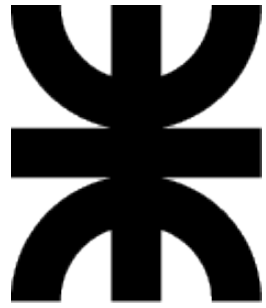
Informática II
Programación en C bajo GNU/Linux

Gonzalo F. Perez Paina



Universidad Tecnológica Nacional
Facultad Regional Córdoba
UTN-FRC

– 2019 –

Programar en lenguaje C

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola␣mundo.\n");
  return 0;
}
```

Programar en lenguaje C

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola␣mundo.\n");
  return 0;
}
```

- ¿Qué representa el texto de arriba? 

Programar en lenguaje C

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

- ¿Qué representa el texto de arriba?
- ¿Qué hay que hacer para obtener los sig. en la terminal?

```
Hola mundo.
```

Programar en lenguaje C

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

- ¿Qué representa el texto de arriba?
- ¿Qué hay que hacer para obtener los sig. en la terminal?

```
Hola mundo.
```

- ¿Qué herramienta utilizan?

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software.

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

1. Editor de código fuente

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

1. Editor de código fuente
2. Herramientas de construcción

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

1. Editor de código fuente
2. Herramientas de construcción
3. Depurador (debugger)

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

1. Editor de código fuente
2. Herramientas de construcción
3. Depurador (debugger)

Algunas características de los IDE

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

1. Editor de código fuente
2. Herramientas de construcción
3. Depurador (debugger)

Algunas características de los IDE

- Resaltado de sintaxis

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

1. Editor de código fuente
2. Herramientas de construcción
3. Depurador (debugger)

Algunas características de los IDE

- Resaltado de sintaxis
- Indentado automático

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

1. Editor de código fuente
2. Herramientas de construcción
3. Depurador (debugger)

Algunas características de los IDE

- Resaltado de sintaxis
- Indentado automático
- Plegado de código

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

1. Editor de código fuente
2. Herramientas de construcción
3. Depurador (debugger)

Algunas características de los IDE

- Resaltado de sintaxis
- Indentado automático
- Plegado de código
- Autocompletado

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

1. Editor de código fuente
2. Herramientas de construcción
3. Depurador (debugger)

Algunas características de los IDE

- Resaltado de sintaxis
- Indentado automático
- Plegado de código
- Autocompletado
- Administración de proyecto

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

1. Editor de código fuente
2. Herramientas de construcción
3. Depurador (debugger)

Algunas características de los IDE

- Resaltado de sintaxis
- Indentado automático
- Plegado de código
- Autocompletado
- Administración de proyecto
- Terminal embebida

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

1. Editor de código fuente
2. Herramientas de construcción
3. Depurador (debugger)

Algunas características de los IDE

- Resaltado de sintaxis
- Indentado automático
- Plegado de código
- Autocompletado
- Administración de proyecto
- Terminal embebida
- Etc.

IDE – Integrated Development Environment

Entorno de desarrollo integrado

Incluye un conjunto de herramientas informáticas para facilitar el desarrollo de software. Normalmente consiste en:

