Elementi di Informatica e Applicazioni Numeriche T

Architettura di un Elaboratore

Elaboratore come Macchina

Ricordate la nostra definizione di elaboratore?

Un elaboratore è una entità che può:

- Memorizzare informazioni
- Eseguire sul tali informazioni alcune operazioni elementari

Elaboratore come Macchina

Per questa lezione la modifichiamo un po'

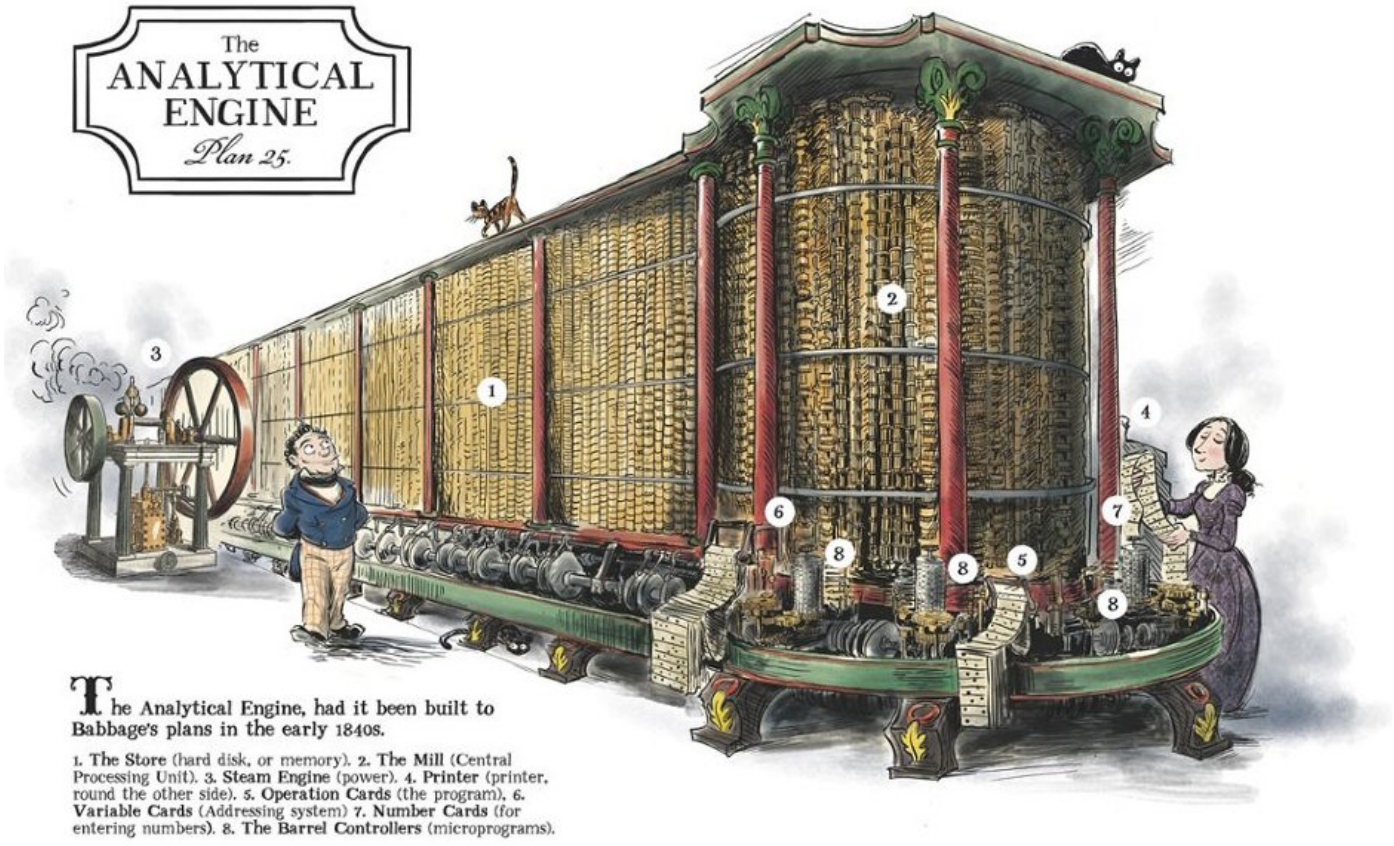
Un elaboratore è una macchina che può:

- Memorizzare informazioni
- Eseguire sul tali informazioni alcune operazioni elementari

- Ci concentreremo sull'elaboratore come macchina...
- ...in particolare su quello che chiamiamo computer (o calcolatore)

Cominciamo con un po' di storia...

Elaboratori Storici - Analytical Engine

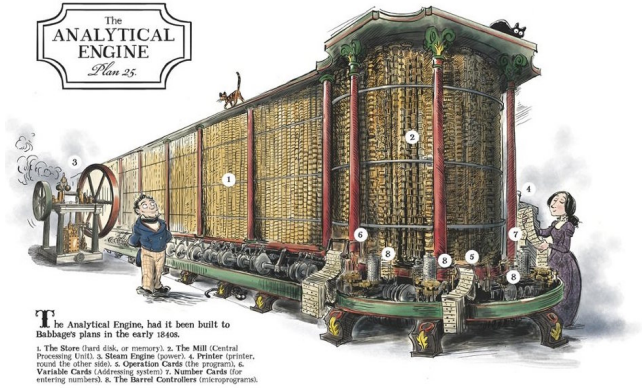


The Analytical Engine, had it been built to Babbage's plans in the early 1840s.

1. **The Store** (hard disk, or memory). 2. **The Mill** (Central Processing Unit). 3. **Steam Engine** (power). 4. **Printer** (printer, round the other side). 5. **Operation Cards** (the program). 6. **Variable Cards** (Addressing system) 7. **Number Cards** (for entering numbers). 8. **The Barrel Controllers** (microprograms).

Elaboratori Storici - Analytical Engine

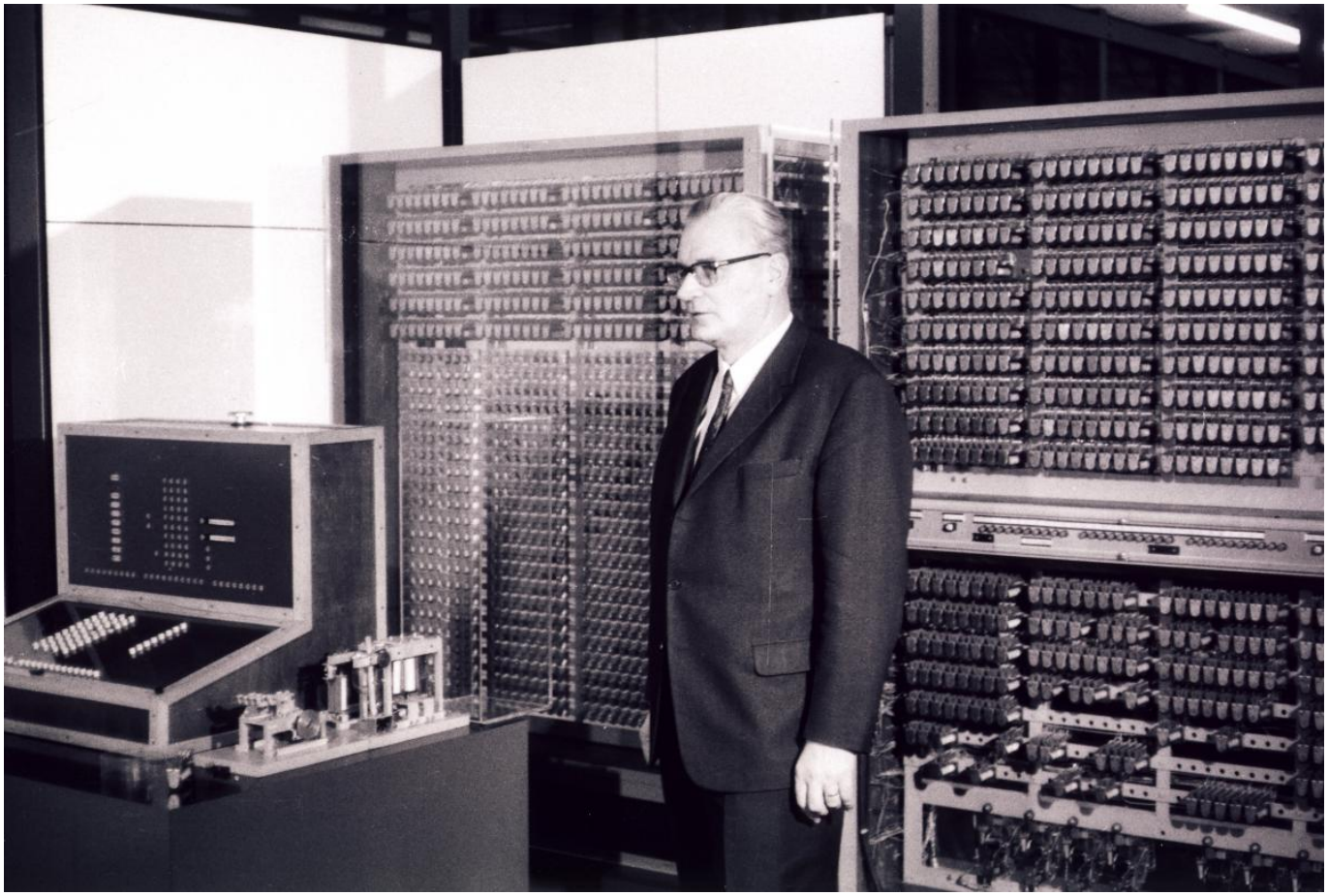
L'Analytical Engine di Charles Babbage (1837)...



