

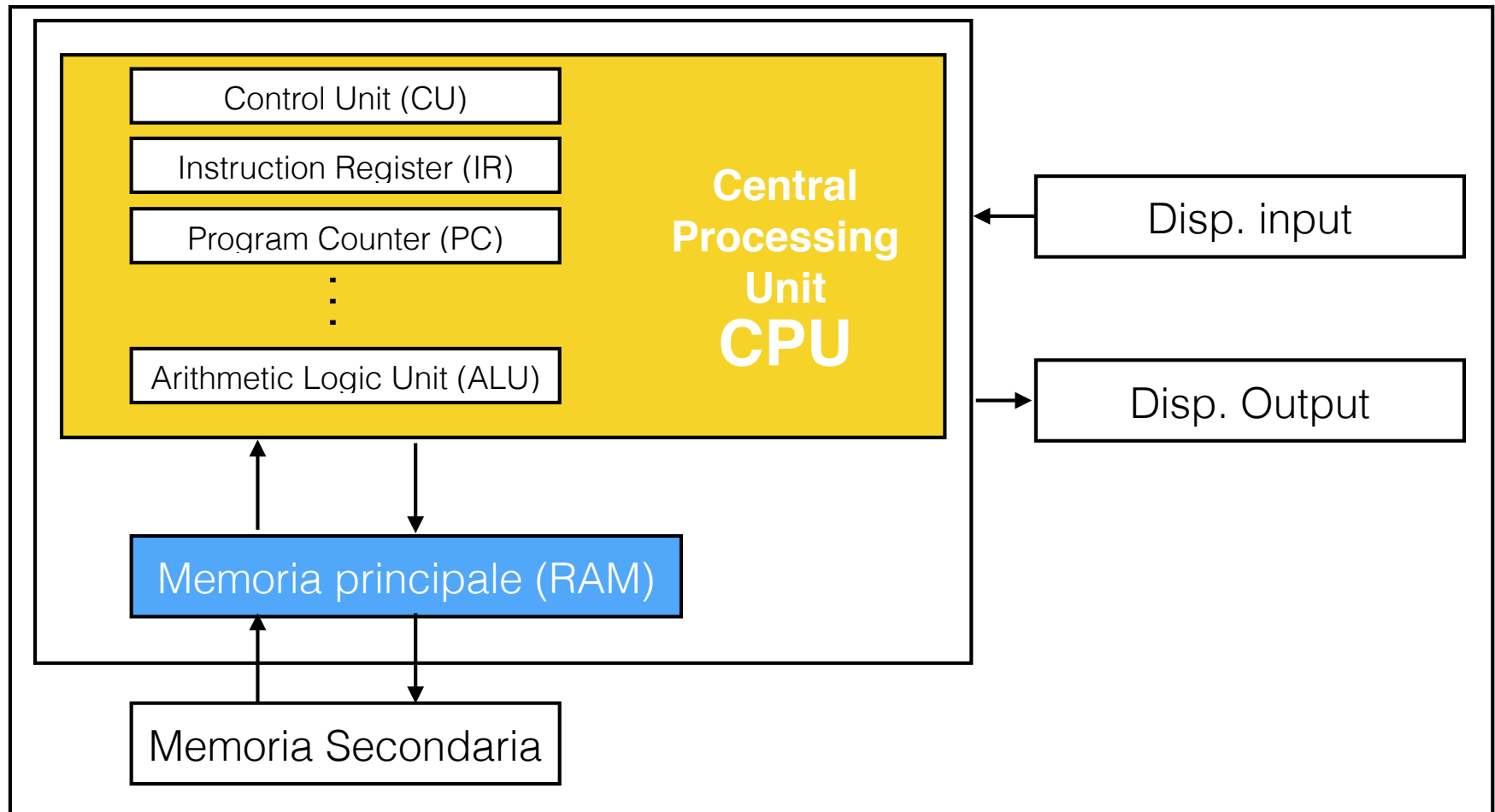
# Realizzare ed eseguire programmi

introduzione alla programmazione


# Sommario

- elementi di un sistema di calcolo
- compilazione

# Elementi di un sistema di calcolo



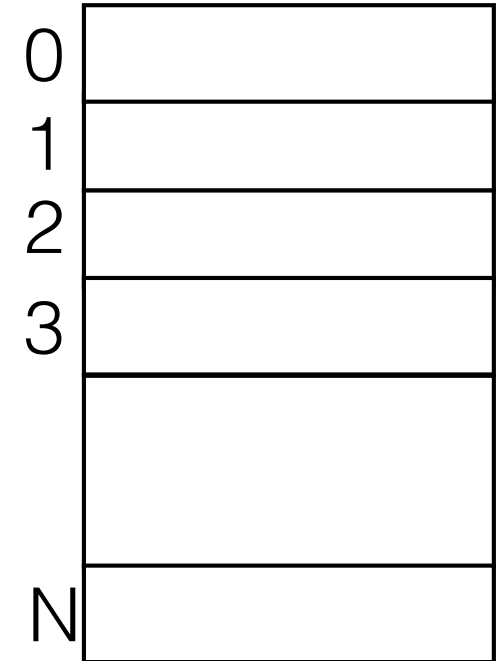
# Macchina di Von Neumann (cenni)

- lo schema logico che abbiamo appena visto è stato introdotto da John Von Neumann negli anni 40
- Elementi
  - Unità di elaborazione (CPU) 

Unità di controllo  
Unità aritmetico-logica  
Registri
  - Memoria (RAM)
  - Input / output

# Macchina di Von Neumann (cenni)

- **RAM** (Random Access Memory, memoria ad accesso arbitrario mediante indirizzo)
- Realizza la memorizzazione di un vettore di numeri interi.
- Sia la **dimensione** del vettore (numero di elementi componenti) che il **massimo valore memorizzabile** in ogni elemento del vettore sono *predeterminati* al momento della costruzione e/o assemblaggio del dispositivo.



# Macchina di Von Neumann (cenni)

- **Input** (unità di ingresso) Permette all'utente di interagire con la macchina (per esempio attraverso l'uso di una tastiera numerica) per l'introduzione di valori di tipo intero, uno alla volta.
- **Output** (unità di uscita) Permette alla macchina di stampare in un formato numerico leggibile dall'utente un valore intero per volta.

# Macchina di Von Neumann (cenni)

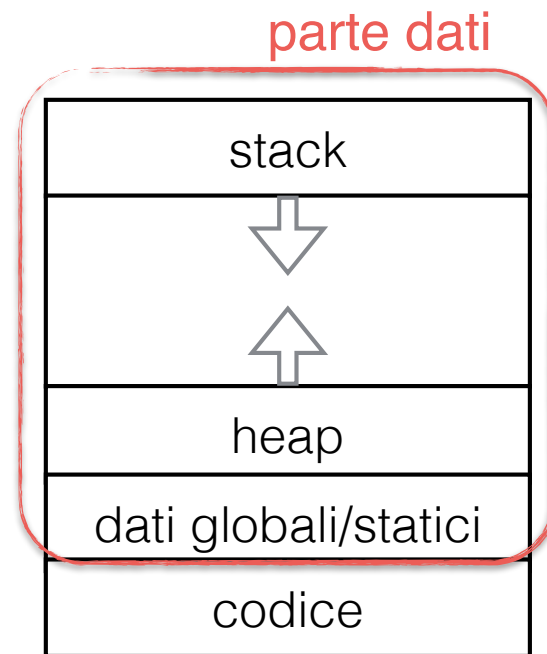
- **Control Unit** (unità di controllo) realizza il funzionamento della macchina.
- Contiene:
- Due registri che memorizzano valori interi:
  - Accumulatore (ACC)
  - Instruction Register (IR)
- Un registro che memorizza l'indirizzo di una cella della RAM:
  - Program Counter (PC): memorizza l'indirizzo della prossima istruzione

# Macchina di Von Neumann (cenni)

- La MVN codifica istruzioni in forma numerica
- Istruzioni e dati vengono inseriti insieme nella RAM
- Questa caratteristica (basata su lavori teorici di Turing) viene usata ancora oggi nella realizzazione di sistemi di calcolo.



# Organizzazione della memoria primaria (RAM) a run time (a tempo di esecuzione)



# Struttura dei sistemi di calcolo (stratificazione)

- **L5 Linguaggi di alto livello** (C, C++, Java, ...) - è il livello di macchina virtuale normalmente usato dai programmatori
- **L4 Assembler e Librerie** - è il livello di macchina virtuale più basso utilizzabile dal programmatore
- **L3 Nucleo del sistema operativo** - permette l'attivazione di processi e l'uso di risorse fisiche del sistema
- **L2 Macchina convenzionale** (microprocessori) - livello di definizione delle istruzioni base del computer
- **L1 Microarchitettura** - Livello di definizione del funzionamento dei singoli componenti fisici in termini di interconnessione e spostamento di informazioni tra circuiti logici
- **L0 Logica Circuitale** - Livello di realizzazione dei circuiti logici elementari
- **L-1 Elettronica/Fotonica** - Livello di progettazione dei dispositivi fisici
- **L-2 Fisica dello stato solido** (semiconduttori - quantistica) - progettazione e realizzazione dei circuiti integrati