1. Editor de código fuente
2. Herramientas de construcción
3. Depurador (debugger)

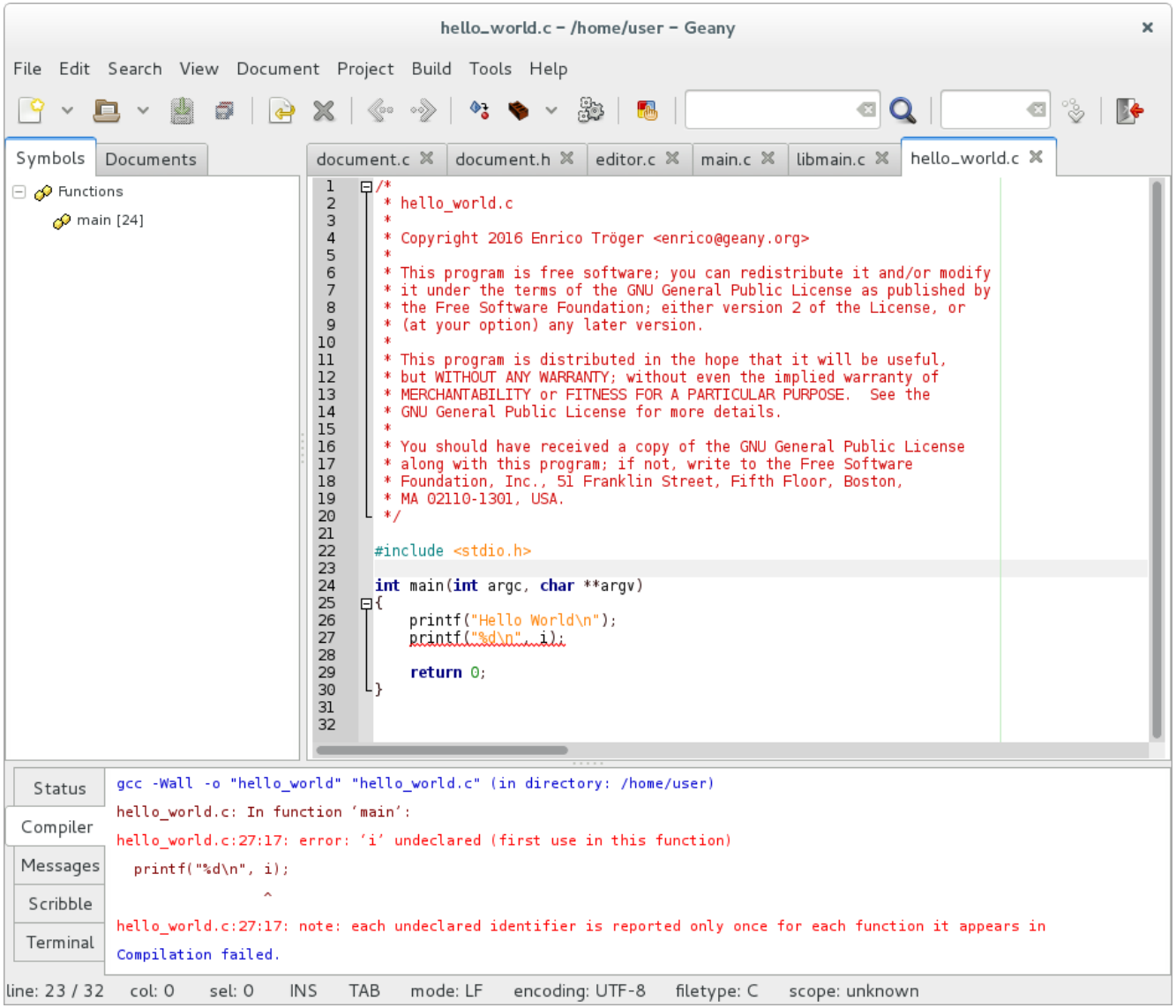
Algunas características de los IDE

- Resaltado de sintaxis
- Indentado automático
- Plegado de código
- Autocompletado
- Administración de proyecto
- Terminal embebida
- Etc.

¿Cuáles conocen?

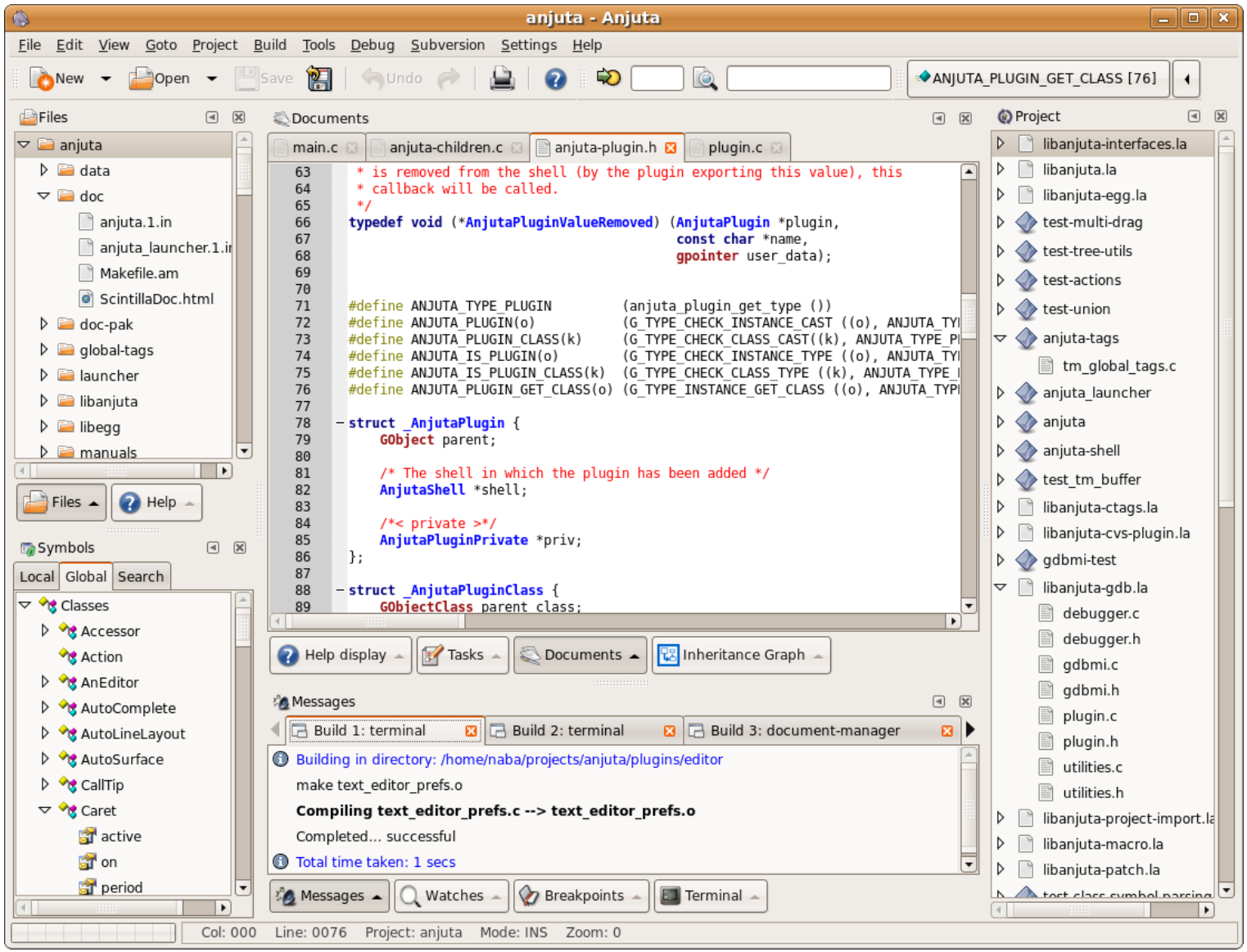
IDE – Integrated Development Environment

