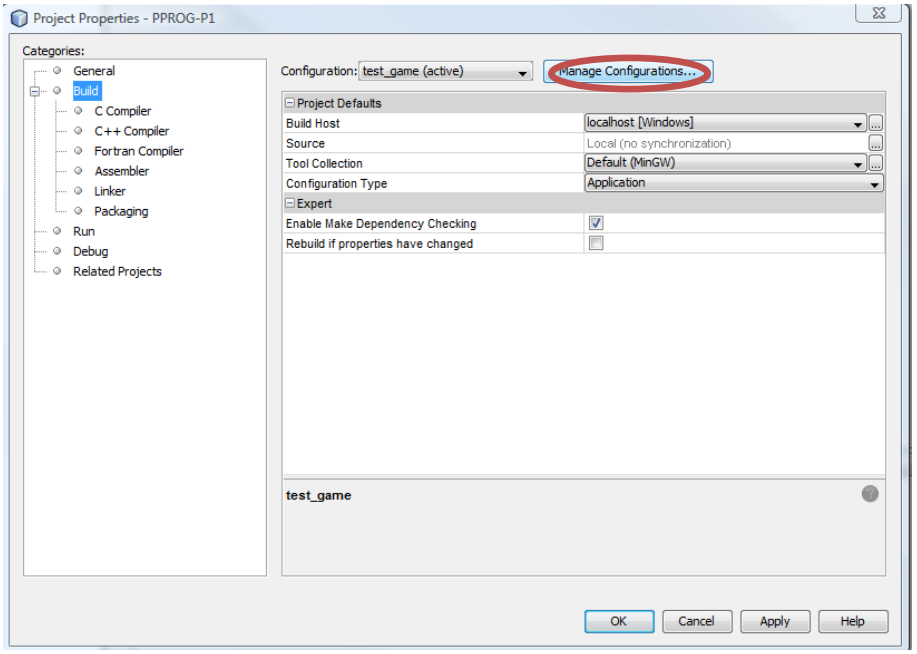
1) Creación de un proyecto (este es el único proyecto que se usará durante el transcurso de la práctica):
   a) Elegir con el ratón "File > New Project…"
   b) Elegir "C/C++ Application".
   c) Si ahora queremos implementar, por ejemplo, la práctica 1 y somos la pareja 5 del grupo 2123, crearemos el siguiente nuevo proyecto: "**P1_Prog2_G2123_P05**".
   d) Añadir archivos fuente y de cabecera como se explicó anteriormente.

2) Creación de una configuración (una para cada ejercicio de la práctica):

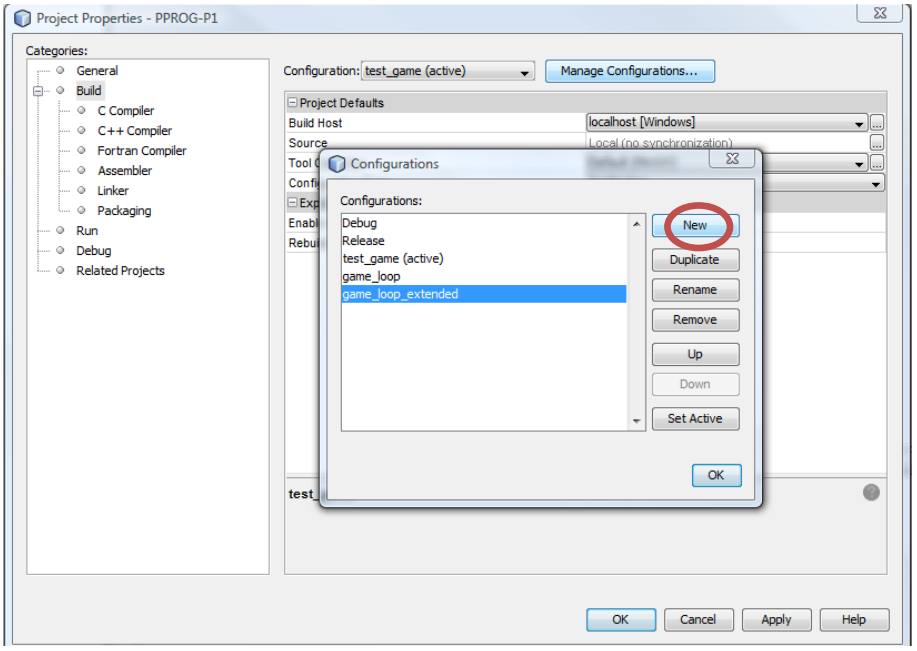
   a) Colocar el ratón sobre el proyecto y con el botón derecho seleccionar *"Set configuration"* y en la lista que aparece *"Customize"*, como en la imagen:



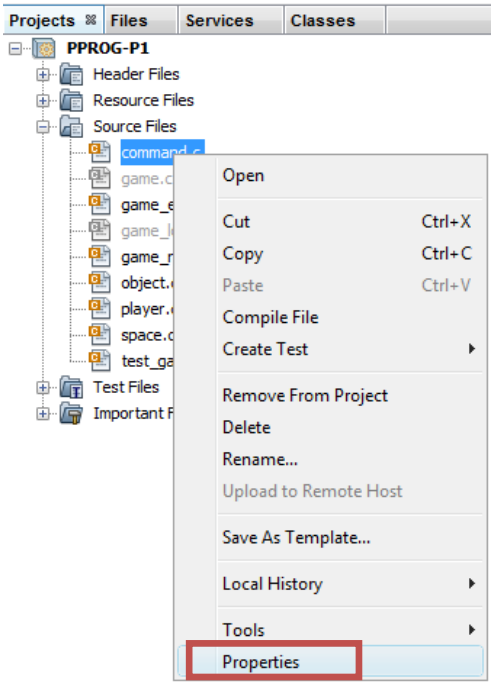
b) Pulsar sobre *"Manage Configurations"* en la ventana que se acaba de abrir:



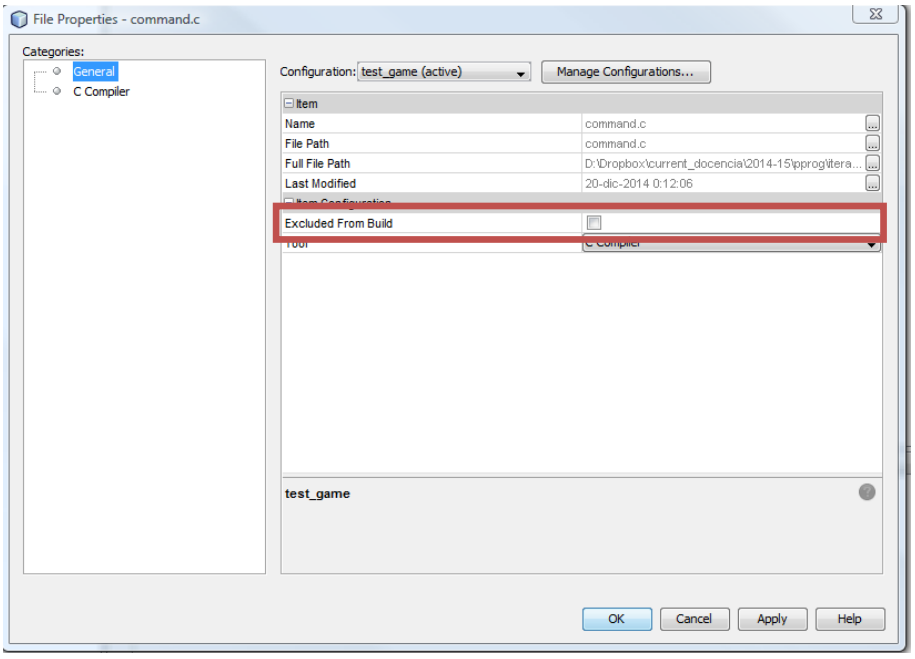
c) Pulsar sobre *"New"* para crear una configuración desde cero. Si la configuración que se quiere crear es parecida a alguna ya existente, se puede duplicar esa misma y luego cambiarle el nombre. Siguiendo con el ejemplo anterior, si estamos implementando el ejercicio 2 de la práctica 1, esta configuración se llamaría "**P1_E2**".