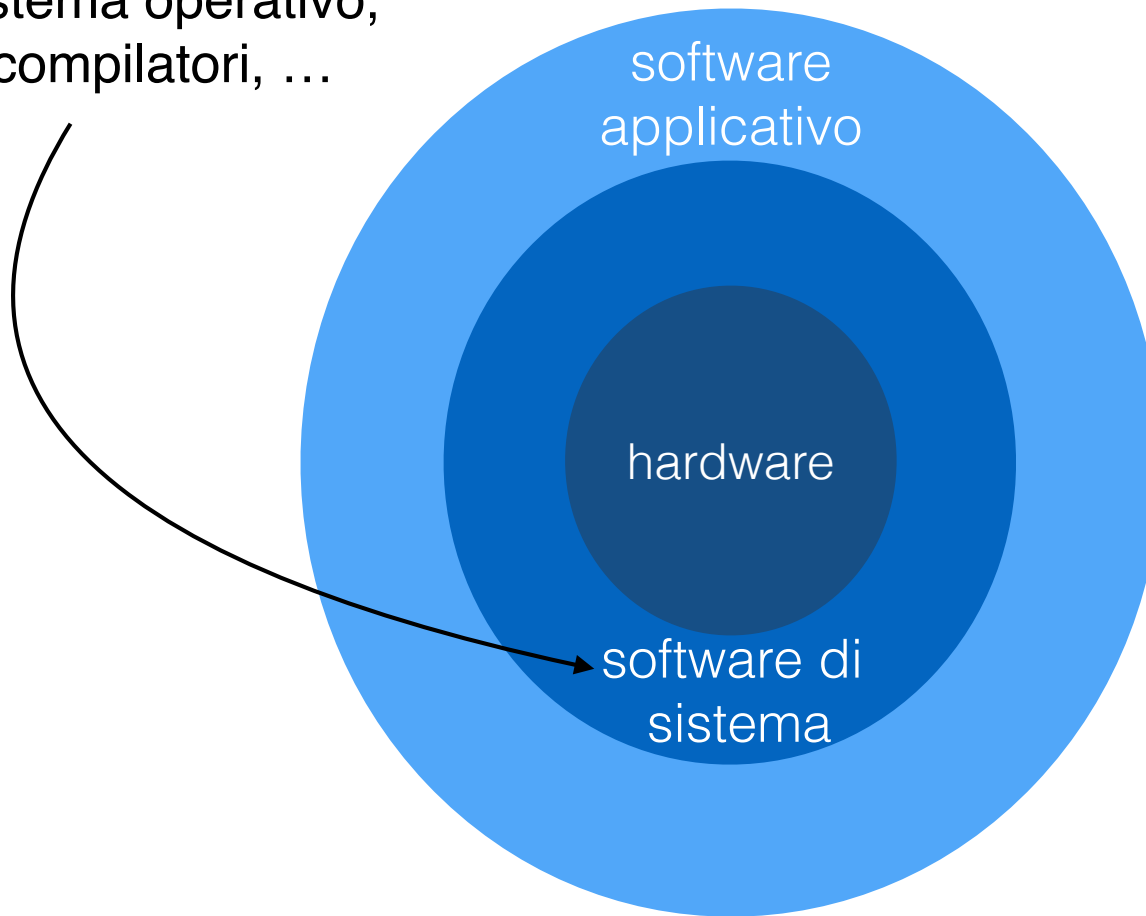
IP, ...

laurea  
ing elettronica

FISICA

# Stratificazione (semplificata)

sistema operativo,  
compilatori, ...



# compilazione

- I linguaggi di alto livello forniscono un supporto logico che permette al programmatore di astrarre rispetto ai dettagli del codice macchina

