

Dispense di sistemi automatici 3At

Architettura del computer e periferiche (prof.ssa Busso)

Il computer nacque quando fu possibile costruire circuiti abbastanza complessi in **logica programmata** da una parte e, dall'altra, pensare, (questo è dovuto a Von Neumann) che **i dati e i programmi** potessero essere conservati in una **memoria**, separata dall'unità operativa.

Logica cablata e logica programmata.

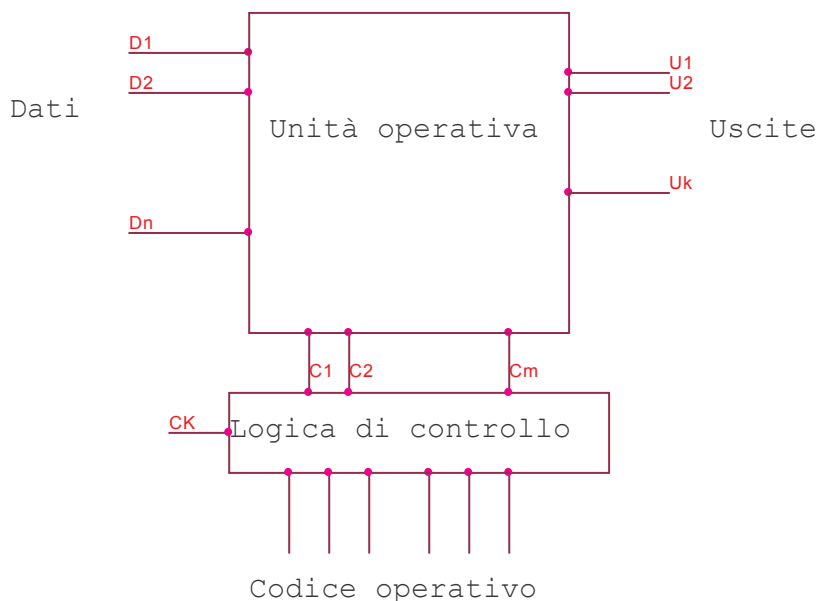
Un circuito è in **logica cablata** quando, fissata la funzione che il circuito deve svolgere, si costruiscono i collegamenti hardware (i cavi, i fili, le piste in rame) tra i diversi integrati e la relazione che collega l'ingresso con l'uscita è fissata una volta per tutte.

Nella **logica programmata**, invece, il circuito possiede anche degli ingressi di controllo e, a seconda del valore di questi ingressi , cambia la funzione svolta dal circuito. In questo caso, con 1 solo ingresso di controllo è possibile scegliere tra due funzioni; se gli ingressi di controllo sono m si può scegliere tra 2^m funzioni differenti.

LA CPU e la scheda madre

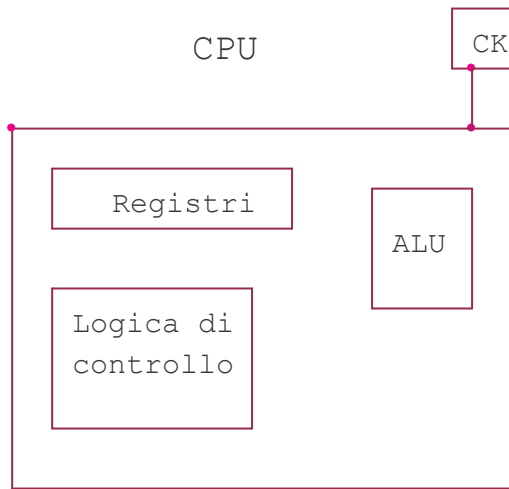
Il cuore del computer (la CPU vedi figura) è un circuito elettronico in logica programmata. Oltre all'unità operativa c'è un circuito di logica di controllo che fa passare da una funzione all'altra con una tempistica scandita dal **clock**, un temporizzatore che permette di controllare la velocità della sequenza delle operazioni. La sequenza delle operazioni è il **programma** (che è conservato in una memoria esterna alla CPU).

Ciascuna **istruzione** del programma, che fa svolgere una determinata operazione, è costituita da due parti: l'insieme dei dati (le variabili D1,D2, Dn nel circuito in figura) e il codice operativo, cioè la combinazione degli ingressi di controllo (C1,C2, Cm) utili per svolgere quella determinata operazione.



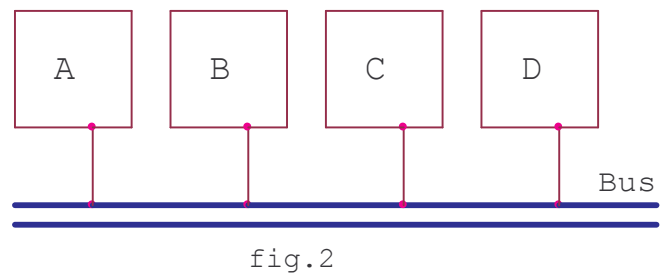
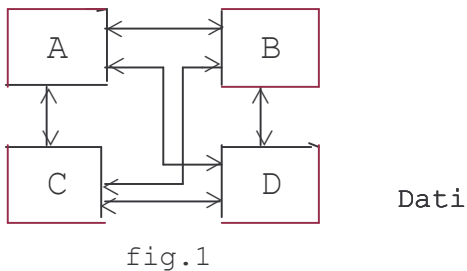
In realtà a questo schema occorre ancora aggiungere una piccola memoria che viene chiamata registro, che conserva i dati di ingresso e di uscita su cui si fanno svolgendo le operazioni.

L'insieme della logica di controllo, dei registri e dell'unità operativa (detta ALU arithmetic-logic unit) è la CPU o microprocessore. (In figura uno schema a blocchi della CPU)



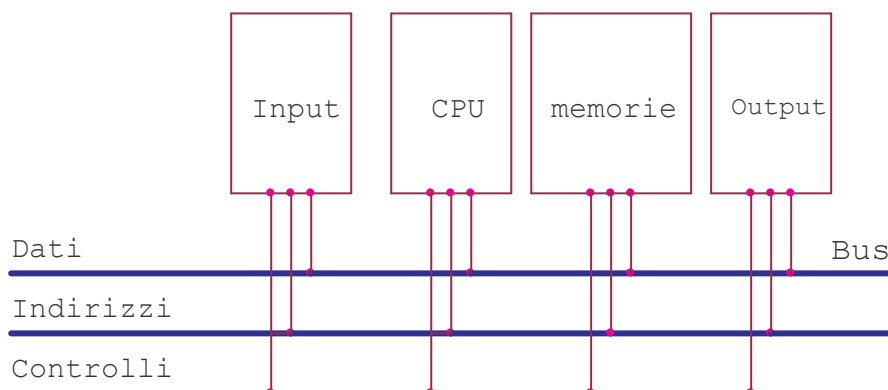
Nel computer non esiste solo la CPU, ma anche le **memorie** e i **circuiti** che permettono di interagire (**interfacce**) con le periferiche di **ingresso** e di **uscita**. I collegamenti tra queste parti non sono **punto a punto**, ma a **buses**.

In un **collegamento punto a punto** ciascun blocco A, B, C, D è collegato agli altri con un filo dedicato (fig.1). La comunicazione è veloce, ma il numero di fili è grande.



Con il bus (fig.2) il collegamento è unico per i 4 blocchi, ma, se comunicano A e C, gli altri blocchi devono aspettare la fine della comunicazione per prendere il controllo del bus: il numero di fili diminuisce drasticamente, ma diminuisce anche la velocità.

Nel Computer i **buses** sono tre : **dati, controlli, indirizzi**. Ciascun blocco deve aver un codice (un indirizzo) per poter essere riconosciuto e devono inoltre essere trasmessi i giusti comandi (i controlli) per le operazioni da svolgere sui dati.



Caratteristiche generali di una CPU commerciale e scheda madre.

Le attuali Cpu , pur avendo una schema di principio simile a quello precedentemente descritto, sono in realtà molto più complesse.

Non esiste una sola **unità operativa** (il circuito che svolge le operazioni; nello schema di principio la ALU) ma da 2 a 4 unità operative che funzionano in parallelo (nei desktop professionali anche 8).

I **registri** sono sostituiti da ben tre livelli di memorie (si chiamano **memorie cache**) che permettono di accumulare all'interno della CPU le operazioni successive e i dati necessari per svolgerle in modo tale che le unità operative non si fermino mai.

La velocità dei buses interni alla CPU è molto alta (oggi 3-4GHz).

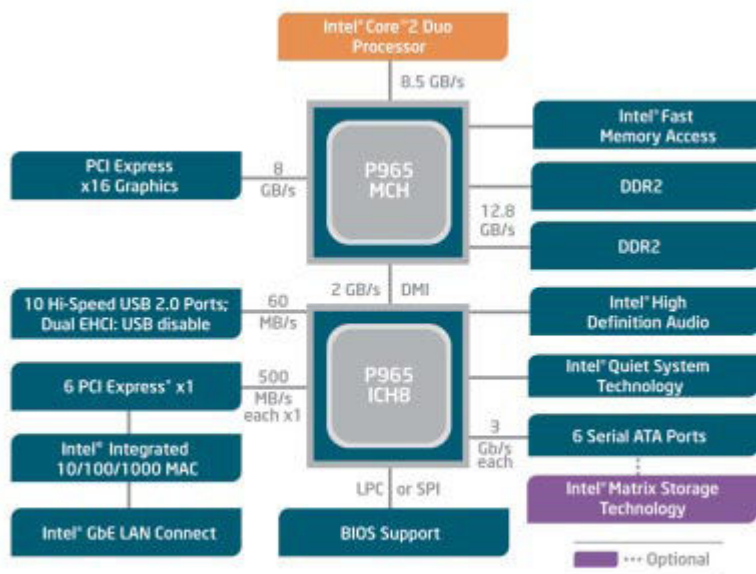
La scheda madre è invece un circuito stampato che serve a collegare con dei buses la CPU e una serie di circuiti integrati , di zoccoli (slot) per l'inserimento di altre schede, di memorie, necessari al coordinamento delle parti del computer.

Gli elementi sempre presenti in una scheda madre sono:

- Lo zoccolo per la CPU e la CPU stessa
- Gli slot per le memorie DRAM
- I chipset , due integrati chiamati anche northbridge e southbridge che servono a distribuire i segnali e i dati alle diverse parti della scheda madre.
- Gli zoccoli per le schede di espansione o PC Express (che permettono di collegare una scheda video esterna) e quelli PCI per altri tipi di schede ad esempio la Ethernet (di collegamento al modem) o una scheda audio esterna.

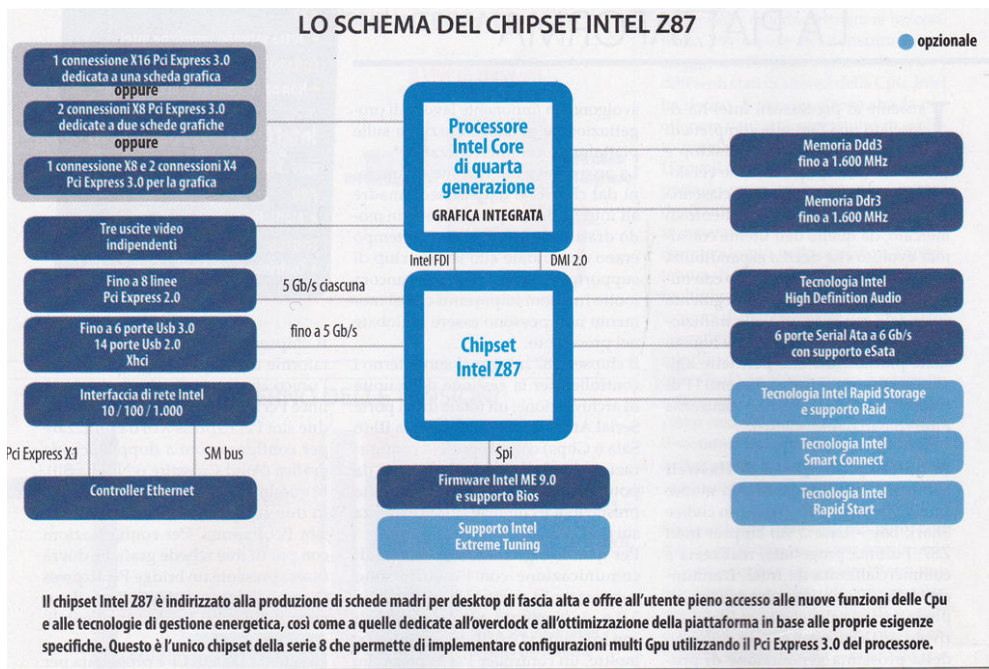
Lo schema funzionale di una **scheda madre classica** è come quello in figura:

vicino alla CPU c'è il northbridge (P965MCH) a cui sono collegati la scheda video esterna o integrata e le memorie DRAM (DDR2), i due blocchi che hanno bisogno della velocità più alta possibile per poter funzionare adeguatamente. I buses che li collegano hanno oggi (2008) una velocità di 1,3 GHz. Un terzo bus raggiunge il southbridge (P965ICH8) da cui partono i bus per tutte le periferiche: i collegamenti seriali (SATA) (per l'hard disk, il lettore ottico, il masterizzatore), le porte usb, la scheda di rete etc.



La scheda madre è strettamente legata al tipo di CPU e al tipo di memorie, quindi un aggiornamento della CPU o un'aggiunta di nuove memorie non è sempre possibile.