Può essere considerato il primo vero computer mai ideato

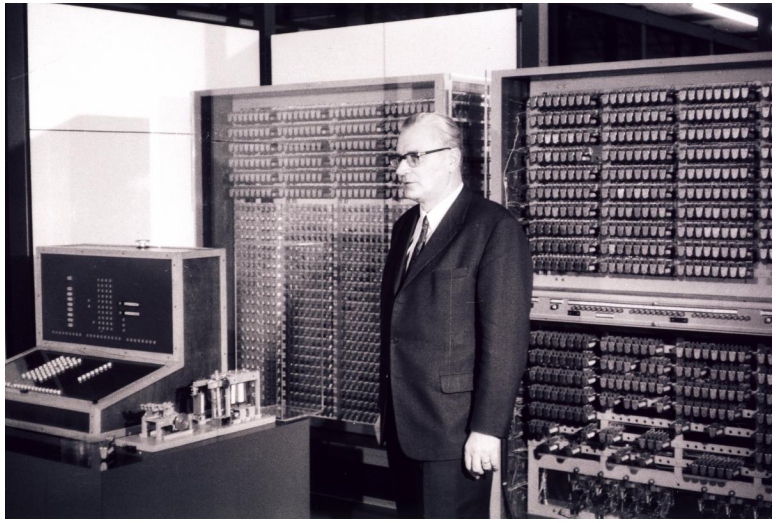
- Supportava cicli, istruzioni condizionali ed operazioni aritmetiche
- Funzionava grazie ad un motore a vapore
- Non è stato mai costruito (la tecnologia era inadeguata)

Elaboratori Storici - Z3



Elaboratori Storici - Z3

Il primo computer funzionante è stato lo Z3 (1941):



- Progettato dal Konrad Zuse, a Berlino
- Era un macchina elettromeccanica (basata su relays)
- Sviluppo interrotto durante la guerra per taglio di fondi

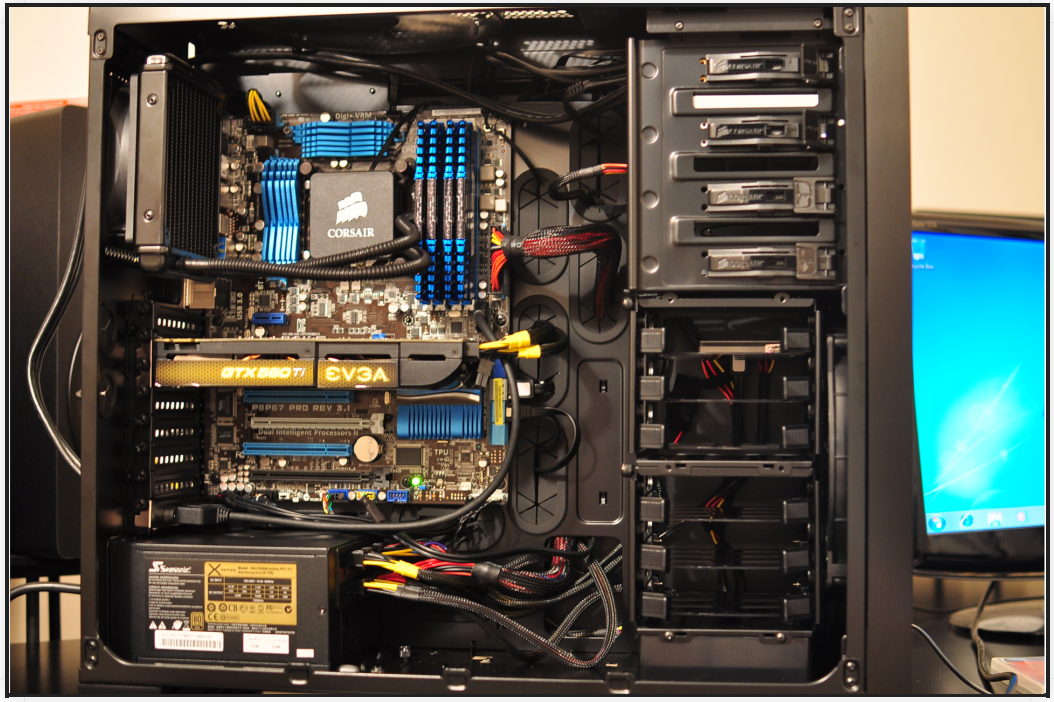
Elaboratori Moderni

Oggi un computer si presenta così (più o meno)



Elaboratori Moderni

In questa lezione ci interessa però come sia fatto all'interno



Elaboratori Moderni

In questa lezione ci interessa però come sia fatto all'interno



Questo componente si chiama scheda madre

Elaboratori Moderni

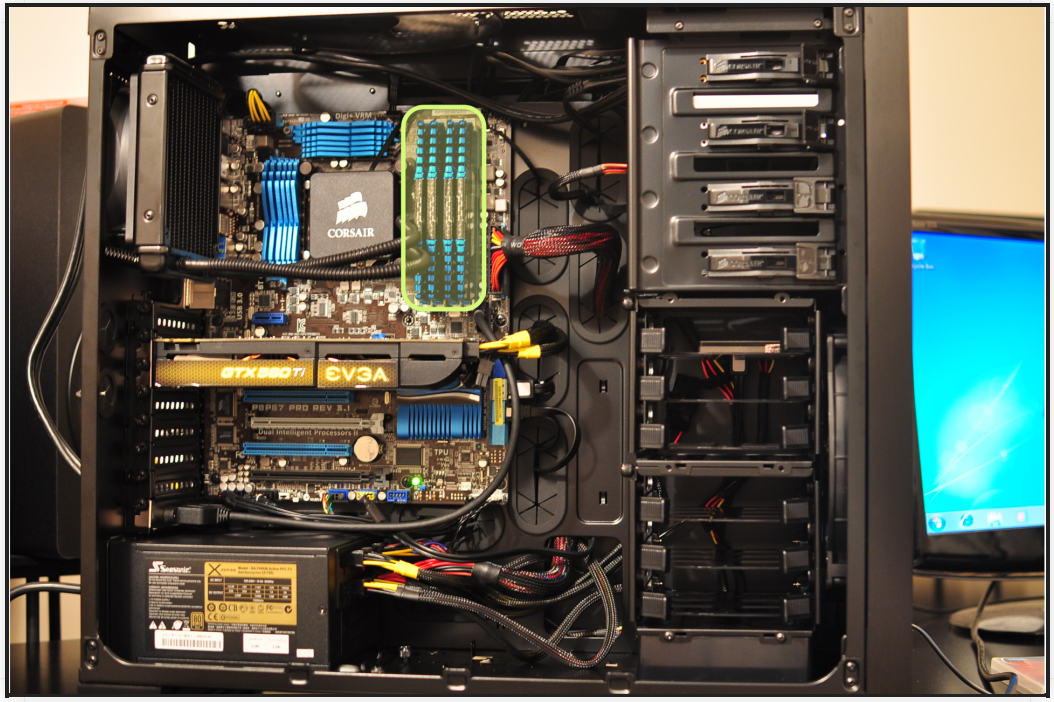
In questa lezione ci interessa però come sia fatto all'interno