Geany



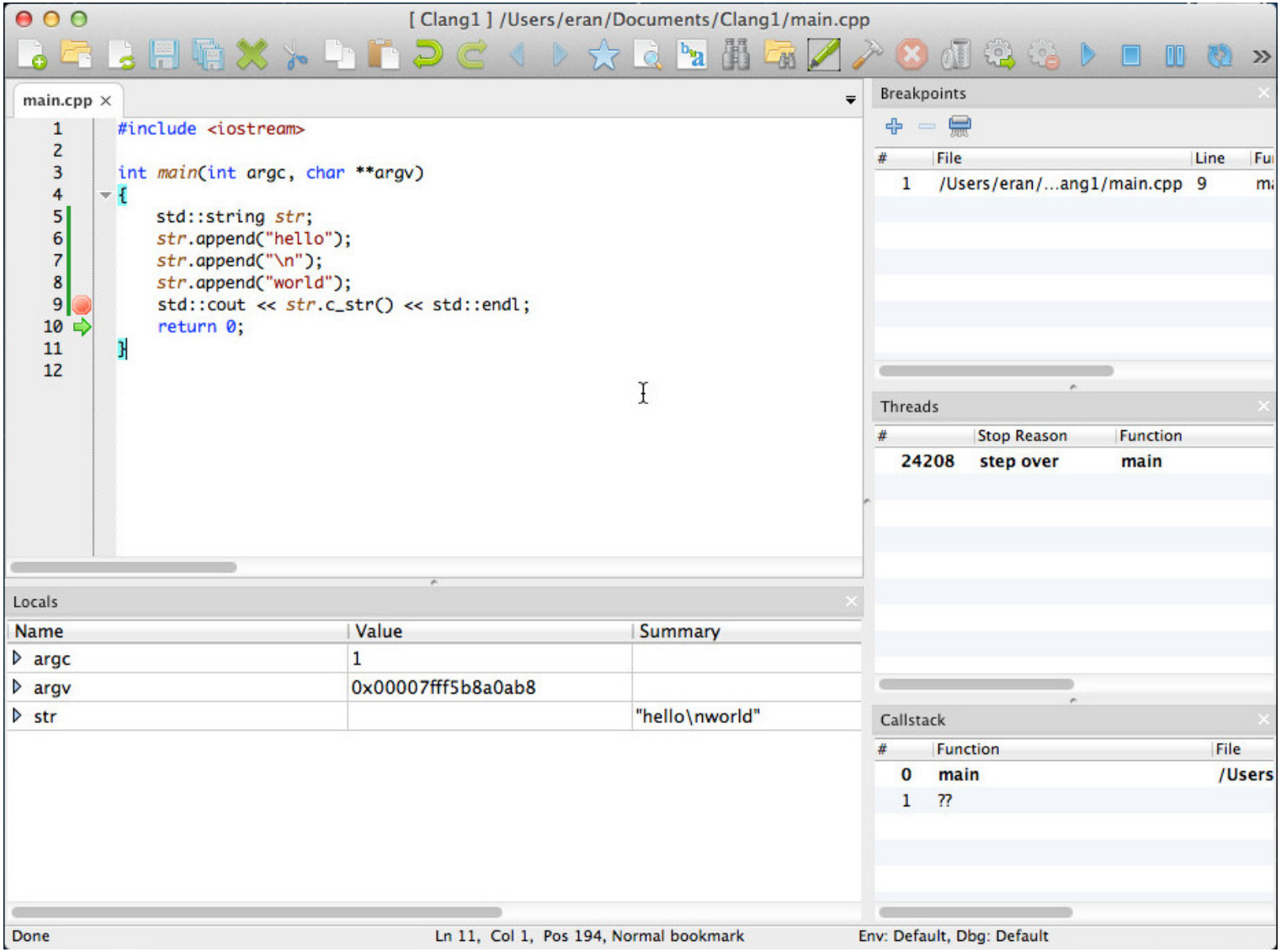
IDE – Integrated Development Environment

