

Rete Nazionale Book in Progress

Informatica

INFORMATICA 1

BOOK IN PROGRESS

CODIFICA DELL'INFORMAZIONE

A cura dei proff.
Alessandra D'Orazio
Giorgia Martina
Paolo Lillo

Competenze	Abilità	Conoscenze
Saper codificare un'informazione numerica	Riconoscere i diversi tipi di informazioni	Dato e informazione
Saper codificare un'informazione alfanumerica		Analogico e digitale
Saper individuare le tecniche di codifica conformi ai dati coinvolti		Bit e byte
		Codifica dei dati

INDICE

- **Introduzione. Il bit** pag. 3
- **Codifica dei numeri** pag. 6
- **Codifica dei caratteri** pag. 14
- **Codifica delle immagini (fisse)** pag. 16
- **Codifica dell'audio** pag. 26
- **Codifica del video** pag. 27
- **Classificazione dell'informazione** pag. 28
- **Approfondimento: aritmetica binaria** pag. 30

Introduzione. Il bit

Diamo una definizione di "INFORMATICA":

"L'informatica è la disciplina che si occupa della **codifica**, della *memorizzazione*, della *trasmissione* e dell'*elaborazione* dell'INFORMAZIONE"

Intendiamo per INFORMAZIONE

"Ogni messaggio o elemento conoscitivo che è possibile comunicare o acquisire"



La parola e le espressioni del corpo, i segni (dai graffiti preistorici alla scrittura) e in generale tutto ciò che può essere percepito dai sensi, hanno permesso all'uomo nel corso della storia di comunicare, cioè di trasferire l'informazione non solo da uomo a uomo ma anche da generazione a generazione. I modelli culturali di oggi sono il risultato della lunga evoluzione del processo.

Se la comunicazione è alla base del trasferimento delle informazioni, il ragionamento e quindi l'uso della capacità logica-deduttiva hanno da sempre avuto il compito di elaborare l'informazione attraverso l'uso di modelli interpretativi che con lo scorrere del tempo sono divenuti sempre più articolati e raffinati.

L'uomo, del resto, si è sempre ingegnato per superare i suoi limiti utilizzando tecniche e strumenti per rendere sempre più efficace la sua capacità di comunicare: dall'uso dei tam-tam e dei segnali di fumo alle moderne tecnologie.

Il lettore avrà certamente sentito parlare del "codice Morse": quando nel 1838 si realizzarono i primi telegrafi via cavo (e successivamente via radio) gli unici "segnali" che era possibile produrre e distinguere in modo chiaro e immediatamente comprensibile erano due segnali acustici di lunghezza lunga e breve, rispettivamente linea e punto; la produzione dei segnali avveniva da un



lato della linea telegrafica (trasmettitore), agendo su un interruttore a molla: ad una

pressione breve corrispondeva un segnale del tipo punto mentre ad una pressione lunga faceva seguito un segnale del tipo linea; dall'altro lato della linea telegrafica il segnale elettrico ricevuto attivava un elettromagnete, il cui movimento portava in contatto una penna su una striscia di carta trascinata da un meccanismo a orologeria.

Il messaggio più famoso del codice Morse è certamente SOS, il segnale di richiesta di soccorso (“Save Our Souls” cioè “Salvate le nostre anime”) corrispondente alla sequenza:



dove i primi tre punti stanno per “S”, le tre linee stanno per “O” e gli ultimi tre punti stanno nuovamente per “S”. Punto e linea rappresentano quindi i 2 simboli grazie ai quali è possibile rappresentare i messaggi dell'alfabeto Morse.

Quindi, una volta stabilita la corrispondenza tra l'insieme di punti e linee e le lettere dell'alfabeto, le cifre, i caratteri di interpunzione e alcuni segni convenzionali (ovvero una volta stabilita la **codifica** nell'alfabeto Morse di ogni singolo carattere che compone un testo scritto), un qualsiasi brano può essere codificato utilizzando punti e linee; d'altro canto, chiunque abbia a disposizione una tabella di conversione

punti-linee → carattere

sarà in grado di leggere il testo originale.

L'alfabeto Morse è un esempio di CODIFICA; è chiaro che per la rappresentazione codificata dell'informazione è necessaria una CONVENZIONE (detta anche PROTOCOLLO), cioè un accordo tra le parti che devono comunicare sul SIGNIFICATO da dare alle sequenze di simboli: tanto più diffusa è tale convenzione tanto più estesa sarà la platea dei possibili fruitori.