Qui c'è la Central Processing Unit (CPU)

Elaboratori Moderni

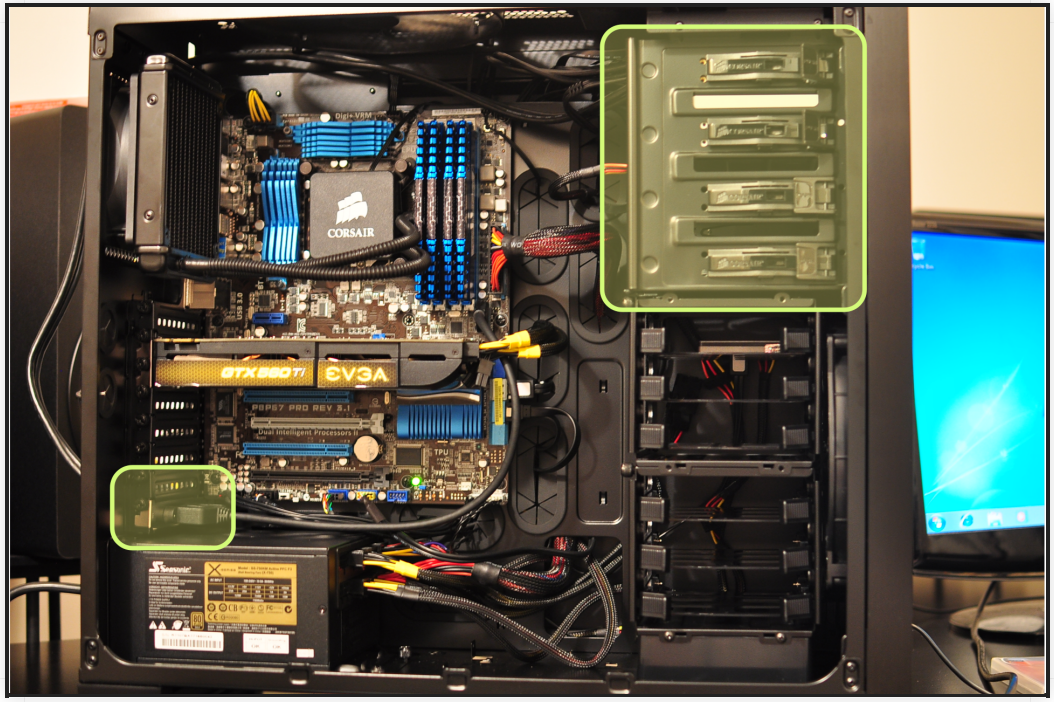
In questa lezione ci interessa però come sia fatto all'interno



Questa è la memoria centrale (in particolare la RAM)

Elaboratori Moderni

In questa lezione ci interessa però come sia fatto all'interno



Qui vedete alcune porte e periferiche (disco rigido e lettore dischi ottici)

Elaboratori Moderni

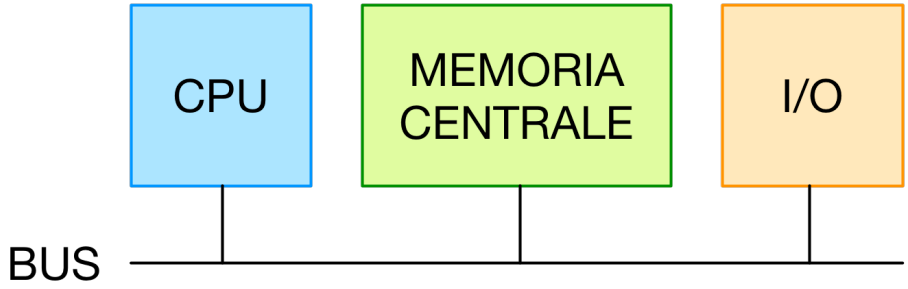
In questa lezione ci interessa però come sia fatto all'interno



Questa è una Graphics Processing Unit (GPU)

Macchina di Von Neumann

Dal punto di vista logico, lo schema è grossomodo il seguente:



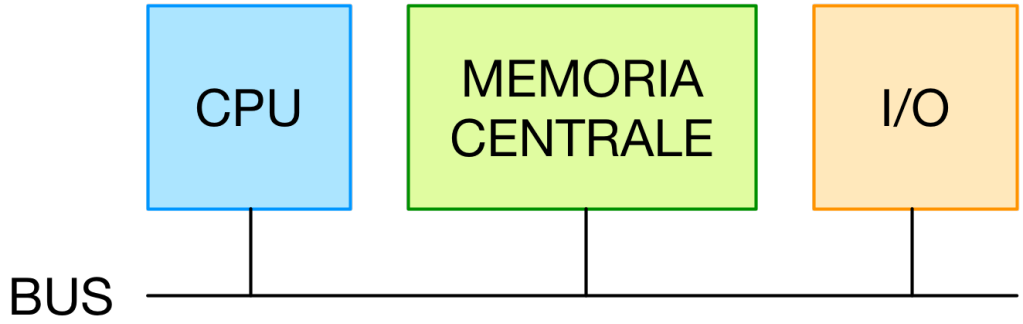
Ci sono tre componenti principali:

- La CPU
- La memoria centrale
- E le periferiche di I/O

Connesse da un fascio di connessioni elettriche chiamato bus

Macchina di Von Neumann

Dal punto di vista logico, lo schema è grossomodo il seguente:



Questa architettura è nota come macchina di Von Neumann

- Il ruolo del BUS è svolto dalla scheda madre...
- ...Che è (parecchio) più complessa di "un fascio di connessioni"...
- ...Ma consente sempre agli altri componenti di comunicare

Central Processing Unit (CPU)



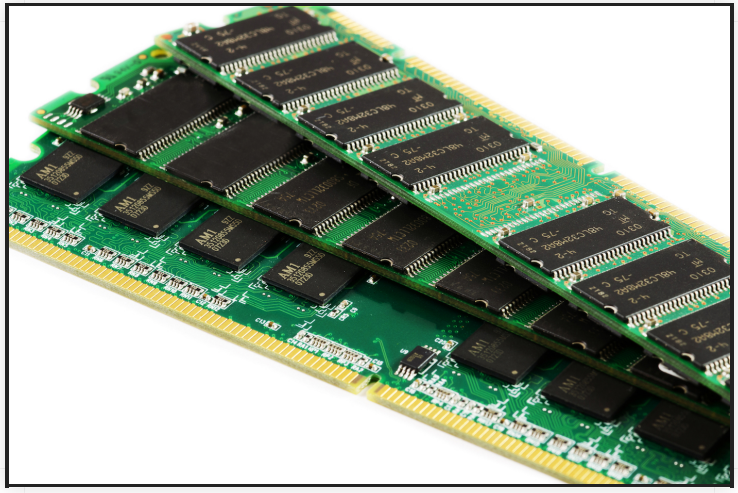
La CPU è un componente HW in grado di eseguire istruzioni

- È il componente più importante di un computer
- Viene chiamata anche processore o microprocessore

Oggi è spesso affiancata da processori dedicati (acceleratori)

- Es. GPU per operazioni grafiche

Memoria Centrale



La memoria centrale (RAM e ROM) ha il compito di

- Memorizzare le istruzioni dei programmi in esecuzione
- Memorizzare i dati su cui essi stanno operando

Ha dimensioni limitate, ma vi si accede molto velocemente

Dispositivi di Input/Output





I dispositivi di I/O si chiamano anche periferiche e comprendono:

- Tutti gli altri dispositivi (monitor, tastiera, stampanti...)
- Le interfacce di comunicazione (rete ethernet, rete wireless, USB...)

Una classe di periferiche importante è data dalla memoria di massa...