Anjuta



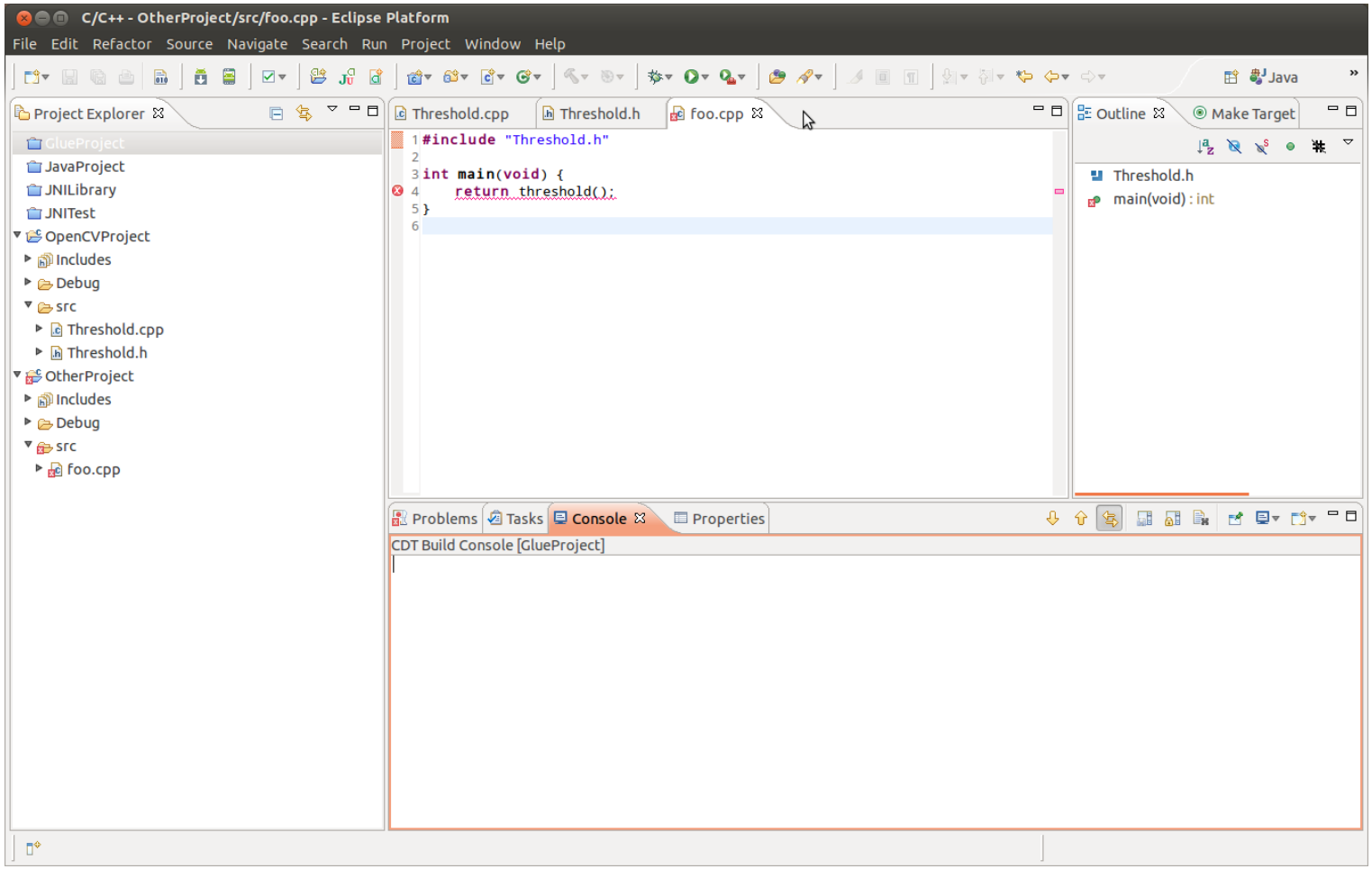
IDE – Integrated Development Environment

CodeLite



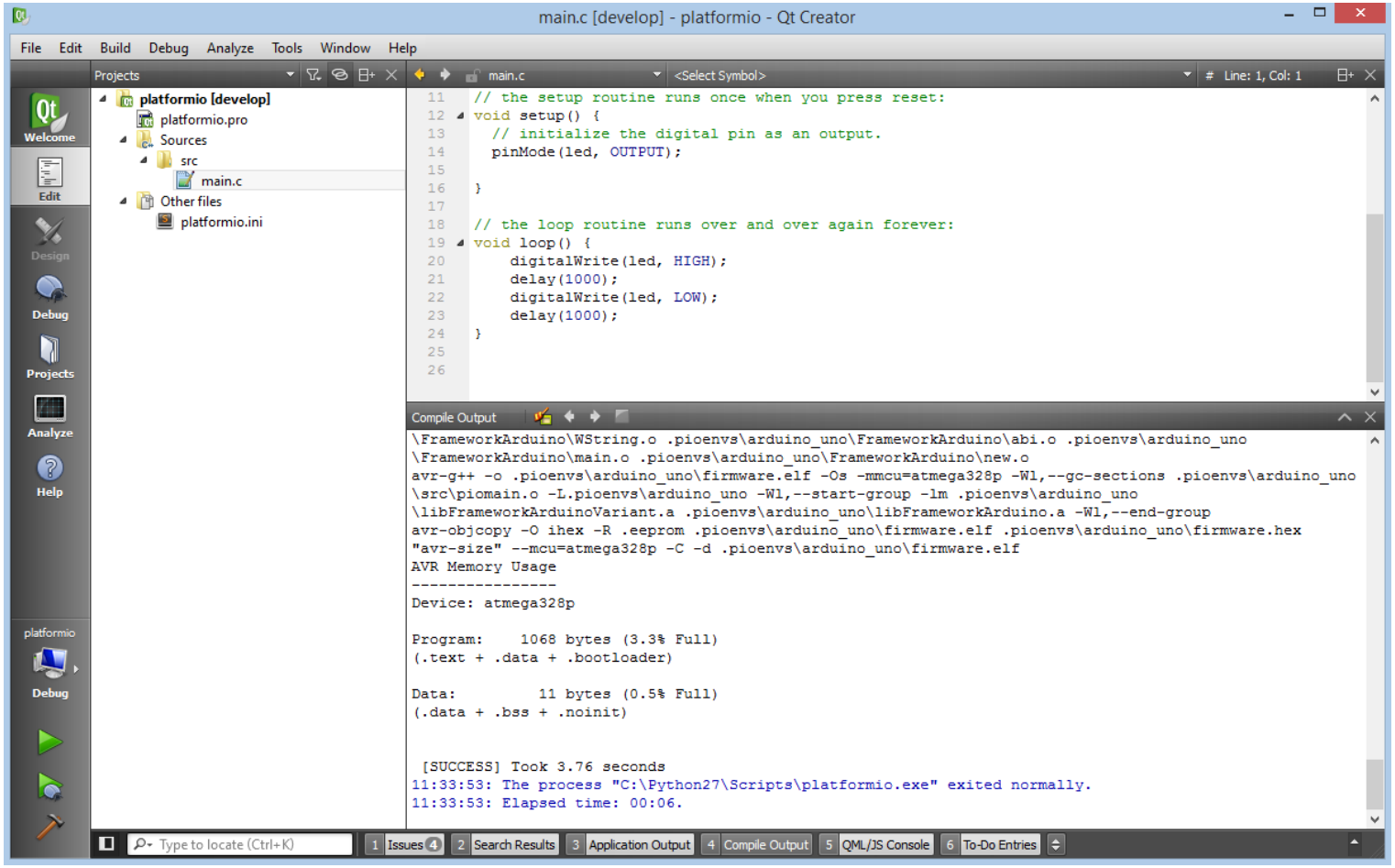
IDE – Integrated Development Environment

