

Chaîne de compilation

Lycée Thiers

Du fichier C à l'exécutable

Un programme écrit en C n'est pas directement exécutable par le processeur. Il passe par plusieurs étapes :



En pratique

gcc pilote toute cette chaîne et peut effectuer toutes les étapes en une seule commande.

Un programme exemple

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5     int i = 10;
6     printf("Hello world!\n");
7     printf("i=%d\n", i);
8     return 0;
9 }
```

Compiler puis exécuter

La commande usuelle est :

```
$ gcc hello.c -o hello
$ ./hello
Hello world!
i=10
```

- l'option **-o hello** donne le nom **hello** à l'exécutable ;
- sans option particulière, **gcc** enchaîne prétraitement, compilation, assemblage et édition de liens.

1. Prétraitement

Le préprocesseur effectue des **transformations textuelles** avant la compilation.

- Les directives commencent par **#**.
- **#include** insère le contenu d'un fichier.
- **#define** permet notamment des substitutions textuelles.

On peut s'arrêter après cette étape :

```
$ gcc -E hello.c -o hello.i
```

Le fichier **hello.i** est encore un fichier texte contenant du C, mais les directives du préprocesseur ont été traitées.

2. Compilation et assemblage

Compilation traduction du C prétraité en **assembleur**.

Assemblage traduction de l'assembleur en **code machine** placé dans un fichier objet.

Pour observer les fichiers intermédiaires :

```
$ gcc -S hello.c # produit hello.s  
$ gcc -c hello.c # produit hello.o
```

Attention

hello.o contient déjà du code machine, mais ce n'est pas encore un programme exécutable : il reste à effectuer l'édition de liens.

3. Édition de liens

L'**éditeur de liens** construit l'exécutable à partir des fichiers objets et des bibliothèques nécessaires.

- Il rassemble les différentes parties du programme.
- Il résout les références vers des fonctions définies ailleurs.
- Par exemple, le code de **printf** n'est pas contenu dans notre fichier **hello.o** : il provient de la bibliothèque C.

Conséquence

Un appel à une fonction inexistante peut être accepté lors de la compilation d'un fichier, puis provoquer une erreur lors de l'édition de liens : *undefined reference*.

Remarque : le code de démarrage **crt0**

Avant d'exécuter notre fonction **main**, il faut effectuer quelques opérations de démarrage : préparer l'environnement d'exécution, puis transmettre le contrôle à **main**.

crt0

On appelle traditionnellement **crt0** (*C runtime startup*) ce code de démarrage, ajouté automatiquement lors de l'édition de liens. Il fournit notamment le point d'entrée initial du programme avant l'appel à **main**.

Sur un système Linux moderne, ce rôle est en pratique réparti entre plusieurs fichiers de démarrage, par exemple **crt1.o**, **crti.o** et **crtn.o**. Le nom **crt0** reste utile pour désigner l'idée générale.

Bibliothèques et option **-lm**

Certaines fonctions sont fournies dans des bibliothèques séparées. Pour les fonctions mathématiques, on écrit par exemple :

```
1 #include <math.h>
2
3 /* ... */
4 double y = sqrt(x);
```

puis on compile avec :

```
$ gcc calcul.c -lm -o calcul
```

- **math.h** fournit les **déclarations** des fonctions ;
- **-lm** demande à l'éditeur de liens d'utiliser la bibliothèque mathématique **libm**.

À retenir

Inclure un fichier **.h** et lier une bibliothèque sont deux opérations différentes.

Quelques options utiles de **gcc**

- Wall** active de nombreux avertissements utiles.
- Wextra** active quelques avertissements supplémentaires.
- Wvla** signale l'utilisation d'un tableau de taille variable (*Variable Length Array*).
- Werror=vla** transforme cet avertissement en erreur : on peut ainsi interdire les VLA dans les programmes demandés.
- Dnom=valeur** définit une macro du préprocesseur depuis la ligne de commande, comme avec un **#define**.
- g** ajoute les informations utiles à un débogueur comme **gdb**.

Par exemple :

```
$ gcc -Wall -Wextra -Werror=vla prog.c -o prog
$ gcc -DDEBUG=1 prog.c -o prog
```

Conseil

Pour les programmes écrits pendant le cours, on utilisera au minimum

-Wall -Wextra.

Les commandes à retenir

gcc -E hello.c -o hello.i	prétraitement seulement
gcc -S hello.c	jusqu'à l'assembleur hello.s
gcc -c hello.c	jusqu'au fichier objet hello.o
gcc hello.o -o hello	édition de liens
gcc hello.c -o hello	chaîne complète

Idée essentielle

.c → **.i** → **.s** → **.o** → exécutable