Dispositivi di I/O: Memoria di Massa



La memoria di massa comprende periferiche che

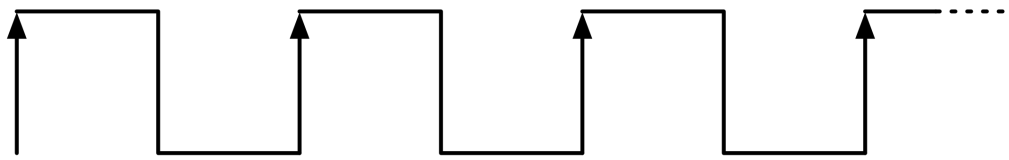
- Memorizzano grosse quantità di dati
- In modo persistente
- È molto più capiente della memoria centrale ($\times 10 - \times 10,000$)...
- ...Ma anche molto più lenta ($\times 1,000 - \times 100,000$)

Come Funziona un Computer?

Vediamo giusto qualche dettaglio

Tutto comincia con il "clock"

- Un circuito sulla scheda madre genera un'onda quadra
- Di solito alto = 1.25 Volt, basso = 0 Volt

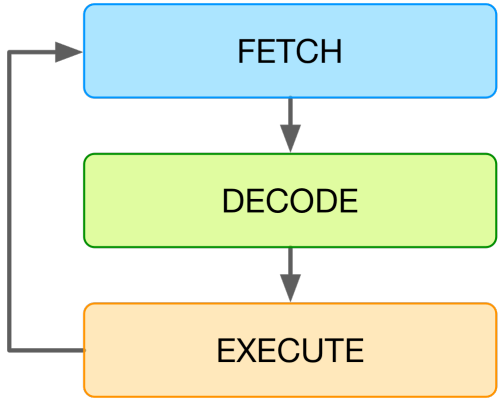


Questo segnale si chiama clock

- La frequenza del clock si misura in Hz (1/sec)
- E.g. 2.3GHz = 2,300,000,000 oscillazioni al secondo

Ciclo Fetch/Decode/Execute

- Ad ogni fronte del clock...
- ...La CPU svolge un passo del ciclo di elaborazione delle istruzioni:



Si chiama ciclo fetch/decode/execute

- Comprende tre fasi
- Descrive l'attività principale svolta dalla CPU

Ciclo Fetch/Decode/Execute

Fase FETCH:

- La CPU accede alla memoria centrale...
- ...e recupera una istruzione

Fase DECODE:

- In base al contenuto dell'istruzione...
- ...l'HW viene predisposto alla sua esecuzione

Fase EXECUTE:

- L'istruzione viene eseguita

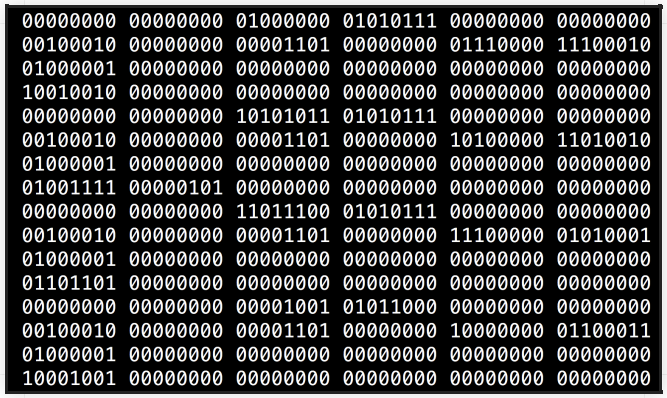
A prima vista, somiglia un po' a quello che fa Matlab...

Rappresentazione Binaria

...Solo che le istruzioni si presentano così:

```
00000000 00000000 01000000 01010111 00000000 00000000
00100010 00000000 00001101 00000000 01110000 11100010
01000001 00000000 00000000 00000000 00000000 00000000
10010010 00000000 00000000 00000000 00000000 00000000
00000000 00000000 10101011 01010111 00000000 00000000
00100010 00000000 00001101 00000000 10100000 11010010
01000001 00000000 00000000 00000000 00000000 00000000
01001111 00000101 00000000 00000000 00000000 00000000
00000000 00000000 11011100 01010111 00000000 00000000
00100010 00000000 00001101 00000000 11100000 01010001
01000001 00000000 00000000 00000000 00000000 00000000
01101101 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00001001 01011000 00000000 00000000
00100010 00000000 00001101 00000000 10000000 01100011
01000001 00000000 00000000 00000000 00000000 00000000
10001001 00000000 00000000 00000000 00000000 00000000
```

Rappresentazione Binaria



Ogni informazione su un computer
è rappresentata utilizzando solo le cifre 0 e 1

- Infatti, è facile associarle a segnali elettrici (e.g. 0 e 1.25 Volt)
- Questa rappresentazione si chiama codice binario

Elementi di Informatica e Applicazioni Numeriche T

Rappresentazione dei Numeri Naturali

Codice Binario

Quali informazioni possiamo rappresentare con il codice binario?

- Di fatto, tutte!
- Noi ci focalizzeremo sui numeri

Altri tipi di informazione sono rappresentabili mediante numeri

- E.g. immagine in B/W = matrice di numeri (luminosità)
- E.g. suono = sequenza di numeri (ampiezza dell'onda sonora)

In particolare, vedremo come rappresentare i numeri naturali

- Vedrete come sono gestiti i reali nel corso di analisi numerica

Rappresentazione dei Numeri Naturali

Consideriamo per esempio:

1265

Che cos'è?

Rappresentazione dei Numeri Naturali

Consideriamo per esempio:

1265

Che cos'è?

- Non un numero, ma una **rappresentazione**...
- ...Che corrisponde al numero:

$$\mathbf{1} \times 10^3 + \mathbf{2} \times 10^2 + \mathbf{6} \times 10^1 + \mathbf{5} \times 10^0$$

- Utilizziamo il numero 10 come base
- I simboli (i.e. cifre) 0-9 per denotare i numeri naturali minori di 10

Diamo un peso alle cifre in base alla loro posizione

Notazione Posizionale

Il metodo si chiama notazione posizionale

- Data una base B, le cifre vanno da 0 a $B-1$)
- Tra rappresentazioni e valori vale la corrispondenza:

$$d_{n-1}\, d_{n-2}\, \ldots\, d_1\, d_0 \longrightarrow \sum_{k=0}^{n-1} d_k\, B^k$$

Un paio di commenti importanti:

- Possiamo esprimere qualunque numero in qualunque base
- In base due, ci sono solo due cifre: `0` e `1`, quelle del calcolatore!
- Una cifra binaria si chiama bit (binary digit)

Rappresentazione in Base Due

Un esempio in base due (si riconosce dal $_2$ a pedice):

Consideriamo il numero in base due:

$$111_2$$

A cosa corrisponde?

Rappresentazione in Base Due

Un esempio in base due (si riconosce dal $_2$ a pedice):

Consideriamo il numero in base due:

$$111_2$$

A cosa corrisponde?

$$\mathbf{1} \times 2^2 + \mathbf{1} \times 2^1 + \mathbf{1} \times 2^0$$

Il risultato è:

$$6 + 4 + 1 = 7$$

Rappresentazione in Base Due

Un altro esempio in base due (si riconosce dal $_2$ a pedice):

Consideriamo il numero in base due:

$$1100_2$$

A cosa corrisponde?

Rappresentazione in Base Due

Un altro esempio in base due (si riconosce dal $_2$ a pedice):

Consideriamo il numero in base due:

$$1100_2$$

A cosa corrisponde?

$$\mathbf{1} \times 2^3 + \mathbf{1} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{0} \times 2^0$$

Il risultato è:

$$8 + 4 = 12$$

Rappresentazione in Base Due

Un terzo esempio in base due:

There are only
10
types of people:
those who understand binary
and those who don't

Numeri Rappresentabili

Limiti pratici della rappresentazione:

- Un calcolatore ha una quantità finita di memoria
- Quindi possiamo usare solo un numero finito di cifre

In particolare, se usiamo N cifre:

- Minimo (tutte le cifre a `0`) $\longrightarrow 0$
- Massimo (tutte le cifre a `1`) $\longrightarrow B^N - 1$

Qualche esempio:

- Con 16 bit: $0..65,535$
- Con 64 bit: $0..1.84467441 \times 10^{19}$

Conversione di Base

Come si ottiene la rappresentazione di un numero in base B?

