

Gli elementi di un sistema di calcolo ram,cpu,alu,ir,cu,disp input,disp output,memoria secondaria,cpu. Spiegazione di tutte

Disegno ALU, a-b-c-o, input,output chi controlla cosa.

Control unit cosa fa(salti,if,esecuzione etc).

Von neumann idea così bene ancora usano questa l'architettura,/mobile no

Unità di controllo controlla tutto,esempio messaggio input,memoria.

processore una macchina che sa fare i calcoli senza dati non riesce a fare niente dobbiamo organizzare un deposito per il processore così pesca,le combina e produce e le rimette nella scatola gialla.

Processore pesca input voglio ottenere una certa informazione , processore si trova dentro e memoria principale anche , input,out e memoria secondaria si trova in periferia.

Organizzazione della memoria, le memorie sono fatte come vettori- vuol dire una sequenza in informazioni ordinate. 1-2-3-4 etc . tante informazioni dentro e ordinate macchina di von neumann, la prima e sempre numerata 0.i vettori che utilizziamo noi sarà sempre roba in cui diciamo voglio vettori in cui le informazioni sono .. in cui vuol dire da zero a n-1.

Flussi soldi dati. Controlli ordini che control unit manda.

Quando il programma che funzionano hanno bisogno della memoria e vedemo che nella memoria c'è tutto, tutto il resto della memoria potrà essere utilizzato per mantenere dati.

Struttura dei sistemi di calcolo. Livello più alto L5.

Traduzione,compilazione,compilatore.

Codice;sintassi,segni, importanza del codice come viene scritto difficoltà del compilatore, non è intelligente, etc.

“.cpp “estensione del linguaggio c++.

Librerie cosa sono. Le Collezioni risolvono dei sotto problemi che possiamo usare dentro il nostro programma.

```
#include <iostream> // per la libreria include//
int main(){
    std::cout<<"Hello world!\n";
    return 0; // reutrn 0 its true so the program works it c an be 1 as
well but the program wont finish //
}
```

Chat gpt appunti ordinati

Elementi di un sistema di calcolo

Componenti principali

- **CPU (Central Processing Unit)** → il “cervello” del computer. Formata da:
 - **ALU (Arithmetic Logic Unit)** → esegue operazioni aritmetiche (somma, sottrazione, moltiplicazione) e logiche (AND, OR, NOT).

- **CU (Control Unit)** → unità di controllo, coordina il funzionamento di tutti i componenti.
- **IR (Instruction Register)** → contiene l'istruzione che la CPU deve eseguire in quel momento.
- **RAM (Random Access Memory)** → memoria volatile, conserva i dati e le istruzioni temporaneamente mentre il computer lavora.
- **Memoria secondaria** → hard disk, SSD, memorie permanenti che mantengono i dati anche a computer spento.
- **Dispositivi di input** → tastiera, mouse, microfono.
- **Dispositivi di output** → monitor, stampante, casse.

● (Aggiunta: oggi ci sono anche dispositivi *I/O misti*, come touchscreen che sono input e output insieme).

ALU (Arithmetic Logic Unit)

- **Input:** due operandi (A e B) + un segnale di controllo (C) che dice quale operazione eseguire.
- **Output (O):** il risultato dell'operazione.
- **L'ALU non decide da sola cosa fare** → viene controllata dalla **Control Unit**.

● (Aggiunta: l'ALU può anche generare dei **flag**, cioè segnali che dicono se il risultato è zero, negativo, se c'è overflow, ecc.)

Control Unit (CU)

- Coordina tutte le attività della CPU.
- Ciclo di esecuzione (Fetch-Decode-Execute):
 1. **Fetch:** prende l'istruzione dalla memoria.
 2. **Decode:** interpreta l'istruzione.

3. **Execute**: esegue l'istruzione tramite ALU o altre unità.

- Gestisce **salti e condizioni** (es. IF, cicli).
 - Controlla i **flussi di dati** tra CPU, memoria e periferiche.
-

Architettura di Von Neumann

- Concetto introdotto da **John von Neumann (1945)**.
- Tutte le istruzioni e i dati sono memorizzati nella stessa memoria.
- La CPU pesca dalla memoria, elabora e scrive nuovamente in memoria.
- Ancora oggi la maggior parte dei computer usa questa architettura.

● (Aggiunta: nei dispositivi mobili spesso si usa una variante chiamata *Harvard modificata*, in cui i dati e le istruzioni possono essere in memorie separate per velocizzare).

Memoria

- Organizzata come un **vettore** (array): ogni cella ha un **indirizzo numerico** che parte da 0 fino a N-1.
 - Dentro ci sono sia i **dati** sia i **programmi**.
 - La CPU accede tramite indirizzi, non tramite "nomi".
 - Es.: indirizzi 0, 1, 2, 3... contengono byte consecutivi.
-

Flussi nel sistema di calcolo

- **Flusso dei dati** → dati che vanno da input → CPU → memoria → output.
- **Flusso di controllo** → ordini inviati dalla Control Unit per coordinare tutto.

Compilazione e codice

- Il processore capisce solo **linguaggio macchina** (0 e 1).
- Noi scriviamo in **linguaggi di alto livello** (C, C++, Python...).
- Serve un **compilatore** → traduce il codice sorgente in linguaggio macchina.
- **Codice sorgente**: contiene istruzioni scritte dal programmatore.
- **Sintassi**: regole del linguaggio (es. parentesi, punto e virgola).

Librerie

- Raccolte di **funzioni e strumenti già pronti**.
- Includendole, possiamo usare funzioni senza doverle riscrivere.
- Esempio: in C++ la libreria **iostream** serve per gestire input/output (cin, cout).


Esempio in C++

```
#include <iostream>    // Libreria standard per input/output (cout, cin)

int main() {
    std::cout << "Hello world!\n"; // Stampa a schermo il messaggio
    return 0;
}
```

Spiegazione riga per riga

- `#include <iostream>` → include la libreria standard di input/output.
- `int main()` → funzione principale, punto di ingresso del programma.

- `std::cout` → standard output, serve per scrivere su schermo.
- `<<` → operatore di inserimento (manda il testo verso cout).
- `"Hello world!\n"` → stringa che viene stampata, `\n` va a capo.
- `return 0;` → indica che il programma è terminato correttamente.
-  (Aggiunta: `std` è lo *namespace standard* di C++, dentro cui sono definite tutte le funzioni e classi principali).