L5

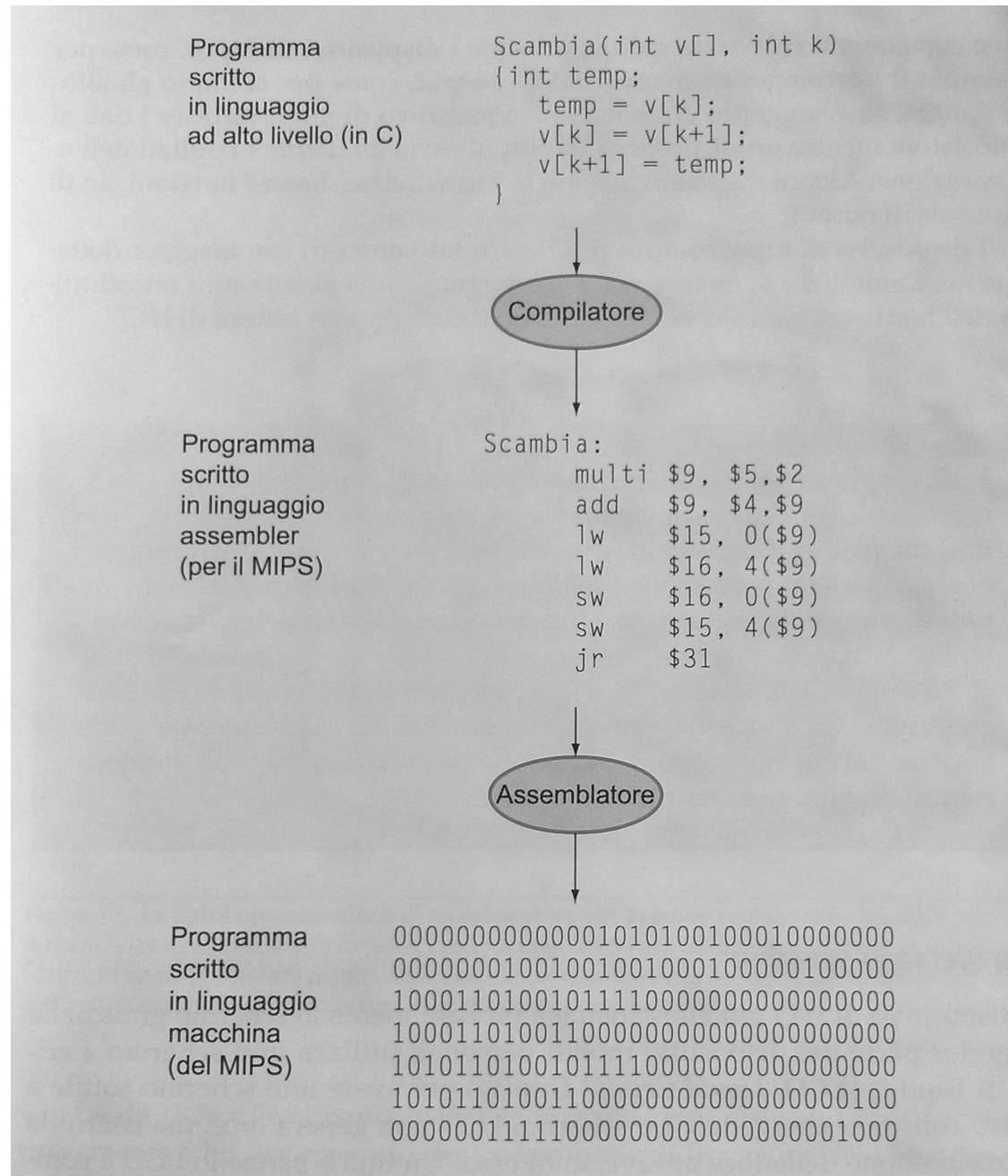
*traduzione*  
(compilatore)



- i programmi che “girano” su un sistema di calcolo sono sequenze di istruzioni e dati in codice macchina