- Si utilizza il metodo delle divisioni ripetute...
- ...che si basa sulla divisione intera

Divisione intera

Dato un numero x ed un divisore B, la divisione intera denota:

- Un quoziente intero q
- Un resto (o modulo) intero $r < B$
- Tali che: $x = q\,B + r$

Sembra chissà che, ma è la divisione delle elementari :-)

Conversione di Base

Metodo delle divisioni ripetute

Questa è la descrizione del metodo in pseudo-codice:

$k = 0$
while $x > 0$
    d_k = resto della divisione intera x/B
    x = quoziente della divisione intera x/B
    $k = k + 1$
end

- Si divide ripetutamente x per la base
- Ad ogni passo si ottiene la k-ma cifra, i.e. d_k

Conversione di Base - Un Esempio

Convertiamo il numero 42 in base due:

x	x/2	x mod 2	
42	21	0	$\rightarrow d_0 = 0$
21	10	1	$\rightarrow d_1 = 1$
10	5	0	$\rightarrow d_2 = 0$
5	2	1	$\rightarrow d_3 = 1$
2	1	0	$\rightarrow d_4 = 0$
1	0	1	$\rightarrow d_5 = 1$

Quindi 42 equivale a 101010_2 !

Operazioni con i Numeri Naturali

Le operazioni in base 2 si fanno come quelle in base 10

Basta ricordarsi che:

- Le cifre sono solo 0 e 1
- Il riporto può essere solo 0 o 1

Come esempio, vediamo una somma:

$$\begin{array}{cccc} 1 & 0 & 1 & 0 \\ & 1 & 1 & 1 \\ \hline \end{array} \quad \begin{array}{c} + \\ = \end{array}$$

Operazioni con i Numeri Naturali

Le operazioni in base 2 si fanno come quelle in base 10

Basta ricordarsi che:

- Le cifre sono solo 0 e 1
- Il riporto può essere solo 0 o 1

Come esempio, vediamo una somma:

$$\begin{array}{cccc} 1 & 0 & 1 & 0 \\ & 1 & 1 & 1 \\ \hline & & & 1 \end{array} \quad \begin{array}{c} + \\ = \end{array}$$

Operazioni con i Numeri Naturali

Le operazioni in base 2 si fanno come quelle in base 10

Basta ricordarsi che:

- Le cifre sono solo 0 e 1
- Il riporto può essere solo 0 o 1

Come esempio, vediamo una somma:

$$\begin{array}{cccccc} & & 1 & & & \\ 1 & 0 & 1 & 0 & & + \\ & & 1 & 1 & 1 & = \\ \hline & & & 0 & 1 & \end{array}$$

Operazioni con i Numeri Naturali

Le operazioni in base 2 si fanno come quelle in base 10

Basta ricordarsi che:

- Le cifre sono solo 0 e 1
- Il riporto può essere solo 0 o 1

Come esempio, vediamo una somma:

$$\begin{array}{cccccc} & 1 & 1 & & & \\ 1 & 0 & 1 & 0 & & + \\ & 1 & 1 & 1 & & = \\ \hline & 0 & 0 & 1 & & \end{array}$$

Operazioni con i Numeri Naturali

Le operazioni in base 2 si fanno come quelle in base 10

Basta ricordarsi che:

- Le cifre sono solo 0 e 1
- Il riporto può essere solo 0 o 1

Come esempio, vediamo una somma:

$$\begin{array}{cccccc} 1 & 1 & 1 & & & \\ & 1 & 0 & 1 & 0 & + \\ & & 1 & 1 & 1 & = \\ \hline & 0 & 0 & 0 & 1 & \end{array}$$

Operazioni con i Numeri Naturali

Le operazioni in base 2 si fanno come quelle in base 10

Basta ricordarsi che:

- Le cifre sono solo 0 e 1
- Il riporto può essere solo 0 o 1

Come esempio, vediamo una somma:

$$\begin{array}{ccccc} 1 & 1 & 1 & & \\ & 1 & 0 & 1 & 0 \\ & & 1 & 1 & 1 \\ \hline 1 & 0 & 0 & 0 & 1 \end{array} \quad \begin{array}{l} \\ + \\ = \\ \\ \end{array}$$

Elementi di Informatica e Applicazioni Numeriche T

Numeri Decimali

Numeri Decimali in Base Due

- Abbiamo visto come rappresentare numeri naturali
- Non vedremo come rappresentare i numeri interi (in $\mathbb{Z}$)

Ci occuperemo invece dei numeri decimali

La notazione decimale è un modo per rappresentare numeri frazionari

- Si presentano per esempio come 5.375
- O più in generale nella forma:

$$\underbrace{d_{n-1}d_{n-2}\ldots d_0}_{\text{parte intera}} . \underbrace{d_{-1}d_{-2}\ldots d_{-m}}_{\text{parte frazionaria}}$$

- Con n cifre nella parte intera ed m cifre nella parte frazionaria

Numeri Decimali in Base Due

Data una base B, la rappresentazione:

$$d_{n-1}d_{n-2}\ldots d_0 \bullet d_{-1}d_{-2}\ldots d_{-m}$$

Corrisponde a:

$$\sum_{k=-m}^{n-1} d_k\, B^k$$

- È sempre la notazione posizionale...
- ...ma stavolta gli esponenti possono essere anche negativi

Per esempio, 5.375 viene interpretato come:

$$\mathbf{5}\times 10^{0} + \mathbf{3}\times 10^{-1} + \mathbf{7}\times 10^{-2} + \mathbf{5}\times 10^{-5}$$

Conversione di Base

Come convertire in base 2 di un numero decimale?

- La base B è sempre > 1
- La parte intera è sempre ≥ 1
- La parte frazionaria è sempre < 1

Quindi, qualsiasi sia B:

- La parte intera resta sempre intera
- La parte frazionaria resta sempre frazionaria

Vuol dire che le possiamo convertire separatamente!

La parte intera la sappiamo convertire: e quella frazionaria?

Conversione di Base

Per la parte fraz. usiamo il metodo delle moltiplicazioni ripetute

Il metodo funziona così (in pseudo-codice):

$k = -1$
while $x > 0$
    $y = x \times B$
    $d_k =$ parte intera di y
    $x =$ parte frazionaria di y
    $k = k - 1$
end

- Si moltiplica ripetutamente x per la base
- Ad ogni passo, la parte intera corrisponde ad una cifra d_k

Conversione di Base - Un Esempio

Convertiamo il numero 0.375 in base due:

x	$x \times 2$	$\lfloor x \times 2 \rfloor$	
0.375	0.75	0	$\rightarrow d_{-1} = 0$
0.75	1.5	1	$\rightarrow d_{-2} = 1$
0.50	1.0	1	$\rightarrow d_{-3} = 1$
0.0			

Quindi 0.375 corrisponde a 0.011_2

- $\lfloor \cdot \rfloor$ rappresenta l'arrotondamento per difetto
- Il numero 5.375 (per esempio) corrisponde a 101.011_2

Limiti della Rappresentazione Decimale

Guardiamo un attimo la formula per valutare una rappresentazione

$$\sum_{k=-m}^{n-1} d_k \, B^k$$

- Costruisce un numero sommando potenze di B...
- ...Per $k < 0$ queste sono frazioni di B

Alcune conseguenze:

- Possiamo rappresentare solo numeri razionali (in $\mathbb{Q}$)
- Se la parte frazionaria non è esprimibile utilizzando potenze di B...
- ...Allora la rappresentazione è infinita e periodica

Numeri Periodici: un Esempio

Il numero $1/3$ è periodico in base 10 ed anche in base 2

x	$x \times 2$	$\lfloor x \times 2 \rfloor$	
$1/3$	$2/3$	0	$\rightarrow d_{-1} = 0$
$2/3$	$4/3$	1	$\rightarrow d_{-2} = 1$
$1/3$	$2/3$	0	$\rightarrow d_{-3} = 0$
$2/3$	$\ldots$		

Il risultato è $0.\overline{01}_2$

- La "barra" indica le cifre periodiche

Numeri Periodici

Come rappresentare un numero periodico sul calcolatore?

