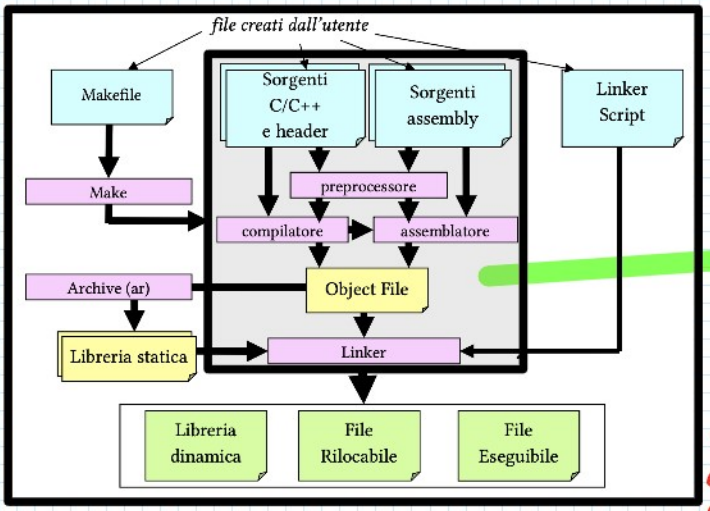
PROCESSO DI COMPILAZIONE

giovedì 17 novembre 2022 11:13



È quello che nel linguaggio del "POPOLO" viene chiamato compilatore.

Si dovrebbe chiamare "TOOLCHAIN" di compilazione. È una catena di STRUMENTI.

IL TERMINE COMPILATORE CHE COMUNEMENTE VIENE UTILIZZATO, È UN TERMINE SCORRETTO.
IL COMPILATORE È UN INSIEME DI PROGRAMMI, È UNA CATENA DI STRUMENTI;

QUESTI STRUMENTI SONO:

- PREPROCESSORE.
- COMPILATORE.
- ASSEMBLATORE.
- LINKER.

→ SONO questi 4 STRUMENTI che permettono di generare l'Eseguibile che poi noi schiacciamo;

Quando SCRIVIAMO un PROGRAMMA in C/C++/ASSEMBLY, abbiamo la necessità di TRADURRE i NOSTRI codici, nell'ISTRUZIONE MACCHINA, ossia caratteri di 0 e 1.

SORGENTE IN C VS SORGENTE IN ASSEMBLY

L'ISTRUZIONE ASSEMBLY ha una CORRISPONDENZA 1A1 CON UN'ISTRUZIONE macchina.

ASSEMBLATORE = Legge UN SORGENTE IN ASSEMBLY E LO TRADUCE NEL CORRISPETTIVO LINGUAGGIO MACCHINA

COMPILATORE = Legge IL codice SORGENTE IN C e LO TRADUCE NELLA SUA EQUIVALENTE ISTRUZIONE ASSEMBLY.

PREPROCESSORE = IL PREPROCESSORE È un altro programma che legge IL NOSTRO codice sorgente e INTERROGA delle cosiddette direttive, ossia dei comandi che devono essere INTERPRETATI dal PREPROCESSORE, e il suo ruolo è quello di TRADURRE da codice sorgente a codice sorgente.

Le DIRETTIVE sono quelle RIGHE che cominciano con #.

IL file IN OUTPUT del PREPROCESSORE È UN FILE IN CUI CIASCUNA DIRETTIVA È TALE PERCUI IL PREPROCESSORE la SOSTITUIRÀ CON ALTRO TESTO

È IN grado di SOSTITUIRE una RIGA col contenuto di un altro file.

Ecco come vengono incluse le librerie.

→ L'OUTPUT È un file sorgente in C nel caso di un programma SCRITTO IN C, oppure un file ASSEMBLY nel caso IN CUI GIÀ SCRITTO IN ASSEMBLY.