L2

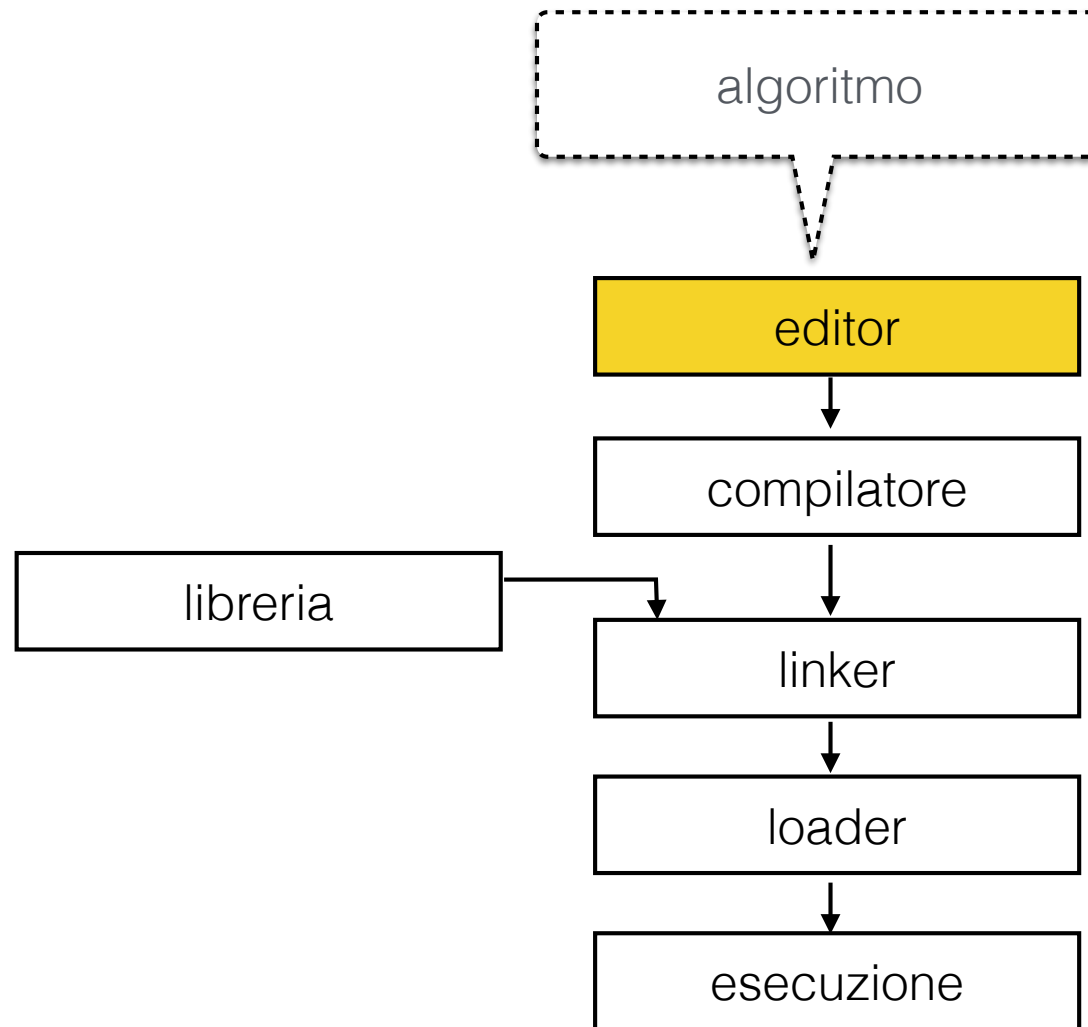
# compilazione



# compilazione

- il compilatore è un programma che prende in input il **codice sorgente** scritto in un linguaggio di alto livello e lo traduce (compila) in **codice eseguibile**
- il processo di compilazione coinvolge un altro programma (il **linker**) che unisce vari pezzi di codice pre-compilato
  - incluse librerie e altri moduli sviluppati dal programmatore