Soluzione 1: "tener traccia" della parte periodica (come in matematica)

E.g. $0.\overline{3}$, $0.\overline{01}_2$

- Così facendo, però fare le operazioni diventa molto complicato
- Di fatto, questa soluzione non si usa quasi mai

Soluzione 2: Troncare, una volta usate tutte le cifre disponibili:

E.g., con quattro cifre: 0.3334 , 0.0101_2

- In pratica, questa rappresentazione approssimata è usatissima
- **Attenzione:** un numero può essere periodico solo in certe basi

Numeri Periodici Solo in Determinate Basi

E.g. il numero 0.1 non è periodico in base 10 ma lo è base 2

x	$x \times 2$	$\lfloor x \times 2 \rfloor$	
0.1	0.2	0	$\rightarrow d_{-1} = 0$
0.2	0.4	0	$\rightarrow d_{-2} = 0$
0.4	0.8	0	$\rightarrow d_{-3} = 0$
0.8	1.6	1	$\rightarrow d_{-4} = 1$
0.6	1.2	1	$\rightarrow d_{-5} = 1$
0.2	…		

Il risultato è $0.\overline{00011}_2$

Elementi di Informatica e Applicazioni Numeriche T

Sistema Operativo

Gestione della

- Abbiamo visto come si possono rappresentare infomazioni…
- …ma gestire un calcolatore resta una faccenda complicata!

Una possibile soluzione

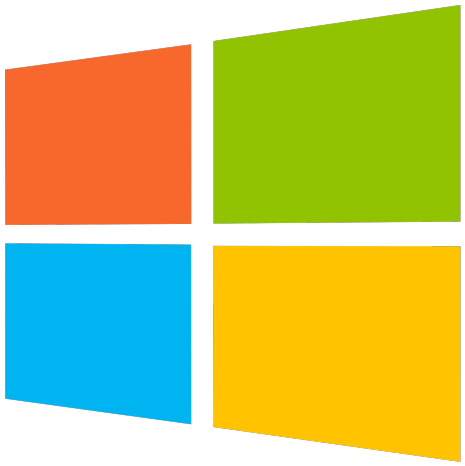
Scrivere dei programmi (componenti SW) che:

- Possono essere utilizzabili da altri programmi
- Permettono di effettuare operazioni complesse…
- …Sulle risorse del calcolatore

Questa collezione dei componenti SW che facilita la gestione del calcolatore si chiama Sistema Operativo

Esempi di Sistema Operativo

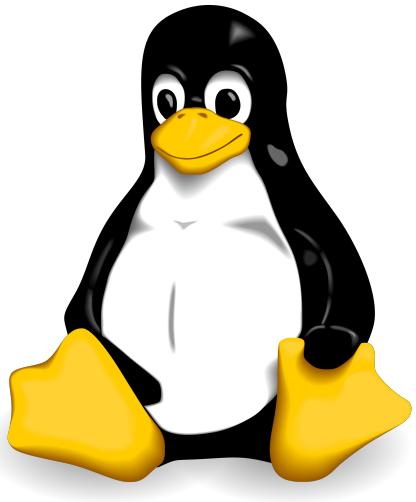
Microsoft Windows



Apple macOS



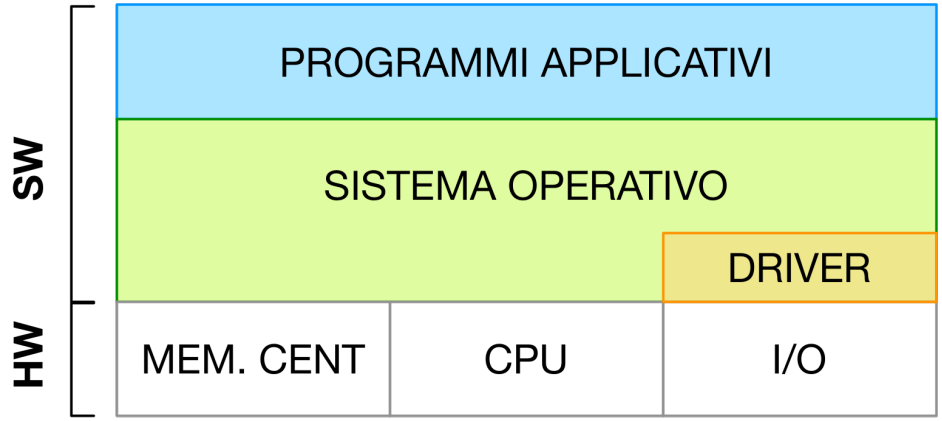
Linux



Organizzazione del Software a Livelli

Il SO fornisce le basi per lo sviluppo e l'esecuzione di altri programmi

Conseguenza: il SW su un calcolatore è organizzato a livelli

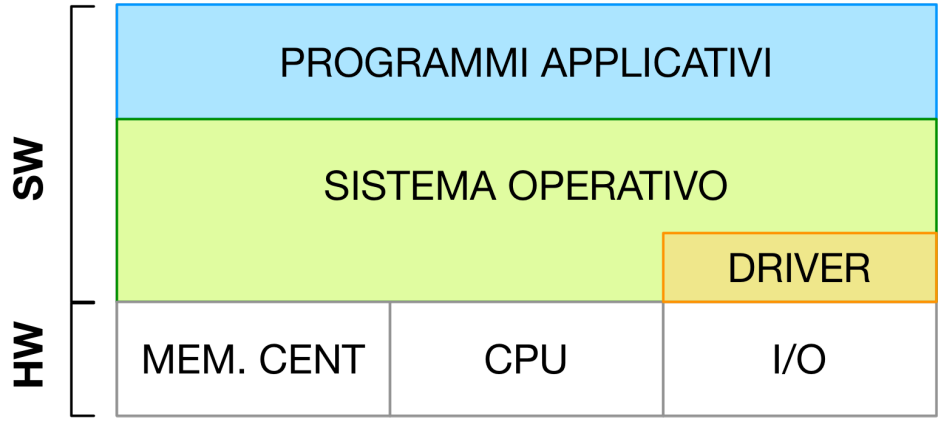


- I programmi applicativi sono i programmi "normali"
- Si basano sulle funzionalità offerte dal sistema operativo

Organizzazione del Software a Livelli

Il SO fornisce le basi per lo sviluppo e l'esecuzione di altri programmi

Conseguenza: il SW su un calcolatore è organizzato a livelli



- I driver di dispositivo sono altri componenti SW
- Permettono al SO di interagire con alcune periferiche
  - Es. scanner e stampanti, mouse...

Compiti del Sistema Operativo

Il sistema operativo ha molti compiti

Ma noi ne accenneremo solo tre!

- L'interazione con l'utente
- La gestione dei processi
- La gestione della memoria di massa

I corrispondenti programmi che se ne occupano sono:

- L'interfaccia utente
- Il kernel
- Il file system

Interfaccia Utente di un SO

Si chiama interfaccia di un SO un programma che permette di interagire con l'utente