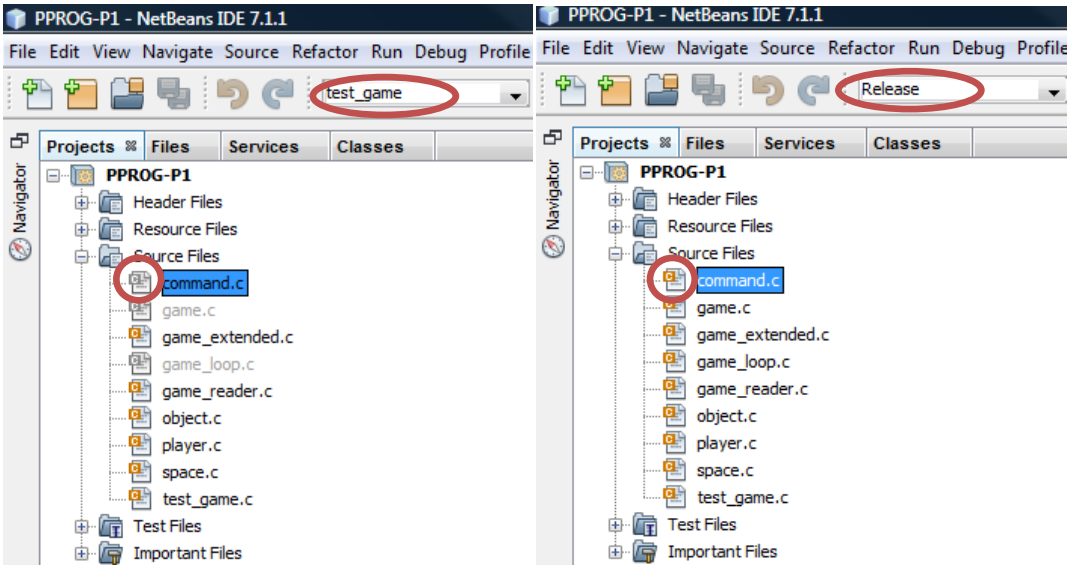
d) Una vez que se ha seleccionado una configuración, hay que descartar los ficheros que pertenecen en ese momento al proyecto pero que no tienen que ver con dicho ejercicio. Esto se hace seleccionando el fichero (tanto .h como .c) en cuestión, pulsando botón derecho sobre él y seleccionando *"Properties"*:



e) En ese menú, seleccionad *"Excluded From Build"*:



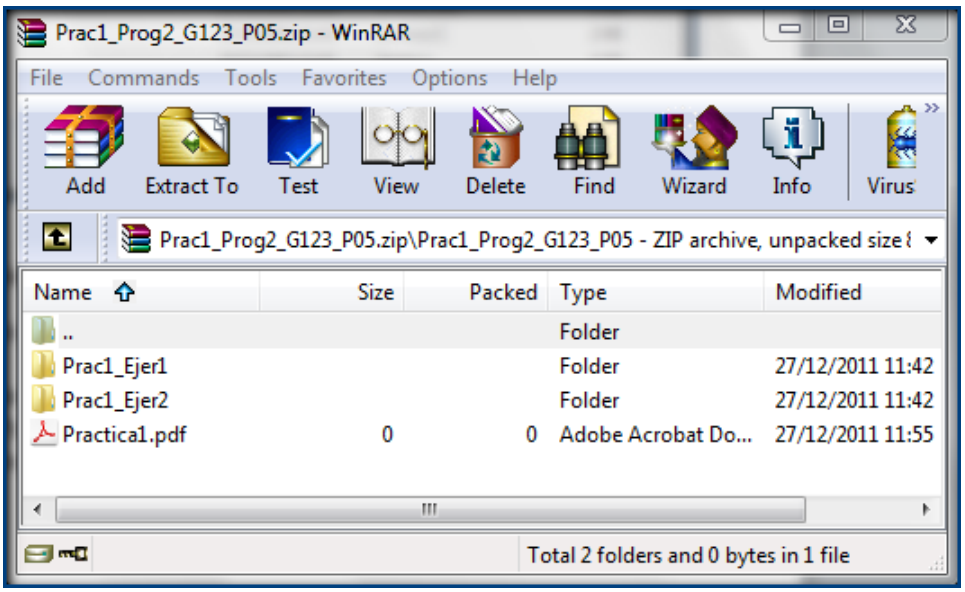
Como resultado, ese fichero aparecerá en gris cuando esa configuración esté seleccionada:



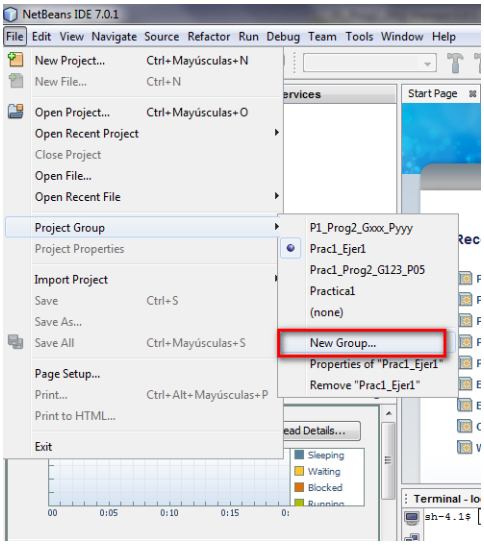
ENTREGA DE LA PRÁCTICA

Para entregar la práctica, si se han hecho bien todos los pasos anteriores:

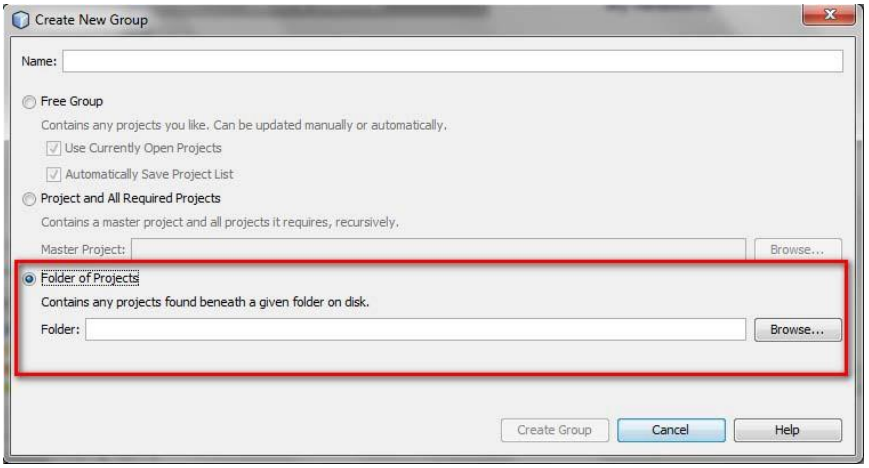
1. Limpiar el proyecto para eliminar los ficheros objeto y los ejecutables. Hay dos formas de hacerlo: eliminar las subcarpetas build y debug del directorio de cada proyecto. O bien, desde el entorno de desarrollo, pulsar con el botón derecho del ratón sobre la carpeta del proyecto y seleccionar **More Build Commands > Clean Project**.
2. Añadir, en el explorador, **a la carpeta de la práctica** (la que contiene una subcarpeta por ejercicio en la opción de grupo de proyectos) el fichero con la memoria (formatos *.pdf*, *.doc* ó *.docx*).
3. Comprimir **dicha carpeta** en un archivo *comprimido (.zip, .rar o .tgz*) de nombre **Px_Prog2_Gx_Py.zip** o **Px_Prog2_Gx_Py.rar**
4. Entregar en Moodle dicho archivo comprimido en el plazo estipulado.



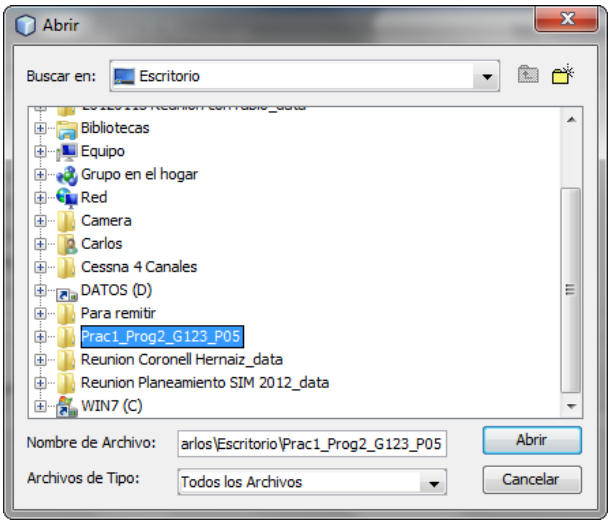
Si se ha usado la opción de grupo de proyectos y se quiere abrir desde otro ordenador la práctica, bastará con abrir el proyecto guardado ("File -> Open Project"); en otro caso, descomprimir el fichero anteriormente creado y desde el entorno de NetBeans seleccionar "Project Group -> New Group":