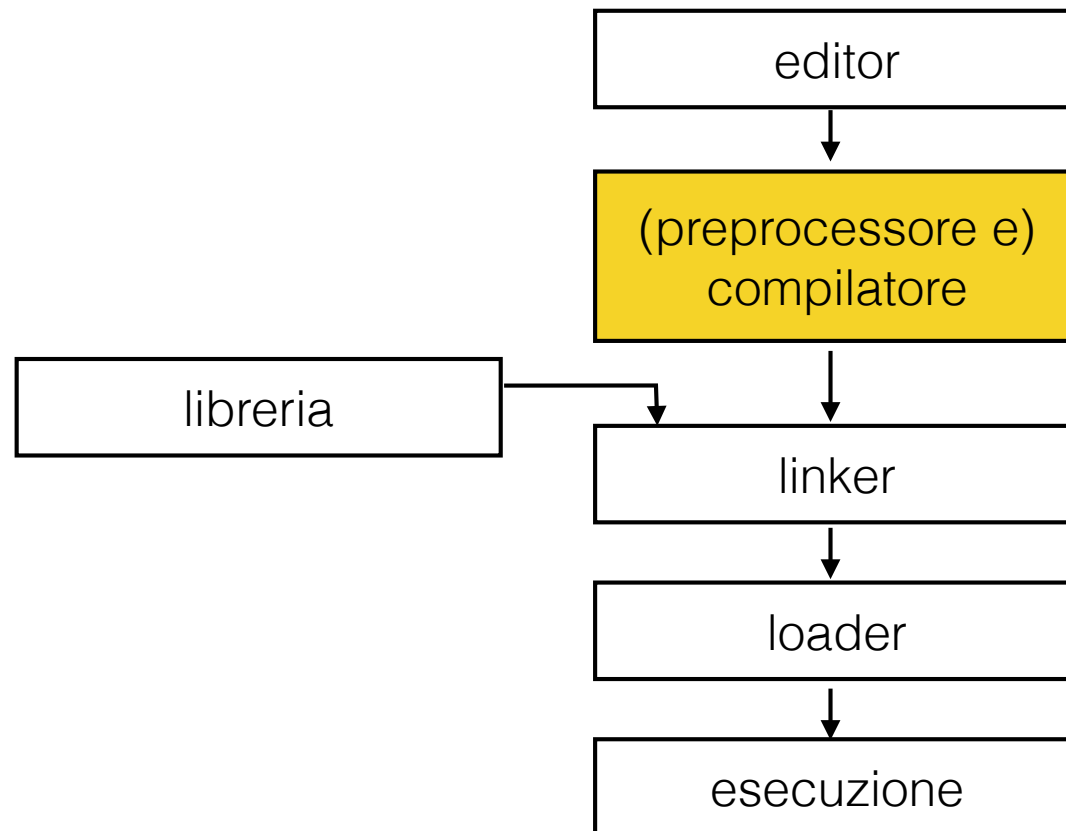
# elaborazione di un programma



usiamo un editor di testo per scrivere un programma, seguendo la sintassi del linguaggio di programmazione scelto. Il programma viene memorizzato (*salvato*) in un file di testo il cui nome deve avere un'estensione opportuna

[in C++ l'estensione è *.cpp*]

# elaborazione di un programma

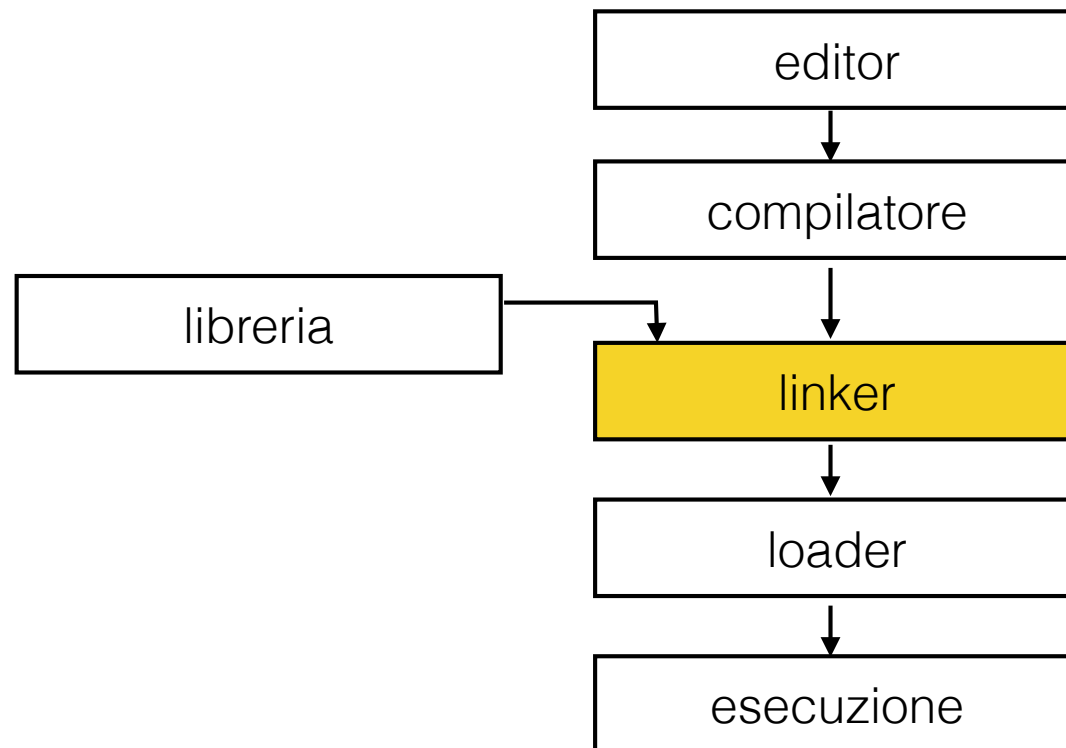


Il compilatore controlla che il programma sorgente *non contenga errori di sintassi* e successivamente effettua la traduzione in **codice oggetto**