Nei nuovi ultrabook (notebook dal peso di circa 1,5kg), nei tablet e negli smartphone (in modi differenti a seconda del livello di prestazione richiesto) la **scheda madre si dice integrata**: la CPU contiene , oltre alle unità di elaborazione e alle memorie cache anche la scheda grafica (la GPU graphic processor unit) , i chipset e, qualche volta anche i controller per le porte USB. Ci sono CPU delle dimensioni di 1 € che svolgono i compiti di oltre metà della vecchia scheda madre.



L'alimentatore .

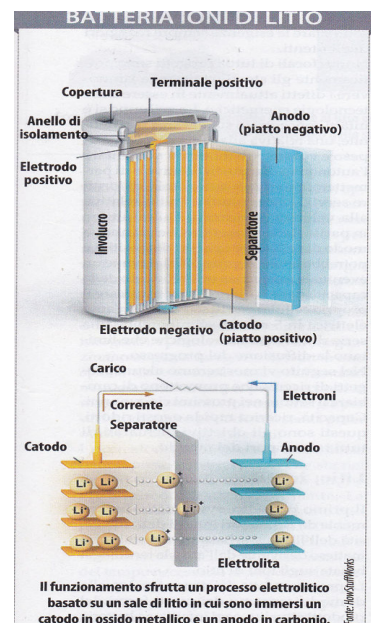
L'alimentatore è il dispositivo, generalmente interno al telaio del PC che serve a trasformare la tensione di rete alternata a 220 V in almeno 3 valori di tensione continua 12V, 5V, 3,3V, necessari ad alimentare i circuiti interni del PC. La potenza di un alimentatore richiesta attualmente parte dai 450-500Watt.

Il valore di tensione a cui lavora oggi una CPU è intorno all'1,5 V (ci sono sulla scheda madre circuiti che abbassano il valore della tensione fornita dall'alimentatore) e la ricerca tecnologica lavora per ridurla ulteriormente in quanto una bassa tensione diminuisce il riscaldamento della CPU stessa.

Le batterie

Se il PC è portatile, l'alimentazione è fornita da batterie ricaricabili (cioè degli accumulatori): celle al litio. Si chiamano celle, perchè la batteria è composta da più elementi in serie (ciascuna con $V = 3,6$ - $3,7V$ (ioni litio) o $V = 3,2V$ nel caso di batteria ai polimeri di litio).

Sono utilizzati dei sali di litio perché il litio è un elemento leggero con 3 elettroni, 2 nel primo orbitale e 1 nel secondo. Cedendo facilmente l'elettrone esterno diventa uno ione positivo. Il catodo (ossido metallico) e l'anodo (carbonio) sono immersi in una soluzione elettrolita di sali di litio. Nella

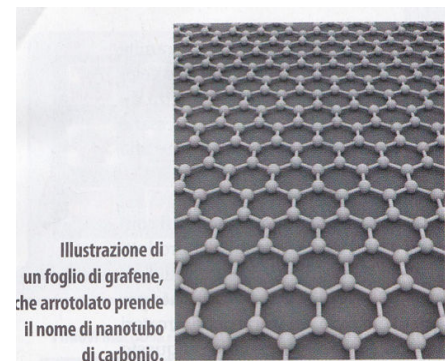


ricarica gli ioni litio vengono assorbiti e trattenuti dall' anodo di carbonio fino a quando necessario; nella scarica vengono rilasciati , creando corrente.

I parametri importanti in una batteria sono:

1. **La capacità** fornita in **Ah** (ampère –ora) indica il prodotto tra la durata temporale e la corrente offerta. (es 10Ah vuol dire 10 A per 1 ora o 1° per dieci ore; in realtà la scala non è lineare per alte correnti il tempo diminuisce. Oggi di norma 2000mAh.)
2. **L'energia specifica** (Wh/kg) , cioè l'energia immagazzinata per unità di massa (attualmente 160Wh/kg)
3. **L'efficienza di carica e scarica** (cioè quanta energia immessa viene effettivamente utilizzata)
4. Il ritmo di **autoscarica**
5. il **numero di cicli di carica / scarica** permessi prima che la carica massima scenda al di sotto di una determinata soglia (circa 2000)
6. I **tempi di carica e scarica**

L'energia specifica è importante perchè è legata al peso dell'oggetto (il peso della batteria è una parte importante del peso di un qualunque portatile, mentre gli altri parametri definiscono l'autonomia che il portatile possiede). La ricarica attuale è dell'ordine delle ore (perchè è lento il processo di incameramento nell'anodo degli ioni litio). Si stanno studiando anodi di **grafene** (atomi di carbonio in molecole a struttura esagonale strutturati in fogli dallo spessore di un solo atomo) che permetterebbero tempi di ricarica di minuti. La differenza tra l'anodo di carbonio e di grafene nell'assorbimento di ioni litio è la stessa che c'è tra legno e una spugna nell'assorbimento dell'acqua.



Memorie

Nel computer esistono più tipi di memorie:

1. **memorie ROM** (Read only memory) . Sono memorie programmate all'atto della fabbricazione e che conservano i programmi base del computer tipo il programma per fare il test di tutte le periferiche che dovrebbero essere collegate all'inizio (tastiera, mouse, hard disk, floppy, cd-rom etc) e che visualizzano sullo schermo ad esempio la mancanza della tastiera e la base del BIOS (Basic Input Output System). Non si cancellano quando si spegne il computer.
2. **memorie RAM C-MOS** (Random Access Memory _Complementary MOS .tipo di tecnologia) : sono piccole memorie a lettura e scrittura che rimangono attive anche se si spegne il computer , perché dotate di una batteria a pastiglia sulla scheda madre in cui sono conservate ad esempio la data, la password per entrare nel BIOS etc. Scollegando la batteria si possono cancellare e quindi nuovamente riscrivere .
3. **memorie DRAM** (Dinamic Random Access Memory) : sono le memorie a lettura e scrittura che si cancellano quando si spegne il PC in cui vengono trasferiti i programmi e i dati (contenuti nell'hard disk) su cui si lavora. Sono oggi dell'ordine dei 2Gbyte - 4GigaByte, ma all'interno del PC si possono inserire fino a 4 moduli e sono abbastanza veloci)
4. **memorie SRAM** (Static RAM) sono le memorie cache a lettura e scrittura , molto più veloci e più costose delle DRAM, situate all'interno della CPU (pochi Mbyte , perché servono a conservare solo i dati e le istruzioni su cui si sta lavorando e al massimo quelle che si "prevede " di usare in un breve spazio di tempo).
5. **memorie di massa** : tutte quelle memorie di tipo magnetico (hard disk o floppy) o ottico (CD ROM o DVD) o memorie $E^2 prom$ o flash (che si cancellano e riscrivono, ma che conservano i dati , anche se si toglie l'alimentazione tipo le pen drive) interne od esterne al computer che servono a contenere grosse quantità di dati e di programmi.

Periferiche

All'interno del telaio del PC o esternamente ci sono circuiti o dispositivi collegati al computer : le **periferiche di input output** .Le principali verranno analizzate in seguito.

Dischi rigidi

I dischi rigidi sono la principale memoria di massa del computer per conservare **i dati , i programmi, il sistema operativo.**

Sono costituiti da una serie di piattelli di alluminio o di vetro , coperti di materiale magnetico, impilati sull'albero di un motorino. Vengono mantenuti sempre in rotazione, per non diminuire il tempo di accesso. Ci sono due testine, una di lettura e l'altra di scrittura, che funzionano con fenomeni fisici differenti. Non toccano i piattelli ma sono mantenute sollevate dal cuscino d'aria creato dai dischi che girano. I piattelli e le testine sono in un contenitore che comunica con l'esterno attraverso un filtro per evitare che polvere e particelle righino i dischi e nello stesso tempo per mantenere la pressione interna equilibrata.

La testina di scrittura, sfruttando il fenomeno dell'induzione, è percorsa da una corrente, il cui valore determina la modificazione dello strato magnetico, mentre quella di lettura è costituita da materiale magnetoresistivo (strati di nickel , ferro, cobalto e rame) che cambia la sua resistività per effetto del campo magnetico (GMR giant magneto-resistive).

A tutt'oggi la capacità del singolo piattello arriva a 320 Gbyte; esiste un limite fisico alla densità areale (numero di bit per pollice²), determinato dalla minima dimensione delle testine e dall'interferenza di dati vicini.

Parametri:

_Capacità: oggi si arriva a più di 1000GByte (1 Terabyte) , anche se 500 Gbyte sono sufficienti per un home computer

_Velocità di rotazione: nei desktop dai 5400 ai 7200 rotazioni per minuti rpm. (si arriva ai 10000 nei dischi più cari)

_Buffer di memoria: mediamente 8 MByte (occorre una memoria per avere i dati sempre a disposizione per il trasferimento)

_Tempo di ricerca: 8-9 msec circa

_Tecnologia di comunicazione tra scheda madre e disco rigido: fino a due o 3 anni fa l'unico tipo di collegamento era parallelo, ad 80 fili con tecnologia EIDE (enhancement integrated drive electronics) ATA (advanced technology attachment) che ha raggiunto una velocità di 133MByte/sec. La nuova tecnologia si chiama **Serial ATA** e ha le seguenti caratteristiche:

- velocità 300MByte /sec
- variazioni del segnale di 250mV invece di 5V
- trasmissione seriale con pochi fili invece del flat cable a 80 pin e lunghezza max 1m (invece di 40 cm)
- scomparsa dei master slave (2 dispositivi collegati sullo stesso cavo es . lettore Cd e disco rigido), ma un collegamento 1 a 1
- supporto per l'hot plugging (installazione e rimozione a caldo).

La velocità di rotazione non è aumentata negli ultimi anni, ma sono migliorati i metodi (software) di lettura e scrittura dei dati ad es. il software Native Command Queuing che riorganizza le richieste in modo da far percorrere alla puntina il minor spazio possibile.

La tecnologia (piattelli, testine) è sempre la stessa da più di 50 anni; solo da poco sono stati introdotti sul mercato delle unità (non si chiamano più dischi, in realtà sono memorie tipo flash) frutto di una nuova tecnologia: gli **SSD , unità allo stato solido** (con struttura tipo le pen drive) , ben 100 volte più veloci (100µs, perché non esiste quasi più il tempo di accesso al dato), più

silenziosi, meno ingombranti, ma altrettanto care (da 0,5-1€ a Gbyte). Non hanno inoltre necessità di parti meccaniche e motorini per far girare i dischi. Come il disco rigido ha una memoria cache (per conservare una parte di dati e permettere all'unità centrale di continuare a lavorare mentre sta salvando o cercando un dato) e un controller (un piccolo microprocessore per organizzare i dati). Le unità SSD hanno **capacità di 128-256Gbyte**; vengono utilizzate ora sui portatili da sole o insieme ad un hard disk. L'SSD conserva il sistema operativo e l'altro, più grande, ma più lento, serve per la conservazione di tutti gli altri programmi e dati.



Un comune disco rigido (a sinistra) confrontato con un'unità a stato solido (a destra)

Tipi di trasmissione.

I due tipi di trasmissione all'interno del PC e su un canale di comunicazione esterno sono :

Parallela. I dati vengono trasmessi un certo numero di bit alla volta (almeno 8), ad ogni impulso di clock, un bit per ogni filo. In realtà oggi i dati sono trasmessi in parallelo solo sulla bread board. Anche solo con l'hard disk il collegamento è ora seriale.

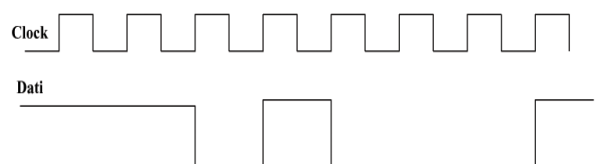
Seriale. I dati vengono trasmessi uno dopo l'altro su un solo filo.

Ci sono due tipi di trasmissione seriale :

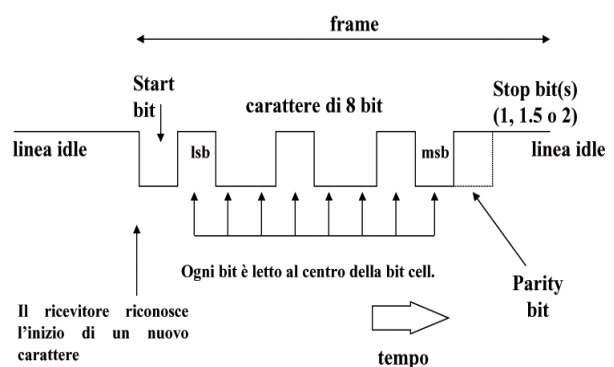
1. **Sincrona** : occorrono due fili: uno per il clock (il segnale di temporizzazione) e l'altro per i dati (il dato è 11010001) che vengono trasmessi in modo sequenziale su una linea, uno per ogni colpo di clock. Il trasmettente e il ricevente vengono sincronizzati dallo stesso clock e quindi non ci possono essere errori di durata del singolo bit.
2. **Asincrona** : c'è un solo un filo per i dati, ma questi sono inseriti tra un bit di start e uno di stop (più un bit di parità per rilevare l'errore di trasmissione). Il computer che riceve capisce quando è iniziata la trasmissione dei dati e quando finisce, inoltre conosce la lunghezza della "parola" trasmessa.

La seriale è più lenta della parallela e deve avere due dispositivi (il cui nome è USART Universal synchronous, Asynchronous Receiver Transmitter), uno in trasmissione e l'altro in ricezione, per trasformare i dati, generalmente in parallelo

Trasmissione sincrona



Trasm issione A sincrona



dentro il pc, in dati seriali e viceversa.

Tutte le connessioni sono costruite su uno di questi due tipi di trasmissione.

Tipi di connessioni alle periferiche.

