

- ¿Qué ocurre cuando se invoca el comando `cpp` con esos argumentos?
me crea un archivo llamado `programa.i` que parece ser un preprocesado de compilación
- ¿Qué similitudes encuentra entre los archivos `programa.c` y `programa.i`?
la única cosa en común es el `main` hasta abajo del `programa.i`, más allá de eso parece estar relacionado con la compilación del programa
- ¿Qué pasa con las macros y los comentarios del código fuente original en `programa.i`?
no aparecen
- Compare el contenido de `programa.i` con el de `stdio.h` e indique de forma general las similitudes entre ambos archivos.
pues son bastante similares al establecer funciones de código tipo `c`
- ¿A qué etapa corresponde este proceso?
corresponde al Programa Fuente modificado
- ¿Para qué sirve la opción `-Wall`?
se encarga de detectar errores comunes
- ¿Qué le indica a `gcc` la opción `-S`?
que se detenga en el archivo tipo ensamblador
- ¿Qué contiene el archivo de salida y cuál es su extensión?
de tipo ensamblador con extensión `.s`
- ¿A qué etapa corresponde este comando?
al Programa objeto en lenguaje
- Antes de revisarlo, indique cuál es su hipótesis sobre lo que debe contener el archivo con extensión `.o`.
teorizo que es el código en lenguaje máquina
- Diga de forma general qué contiene el archivo `programa.o` y por qué se visualiza de esa manera.
está en binario, en el lenguaje de las máquinas
- ¿Qué programa se invoca con `as`?
el ensamblador
- ¿A qué etapa corresponde la llamada a este programa?
Programa objeto en lenguaje
- `ld-linux-x86-64.so.2`
`\\wsl.localhost\Ubuntu\mnt\wslg\distro\usr\lib64`
- `Scrt1.o` (o bien, `crt1.o`)

- `\\wsl.localhost\Ubuntu\mnt\wslg\distro\usr\lib\x86_64-linux-gnu`
- `crti.o`
`\\wsl.localhost\Ubuntu\mnt\wslg\distro\usr\lib\x86_64-linux-gnu`
- `crtbeginS.o`
`\\wsl.localhost\Ubuntu\mnt\wslg\distro\usr\lib\gcc\x86_64-linux-gnu\13\crtbegin.o`
- `crtendS.o`
`\\wsl.localhost\Ubuntu\mnt\wslg\distro\usr\lib\gcc\x86_64-linux-gnu\13\crtendS.o`
- `crtn.o`
`\\wsl.localhost\Ubuntu\usr\lib\x86_64-linux-gnu\crtn.o`
- En caso de que el comando `ld` mande errores, investigue cómo enlazar un programa utilizando el comando `ld`. Y proponga una posible solución para llevar a cabo este proceso con éxito.
después de investigar llegué a que el comando para mi caso, que ando en mi casa con Windows, es este:
`gcc programa.o -o ejecutable`

- Describa el resultado obtenido al ejecutar el comando anterior.
se creó un archivo ejecutable
- Una vez que se enlazó el código máquina relocizable, podemos ejecutar el programa con la siguiente instrucción en la terminal: `./ejecutable`

```
alexz@DESKTOP-7R91J04:~/coding/comp12$ cpp programa.c programa.i
alexz@DESKTOP-7R91J04:~/coding/comp12$ gcc -Wall -S programa.i
alexz@DESKTOP-7R91J04:~/coding/comp12$ ls
programa.c  programa.i  programa.s
alexz@DESKTOP-7R91J04:~/coding/comp12$ as programa.s -o programa.o
alexz@DESKTOP-7R91J04:~/coding/comp12$ gcc programa.o -o ejecutable
alexz@DESKTOP-7R91J04:~/coding/comp12$ ./ejecutable
Hola Mundo!
Resultado : 28.274401
alexz@DESKTOP-7R91J04:~/coding/comp12$ |
```

- Genere nuevamente el archivo.i. De preferencia asigne un nuevo nombre. ¿Cambia en algo la ejecución final?
solo cambia un poco el valor, el resto sigue igual
- Directivas de c
- 1. Directiva `#if` con operadores lógicos
- Permite evaluar expresiones constantes. Si la condición falla, detiene la compilación con un mensaje personalizado.
- 2. Operador `#`
- Es un operador especial del preprocesador que envuelve un parámetro entre comillas dobles.
- 3. `undef`
- "Des-define" una macro o constante definida previamente con `#define`
- 4. `pragma`
- Ofrece instrucciones especiales al compilador que no están definidas en el estándar de C.