**NB: verifica correttezza sintattica!**



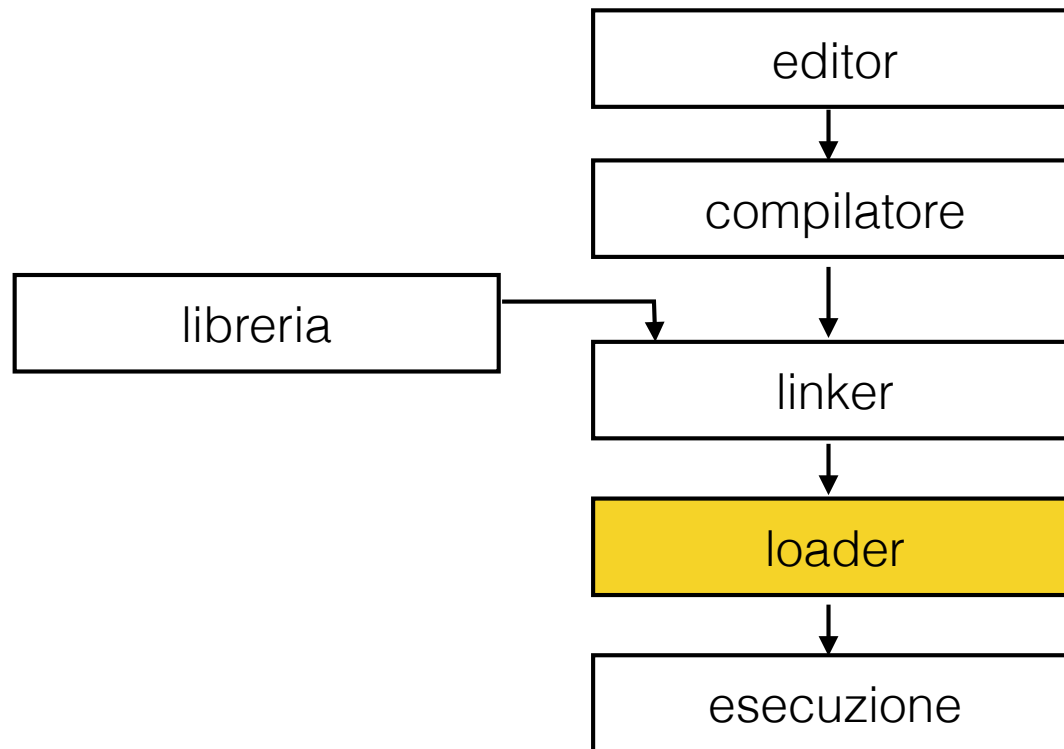
# elaborazione di un programma



il linker combina il codice oggetto con librerie o altri codici pre-compilati, crea l'eseguibile e lo memorizza su disco

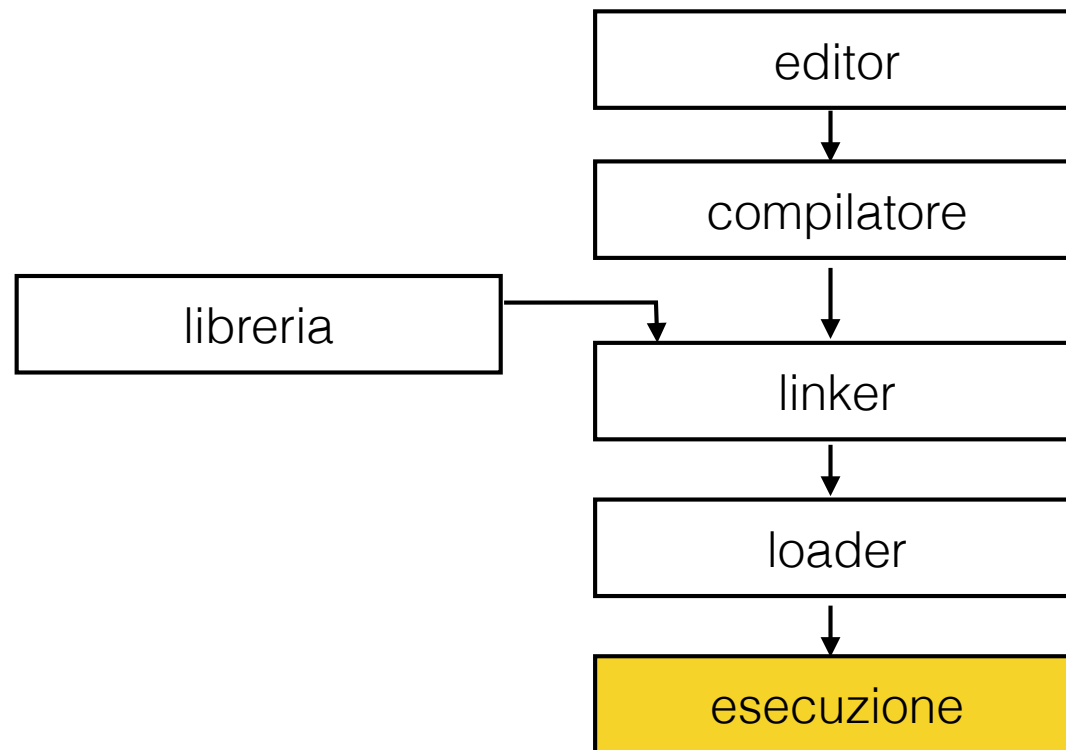
Ne ripareremo in seguito:  
principi di Programmazione Strutturata