Le connessioni (in gergo **porte**) non sono solo i connettori , ma anche i circuiti elettronici presenti per modificare i segnali in ingresso e in uscita dal PC. Le caratteristiche più importanti sono **il tipo di collegamento**, la **velocità di trasferimento dati** (bit/s),la **distanza** raggiungibile e la **potenza**.

Collegamenti via cavo.

1. **Seriale** vecchia porta COM (caratteristiche: la connessione è con lo standard RS232 (segnale +12 -12V) con velocità di trasmissione di 115.2kbit/s. Distanza max 30m; connettore a 9 pin.) ; oggi usata solo più per connessioni specifiche , ad es. collegamento di schede elettroniche
2. **Parallela** (velocità 4Mbit/s, max distanza 10 m,connettore Centronix a 25 pin). Usata nelle stampanti o negli scanner , ora sostituita con la USB.
3. **PS/2** vecchia connessione per mouse e tastiera oggi sostituita dall'USB
4. **USB** Universal serial bus. Ha 4 pin Gnd (ground = massa), dati+, dati-, 5V e quindi può fornire anche l'alimentazione alle periferiche 5V, 100mA (il valore della corrente è una limitazione: una fotocamera digitale ne richiede 500). E' possibile attaccare con appositi Hub (tipo di presa per il collegamento di più fili) fino a 127 periferiche e funziona hot plugging (cioè le periferiche si possono attaccare con il PC funzionante). Ne esistono di 3 tipi: la 1.1 con velocità max 112Mbit/s e la 2.0 con velocità max 480Mbit/s e la 3.0 fino a 5Gbit/s (reale 3,26Gbit/s) La 3.0 ha una distanza pratica di 3 m.. Sostituisce la parallela per stampanti e scanner avendo la stessa velocità.
5. **Firewire**.Connessione seriale ad alta velocità su cavo, è hot plugging con velocità di trasferimento 100-200-400Mbit/s e può fornire alle periferiche una tensione da 8 a 40 V e una corrente fino a 1,5A. La distanza max è 4,5 m senza ripetitore, 72m con i ripetitori e si possono attaccare 63 dispositivi. La trasmissione può essere sincrona (viene trasmesso anche il segnale di clock) o asincrona (ci sono bit di start e bit di stop per indicare quando inizia e quando finisce il dato). Si usa anche su fibra ottica con una velocità di 3,2Gbit/s e 500m.

Collegamenti wireless.

Sono collegamenti senza fili e possono essere di due tipi :

1. infrarossi.

Irda. Porta a infrarossi utilizzata nei palmari, nei notebook e nei cellulari con velocità 115.2Kbit/s ,la fast Irda arriva a 4Mbits /s (il collegamento deve essere diretto)

2. via radio

Bluetooth (Blatand) .La connessione è via radio con frequenza di portante nella banda da 2,4 a 2,48 GHz. Serve a collegare dispositivi di tipo diverso su distanze o di 10 o di 100m e collega fino a 8 dispositivi .La caratteristica fondamentale è la bassa potenza (partono da 1mW) e il basso costo e serve a costruire le PAN Personal Area Network

Wi-Fi (wireless fidelity). Lavora in una banda a 2,45GHz con una velocità fino 11Mbit/s (lenta) e con potenze più alte(100mW) della Bluetooth. Non consente collegamenti in movimento, ma serve per estendere o collegare reti esistenti. Ha grande ampiezza di banda e , a seconda del tipo di antenna usata e degli ostacoli presenti, copre di stanze tra i 150m e il Km.

CD ROM e DVD


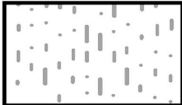

Sono memorie di massa di tipo ottico, perché i dati vengono archiviati e poi letti attraverso un raggio laser che incide (brucia), nel caso della scrittura, lo strato del disco in modo differente (costruendo avvallamenti (pits) o lasciando zone piane (lands)) a seconda se il dato è uno 0 o un 1. Nel caso della lettura la luce laser si riflette in modo differente su pits e lands.

Nei **CD ROM** i dati (durata 80 minuti/capacità 700Mbyte) vengono archiviati su una traccia a spirale. Il disco (12 cm diametro 1,2mm di spessore) è formato da uno strato di policarbonato rivestito da una pellicola riflettente di alluminio o d'oro. Il **laser a 780 nm (infrarosso)** attraversa il policarbonato va ad incidere (scrittura) o viene riflesso (lettura) dal metallo. Il fotorivelatore rileva variazioni di intensità luminosa poiché i pits e i lands riflettono in modo differente la luce.

La tecnologia **DVD** (Digital Versatile Disc) utilizza un diodo laser a **650 nm (rosso visibile)**, La lunghezza d'onda è minore, quindi i pits e i lands possono essere più vicini, così pure gli avvolgimenti della spirale: si arriva nel DVD singola faccia, singolo strato a 4,37 GByte.

Formati ottici a confronto

Con l'evolversi della tecnologia ottica, la lunghezza d'onda del laser diminuisce e la densità dei dati aumenta. Ciò permette di moltiplicare la quantità delle informazioni archiviabili sui dischi di generazione più recente.

Formato:	DVD	HD-DVD	BLU-RAY
Spessore del substrato	0,6 mm	0,6 mm	0,1 mm
Lunghezza d'onda e colore del laser	650 nm/ rosso	405 nm/ blu-viola	405 nm/ blu-viola
Densità dei dati			
Capacità del disco a strato singolo strato	4,7 GByte	15 GByte	25 GByte
Capacità del disco a doppio strato	8,5 GByte	30 GByte	50 GByte
Velocità di trasferimento a 1x	11 Mbps	36,55 Mbps	Video BD-ROM: 54 Mbps; tutti i restanti formati: 36 Mbps
Diametro del laser	1.320 nm	620 nm	480 nm
Track pitch	740 nm	400 nm	320 nm
Densità di registrazione	2,2 Gbit/pollice quadrato	8,8 Gbit/pollice quadrato	14,73 Gbit/pollice quadrato
Risoluzione video	720 x 576	1.920 x 1.080	1.920 x 1.080
Sostenitori principali	DVD Forum, DVD+RW Alliance	DVD Forum (in primis NEC e Toshiba)	Blu-ray Disc Association
Tempo di riproduzione HD	-	Strato singolo: circa 4 ore Strato doppio: circa 8 ore	Strato singolo: circa 6 ore Strato doppio: circa 13 ore

Nella doppia faccia il disco è formato da due substrati da 0,6mm disposti dorso a dorso e separati dallo strato riflettente e arriva a immagazzinare 7,95Gbyte. Ciascun substrato può ospitare ancora due strati (layer) di dati (letti attraverso una differente messa a fuoco del laser) e quindi questo tipo di DVD immagazzina quasi 16 GByte (1-10€).

I DVD registrabili sono dischi come i CD ROM, ma lo strato è composto da un colorante organico fotosensibile e il laser ne modifica in modo permanente le proprietà ottiche. I riscrivibili funzionano nello stesso modo, ma lo strato è in materiale policristallino (costituito non da un unico cristallo) e la tecnologia è a cambiamento di fase.

Il DVD video riesce a contenere un intero film con la tecnologia di compressione Mpeg2 e risoluzione 720x480 punti e 30 quadri al secondo (NTSC) o 720x576 con 25 quadri al sec (PAL), ma non può contenere film in alta definizione in modalità interlacciata (disegna le righe pari e poi le dispari) o progressiva (1280x720 o 1920 x 1080 punti); in questo caso occorrerebbero 11GByte per un'ora di film.(per la compressione dei punti e dei quadri vedi il paragrafo del monitor)

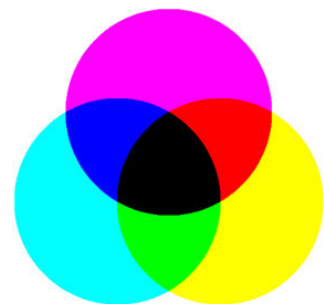
I due formati della nuova generazione (inizio 2006) il **HD-DVD** e il **BLU-RAY** lavorano tutti con una luce blu viola di 405 nm e possono raggiungere i 15-20GByte per strato (HD) e i 25 (BLU).Gli HD però usano una tecnologia a 0,6mm per faccia come i DVD (non bisogna cambiare le linee di produzione) mentre nei BLU i dati sono stampati sul lato esterno del disco e protetti da un rivestimento di 0,1mm .Sui nuovi formati esiste una nuova tecnologia anticopia .

I lettori attuali hanno velocità di lettura , di scrittura e di riscrittura che dipendono dal supporto (più veloci sui cd :48x/48x /32x [1x vuol dire 150KByte/s], meno sui DVD, 16x/ 16/ 6 o 8x), un buffer di memoria di 2 MByte o più, un'interfaccia o ultra ATA o SATA. Il **buffer** di memoria serve per impedire l'arresto della lettura o, soprattutto della scrittura; anzi esiste una tecnologia (Burn proof, a prova di svuotamento del buffer) che mette il masterizzatore in attesa qualora si verificano circostanze che potrebbero causare lo svuotamento del buffer(si chiama buffer underrun).

STAMPANTI

I colori.

I tre colori primari (sottrattivi) per la stampa, cioè nel caso di colori coprenti su ,ad esempio, carta, sono il giallo, il rosso (magenta), il blu (ciano).Gli altri colori sono ottenuti mescolando , o nel caso della stampa, accostando minuscole goccioline dei 3 primari. Per chiarire o per scurire si usa il nero. L'insieme, lo spazio, di questi colori si chiama **Cmyk**, dalle iniziali in inglese dei 3 colori primari + il nero.



Spazio Cmyk

Quando si tratta di luci (quindi non ci occupiamo più di stampanti, ma di **monitor e schermi TV**) i **tre colori primari (additivi)** sono: il rosso (red), il verde (green), il blu (blue) e gli altri si ottengono per sovrapposizione di luci di questi 3 colori fondamentali. L'insieme , lo spazio, di questi 3 colori si chiama **RGB**.

I colori primari sono differenti perchè il nostro occhio nel caso della stampa riceve solo le lunghezze d'onda che non sono assorbite dalla carta (da questo il nome sottrattivi), mentre per le luci riceve i colori direttamente.



Spazio RGB

Non è detto che i due spazi , quello di visualizzazione e quello di stampa si sovrappongano esattamente ; questo vuol dire che ci sono colori visibili a schermo e non stampabili e viceversa. Ogni dispositivo ha il suo **gamut** termine inglese che indica **l'insieme dei colori che il dispositivo**

o la periferica è in grado di produrre, riprodurre o catturare ed è un sottoinsieme dei colori visibili.

GAMUT (da wikipedia)

Poiché per descrivere i colori esistono diversi modelli, detti modello di colore, come ad esempio i modelli RGB, CMYK, Hue Saturation Brightness (HSB), Hue Saturation Lightness (HLS), CIE L*a*b*, YIQ, con gamut di un modello di colore si intende l'insieme di tutti i colori descrivibili da quel particolare modello di colore. Quando un colore non può essere descritto da un certo modello di colore, si dice che, rispetto a quel modello di colore, è fuori gamma (o fuori gamut: *out of gamut* in inglese).

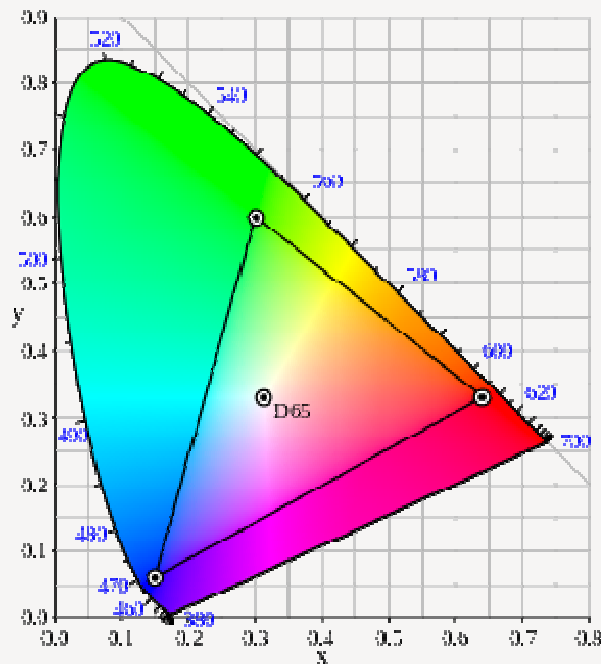


Diagramma delle cromaticità CIE 1931 (colori percepibili dall'osservatore standard) e proiezione del gamut dello spazio colore sRGB (il triangolo interno, standard RGB).

I primari scelti dalla CIE per generare tutti i colori visibili sono tinte ipersature: colori (in realtà, non essendo visibili, non dovrebbero essere indicati come tali) più saturi di quanto i nostri fotorecettori retinici siano in grado di decifrare.

I tre "primari immaginari" sono stati denominati X, Y, e Z. X corrisponde a un rosso violaceo ipersaturo contraddistinto da due picchi nello spettro cromatico rispettivamente intorno ai 450nm e ai 600 nm (quest'ultimo molto superiore al primo), Y e Z corrispondono a tinte spettrali - sempre irrealisticamente ipersature - con lunghezza d'onda dominante rispettivamente di 520 e 477 nanometri.