Ci sono due grandi categorie di interfacce

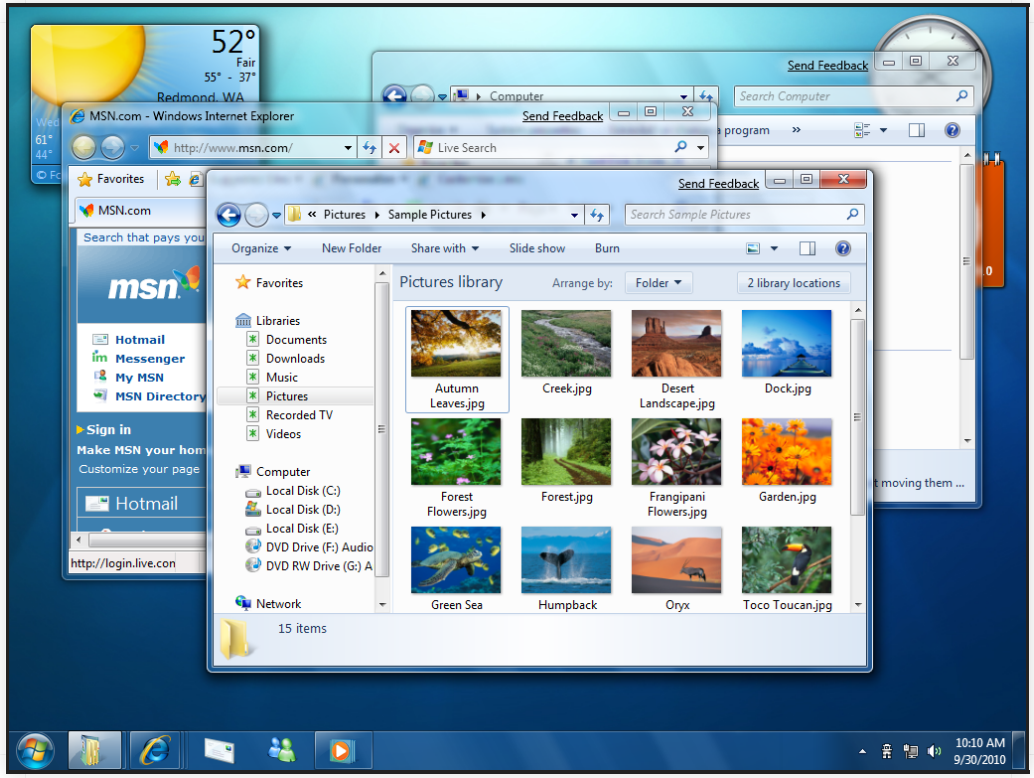
- Interfacce testuali: sono degli interpreti di comandi
- Interfacce grafiche: tipicamente basata su "finestre" e "click"

Windows, OS X e Linux le offrono entrambe

- Linux enfatizza l'interfaccia testuale
- OS X e (soprattutto) Win enfatizzano quella grafica

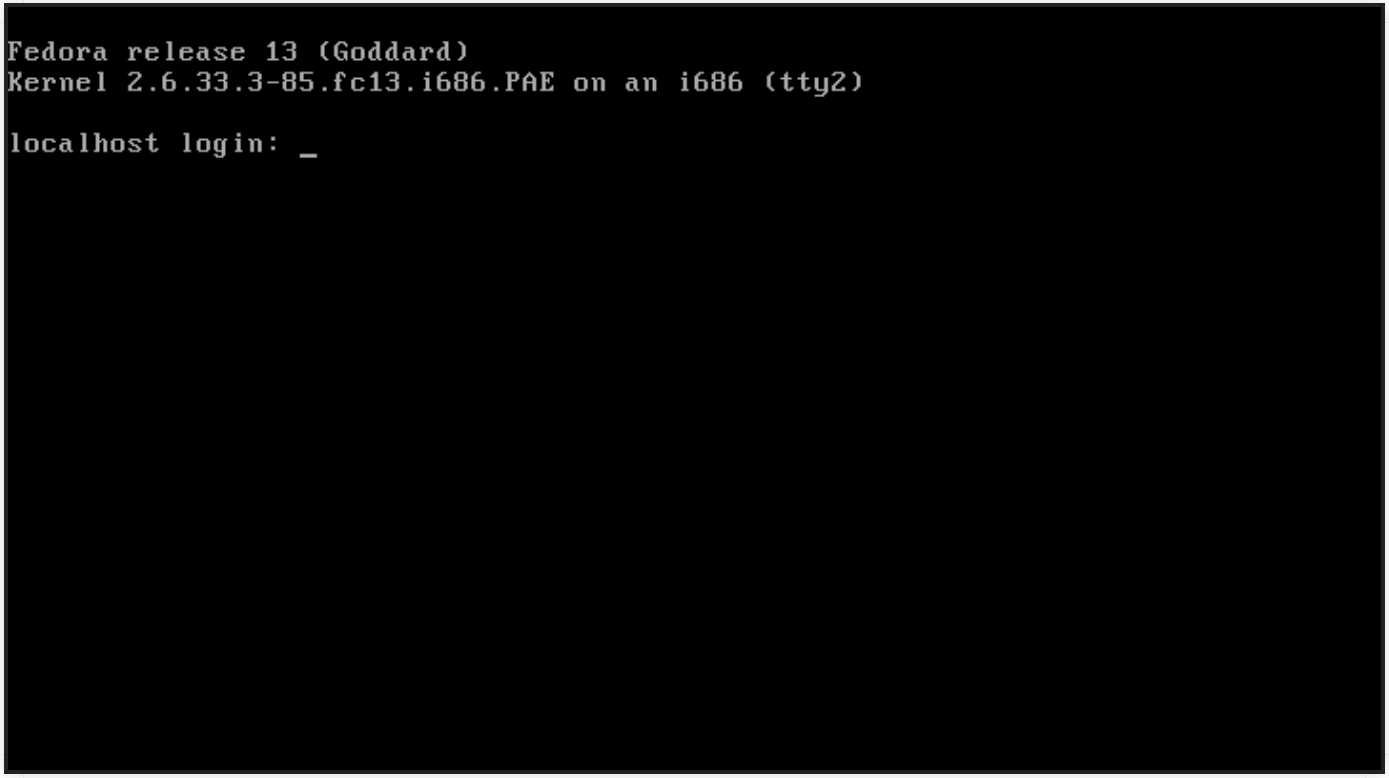
Esempi di Interfaccia

Un esempio: interfaccia grafica di Win7



Esempi di Interfaccia

Un esempio: interfaccia testuale di una distribuzione di Linux



Gestione dei Processi

Il kernel è il primo componente del SO ad avviarsi:

- Esso permette di eseguire altri programmi
- Fornisce l'astrazione fondamentale di processo

Un processo è una astrazione che corrisponde ad un programma in esecuzione

Quando "aprite un programma" il SO:

- Sposta il codice del programma in memoria centrale
- Predispone alcuni dati per gestirne l'esecuzione
- Fa sì che la CPU inizi ad eseguire le istruzioni del programma

Operazioni sui Processi

Alcuni esempi di azioni che si possono effettuare sui processi:

- Creazione
  - E.g. eseguire un programma
- Visualizzazione
  - E.g. Elenco nel Task Manager di Win
- Terminazione
  - E.g. Chiusura forzata dal Task Manager di Win
- Messa in background
  - E.g. Riduzione ad icona

Ce ne sono altre, ma non le vedremo

Gestione della Memoria di Massa

- Per quanto riguarda la gestione della memoria di massa...
- ...Il SO fornisce una astrazione fondamentale di file

Si chiama file una astrazione che corrisponde ad una sequenza di bit in memoria

Senza SO non esistono i file, ma solo bit sul disco rigido

- Il "programma" che realizza tale astrazione si chiama File System
- Alcuni file system "famosi": FAT32, NTFS, exFAT, hfs, ext3
- Quando si predispone un disco rigido per un certo file system...
- ...Si dice che il disco viene formattato

File: Nome ed Estensione

Ogni file ha un nome

Per esempio: `l'isola del tesoro.pdf`

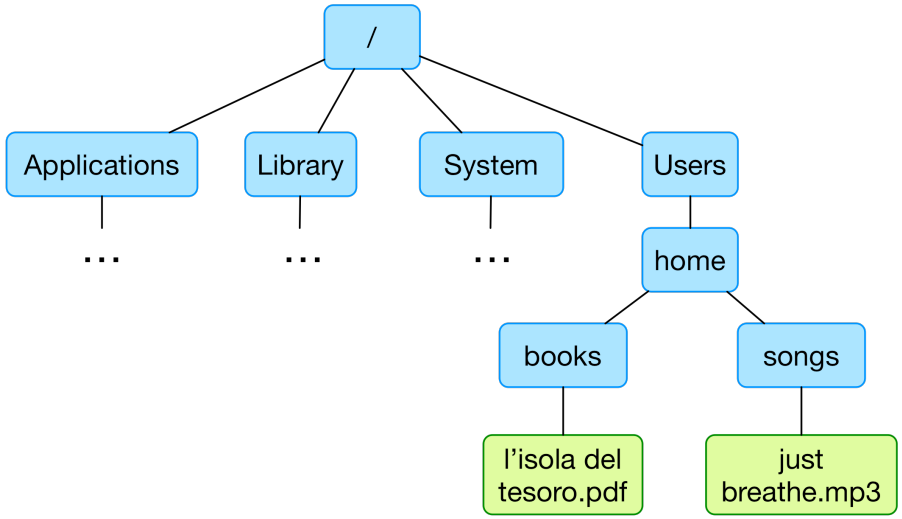
- Di solito il nome di un file termina con una estensione
- Es. ".pdf", ".docx", ".txt", ".m"

A che cosa serve l'estensione?