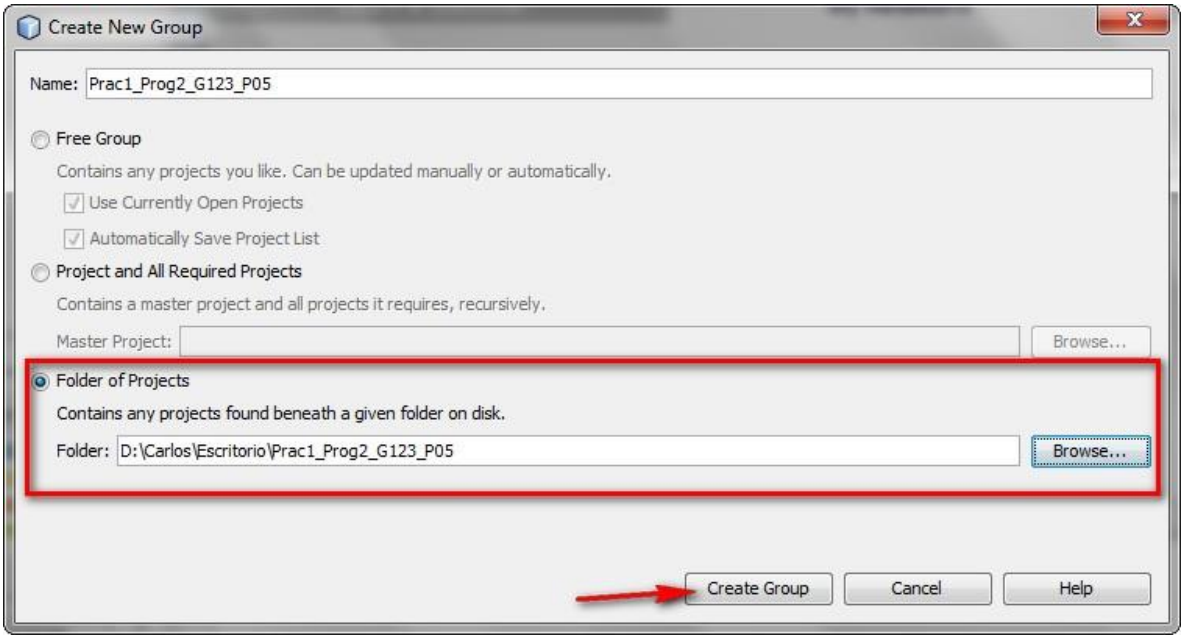
Seleccinar "Folder of Projects":