Inoltre la tinta Y (quella corrispondente al "verde ipersaturo") ha un andamento proporzionale alla nostra sensibilità alla luminosità delle tinte. Scelti i tre primari tramite i quali è possibile ottenere, per sintesi additiva, qualsiasi tinta reale è possibile a questo punto utilizzare uno spazio tridimensionale, avente per assi i tre primari utilizzati, per catalogarle tutte. La variabile z può essere pensata:

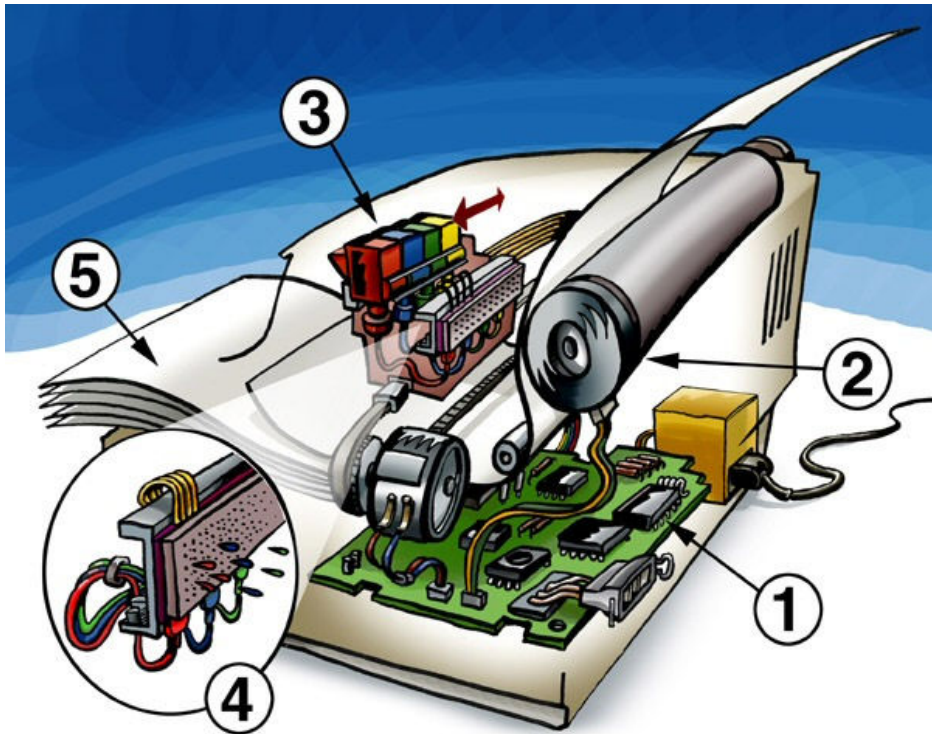
$$z = 1 - x - y$$

ed è dunque possibile utilizzare due sole coordinate cromatiche (x e y, ad esempio) per identificare un colore essendo la terza (z, in questo caso) ricavabile sottraendo all'unità le altre due. Il vantaggio è evidente: normalizzando i colori col meccanismo della somma costante (uguale a 1) è possibile utilizzare un grafico bidimensionale per catalogare qualitativamente (e non quantitativamente) tutte le tinte reali.

Caratteristiche stampanti.

Le caratteristiche di cui bisogna tener conto sono:

1. **la risoluzione** (la risoluzione mi indica quante goccioline ,dots (punti), è possibile avere in un pollice (in inglese inch= 2,54cm) si misura in dpi (dots per inch). La risoluzione è importante nella stampa fotografica a colori: ora raggiunge i 4800x2400 dpi o anche di più, mentre le stampanti laser raggiungono i 1200x1200 (ma qui è più importante la velocità di stampa, quando la qualità del testo o la grafica è buona).
2. **la velocità di stampa** : ppm (pagine per minuto). Varia a seconda del tipo di documento, della risoluzione, del prezzo. E' importante se la stampante è utilizzata per grandi quantità di documenti o è una stampante di rete , condivisa da più computer
3. **la capacità del buffer** di stampa e il tipo di processore (dipende principalmente dall'uso).
4. **le caratteristiche di stampa**: fronte retro, formato del foglio A6, A4, A3, monocromatica , a colori, set di caratteri disponibili, numero di cartucce (attenzione al costo: 10-15 € a singolo colore e fino a 50 per i 3 colori)
5. **tipo di connessione** al computer (parallela o USB 1.1 o 2) o alla macchina fotografica (principalmente A6 con o senza display) .
6. **Il tipo di tecnologia**
 - a. **inkjet** La testina di stampa controlla la fuoriuscita degli inchiostri dagli ugelli attraverso un meccanismo di piezoelettricità inversa (deformazione dei cristalli dovuta al passaggio della corrente) La minima emissione attuale è un goccia da 2 picolitri (10^{12}). E' la tecnologia più adatta per la stampa a colori. Il numero di inchiostri varia da 4 (Ciano, Magenta, Yellow, Black, quello che si chiama lo spazio colore delle stampanti .Cmyk) ai 6 con l'aggiunta del yellow chiaro o ciano chiaro e magenta chiaro (le attuali buone stampanti fotografiche) agli 8 colori col aggiunta del verde e del rosso (le migliori nuove con una risoluzione di 4800x2400dpi).La grandezza degli ugelli può essere variata a seconda della grandezza delle zone di colore). **Problemi** : dominanti di colore e banding (strisce).



La tipica stampante a getto d'inchiostro presenta un carrello (3) che si muove avanti e indietro per tutta la larghezza del foglio(5), il quale a sua volta procede in direzione perpendicolare al carrello mediante un sistema di rulli (2) che lo trascina. Sul carrello sono fissate le testine di stampa⁸⁴, il cui compito è quello di proiettare sul foglio microgocce di inchiostro del volume di pochi picolitri attraverso minuscoli forellini detti ugelli. Il meccanismo di eiezione delle gocce può essere di due tipi:

- termico: in corrispondenza di ogni ugello è posizionato un resistore attraverso il quale vengono fatti passare impulsi di corrente (1); ad ogni impulso il resistore si riscalda alla temperatura di alcune centinaia di gradi in pochi microsecondi e genera nell'inchiostro a contatto con esso una bolla di vapore. L'espansione di quest'ultima provoca l'espulsione della goccia dall'ugello soprastante; questa è la tecnologia più diffusa nell'ambito home/office ed è utilizzata da Hewlett-Packard, Canon, Lexmark e Olivetti;
- piezoelettrico: sotto ogni ugello è sistemato un canalino circondato da un cristallo piezoelettrico; un impulso elettrico (1) provoca la deformazione del cristallo e conseguentemente la repentina strozzatura del canalino e l'eiezione dell'inchiostro; è la tecnologia utilizzata da Epson.

L'inchiostro viene prelevato da serbatoi chiamati cartucce le quali possono contenere inchiostro libero o trattenuto da una spugna.

b. **laser** Il funzionamento è basato sulla presenza di un tamburo rotante con un foglio fotosensibile che si carica di elettricità statica negativa dove è colpito dalla luce laser, quindi la luce laser “scrive” i caratteri sul foglio; poi viene rilasciato il toner (inchiostro solido) carico positivamente, si deposita elettricamente dove è passato il laser. Il foglio scorre sotto il rullo e attira la polvere del toner; la polvere viene poi fusa per mezzo di un paio di rulli surriscaldati: la polvere si fonde e penetra nella carta lasciando un foglio

senza sbavature. **Problemi:** ghosting (immagine sbiadita del foglio precedente) e contrasto, soprattutto all'interno dei caratteri.

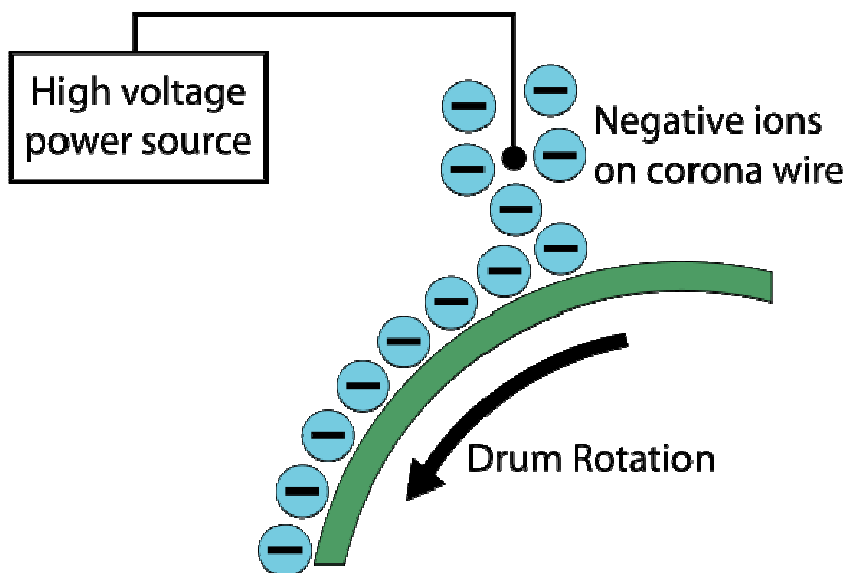
Il processo è più complesso per una stampa a colori: 4 tamburi o uno solo con le 4 cartucce allineate una dopo l'altra; in tutti e due i casi il processo o è molto più lungo o si creano problemi di allineamento degli inchiostri. Ora ci sono stampanti laser a 4 colori, ma il pregio maggiore è comunque legato alla velocità con stampa monocromatica.

Esiste anche una tecnologia parallela a Led (una matrice, una rete di righe e colonne, che sostituisce il laser).

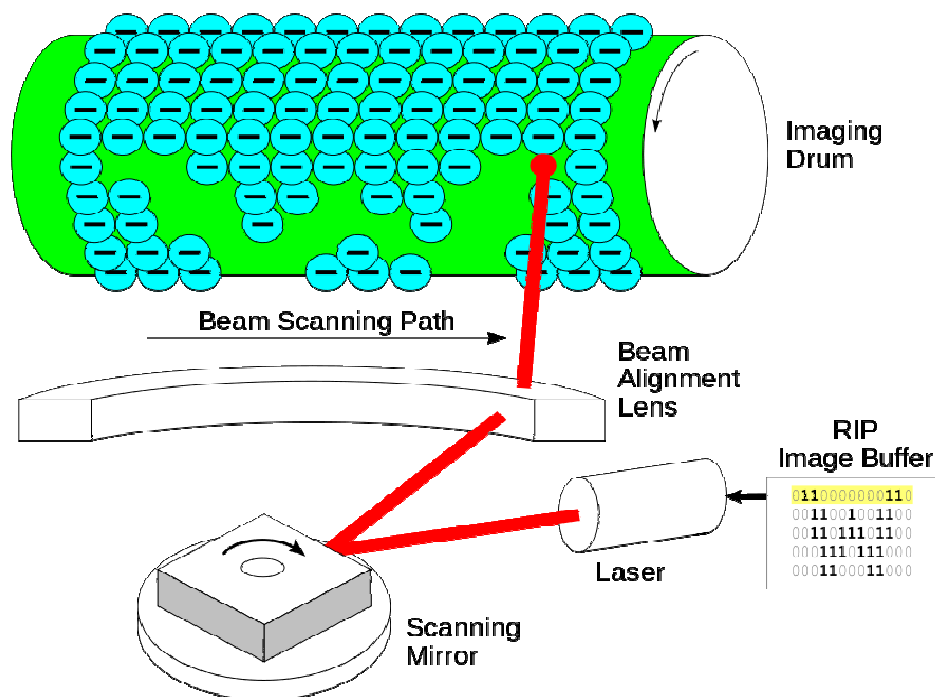
L'immagine da riprodurre (proveniente da supporto cartaceo o elettronico) è riportata da un laser su un cilindro di selenio reso fotosensibile, detto "tamburo" o "rullo magnetico", che con la luce si carica, acquisisce l'immagine in negativo, immagazzina il toner e lo riporta poi sulla carta.

Il procedimento consta di 7 fasi.

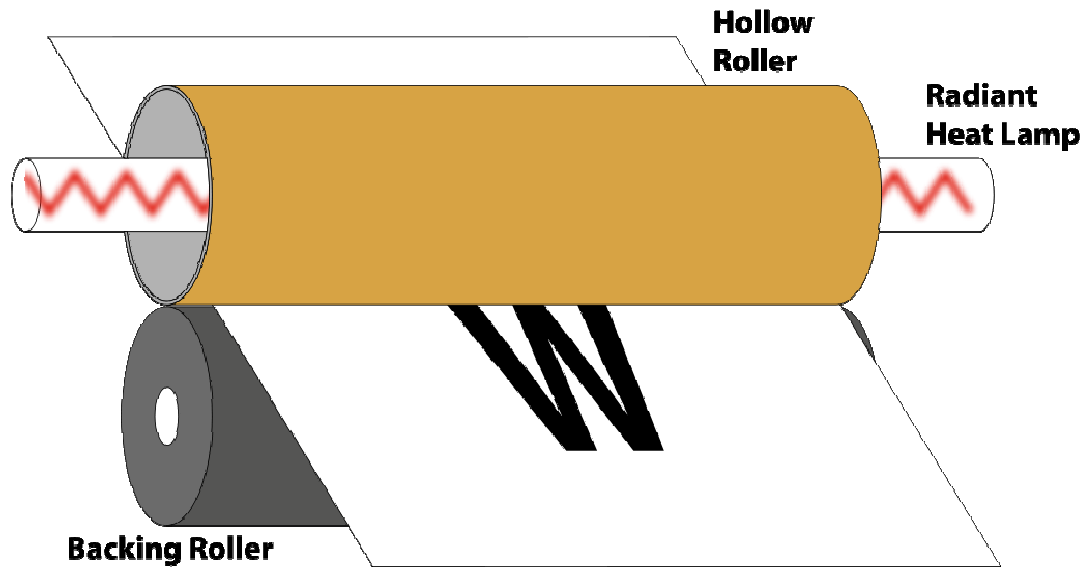
1. **caricamento** statico del tamburo attraverso il PCR (rullo primario caricabatteria). Il flusso continuo di corrente elettrica proveniente dal PCR genera uno stato di carica negativa sulla superficie fotosensibile del tamburo.