L'introduzione dell'informatica ha avviato un'epoca in cui l'informazione e il trattamento dell'informazione, grazie all'evoluzione della tecnologia elettronica, hanno assunto forme sempre più automatiche. Al fine di potere essere “gestita” da processi automatici è stato indispensabile rendere l'informazione sempre più autonoma cioè indipendente dalla capacità interpretativa dell'uomo.

Prima di entrare nel vivo della trattazione sulla codifica dei più comuni tipi di informazione (spiegheremo tra un po' cosa intendiamo per “tipi”), soffermiamo la nostra attenzione sulla definizione e sulla codifica di un'informazione in ambiente più specificatamente informatico.

Prima domanda: se l'informazione è l'oggetto intorno a cui ruota la disciplina Informatica, che cosa è un dato e che relazione intercorre tra un dato e un'informazione?

Seconda domanda: nel campo dell'informatica, come si può memorizzare un'informazione e in quanti modi diversi è possibile rappresentare la stessa informazione?

Rispondiamo alla prima domanda: i termini **dato** e **informazione** sono a volte utilizzati come sinonimi, ma i due concetti non coincidono. Facciamo alcuni esempi:

- il numero intero 16: è un dato o un'informazione? È un dato, diventa un'informazione se aggiungo che 16 è l'età di una ragazza oppure che 16 sono gli amici che ho invitato alla mia festa.
- Maria ha i capelli rossi: è un dato o un'informazione? È un'informazione: il dato, colore rosso, è “diventato” un'informazione in quanto è stato associato ai capelli di Maria.

Quindi: un dato è una parte dell'informazione, la quale a sua volta può essere considerata come un dato a cui è associato il significato che ha nel contesto di studio. Più precisamente, possiamo considerare l'informazione costituita da tre elementi:

- il valore (il dato)
- il tipo (carattere, numero intero, stringa,...) utilizzato per esprimere il valore
- il significato (la semantica) da associare al valore

Se l'informatica si occupa, tra le varie cose, così come detto nella definizione iniziale, di memorizzare ed elaborare le informazioni, che cosa verrà trascritto ovvero memorizzato e che cosa verrà elaborato? La risposta è: i dati!

Possiamo ora rispondere alla seconda domanda, iniziando con il constatare che i dati sono memorizzati nelle memorie (di massa, centrale, RAM, ROM, ..) ed elaborati dai computer: memorie e computer sono supporti fisici costituiti da dispositivi in grado di distinguere tra due (e solo due) diversi valori di riferimento di una grandezza fisica (assenza/presenza di tensione, luminoso/non luminoso). Tali dispositivi sono detti elettricamente bistabili.

Se associamo a ciascun valore un simbolo, possiamo rappresentare i due diversi stati del dispositivo in esame con due simboli distinti; i due simboli utilizzati sono {0, 1}, che costituiscono l'alfabeto binario. Attenzione: 0 e 1 non sono numeri ma simboli, da non confondere quindi con i primi due numeri naturali.

A questo punto possiamo definire il concetto informatico di BIT (BInary digiT = cifra binaria):

“Il bit è l'unità elementare di informazione cioè la minima quantità di informazione codificabile”

Il bit è quindi il più piccolo dato memorizzabile in un elaboratore e può rappresentare

una sola informazione binaria. Qualsiasi altro tipo di informazione è rappresentato come sequenze finite di 0 e 1; come questi 2 simboli sono combinati tra loro dipende dal codice usato, ovvero dalla relazione che trasforma ogni dato in una stringa di bit.

Ma una sequenza di bit cosa rappresenta? A quale possibile informazione è associata? Nei paragrafi successivi daremo una risposta a questi quesiti; più precisamente affronteremo il problema di come è possibile codificare:

- numeri
- caratteri
- immagini
- suoni
- video

Codifica dei numeri

Iniziamo con il ricordare alcune definizioni:

- **Numero:** oggetto o ente astratto
- **Numerale:** insieme di simboli che rappresenta un numero in un dato sistema di numerazione
- **Sistema di numerazione:** un insieme di simboli e un insieme di regole; i simboli servono per rappresentare un numero, le regole stabiliscono come scrivere e come operare con i numerali.

Di conseguenza possiamo rappresentare un numero e scrivere il suo numerale solo dopo aver stabilito il sistema di numerazione che intendiamo utilizzare: se cambiamo sistema di numerazione, cambia la rappresentazione del numero.

Esempio: il numerale 15 nel sistema decimale diventa XV in numeri romani.