Eclipse CDT



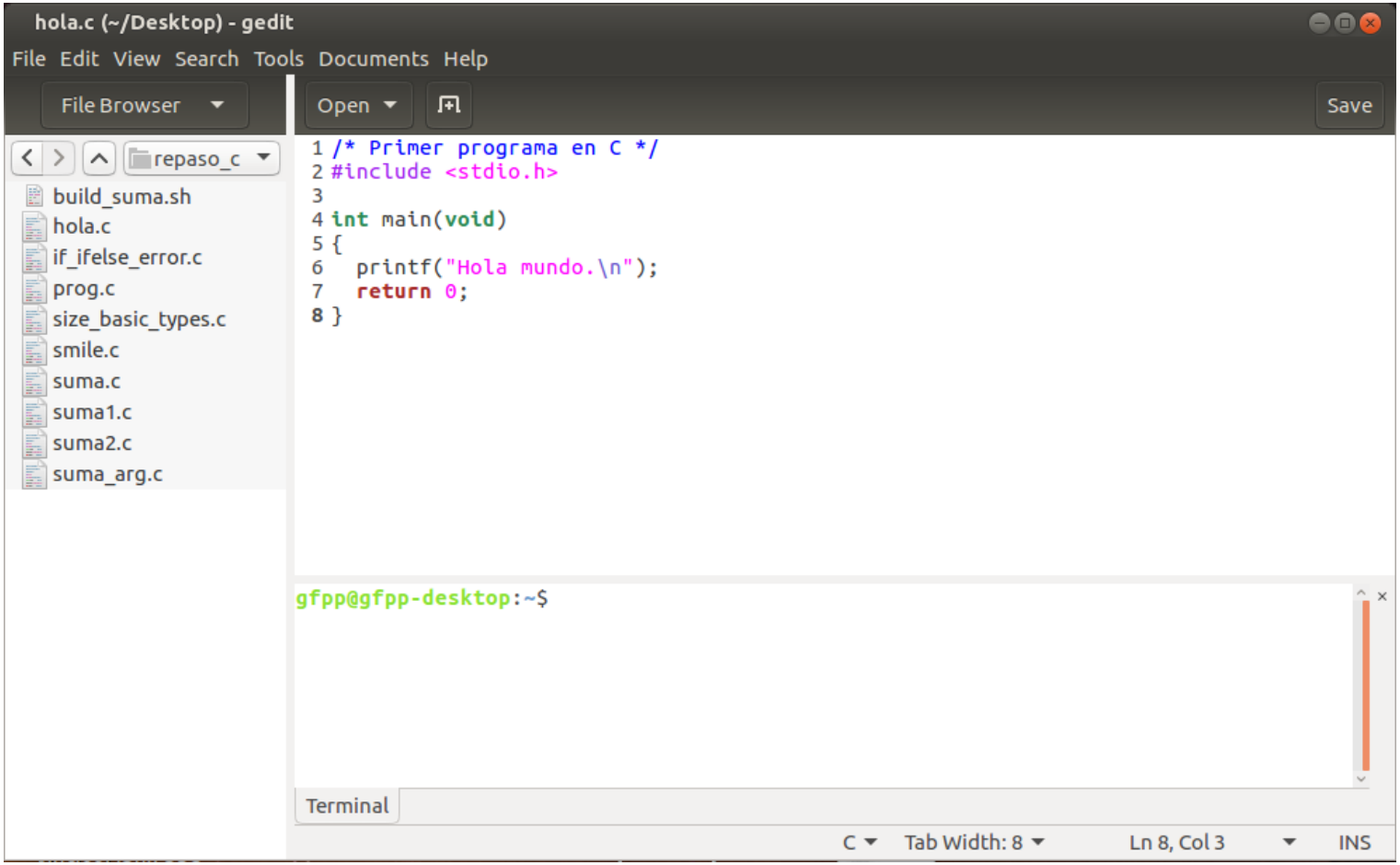
IDE – Integrated Development Environment