2. **esposizione:** Il raggio laser legge l'immagine da riprodurre e scansiona la superficie del tamburo togliendo la carica negativa nelle aree dove il toner non dovrà collocarsi, nella successiva fase di sviluppo. Sulla superficie del tamburo a questo punto risulta esserci un'immagine "nascosta" che non è altro che la riproduzione in negativo dell'immagine da stampare.



3. **sviluppo:** la fase di sviluppo avviene attraverso l'immagazzinamento, da parte del tamburo magnetico, della polvere del toner sull'immagine nascosta. La polvere del toner, proveniente dal proprio serbatoio (le cui aperture sono adiacenti al tamburo) viene attratta sul tamburo stesso e si attacca alle aree rimaste sensibili alla carica negativa, cioè all'immagine nascosta. Nel procedimento interviene anche una lama dosatrice (racla) che regola la quantità di polvere toner nella zona del rullo magnetico. Il suo compito è quello di livellare l'acquisizione del toner togliendo il superfluo.
4. **trasferimento:** il passaggio successivo è lo spostamento del toner sul foglio di carta. Durante il suo scorrimento il foglio viene caricato positivamente dal rullo di spostamento, per far sì che il toner sul tamburo, con carica negativa, si trasferisca sul foglio di carta creando l'immagine da stampare.
5. **fusione:** le particelle di toner che sono già presenti sul foglio di carta non sono ancora attaccate perfettamente ad esso, hanno bisogno della fase di fusione. Il foglio di carta deve passare attraverso una zona della stampante detta "forno di cottura", composto dal rullo fusore e dal pressore. Il rullo pressore comprime il foglio sul rullo fusore, che con il calore emesso salda il toner alla carta.



6. **pulizia:** nella cartuccia, mentre avviene la rotazione, non tutto il toner collocato sul tamburo si trasferisce sul foglio di carta, perciò c'è necessità di un meccanismo di pulizia per consentire alla macchina di ripetere il procedimento di stampa. La lama di pulizia ha il compito di pulire il tamburo dal toner rimasto, non più riutilizzabile, trasportandolo in un serbatoio specifico. Esiste un'altra lama detta di recupero, che ha la mansione di impedire al toner rimasto di cadere davanti al tamburo, impedendogli quindi di uscire dalla cartuccia.
7. **cancellazione:** dopo che la lama di pulizia ha tolto il residuo di toner, la macchina cancella le impronte di carica negativa ancora presenti sul rullo magnetico.

c. **inchiostro solido** Sono prodotte da Xerox (stampanti di rete): vengono utilizzati blocchi di inchiostro a base resinosa portati alla temperatura di fusione da elementi riscaldanti: una testina di stampa con ugelli espelle l'inchiostro. Sono meno costose di quelle laser e più ecologiche (4 colori).

Schermi e monitor

L'occhio umano e l'immagine in movimento.

Nel cinema su pellicola si vedeva un movimento fluido e senza scatti quando il tempo tra un fotogramma e il successivo era $<$ di 20 msec (per la persistenza dell'immagine sulla retina).

Il monitor del computer e la televisione, pur non avendo i fotogrammi devono soddisfare lo stesso requisito.

Nel caso delle vecchie televisioni (o, nei laboratori, gli oscilloscopi) con il tubo a raggi catodici si usava una sostanza, i fosfori, che emettono luce per un tempo dell'ordine dei msec, poi si

“spengono”. Inoltre i fosfori, dei 3 colori base RGB, erano abbastanza vicini, perché l’occhio umano non li vedesse separati. (circa 0,25 -0,30 mm). Lo schermo era una rete di punti fosforescenti (la prima rete era di 640x480 punti) su cui passava , a intervalli regolari un “qualcosa”, un fascio di elettroni), che eccitava i punti fosforescenti . Se l’intervallo regolare era di 20 msec , il numero di volte al sec (cioè la frequenza) era $f=1/20\text{ms} \Rightarrow 50 \text{ Hz}$ (Hertz), cioè l’eccitazione capitava 50 volte al sec.

Ora i punti fosforescenti sono sostituiti dai cristalli liquidi (LCD) e il fascio di elettroni è diventato un segnale elettrico. Attualmente il numero di punti (pixel) è 1920x1080 (si dice che è un monitor HD in alta definizione) e la frequenza (principalmente per i monitor) è tra i 50 e i 120Hz.

Schermi.

Tutti i monitor e gli schermi oggi in commercio si basano sulla tecnologia a cristalli liquidi, ma esistono almeno 3 tipi di tecnologia che li utilizzano: **LCD** con illuminamento a **lampade fluorescenti**, a **led** e la **tecnologia Ips** (in plane switching).

Tecnologia LCD Liquid Crystal Display.

Il principio su cui si basano gli schermi LCD è la proprietà dei cristalli liquidi (lunghe catene molecolari) di rifrangere la luce in modo differente a seconda dell’angolazione da cui entra e di cambiare orientamento se sottoposti ad una ddp (tensione elettrica).

I cristalli liquidi possono assumere a diverse temperature stati differenti (cioè possono essere liquidi o solidi); a temperatura ambiente formano una struttura filiforme e, se sono presenti delle scanalature, tendono a disporsi in modo ordinato, seguendole. Ora se si usano due piani d’appoggio con scanalature a 90°, l’una rispetto all’altra, e in mezzo i cristalli questi creano delle catene elicoidali (vedi fig.1). Se si applica una ddp alle lastre, l’orientamento dei cristalli può essere modificato (ad es. la catena può raddrizzarsi tra le due lastre, ponendosi in posizione perpendicolare alle stesse fig.2). Ora se esternamente alle lastre si sistemano due filtri polarizzatori (vedi nota1) che lasciano passare la luce quando i cristalli formano le catene elicoidali , se le catene diventano perpendicolari il 2° filtro blocca la luce. Quindi se una fonte luminosa viene disposta dietro una struttura composta di tanti cristalli liquidi, ognuno comandato indipendentemente da una ddp, i cristalli con la ddp attiva non lasciano passare la luce.

In realtà la struttura è ancora più complessa perché esistono, uno per pixel dei piccoli transistor (nella tecnologia Tft thin film transistor, oggi la più usata) molto veloci e che aumentano il contrasto e poi i filtri , 3 per ogni pixel, uno per ogni componente RGB (Red , Green ,Blue) per costruire i colori (vedi fig.3).

Negli LCD classici la retroilluminazione è dovuta a lampade fluorescenti; nei monitor a **LED** è una matrice di Led che sostituisce le lampade con i vantaggi di diminuire lo spessore dello schermo e di avere un nero più nero e un bianco più bianco, aumentando il contrasto.

Nella tecnologia **IPS** i cristalli liquidi sono disposti orizzontalmente e ruotano: il vantaggio è un notevole aumento dell’angolo di visione.

Il **segnale** utilizzato è **digitale**, quindi significa non bisogna convertire in analogico il segnale della scheda grafica e non si hanno perdite e distorsioni; inoltre non c’è emissione elettromagnetica come nel caso dei vecchi schermi e, dal punto di vista visivo, è meno stancante.

Il difetto principale è un angolo di visione piccolo (meno di 45 ° dalla perpendicolare, molto migliorato con gli IPS ; notevolmente cari a tutt’oggi); inoltre, per ogni tipo di diagonale c’è un’unica risoluzione ottimale (per i 15’’ è di 1024x768, se è minore bisogna espandere l’immagine aggiungendo pixel a quelli esistenti, diminuendo la nitidezza).

Una caratteristica di cui bisogna tener conto nell'acquisto di un monitor LCD è il numero di pixel difettosi. I pixel difettosi in un pannello possono essere di tre tipi: sempre accesi, sempre spenti, oppure con un difetto nel subpixel di uno dei 3 colori. Esiste una normativa ISO (International Standard Organization) per quanto riguarda il numero di pixel difettosi che prevede 4 classi : da nessun pixel (classe I) a 50,150 ,500 (classe IV) pixel difettosi. La classe II (2,2, 5 pixel difettosi) è quella più frequentemente adottata.

Oltre alla classe, altri parametri importanti sono: la luminanza (l'intensità luminosa emessa per unità di superficie (in nit = candele /mquadro) e il rapporto di contrasto che indica il rapporto di luminosità tra pixel spenti e pixel accesi (aumenta la nitidezza).

Nel **monitor** possono essere inseriti due speaker , un microfono; il monitor può funzionare da Hub per altre periferiche, può anche avere un ingresso compatibile con il segnale televisivo (ingresso S-video); in quelli di ultima generazione può esserci direttamente la scheda madre del PC. Inoltre può avere la funzionalità pivot che permette di passare dalla modalità landscape (orizzontale) a quella portrait (verticale) utile per lavorare con i documenti formato A4.

Nota 1. La luce solare è formata da raggi con differenti direzioni; i filtri polarizzatori sono filtri che lasciano passare solo i raggi in una determinata direzione.

Fig 1

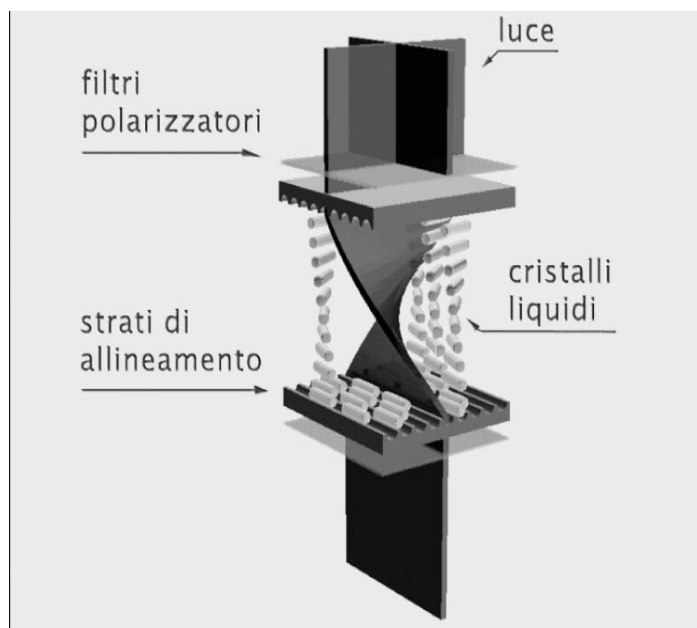


Fig.2

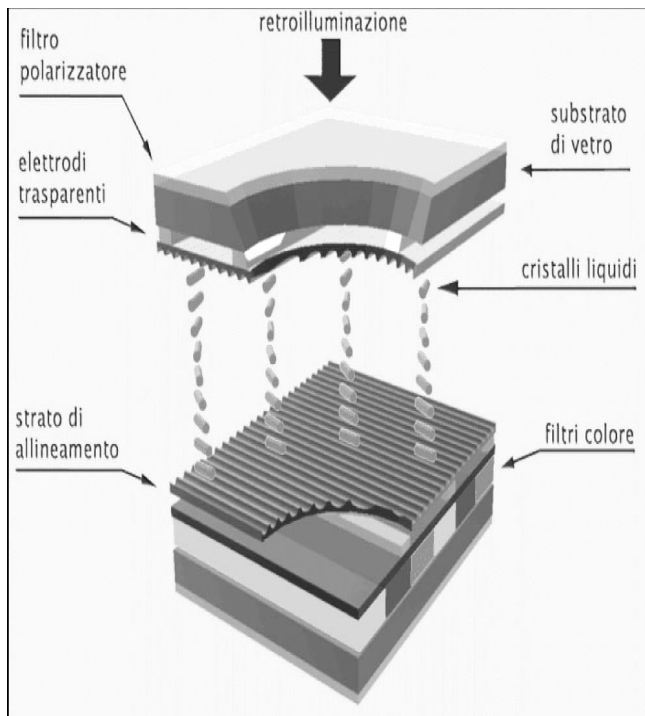
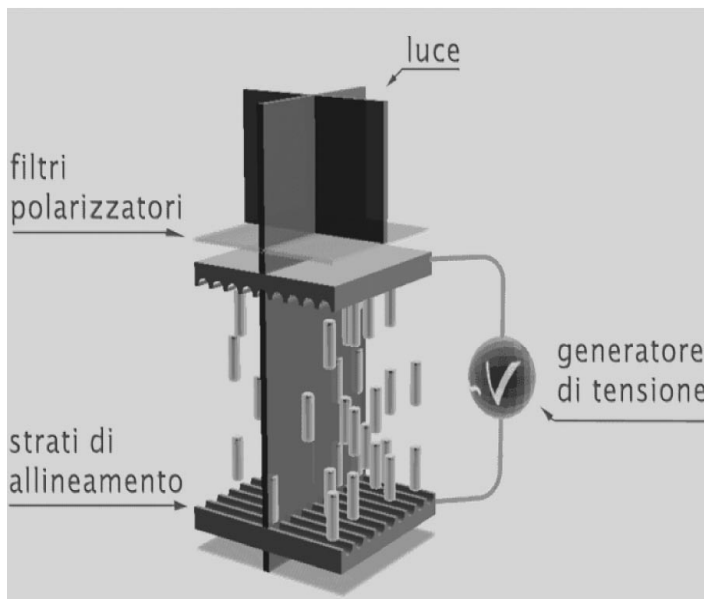


Fig.3



La programmazione.

Algoritmi

Tutte le volte che si vuole far risolvere un problema dal computer occorre fornire ad esso dei dati e dei comandi (istruzioni) e il pc ci fornirà dei risultati.