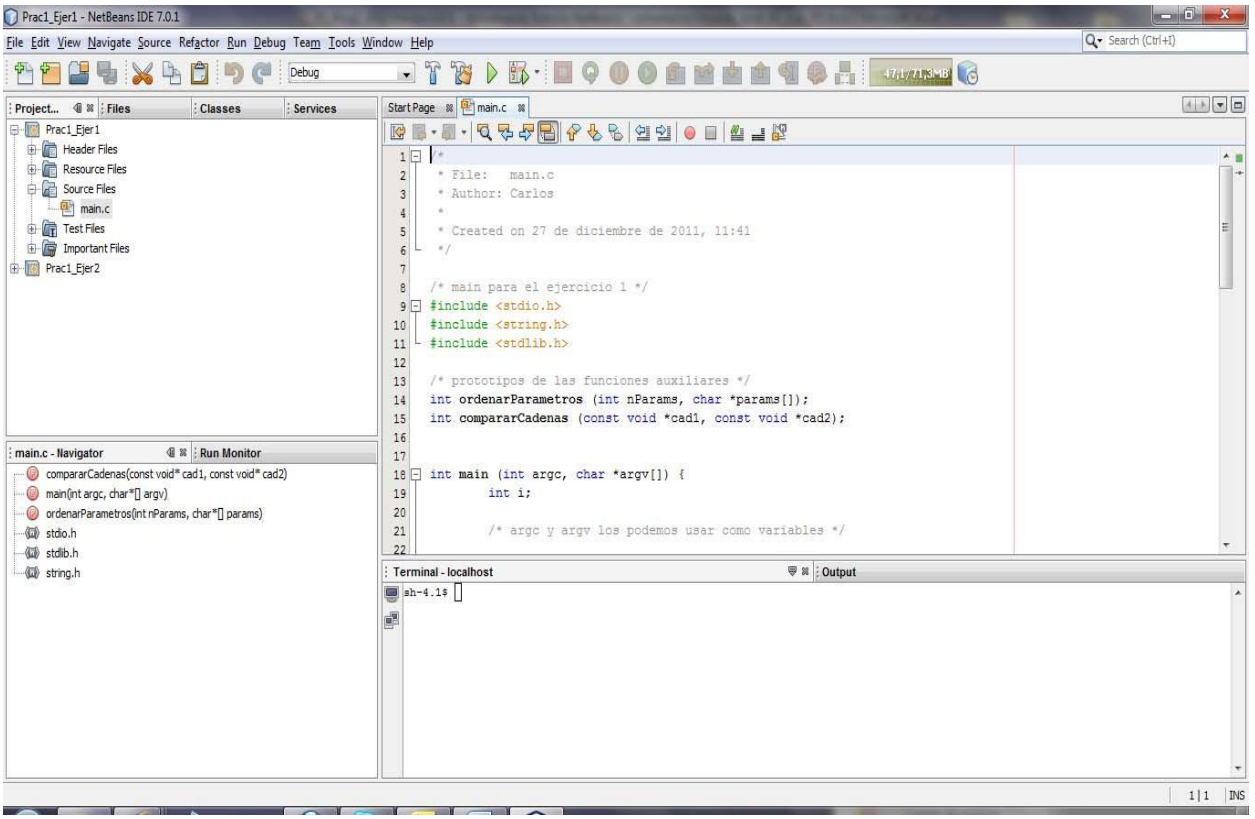
Abrir el explorador y seleccionar la carpeta de la práctica:



Seleccionar “Create Group”:



Y en el entorno NetBeans se abrirá la práctica complete, con todos los proyectos creados hasta el momento:

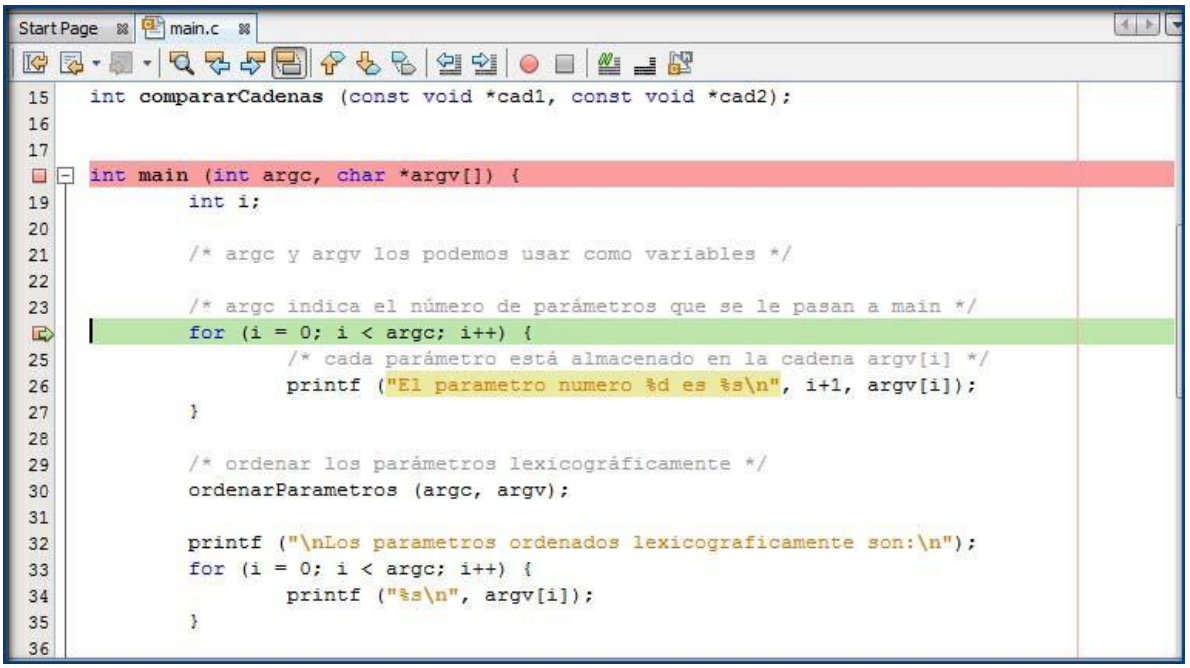


DEPURACIÓN DE UN PROYECTO

NetBeans cuenta con un buen entorno de depuración, el cual resulta muy útil cuando se quiere comprobar por qué un programa no funciona como debería. Para ello, la idea básica es controlar la ejecución del programa mediante puntos en los que ésta se detiene (*breakpoints*) y explotar la posibilidad de ir ejecutando el programa línea a línea, mientras se monitoriza el valor de las variables que esté usando el programa en el punto que actualmente se está ejecutando.

Para poner un *breakpoint* en una determinada línea de código, la opción más directa es hacer click en el margen gris previo a esa línea. Para quitar el *breakpoint,* se volvería a hacer click en el margen. Los *breakpoints* se mantienen como información del proyecto, es decir, que, si se cierra el proyecto y se vuelve a abrir, éstos permanecerán donde se hubieran puesto.

Para iniciar el depurador, hay que elegir en la barra superior "*Debug*" y luego "Debug Main Project" (alternativamente se puede pulsar Ctrl+F5). Una flecha verde indicará en todo momento la línea por la que estemos depurando.



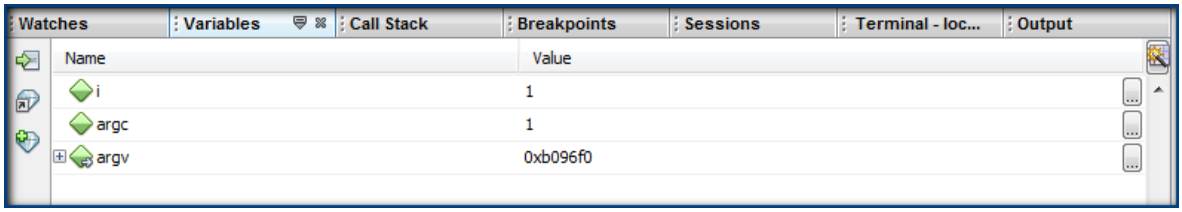
Con la ejecución detenida, existen las siguientes opciones en el menú Debug para continuar la ejecución, bien con la barra de depuración:

O bien con los comandos mediante la pulsación de teclas:

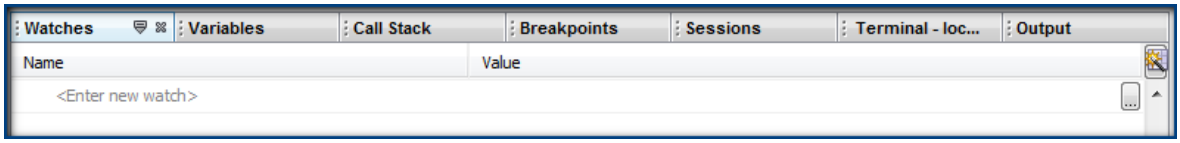
- **"*Continuar*" (F5): la ejecución continúa hasta el siguiente *breakpoint* (o hasta el final si no hay).**
- **"*Paso a paso por procedimientos*" (F8): se ejecuta sólo la línea actual. Si esta línea contiene una llamada a una función, ésta se ejecuta en su totalidad.**
- **"*Paso a paso por instrucciones*" (F7): se ejecuta sólo la línea actual. Si esta línea contiene una llamada a una función, la ejecución pasa a la primera línea de esa función.**

- **"*Paso a paso para salir*" (Ctrl+F7): la ejecución continúa hasta que la función actual termina, deteniéndose tras la línea que llamara a dicha función (o hasta el final si la función es *main*).**

Mientras la ejecución está detenida, en la parte inferior aparece una ventana con los valores que tienen en ese momento las variables del programa.



También podemos inspeccionar variables y expresiones definidas por el usuario. Se hará click en la columna denominada "Watches":



Y seleccionaremos manualmente el valor de una variable. Imaginemos que queremos averiguar la dirección de memoria de la variable entera "i":