# elaborazione di un programma



Il loader carica nella memoria principale il programma eseguibile

# elaborazione di un programma



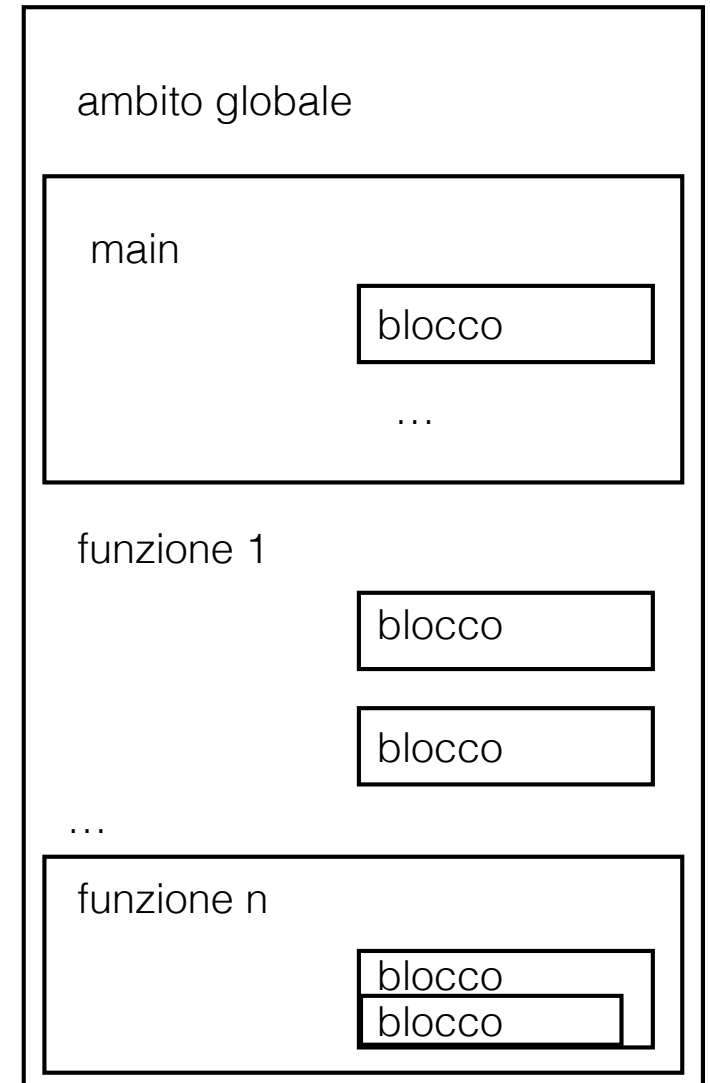
la CPU esegue  
un'istruzione per volta  
immagazzinando in  
memoria i nuovi dati  
ottenuti

# Programmi sequenziali

- considereremo programmi sequenziali, nei quali viene eseguita un'istruzione alla volta in sequenza
- tali programmi sono costituiti da un programma principale (il main in C e C++) più eventuali estensioni procedurali (funzioni)
- nella (buona) pratica le funzioni di solito costituiscono la maggior parte del codice di un programma strutturato

# Struttura dei programmi sequenziali

- i *blocchi* delimitano sia unità logiche di calcolo che ambiti di uso e “vita” delle variabili, governati da precise regole



struttura dei programmi C

# sistema operativo

- il sistema operativo (Operating System, OS) è il programma che viene caricato in fase di avvio del sistema (fase di *bootstrap*)
- l'OS continua a girare e a coordinare l'attività degli altri programmi finché il sistema è in funzione
- Esso può disporre l'esecuzione “contemporanea” di più programmi in esecuzione (*processi*) secondo uno schema di condivisione della memoria RAM

**noi assumeremo di trattare un programma alla volta, con la sua porzione di memoria**