Un **problema** è sempre caratterizzato da una situazione non completamente nota, di cui si cerca di conoscere determinati aspetti, avendo delle conoscenze iniziali .

Le conoscenze iniziali sono i dati in nostro possesso, il modo di ottenere i dati finali è la strategia risolutiva e i dati finali rappresentano l'obiettivo del problema.



Un esempio di matematica: determinare il perimetro di un quadrato , conoscendo l'area $S = 100 \text{ cm}^2$. Il dato iniziale è la superficie, la strategia risolutiva sono le due operazioni radice quadrata e prodotto, l'obiettivo è il perimetro.

Ma se il problema non è matematico , la strategia risolutiva sarà una serie di azioni elementari (es. spiegare ad uno straniero come raggiungere l'aeroporto di Caselle partendo da Mirafiori).

La strategia risolutiva in ogni caso si chiama **algoritmo** .

L'algoritmo è quindi una serie ordinata di azioni necessarie , affinché , partendo dai dati iniziali di un problema , si riesca ad arrivare alla soluzione.

L'algoritmo deve essere eseguibile , cioè deve possedere le tre seguenti proprietà:

1. essere finito
 - a. Considera un numero naturale
 - b. Aggiungi 1
 - c. Se la somma è $<$ di 10, torna al punto a)Non finito (es.):
 - a. Considera un numero naturale
 - b. Aggiungi 1
 - c. Torna al punto a)

2. non ambiguo
 - a. Considera 2 numeri x e y
 - b. Fa il prodotto
 - c. Se $x*y > 100$, va all'istruzione e)
 - d. Torna ad a)
 - e. Eleva $x*y$ al quadrato
 - f. Scrivi il risultato.

Il seguente algoritmo invece è ambiguo:

- a. Considera 2 numeri x e y
- b. Fa il prodotto
- c. Se $x*y$ è grande ,va all'istruzione e) (ci si chiede cosa vuol dire grande)
- d. Torna ad a)
- e. Eleva $x*y$ al quadrato
- f. .Scrivi il risultato.

3. generalizzabile

Se vale in una situazione , dovrà valere in tutte le situazioni simili. Es. la regola per trovare il minimo comune multiplo vale per qualunque numero intero.

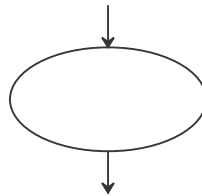
Una volta analizzato il problema e trovato l'algoritmo necessario a risolverlo, occorre descrivere la sequenza di azioni che si devono fare.

Diagramma di flusso.

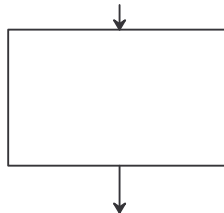
L'algoritmo può essere descritto da un'espressione matematica, oppure da una sequenza di frasi (discorsivo) oppure da **uno schema grafico che visualizza il fluire delle diverse operazioni da compiere. Lo schema grafico si chiama diagramma di flusso.**

Il diagramma di flusso (flow chart) utilizza una serie di simboli convenzionali di cui i principali sono

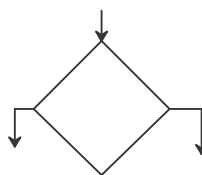
Inizio_ Fine



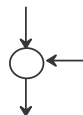
Blocco d'azione



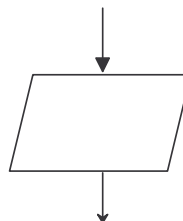
Blocco di controllo
o di selezione



collegamento



Blocco di input/output
(per immettere dati iniziali dall'esterno e per visualizzare o stampare i dati finali)

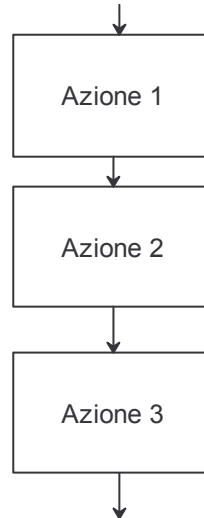


Ne esistono altri , ma per ora per i diagrammi di flusso base sono sufficienti.

Strutture fondamentali .

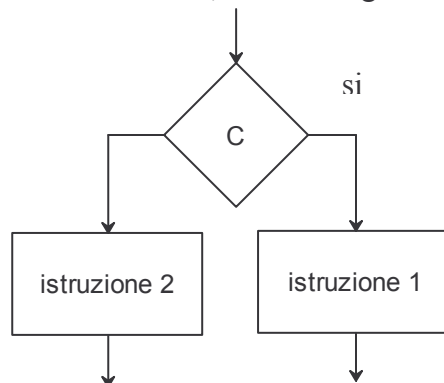
Le strutture fondamentali di un algoritmo e quindi di un diagramma di flusso sono 3 (nell'analisi di un problema ci possono essere tutte o solo qualcuna):

1. **sequenza** Le azioni sono eseguite una di seguito all'altra nell'ordine in cui sono state scritte (questo è lo schema che il computer segue , se non viene specificato altrimenti).

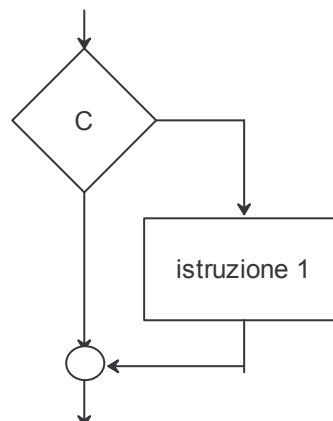


2. **selezione** E' un'istruzione di selezione o di controllo che permette di scegliere tra due percorsi differenti; può essere di due tipi:

a. Se è soddisfatta la condizione c, allora svolgi l'istruzione 1 , se no, allora la 2.

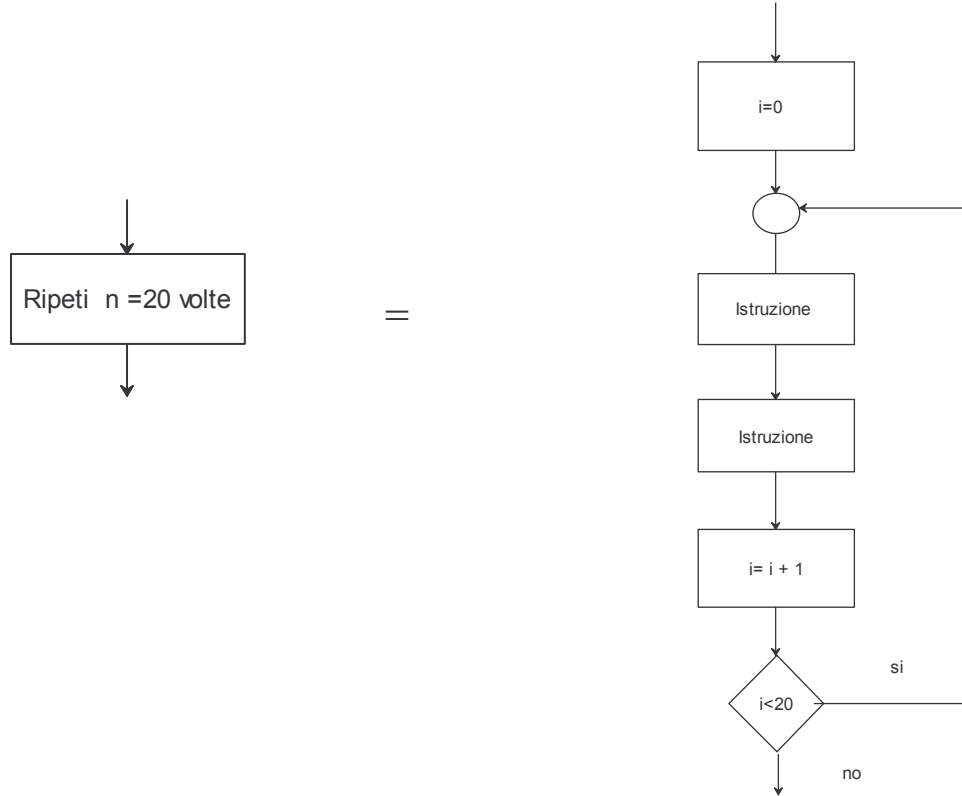


b. se è soddisfatta la condizione C, allora esegui l'istruzione 1, se no , prosegui il programma



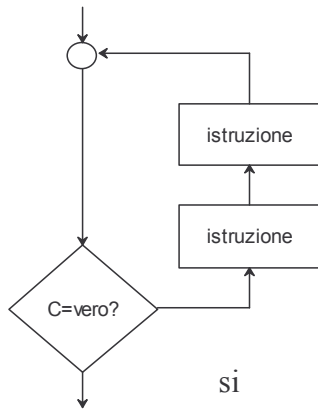
3. **iterazione o ciclo** si ripetono una o più istruzioni un certo numero di volte. Può essere di due tipi:

a. Possiamo conoscere il numero di volte per cui un ciclo si ripete

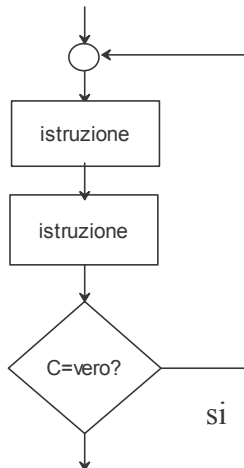


b. Non si conosce il numero di cicli, ma si esce su condizione:

i. con un controllo di testa (il controllo è prima delle istruzioni)



ii. o con un controllo di coda. (il controllo è dopo le istruzioni)



Esistono software appositi (tipo Diagram Designer software libero , scaricabile da Internet) per disegnare i diagrammi di flusso.

Per maggiori dettagli sui diagrammi di flusso e il loro uso consultare il libro in adozione al 3 anno Paolo Guidi_ Sistemi automatici per l'elettronica e l'elettrotecnica vol. 1_Zanichelli Unità 7: Tecniche di programmazione.

Linguaggi di programmazione

Una volta trovato l'algoritmo e disegnato il diagramma di flusso occorre trasformarlo in un linguaggio comprensibile dal PC. **Un linguaggio è un insieme di parole collegate tra di loro da regole sintattiche.** Di linguaggi ad **alto livello** (vicini al nostro linguaggio, non al binario del PC) ne esistono tanti (C,C++, Basic, Pascal, HTML, Java) ciascuno adatto per un determinato compito (dall'analizzare problemi finanziari o matematici, allo scrivere e disegnare un sistema operativo o un pacchetto applicativo, es. Word, al costruire un sito web, ancora a comandare un microcontrollore che programma la lavatrice, o un PLC che controlla il braccio meccanico di una catena di montaggio). Per poter programmare è necessario conoscere un linguaggio di programmazione (cioè le parole, dette istruzioni, e le regole sintattiche) e avere gli strumenti software adatti a tradurre le istruzioni in linguaggio macchina (cioè nella sequenza di numeri binari che il PC può interpretare).

Gli strumenti software si chiamano IDE (Integrated Development Environment) cioè ambiente di sviluppo integrato e sono un insieme di programmi raccolti in un unico software che permettono di:

1. scrivere il programma (un **editor** di testo non formattato come notepad; Word è un editor formattato cioè inserisce altri caratteri oltre a quelli che noi scriviamo ad es. l'allineamento a destra a sinistra , testo giustificato).Il file ottenuto si chiama **file sorgente**.
2. tradurre le istruzioni in un codice interpretabile dal PC .Questa operazione viene svolta da un **compilatore** che crea un file che si chiama **file oggetto**.
3. trovare (contemporaneamente alla compilazione) gli errori lessicali e sintattici che il programmatore ha fatto .Il programma si chiama **debugger**.
4. inserire le librerie (cioè parti di programma standard che non è necessario riscrivere tutte le volte) che il programmatore ha richiesto di includere . Il programma si chiama **linker** e, oltre ad aggiungere le librerie, trasforma il file oggetto in **file eseguibile**.

A questo punto si può far "girare" (to run) il programma cioè vedere se svolge il compito che il programmatore voleva far svolgere.

Il compilatore trova gli errori sintattici e lessicali, ma non quelli **logici** . Se il programmatore ha sbagliato la formula o se non ha messo nel giusto ordine le azioni il risultato verrà sbagliato, perchè il PC è solo un esecutore. Occorre quindi verificare il programma per tutte le possibili soluzioni.

Vedi il libro in adozione al 3 anno Paolo Guidi_ Sistemi automatici per l'elettronica e l'elettrotecnica vol. 1_Zanichelli Unità 7: Tecniche di programmazione

Il linguaggio di programmazione C++

Il linguaggio C++ deriva dal linguaggio C (un linguaggio ad alto livello che permetteva anche interazioni con la CPU) e, pur conservando le istruzioni e i vantaggi del precedente, permette **la programmazione ad oggetti** cioè una programmazione in cui è possibile costruire dei moduli "sicuri"(oggetti) facilmente riutilizzabili.

Sono spiegate solo le istruzioni base necessarie per codificare le strutture indicate nella parte sui diagrammi di flusso.

Prima di andare ad analizzare le principali istruzioni in C++, bisogna definire che cosa sono le **variabili** e quali **tipi di variabili** sono usate dal PC.

Variabili.

In **matematica e in fisica** una **variabile** è un carattere alfabetico che rappresenta un numero arbitrario, non completamente specificato o del tutto sconosciuto. Ad es. nell'equazione di una retta $y=4x+5$ x e y sono 2 variabili a cui io posso assegnare tutti i valori che soddisfano l'equazione. Si dice anche che una variabile può assumere tutti i valori possibili nell'intervallo di definizione. In **informatica** una variabile individua una porzione di memoria (una o più locazioni di memoria) destinata a contenere dei dati, suscettibili di modifica nel corso dell' esecuzione di un programma, cioè è "l'indirizzo" della cella di memoria in cui c'è o verrà inserito un valore . Una **variabile** è un **nome** (solitamente come una sequenza di caratteri e cifre es **voto, media1, b, A123**).