- Fornisce una indicazione sul contenuto del file
- Serve per determinare (e.g.) con che programma aprirlo
- Non ha alcun effetto reale sul contenuto!
- Può essere cambiata liberamente

Organizzazione dei Files

I file sono organizzati in directory (o cartelle)



- File e directory formano una struttura ad albero
- Una directory fa da radice ("C:\" su Win, "/" su Linux e maxOS)
- Domani sfrutteremo queste informazioni in Matlab

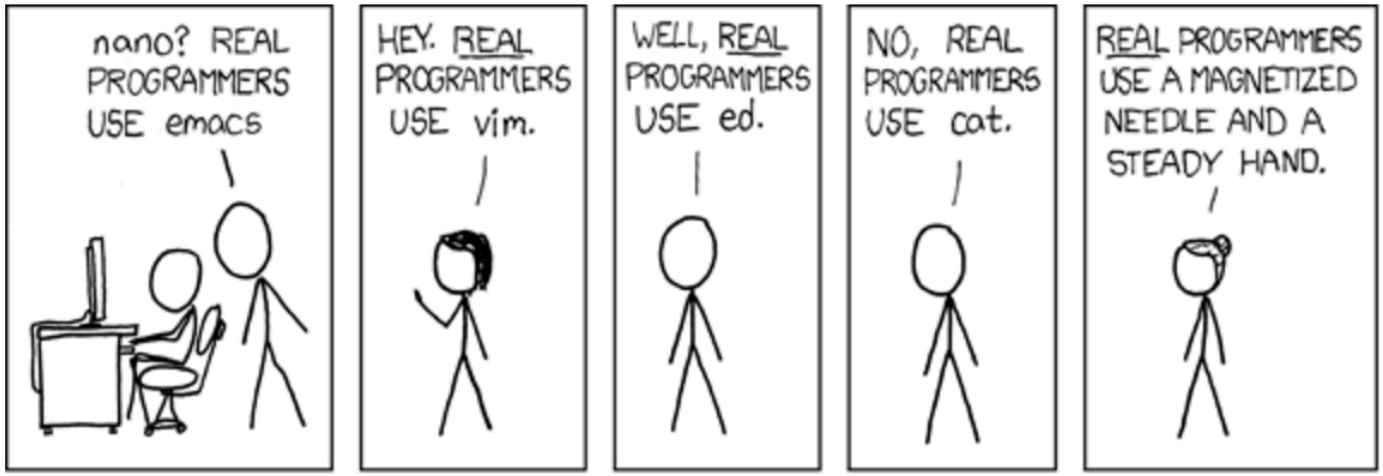
Elementi di Informatica e Applicazioni Numeriche T

Linguaggi e Traduttori

Programmare un Computer

Come programmare un computer? I primi programmatori:

- Dovevano utilizzare solo le istruzioni native (e.g. binarie!)
- E un supporto che il computer potesse leggere direttamente



P.S.: nano, emacs e vim sono editor di testo in giro dagli anni '70

Il Primo Programmatore

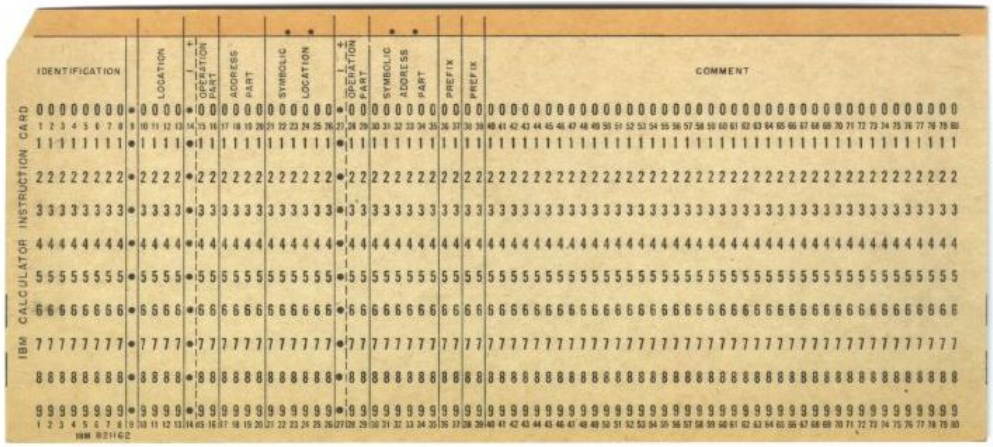
La prima persona a scrivere programmi è stata Ada Lovelace



- La figlia del poeta Byron, fu una collaboratrice di Charles Babbage
- Scrisse programmi per l'analytical engine (lastre di metallo perforate!)

Schede Perforate

Fino agli anni '60 si programmava più o meno nello stesso modo



- Si "scrivevano" istruzioni binarie perforando schede di cartone...
- ...Che venivano inserite in appositi lettori elettromeccanici

Inutile dirlo, programmare così è piuttosto scomodo

Traduttori

Come rendere l'attività di programmazione più semplice?

Possiamo scrivere un programma per usare un linguaggio diverso!

Si chiama traduttore un programma che traduce un linguaggio formale in un altro linguaggio formale

- Grazie ai traduttori, possiamo programmare con un linguaggio...
- ...Che opera ad un livello di astrazione più "alto" rispetto alla macchina
- Un linguaggio di questo tipo di dice di alto livello

Ci sono due grandi categorie di interpreti...

Compilatori ed Interpreti

La prima categoria è quella dei compilatori

Un compilatore traduce un programma prima della sua esecuzione

- Un programma scritto in un linguaggio di alto livello…
- …Viene tradotto in linguaggio macchina (compilato)…
- …A questo punto può essere eseguito direttamente

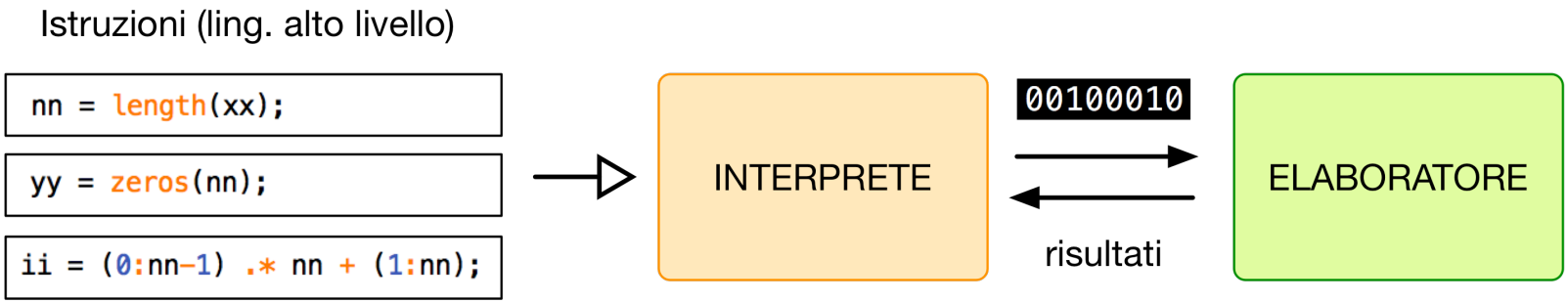


Compilatori ed Interpreti

La seconda categoria è quella degli interpreti

Un interprete traduce un programma durante la sua esecuzione

- Le istruzioni di alto livello vengono considerate una ad una...
- ...Tradotte in linguaggio macchina...
- ...E poi eseguite al momento



Compilatori e Interpreti

Un confronto tra i due approcci:

I programmi compilati sono più efficienti di quelli interpretati

- L'interprete esegue istruzioni addizionali per controllare l'esecuzione
- I compilatori spesso ottimizzazione il codice durante la traduzione

Usare un interprete semplifica la stesura di un programma

- Non serve compilare: è un passaggio in meno
- La compilazione può richiedere tempo (anche parecchio)

Matlab è un interprete: è pensato per sviluppare in fretta

Domani ne vedremo alcune conseguenze!