Esaminiamo allora due caratteristiche del sistema di numerazione che usiamo correntemente, il sistema decimale:

- è in base 10: ogni numero è rappresentato da una combinazione di 10 simboli diversi {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
- è posizionale: esiste una regola che stabilisce un legame tra il simbolo e la sua posizione nel numerale che rappresenta il numero, cioè ogni cifra assume un valore diverso a seconda della posizione occupata nella rappresentazione del numero.

Esempio: numerale = $184 \rightarrow 1 \cdot 10^2 + 8 \cdot 10^1 + 4 \cdot 10^0 = 100 + 80 + 4$
 posizione $\rightarrow$ 2 1 0

(N.B. Il simbolo di moltiplicazione è indicato con "" e non con "x", in conformità con la simbologia adottata in ambiente informatico)*

Come si può osservare, la posizione delle cifre in un numerale è tanto più importante, perché è quella che più contribuisce al valore finale (in gergo “più significativa”), quanto più a sinistra si trova nella sequenza numerica: la cifra MENO SIGNIFICATIVA è quella più a destra, che nel sistema decimale rappresenta le “unità”.

Il sistema decimale deve ad una proprietà anatomica (le 10 dita delle mani) la sua diffusione, ma non risulta altrettanto vantaggioso quando l'elaborazione dei numeri si sposta dall'uomo al computer.

Come già accennato nel paragrafo precedente, un computer in quanto macchina “sente” 2 diversi stati (acceso/spento, presenza/assenza di tensione); conseguenza quasi ovvia: per codificare i numeri che devono essere elaborati da un computer si utilizza il sistema di numerazione in base 2, ovvero il sistema di numerazione posizionale che ha il bit come elemento di base.

Prenderemo in esame altri 2 sistemi di numerazione posizionali, il sistema di numerazione in base 8 (ottale) e il sistema in base 16 (esadecimale), entrambi caratterizzati dal fatto che le basi sono potenze della base 2 (spiegheremo in seguito cosa comporta questa considerazione).

Due annotazioni:

- useremo il termine numero anche quando sarebbe più corretto utilizzare numerale
- ogni sequenza di simboli che scriveremo avrà come pedice la base in cui è espresso il numero.

Esempio: 215_{10} rappresenta un numero espresso in base 10

1011_2 rappresenta un numero espresso in base 2 (si legge **uno zero uno uno**)

Sistema di numerazione in base 2

Il sistema binario utilizza, come già detto, due simboli, 0 e 1, grazie ai quali possiamo rappresentare qualsiasi numero, purché si abbia a disposizione un numero di bit adeguati. Spieghiamo questa affermazione con una tabella in cui riportiamo tutti i valori rappresentabile con un determinato numero di bit

n° di bit	numeri rappresentabili															
1	0	1														
2	00	01	10	11												
3	000	001	010	011	100	101	110	111								
4	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

In generale con n bit si possono rappresentare interi positivi compresi nell'intervallo $[0, 2^n - 1]$.

Attenzione: una volta che è stato fissato n , occorre rappresentare anche gli 0 non significativi.

Sistema di numerazione in base 8

- è un sistema di numerazione posizionale
- ogni numero è espresso come combinazione degli 8 simboli: 0, 1, 2, 3, 4, 5, 6, 7

Sistema di numerazione in base 16

- è un sistema di numerazione posizionale
- ogni numero è espresso come combinazione dei 16 simboli:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Scriviamo i primi 21 numeri naturali in base 10, in base 2, in base 8, in base 16 (per semplicità, non ci preoccupiamo della lunghezza delle rappresentazioni nelle varie basi).

base 10	base 2	base 8	base 16
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14

Ma allora: se non abbiamo a disposizione la tabella precedente, come possiamo affermare che la sequenza 1101 in base 2 equivale a 13 in base 10 ($1101_2 = 13_{10}$)?

E soprattutto: come procediamo per valori maggiori di 20? 34710 in base 10 a quale valore corrisponde in base 2? e in base 16? e in base M?

Esiste una tecnica, semplice e intuitiva, che permette di convertire un numero espresso in base 10 nell'equivalente in un'altra base M (in effetti è una tecnica che permette la conversione da una qualsiasi base N a una qualsiasi altra base M). Ricordiamo che stiamo trattando numeri naturali, quindi senza parte decimale e senza segno.

La regola di conversione di un numero X in base 10, X_{10} , nella base M è la seguente:

1. si esprime M in base 10 (indichiamo il numero ottenuto con M_{10})
2. si procede per divisioni successive, dove il divisore è sempre M_{10} , il dividendo, invece, è X_{10} la prima volta, successivamente diventa il quoziente ottenuto nella divisione precedente
3. si termina il procedimento quando il valore del quoziente è 0
4. si prendono i resti delle divisioni effettuate in ordine inverso, dall'ultimo al primo
5. si convertono, per basi $M > 10$, i resti ottenuti, espressi in base 10, nella corrispondente cifra in base M .