Ci possono essere variabili di tipo diverso : un numero intero, uno con la virgola, un carattere alfanumerico. A seconda del linguaggio di programmazione usato e della lunghezza del numerosi hanno nomi differenti. Sotto è allegata una tabella con i tipi di variabili usate.

Costanti . In **informatica** una costante é un dato non modificabile situato in una porzione di memoria (una o più locazioni di memoria) .

Rappresentazione dei numeri sul PC

Numeri interi. Il numero di bit con cui viene rappresentato dipende dal tipo di applicazione odi linguaggio che si sta usando.

Tipi interi senza segno:

N° byte(occupato in memoria)	intervallo di valori della variabile	nome in C o C++
1byte	0-255	Unsigned char (anche i caratteri alfanumerici)
2byte	0-65535	Unsigned int
4byte	0-4 294 964 295	Unsigned long

Tipi interi con segno:

N° byte (occupato in memoria)	intervallo di valori della variabile	nome in C o C++
1byte	-128-+127	Char (serve anche per i caratteri alfanumerici)
2byte	-32768-+32767	Int
4byte	-2 147 483 648- +2 147 483 647	Long

Numeri non interi. Il numero di bit con cui viene rappresentato dipende dal tipo di applicazione che si sta usando. I numeri **negativi** sono scritti in modulo e segno.

I numeri non interi si dicono anche numeri in **virgola mobile (floating point)** perché in forma normalizzata sono scritti come 0,..... un numero di cifre che corrisponde alla precisione moltiplicato per una potenza di 10 (ad es. 75,46 viene salvato in memoria come $0,7546 * 10^2$) .La parte dopo la virgola si chiama **mantissa** e l'esponente della potenza di 10 **caratteristica**. Se il numero non è normalizzato si può normalizzare come nell'esempio..

Tipi di numeri non interi:

N° byte (occupati in memoria)	segno(bit)	mantissa (bit)	caratteristica(bit)	nome in C o C++
4 (32bit)	1 +	23 +	8	Float
8 (64bit)	1 +	52 +	11	Double
10(80bit)	1 +	64 +	15	Long Double

Float.

I numeri in virgola mobile utilizzano 4 byte 3 byte per la mantissa e 1 per la caratteristica :

1° byte	2° byte	3° byte	4 ° byte
S mantissa	mantissa	mantissa	caratteristica

S indica il segno : 0 se positivo, 1 se negativo.

Istruzioni in C++

La funzione main ()

Nella codifica (cioè nella scrittura di un programma in un determinato linguaggio di programmazione), le istruzioni (gli enunciati) sono racchiuse tra due parentesi graffe che indicano l'inizio e la fine del programma . In C++ la funzione principale si chiama `int main()`

```
int main ( )
{.....
.....
.....}
```

}

Le parentesi graffe (non essendo presenti su tastiera)si ottengono con :

{ Alt+123 (tastierino numerico) oppure Alt Gr+shift+[
} Alt+125 (tastierino numerico) oppure Alt Gr+shift+]

Al di fuori del main, prima, bisogna indicare quali librerie includere nel programma e i tipi di tutte le variabili utilizzate nel programma

Le istruzioni di input (cin>> ...) output (cout<<.....)

Se si desidera inserire da tastiera un valore occorre spiegare all'utente cosa deve inserire (cioè visualizzare sullo schermo una scritta) e poi richiedere il valore da tastiera .Il programma sarà:

```
#include <iostream> //libreria per il flusso di dati di input output
using namespace std; // direttiva per l'utilizzo dei comandi standard
int b; //dichiarazione variabili intere
main () //inizio programma principale
{cout<<"inserisci numero biro"; //visualizza sullo schermo la frase tra " "
cin>>b; //inserisce da tastiera il valore della variabile b
.....
.....
}
```

NB.

1. Tutti gli enunciati finiscono con un ; che ha lo scopo di indicare al compilatore la fine dell'istruzione. Se manca il ; sulla riga successiva ci deve essere una parentesi { per indicare che l'istruzione continua su più righe.
2. il C++ è “**case sensitive**” cioè distingue tra maiuscolo e minuscolo ; le parole riservate (cioè ad es cout) sono sempre **minuscole**. La variabile **A** è diversa dalla variabile **a**.

cout visualizza sullo schermo la scritta tra “ “

cin aspetta e preleva da tastiera il valore della variabile inserito.

Sono state aggiunte 3 righe di codice:

1. #include<iostream> Il C++ non ha istruzioni dirette per controllare la tastiera e lo schermo. iostream è una libreria (un insieme di parti di programma (funzioni, oggetti) già costruite che così non si deve riscrivere)
2. using namespace sdt; istruzione che serve a dire che utilizzo i nomi standard per l'input output
3. int b; questo enunciato si chiama dichiarazione e serve a dire al PC che ci sono 2 byte (è un intero) in memoria per la variabile che si chiama b.

Dopo i ; ci sono // (doppio slash) e poi un testo; il testo è un commento (una spiegazione di cosa fa l'istruzione).

I **commenti** hanno valore soltanto per il programmatore e vengono ignorati dal compilatore. E' possibile inserirli nel proprio codice in due modi diversi:

1. racchiudendoli tra i simboli /* e */
2. facendoli precedere dal simbolo //

Nel primo caso e` considerato commento tutto quello che e` compreso tra /* e */ , il commento quindi si puo` estendere anche su piu` righe e trovarsi in mezzo al codice.

Nel secondo caso, proprio del C++, e` invece considerato commento tutto cio` che segue // fino alla fine della linea, ne consegue che non e` possibile inserirlo in mezzo al codice o dividerlo su piu` righe (a meno che anche l'altra riga non cominci con //).

Istruzioni per l'uso di un ambiente di sviluppo per il C++.

L'IDE, l'ambiente di sviluppo utilizzato, è il Dev C++ scaricabile liberamente da internet. Si apre il programma e si crea un nuovo file sorgente (attenzione non un progetto, perchè è più complicato). Si scrive il programma, lo si salva (con estensione .cpp) lo si compila e, se ci sono degli errori di sintassi, questi appaiono nella parte inferiore dello schermo. Si corregge e si risalva. Quando il programma è giusto si può eseguire: a questo punto appare un piccolo schermo nero in cui si può provare il programma e verificare i risultati (ci possono ancora essere degli errori logici)..

Sotto è scritto un programma che utilizza le istruzioni viste fin qui e altre da spiegare (si può provare su DevC++):

```
/* Cartolaio.cpp Uno studente acquista dal cartolaio quaderni e biro. Costruire un programma che, ricevendo in ingresso il costo e il numero di quaderni e di biro acquistate, stampi (visualizzi) l'importo complessivo */
```

```
#include <iostream>           //libreria per il flusso di dati di input output
using namespace std;        // direttiva per l'utilizzo dei comandi standard
int b,q;                    //dichiarazione variabili intere
float cb,cq,importo;        //dichiarazione variabili reali ( con la virgola)
main ()                      //inizio programma principale
{
    cout<<"inserisci num biro"; //scritta sullo schermo
    cin>>b;                    //immissione da tastiera del valore di una variabile
    cout<<"inserisci num quaderni"; //scritta sullo schermo
    cin>>q;                    //immissione da tastiera del valore di una variabile
    cout<<"inserisci costo biro e costo quaderni "; //scritta sullo schermo
    cin>>cb>>cq;              //immissione da tastiera del valore di due variabili
    importo= b*cb+q*cq;      // assegna alla variabile importo il risultato dell'operazione
    cout<<endl<<" l'importo e'="<<importo<<endl; /*visualizza sullo schermo su una nuova linea
                                                (èndl) la frase tra " " e il valore di importo*/
    system("PAUSE");        //aspetta la pressione di un tasto per chiudere la schermata del programma
}                            // Chiude il programma principale
```

NB: Tutti i programmi devono avere come commento all'inizio il testo del problema e il nome del file sorgente (in questo caso cartolaio. cpp).

Tutte le volte che si apre una parentesi graffa le istruzioni vengono scritte più a destra fino alla chiusura : questo modo di scrivere un programma si chiama **indentazione** e permette una maggior visibilità del flusso di programma con la visualizzazione dei blocchi di istruzioni che vengono trattate insieme.

Le 4 istruzioni nuove sono:

1. **float cb,cq, importo** ; tutte le variabili utilizzate devono essere dichiarate e bisogna anche definirne il tipo (queste sono con la virgola)
2. **importo= b*cb+q*cq;** è un calcolo , ma dal punto di vista della programmazione è un'istruzione di **assegnazione** cioè l' '=' indica che il risultato del calcolo viene posto nella variabile (meglio nelle celle di memoria di nome importo) importo. In questa istruzione il simbolo = sembra avere lo stesso significato di = in matematica. Ma in un programma si può anche scrivere $a=a+1$ espressione che in matematica è sbagliata : qui vuol dire che la a che sta a destra dell'uguale viene aumentata di 1 e il risultato viene di nuovo posto nella variabile a ,quella a sinistra dell'=. (se a valeva 2 prima di quest'istruzione , dopo vale 3)
3. **cout<<endl<<" l'importo e'="<<importo<<endl;** endl (end line) indica solo che scrive il risultato su una nuova riga .E' una visualizzazione sullo schermo un po' più complessa: apparirà l'importo = valore importo. E' questo il cout tipico per la visualizzazione dei risultati.

4. **system("PAUSE");** Il programma gira molto velocemente e quando finisce, il piccolo schermo aperto durante il programma, scompare ; per poter vedere i risultati occorre fermare il programma.

Il programma sopra scritto ha una struttura (per quanto riguarda il diagramma di flusso) di sequenza, cioè le istruzioni sono l'una sotto l'altra nel giusto ordine.

La selezione

Se il problema richiede di scegliere tra due algoritmi o azioni differenti occorre un'altra istruzione **if.....else**. La sintassi dell'istruzione ha due possibili formulazioni (vedi i due tipi di struttura della selezione nei diagrammi di flusso):

```
if (Condizione)
    Istruzione1 ;
```

oppure

```
if ( Condizione )
    Istruzione1 ;
else
    Istruzione2 ;
```

L'**else** e` quindi opzionale e quindi se la condizione è falsa si esegue direttamente la prima istruzione successiva del programma. Se invece l'**else** viene utilizzato nessuna istruzione deve essere inserita tra il ramo **if** e il ramo **else** . In entrambi i casi se **Condizione** e` vera viene eseguita **Istruzione1**, altrimenti nel primo caso non viene eseguito alcunchè e il programma va avanti, nel secondo caso invece si esegue **Istruzione2**.

Un esempio:

```
if ( X==10 )
    cout<<"hai vinto";
else
{
    // le istruzioni sono 2 quindi occorre aprire la {
    Y=Y+1;
    cout "hai perso";
}
```

NB. Gli operatori presenti nella condizione possono essere >, <, >=, <=, =, != (diverso) o anche un'espressione matematica o logica. In questo caso = non è un'operazione di assegnazione, cioè 10 non viene posto nella cella denominata X, ma si **confronta** quello che è nella cella X con 10 ; se il risultato è vero si esegue l'istruzione successiva, altrimenti si passa all'**else**.

È importante sottolineare che, quando un ramo if o un ramo else è composto da più di un'istruzione, è necessario racchiudere tutte le istruzioni che ne fanno parte all'interno di una coppia di parentesi graffe, in modo da formare un blocco.

Un programma con selezione:

```
/* Acqua.cpp. Per la fatturazione della bolletta dell'acqua si calcolano:
15 euro fissi per consumi non superiori ai 40m^3 , 0,80 euro a m^3 oltre i 40, sul totale un'IVA del
19%. Calcolare il totale e visualizzarlo */
```

```
#include <iostream>
using namespace std;
float mcubi, importo, IVA;
int main()
{ cout<<endl<<" Inserisci i mcubi";
  cin>>mcubi;
  if (mcubi>40.00) //se mcubi> 40.00
```



```

    importo= 15+(mcubi-40)*0.8; // allora si somma al fisso il costo dei mcubi in più
else
    //altrimenti
    importo=15; //l'importo è solo il fisso
IVA=importo*19/100;
importo=importo+IVA;
cout<<endl<<" Importo= "<<importo<<endl;
system("pause");
} //le istruzioni già spiegate non sono commentate.

```

Istruzioni iterative.

Facciamo il caso di dover sommare 10 numeri forniti da tastiera . Con le istruzioni fornite fino a questo punto l'unico programma che possiamo scrivere è

```

#include <iostream>
using namespace std;
float a,b,c,e,f,g,h,i,l,m, totale;
int main()
{
    cout<<endl<<" Inserisci numero";
    cin>>a;
    cout<<endl<<" Inserisci numero";
    cin>>b;
    cout<<endl<<" Inserisci numero";
    cin>>c;
    cout<<endl<<" Inserisci numero";
    cin>>d;
    ..... inserisco gli altri numeri
    .....
    .....
    Totale=a+b+c+d+e+f+g+h+i+l+m;
    cout<<" Totale= "<<totale;
    system("pause");
}

```

Cioè ci occorrono 11 variabili e una lunga sequenza di istruzioni per l'inserimento dei valori.

Ma si può usare un altro modo (algoritmo) per fare la somma, quella che si chiama **Somma ricorsiva** (è il modo in cui noi facciamo lunghe somme quando non abbiamo a disposizione un foglio di carta).