Qt Creator

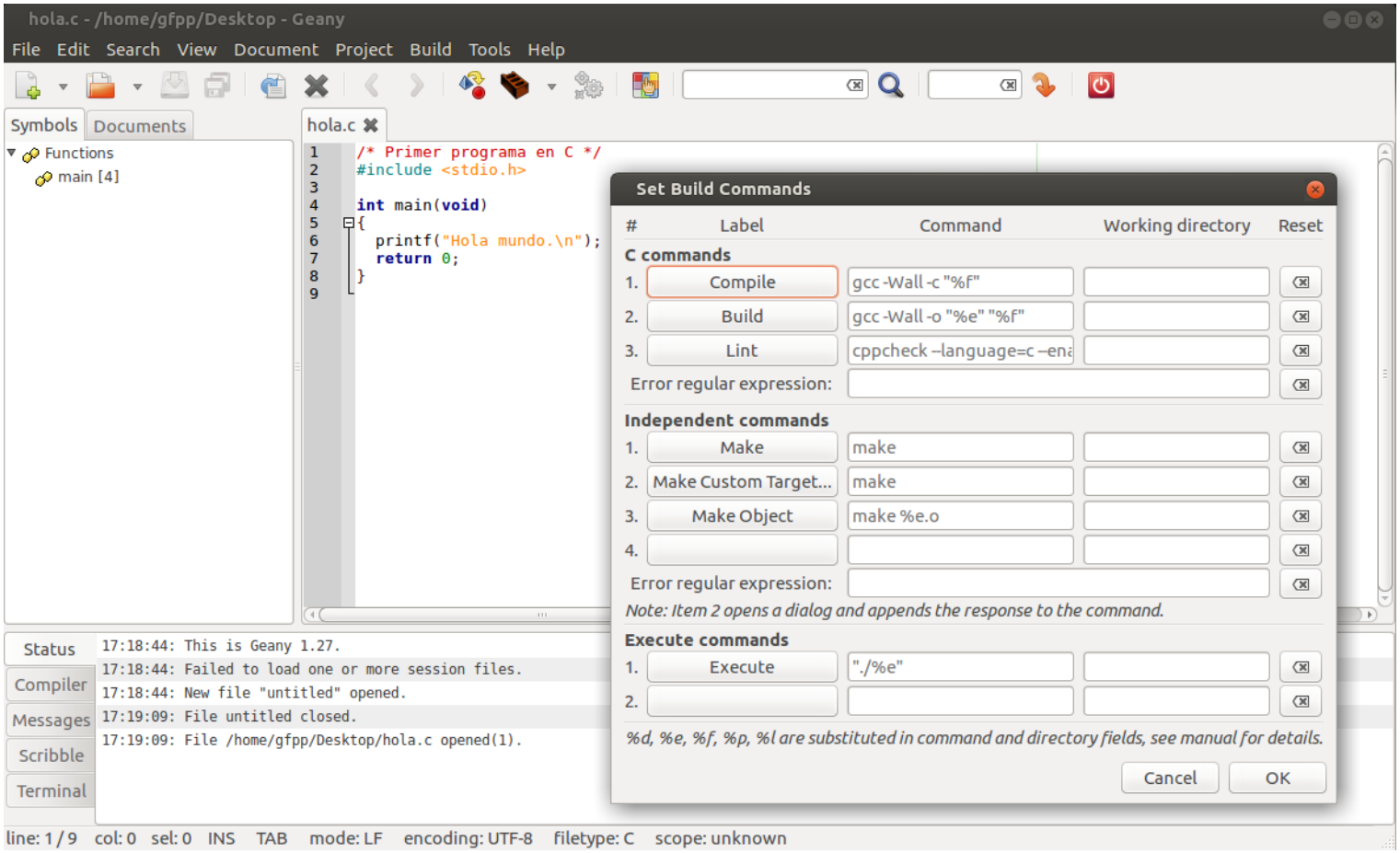


IDE – Integrated Development Environment

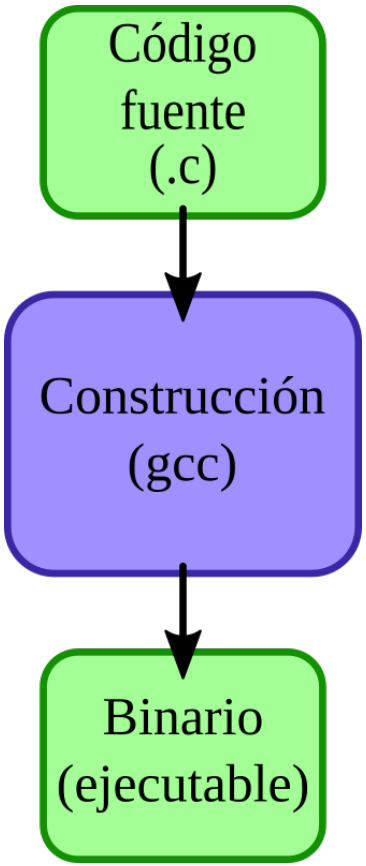
gedit



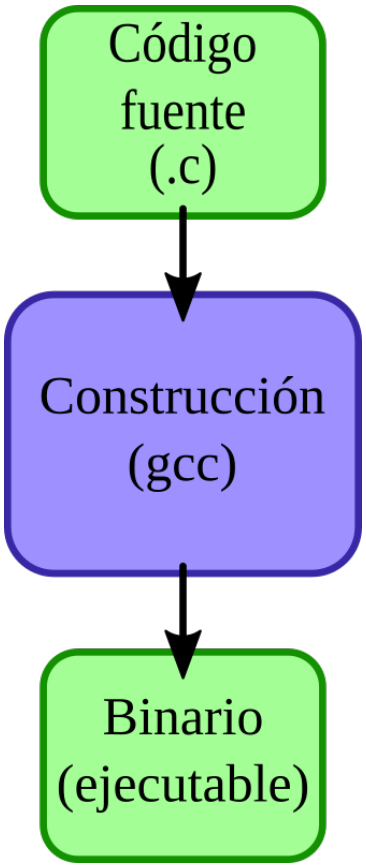
Ejemplo de configuración del IDE Geany



Construcción de un programa



Construcción de un programa

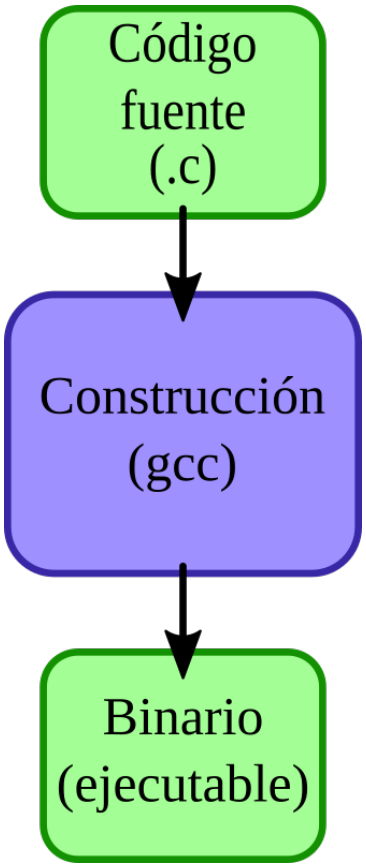


1. Escribir el código fuente

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

Construcción de un programa



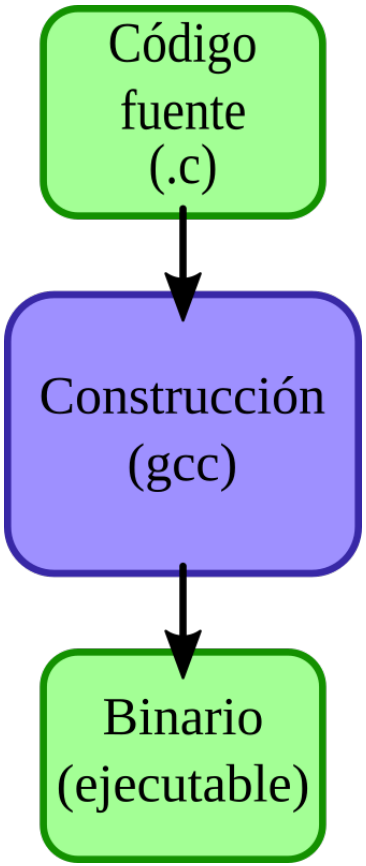
1. Escribir el código fuente

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

2. Guardar con extensión `.c` (nombre sin espacios), p.e. `hola.c`

Construcción de un programa



1. Escribir el código fuente

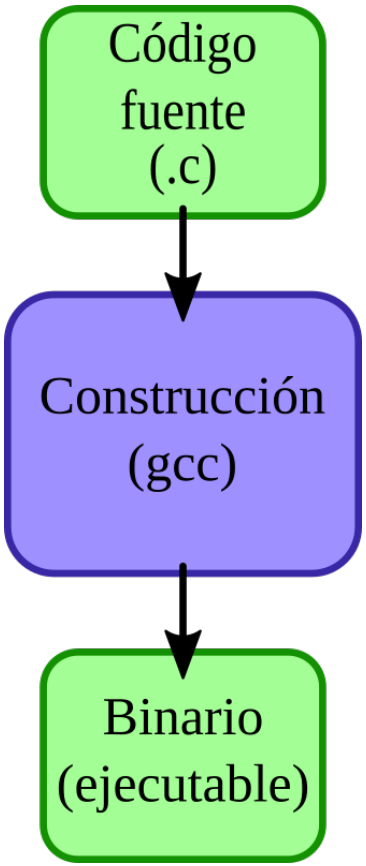
```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

2. Guardar con extensión `.c` (nombre sin espacios), p.e. `hola.c`
3. Construir el programa

```
> gcc hola.c
```

Construcción de un programa



1. Escribir el código fuente

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

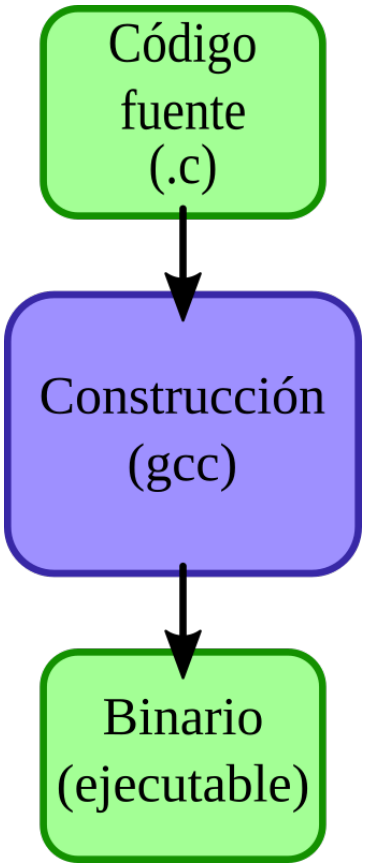
2. Guardar con extensión `.c` (nombre sin espacios), p.e. `hola.c`
3. Construir el programa

```
> gcc hola.c
```

4. Ejecutar el programa

```
> ./a.out
Hola mundo.
```

Construcción de un programa



1. Escribir el código fuente

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

2. Guardar con extensión `.c` (nombre sin espacios), p.e. `hola.c`
3. Construir el programa

```
> gcc hola.c
```

4. Ejecutar el programa

```
> ./a.out
Hola mundo.
```

En la cátedra se utilizará este procedimiento de compilación

Introducción a gcc

hola.c

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

Introducción a gcc

hola.c

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

Compilación

```
> gcc hola.c
```

Introducción a gcc

hola.c

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

Compilación

```
> gcc hola.c
```

(salida `a.out`)

Introducción a gcc

hola.c

```
1 /* Primer programa en C */
2 #include <stdio.h>
3
4 int main(void)
5 {
6   printf("Hola␣mundo.\n");
7   return 0;
8 }
```

Compilación

```
> gcc hola.c
```

(salida `a.out`) Cómo se ejecuta? `> ./a.out`

Introducción a gcc

hola.c

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola␣mundo.\n");
  return 0;
}
```

Compilación

```
> gcc hola.c
```

(salida `a.out`) Cómo se ejecuta? `> ./a.out`

Cómo cambiar el nombre a binario?