Se genero un nuovo sorgente C, verrà dato in pasto al compilatore!
Il risultato viene dato in pasto all'ASSEMBLATORE!

le stringhe binarie del linguaggio macchina vengono salvate su un file **OBJECT.**

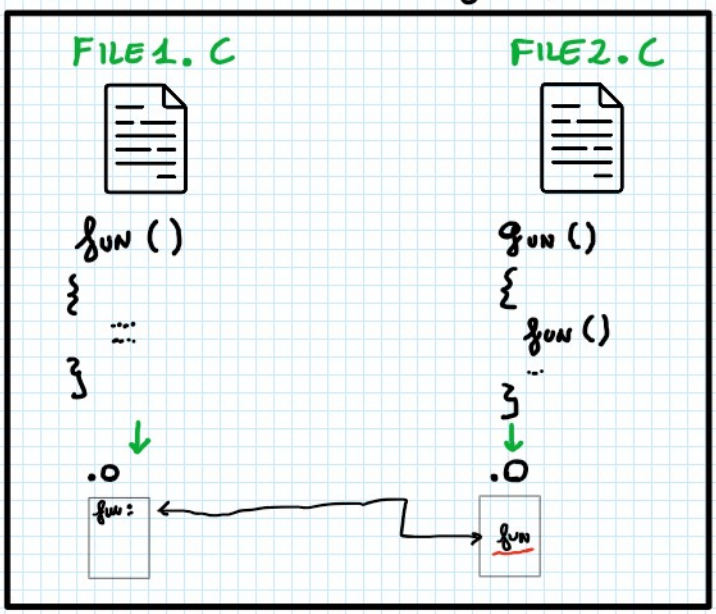
Quindi l'output dell'ASSEMBLATORE, non è proprio un programma, è un **FILE OBJECT.**

Un FILE oggetto è un insieme di dati e ISTRUZIONI che però NON possono essere eseguite in maniera definitiva dal NOSTRO PROCESSORE;

Se ho 10 sorgenti C, e 11 sorgenti assembly del mio progetto, ciascuno di questi verrà compilato uno per volta singolarmente;

Il risultato di ciascuna compilazione di ciascuno dei file sorgente, sarà un object file diverso;

Quindi questi file object li dovrò collegare fra di loro!



Dentro i file .o ci saranno delle ISTRUZIONI macchina che ad un certo punto dovranno effettuare un'invocazione in qualche modo, della funzione fun! MA la funzione fun sta in un altro file!

Il mio compilatore NON INSERISCE la chiamata a funzione fun, ma INSERISCE una specie di segnalibro: quest segnalibro porta con sé dei metadati che dicono che ad un certo punto del mio programma bisognerà inserire l'invocazione alla funzione fun();

L'output dell'assemblatore è un output intrinsecamente incompleto!
→ quando sarai in grado di trovare in memoria l'indirizzo della funzione f allora scrivilo nel file.

questo file oggetto con questi simboli strani, si chiama "file rilocabile", perché poi il **PROCESSO DI RILOCAZIONE** sostituirà quei segnalibri con i riferimenti stessi!

IL LINKER si occupa di prendere questi file RILOCABILI, metterli insieme e generare un eseguibile!
ASSEMBLA file OBJECT attraverso il processo di Rilocazione!
Esiste un linker SCRIPT che comunica al linker, in che modo fare il collegamento!

Il linker trasformerà tutto questo con un output di 3 tipi:
- ESEGUIBILE

Il linker trasformerà tutto questo con un output di 3 tipi:

- ESEGUIBILE
- FILE RIELOCABILE: OBJECT file
- DLL: pezzo di programma che posso collegare ad un altro programma per fornirgli delle funzionalità. (C. So in windows)

Le librerie statiche (.h includo) vanno dentro l'eseguibile, quelle dinamiche no, vengono messe da una parte per consentirne l'esecuzione

Se ho 200 file C è tutto più complicato, non posso fare il procedimento per ognuno.
Posso automatizzare questo processo, esiste uno strumento che si chiama MAKE