Si tiene conto che si parte da 0 (cioè si azzerava una variabile che conterrà la somma). Si prende il primo numero e lo si somma al secondo. Si ricorda solo la somma . A questa si aggiunge il terzo numero. Di nuovo si ricorda la somma e avanti così. C'è sempre solo la somma parziale e un numero da ricordare. Dal punto di vista informatico è

$S=0$ Inserisci val $S=S+val$ Inserisci val $S=S+val$ Inserisci val $S=S+val$ per 10 volte.

Ci sono solo 2 variabili S e val e 2 istruzioni che si ripetono 10 volte .

Istruzione for.

L'istruzione iterativa che permette di ripetere un numero conosciuto di volte una parte del programma è il **for** la cui sintassi è

for (espressione 1; espressione 2; espressione 3)

```

{.....
.....
}

```

dove espressione1 è la condizione iniziale, l'espressione2 è la condizione finale e l'espressione 3 è l'incremento. Nei casi più semplici diventa:

```
for ( i=0; i<n; i=i+1)
{.....
.....
}
```

che significa: per **i** che parte da 0 fino a quando **i<n**, con un **incremento della i di 1**, ripeti la parte di programma tra le parentesi graffe. L'intero **i** si chiama **contatore** e serve a contare le ripetizioni del for.

Con quest'istruzione il programma sulla somma di 10 termini diventa:

```
// somma.cpp Somma 10 numeri dati da tastiera.

#include <iostream>
int i,val,somma = 0;//le variabili possono essere inizializzate durante la dichiarazione
int main()
{
    cout << "Inserire 10 valori da sommare:" << endl;
    for (i = 0; i < 10; i++) //ripeto le istruzioni tra le { }per 10 volte
        { cin >> val;
          somma = somma + val;
        }
    cout << "somma = " << somma << endl;
    system ("Pause");
}
```

Se non conosco quanti valori bisogna inserire si può chiedere la quantità da tastiera prima del for.

```
// somma1.cpp Somma gli n numeri dati da tastiera.

#include <iostream>
int i,n,val, somma = 0;//le variabili possono essere inizializzate durante la dichiarazione
int main()
{
    cout << "inserire quanti valori sommare" << endl;
    cin>>n;
    cout << "Inserire gli n valori da sommare:" << endl;
    for (i = 0; i < n; i++)
        { cin >> val;
          somma = somma + val;
        }
    cout << "somma = " << somma << endl;
    system ("Pause");
}
```

Questo programma è adattabile a più situazioni del precedente, perchè posso cambiare il numero di valori..

Istruzione while.

Esistono però delle situazioni reali in cui non è possibile conoscere il numero di valori prima di fare la somma (es. la cassiera del supermercato non conosce quanti sono i prodotti prima di cominciare a inserire i prezzi).

In questo caso l'istruzione iterativa si chiama while e la codifica è:

```

a=3;
while( a==3) //fino a quando a=3 ripeti le istruzioni tra le { }
{.....
.....
cout<<"Inserisci 3 se vuoi ripetere il ciclo, altro per uscire";
cin>>a;
}

```

Bisogna prima inizializzare una variabile che serve come condizione per ripetere il ciclo; poi si inizia il ciclo che si ripeterà fino a quando a=3. Per poterlo interrompere, bisogna poter cambiare il valore di a (è il cin >>a interno). L'operatore della condizione può essere >, <, >=, <=, !=, = o una qualunque espressione composta.

Se il problema è quindi sommare un numero di valori che non si può conoscere a priori il programma diventa:

// somma2.cpp Somma un numero di valori inseriti da tastiera non conosciuto a priori.

```

#include <iostream>
int i,val, a=3, somma = 0; //le variabili possono essere inizializzate durante la dichiarazione
int main()
{ i=0; //è un contatore che serve per contare il numero di valori che non conosco.

while( a==3) // mentre a=3 ripeti le istruzioni tra le graffe.
{cout << "inserire il valore" << endl;
cin >> val;
somma = somma + val;
i=i+1; // il numero di valori è il numero di ripetizioni del ciclo
cout<<"Inserisci 3 se vuoi ripetere il ciclo, altro per uscire";
cin>a;
}
cout << "somma = " << somma << endl;
cout << "il numero di valori è = " << i << endl; //nella i è contenuto il numero di valori inserito
system ("Pause");
}

```

Variabili strutturate

Tante volte ci sono molti dati dello stesso tipo (costi, numeri, iniziali di prodotti etc) e in questo caso si possono aggregare in una sola variabile chiamata **vettore (array)**.

A differenza di una variabile, che contiene un singolo valore, un array indica una variabile (o meglio un contenitore) che fa riferimento ad un insieme **omogeneo** di valori/oggetti. Ogni array ha un **nome** al quale viene associato un **indice** che individua **i singoli elementi** dell' array. E' possibile definire array di tipo: carattere, intero, float, double etc..

Le proprietà fondamentali di un array sono:

1. Gli oggetti che compongono l'array sono denominati elementi;
2. Tutti gli elementi di un array devono essere dello stesso tipo;
3. Tutti gli elementi di un array vengono memorizzati uno di seguito all'altro nella memoria del PC e l'indice del primo elemento è 0;
4. Il nome dell' array è un variabile che rappresenta l'indirizzo di memoria del primo elemento dell' array stesso.

Per dichiarare un array (analogamente ad una qualsiasi variabile) in C++ è necessario definirne il tipo, seguito da un nome valido e da una coppia di parentesi quadre che racchiudono un'espressione costante rappresentante la dimensione massima dell' array. Es.

```
int prova[100]; // Un array di nome prova di 100 interi
char pippo [20]; // Un array di nome pippo di 20 caratteri
```

Si noti che all'interno delle parentesi quadre non è possibile, in fase di dichiarazione dell' array, utilizzare nomi di variabili, perché la dichiarazione serve a definire il numero di locazioni di memoria occupate e quindi occorre una quantità definita o una costante (potrebbe essere anche una lettera, ma definita come costante).

Un limite dei programmi di somma ricorsiva precedenti con for e while è che le istruzioni iterative non salvano i valori (cioè se devo sommare 10 numeri utilizzo una sola variabile es. **a** per inserire da tastiera i 10 numeri ; alla fine in **a** mi rimane l'ultimo numero inserito, i 9 precedenti non sono stati conservati o meglio sono stati sovrascritti). I vettori risolvono questo problema : questo vuol dire che i valori sono conservati nella memoria del computer e si possono utilizzare per altre operazioni o per visualizzarli in una lista . Ripetiamo il programma con la somma di n valori inserendoli in un vettore e visualizziamo anche la lista dei valori.

```
// somma3.cpp Somma numeri dati da tastiera. Visualizza la somma e l'elenco dei valori.
```

```
#include <iostream>
int i,n,val[50], somma = 0;// si indica un numero di elementi del vettore opportuno(dipende dalla situazione)
int main()
{
    cout << "inserire quanti valori sommare" << endl;
    cin>>n;
    cout << "Inserire gli n valori da sommare:" << endl;
    for (i = 0; i < n; i++)
        { cin >> val[i]; //tra le [] adesso si inserisce i , così ad ogni giro i incrementa di 1 e incrementa di 1
          // anche la posizione dell'elemento in memoria in cui si vuole salvare il valore
          somma = somma + val [i];
        }
    cout << "somma = " << somma << endl;
    cout<<"Lista dei valori"<<endl: // titolo della lista
    for (i=0; i<n; i=i+1) //istruzione iterativa per visualizzare i valori
        cout<<val [i]<<endl; // endl (end line) serve per andare a capo dopo aver scritto un valore,
        // se non ci fosse i valori verrebbero visualizzati attaccati su un'unica riga.
    system ("Pause");
}
```

e lo stesso programma con il while:

```
/*somma5.cpp Somma un numero di valori inseriti da tastiera non conosciuto a priori. Visualizza la somma e l'elenco dei valori.*/
```

```
#include <iostream>
int i,val[50], a=3 somma = 0;//si indica un numero di elementi del vettore opportuno ( adatto al problema)
int main()
{ i=0; //è un contatore che serve per contare il numero di valori che non conosco.

    while( a==3)
        {cout << "inserire il valore" << endl;
          cin >> val[i];
          somma = somma + val[i];
          i=i+1; // il numero di valori è il numero di ripetizioni del ciclo
```

```

        cout<<"Inserisci 3 se vuoi ripetere il ciclo, altro per uscire";
        cin>a;
    }
    n=i; /* devo salvare il valore di i (contiene il numero di valori) in un'altra variabile n , perchè nel for
        successivo riassetto i e non so più quanti sono i valori.*/
    cout << "somma = " << somma << endl;
    cout<<"Lista dei valori"<<endl;
    for (i=0; i<n; i=i+1)
        cout<<val [i]<<endl;    // endl (end line) serve per andare a capo dopo aver scritto un valore
    system ("Pause");
}

```

NB. La visualizzazione dell'elenco si può fare sempre con il for , perchè a questo punto del programma si conosce il numero di elementi.

Le variabili carattere

Ogni simbolo alfanumerico della tastiera corrisponde all'interno della CPU a un numero in binario a 8 bit secondo il codice ASCII (vedi tabella). Il tipo di variabile per un simbolo alfanumerico si chiama **char** (1 byte). Per il PC i caratteri alfanumerici e i corrispondenti codici ASCII sono intercambiabili; solo nella visualizzazione (cioè sullo schermo) occorre distinguere. Es.

/*Visualizzazione dei numeri, delle lettere maiuscole e minuscole e del corrispondente codice ascii */

```

#include <iostream>
using namespace std;
int i;
int main ( )
{   for (i=48;i<91; i++)    // visualizza il carattere e il corrispondente ASCII in decimale ( 48 è=0 ,
                        // 90 =Z vedi tabella ASCII successiva
        cout<<(char)i <<" ="<<i <<endl;// andando a capo per ogni carattere
    system ( "Pause");
}

```

Tabella codice ASCII esteso

Nella tabella seguente ci sono tutti i simboli alfanumerici presenti sulla tastiera (e anche altri che si possono visualizzare con Alt+il numero ASCII corrispondente sul tastierino numerico); ad ogni simbolo corrisponde un numero da 0 a 255 in decimale o da 0 a FF in esadecimale.

N.B. $2^8 = 256$ è il numero max che si può scrivere in un byte (8 bit) ed è la dimensione del tipo **char**.

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char						
0	00	Null	32	20	Space	64	40	@	96	60	`	128	80	Ç	160	A0	á	192	C0		224	E0	
1	01	Start of heading	33	21	!	65	41	A	97	61	a	129	81		161	A1		193	C1		225	E1	
2	02	Start of text	34	22	"	66	42	B	98	62	b	130	82		162	A2		194	C2		226	E2	
3	03	End of text	35	23	#	67	43	C	99	63	c	131	83		163	A3		195	C3		227	E3	
4	04	End of transmit	36	24	\$	68	44	D	100	64	d	132	84		164	A4		196	C4		228	E4	
5	05	Enquiry	37	25	%	69	45	E	101	65	e	133	85		165	A5		197	C5		229	E5	
6	06	Acknowledge	38	26	&	70	46	F	102	66	f	134	86		166	A6		198	C6		230	E6	
7	07	Audible bell	39	27	'	71	47	G	103	67	g	135	87		167	A7		199	C7		231	E7	
8	08	Backspace	40	28	(72	48	H	104	68	h	136	88		168	A8		200	C8		232	E8	
9	09	Horizontal tab	41	29)	73	49	I	105	69	i	137	89		169	A9		201	C9		233	E9	¡
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j	138	8A		170	AA		202	CA		234	EA	¢
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k	139	8B		171	AB		203	CB		235	EB	£
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l	140	8C		172	AC		204	CC		236	EC	¤
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m	141	8D		173	AD		205	CD		237	ED	¥
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n	142	8E		174	AE		206	CE		238	EE	¦
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o	143	8F		175	AF		207	CF		239	EF	§
16	10	Data link escape	48	30	0	80	50	P	112	70	p	144	90		176	B0		208	D0		240	F0	¨
17	11	Device control 1	49	31	1	81	51	Q	113	71	q	145	91		177	B1		209	D1		241	F1	©
18	12	Device control 2	50	32	2	82	52	R	114	72	r	146	92		178	B2		210	D2		242	F2	ª
19	13	Device control 3	51	33	3	83	53	S	115	73	s	147	93		179	B3		211	D3		243	F3	«
20	14	Device control 4	52	34	4	84	54	T	116	74	t	148	94		180	B4		212	D4		244	F4	¬
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u	149	95		181	B5		213	D5		245	F5	­
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v	150	96		182	B6		214	D6		246	F6	®
23	17	End trans. block	55	37	7	87	57	W	119	77	w	151	97		183	B7		215	D7		247	F7	¯
24	18	Cancel	56	38	8	88	58	X	120	78	x	152	98		184	B8		216	D8		248	F8	°
25	19	End of medium	57	39	9	89	59	Y	121	79	y	153	99		185	B9		217	D9		249	F9	±
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z	154	9A		186	BA		218	DA		250	FA	²
27	1B	Escape	59	3B	;	91	5B	[123	7B	{	155	9B		187	BB		219	DB		251	FB	³
28	1C	File separator	60	3C	<	92	5C	\	124	7C		156	9C		188	BC		220	DC		252	FC	´
29	1D	Group separator	61	3D	=	93	5D]	125	7D)	157	9D		189	BD		221	DD		253	FD	µ
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~	158	9E		190	BE		222	DE		254	FE	¶
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F		159	9F		191	BF		223	DF		255	FF	·