Introducción a gcc

hola.c

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

Compilación

```
> gcc hola.c
```

(salida `a.out`) Cómo se ejecuta? `> ./a.out`

Cómo cambiar el nombre a binario?

```
> gcc hola.c -o hola
```

Introducción a gcc

hola.c

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

Compilación

```
> gcc hola.c
```

(salida `a.out`) Cómo se ejecuta? `> ./a.out`

Cómo cambiar el nombre a binario?

```
> gcc hola.c -o hola
```

```
> gcc -Wall hola.c -o hola
```

Introducción a gcc

hola.c

```c
/* Primer programa en C */
#include <stdio.h>

int main(void)
{
  printf("Hola mundo.\n");
  return 0;
}
```

Compilación

```
> gcc hola.c
```

(salida `a.out`) Cómo se ejecuta? `> ./a.out`

Cómo cambiar el nombre a binario?

```
> gcc hola.c -o hola
```

```
> gcc -Wall hola.c -o hola
```

(habilita todas las advertencias/warnings)

Introducción a gcc

mal.c

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5   printf("Dos␣y␣dos␣son␣%f\n", 4);
6   return 0;
7 }
```

Introducción a gcc

mal.c

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5   printf("Dos␣y␣dos␣son␣%f\n", 4);
6   return 0;
7 }
```

¿Qué se imprime al ejecutar el programa?

Introducción a gcc

mal.c

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5   printf("Dos␣y␣dos␣son␣%f\n", 4);
6   return 0;
7 }
```

¿Qué se imprime al ejecutar el programa?

Compilar y ejecutar

```
> gcc mal.c -o mal
> ./mal
Dos y dos son 0.000000
```

Introducción a gcc

mal.c

```c
#include <stdio.h>

int main(void)
{
  printf("Dos␣y␣dos␣son␣%f\n", 4);
  return 0;
}
```

¿Qué se imprime al ejecutar el programa?

Compilar y ejecutar

```
> gcc mal.c -o mal
> ./mal
Dos y dos son 0.000000
```

Error de compilación (`-Wall`)

```
mal.c:5:10: warning: format '%f'
expects argument of type 'double',
but argument 2 has type 'int'
[-Wformat=]
```

Introducción a gcc

mal.c

```c
#include <stdio.h>

int main(void)
{
  printf("Dos y dos son %f\n", 4.); // CORREGIDO
  return 0;
}
```

¿Qué se imprime al ejecutar el programa?

Compilar y ejecutar

```
> gcc mal.c -o mal
> ./mal
Dos y dos son 0.000000
```

Error de compilación (`-Wall`)

```
mal.c:5:10: warning: format '%f'
expects argument of type 'double',
but argument 2 has type 'int'
[-Wformat=]
```

Actividad práctica

1. Reproducir el ejemplo anterior (*Hola mundo!*) en GNU/Linux
   - Editar el archivo `hola.c` con gedit
   - Abrir una terminal (p.e. `gnome-terminal`) e ir al directorio donde se encuentra el archivo fuente (comando `cd`)
   - Compilar el programa con `gcc`
   - Ejecutar

Actividad práctica

1. Reproducir el ejemplo anterior (*Hola mundo!*) en GNU/Linux
   - Editar el archivo `hola.c` con gedit
   - Abrir una terminal (p.e. `gnome-terminal`) e ir al directorio donde se encuentra el archivo fuente (comando `cd`)
   - Compilar el programa con `gcc`
   - Ejecutar
2. Configurar gedit para poder programar en C
   - Habilitar la numeración de líneas y el resaltado de sintaxis
   - Habilitar la terminal embebida

Actividad práctica

1. Reproducir el ejemplo anterior (*Hola mundo!*) en GNU/Linux
   - Editar el archivo `hola.c` con gedit
   - Abrir una terminal (p.e. `gnome-terminal`) e ir al directorio donde se encuentra el archivo fuente (comando `cd`)
   - Compilar el programa con `gcc`
   - Ejecutar
2. Configurar gedit para poder programar en C
   - Habilitar la numeración de líneas y el resaltado de sintaxis
   - Habilitar la terminal embebida
3. Descargar, configurar y probar algunos de los IDE antes mencionados

Actividad práctica

1. Reproducir el ejemplo anterior (*Hola mundo!*) en GNU/Linux
   - Editar el archivo `hola.c` con gedit
   - Abrir una terminal (p.e. `gnome-terminal`) e ir al directorio donde se encuentra el archivo fuente (comando `cd`)
   - Compilar el programa con `gcc`
   - Ejecutar
2. Configurar gedit para poder programar en C
   - Habilitar la numeración de líneas y el resaltado de sintaxis
   - Habilitar la terminal embebida
3. Descargar, configurar y probar algunos de los IDE antes mencionados
4. Codificar un programa para sumar dos números enteros con interacción con el usuario

Actividad práctica

1. Reproducir el ejemplo anterior (*Hola mundo!*) en GNU/Linux
   - Editar el archivo `hola.c` con gedit
   - Abrir una terminal (p.e. `gnome-terminal`) e ir al directorio donde se encuentra el archivo fuente (comando `cd`)
   - Compilar el programa con `gcc`
   - Ejecutar
2. Configurar gedit para poder programar en C
   - Habilitar la numeración de líneas y el resaltado de sintaxis
   - Habilitar la terminal embebida
3. Descargar, configurar y probar algunos de los IDE antes mencionados
4. Codificar un programa para sumar dos números enteros con interacción con el usuario
5. Codificar un programa para sumar dos números enteros sin interacción con el usuario