Rileggiamo quanto esposto con $M=2$.

Conversione da base 10 a base 2

1. (superfluo) si esprime 2 in base 10
2. si procede per divisioni successive del numero dato per 2
3. si termina il procedimento quando il valore del quoziente è 0
4. i resti delle singole divisioni vengono presi in ordine inverso rispetto a quello del calcolo e rappresentano le cifre del numero espresso in base 2 (ricordati che i soli possibili resti saranno 0 o 1).
5. (superfluo)

Esempio: convertire 106_{10} in base 2.

$$106:2 = 53 \text{ resto } 0$$

$$53:2 = 26 \text{ resto } 1$$

$$26:2 = 13 \text{ resto } 0$$

$$13:2 = 6 \text{ resto } 1$$

$$6:2 = 3 \text{ resto } 0$$

$$3:2 = 1 \text{ resto } 1$$

$$1:2 = 0 \text{ resto } 1$$

106	53	26	13	6	3	1	0	← quozienti
0	1	0	1	0	1	1		← resti

quindi

$$106_{10} = 1101010_2$$

Proponiamo altri esempi con base M diversa da 2:

- convertire 106_{10} in base 5

$$106:5 = 21 \text{ resto } 1$$

$$21:5 = 4 \text{ resto } 1$$

$$4:5 = 0 \text{ resto } 4$$

106	21	4	0	← quozienti
1	1	4		← resti

otteniamo: $(106)_{10} = (411)_5$

- convertire 180_{10} in base 16

$$180:16 = 11 \text{ resto } 4$$

$$11:16 = 0 \text{ resto } 11$$

180	11	0	← quozienti
4	11		← resti

Ricorda che i resti sono in base 10, occorre quindi convertirli in base 16 (in generale in base M):

$$(11)_{10} = (B)_{16}$$

otteniamo: $(180)_{10} = (B4)_{16}$

Consideriamo ora le conversioni da base M a base 10. Enunciamo la regola che permette di convertire un numero espresso in una base qualsiasi nella corrispondente rappresentazione decimale (ricorda che i sistemi coinvolti sono tutti sistemi posizionali)

- si moltiplica ogni cifra per il valore della base elevato ad esponente pari alla posizione che ha la cifra nella sequenza di simboli
- si sommano i valori così ottenuti

Rileggiamo quanto esposto con $M=2$.

Conversione da base 2 a base 10

- si moltiplica ogni *bit* per il valore della base elevato ad esponente pari alla posizione che ha il *bit* nella sequenza di simboli
- si sommano i valori così ottenuti.

Esempio: convertire $(11011)_2$ in base 10.

$$\begin{array}{ccccccc} 1 & 1 & 0 & 1 & 1 & = & 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 16 + 8 + 0 + 1 + 1 = 27 \text{ in base 10} \\ \text{posizione} \rightarrow & 4 & 3 & 2 & 1 & 0 & \end{array}$$

Nell'esempio precedente, la cifra meno significativa è il coefficiente della potenza 2^0 , la cifra più significativa (1 in posizione 4) è il coefficiente di 2^4 .

Proponiamo altri esempi con base M diversa da 2:

Conversione dalla base M alla base 10

- convertire 214_6 in base 10
ricordando che la cifra meno significativa indica il coefficiente di 6^0
quella più significativa il coefficiente di 6^2 , segue

$$214_6 = 2 \cdot 6^2 + 1 \cdot 6^1 + 4 \cdot 6^0 = 2 \cdot 36 + 1 \cdot 6 + 4 \cdot 1 = 72 + 6 + 4 = 82_{10}$$

Conversione da base N a base M

- Nel caso generale, con N e M valori qualsiasi, è possibile applicare la regola espressa inizialmente ma, spesso, si preferisce trasformare il numero dalla base N alla base 10 e poi far seguire la conversione da base 10 a base M.

Esempio: $45_7 = \text{numero}_3$

1° passo: si converte 45_7 in base 10 $\rightarrow 45_7 = 4 \cdot 7^1 + 5 \cdot 7^0 = 28 + 5 = 33_{10}$

2° passo: si converte il numero così ottenuto in base 3 $\rightarrow 33_{10} = 1020_3$

- Più semplice è la conversione se la base N è 2 e la base M è 8 oppure 16 (ma vale anche il viceversa): l'identità $8 = 2^3$ permette di convertire ogni gruppo di 3 bit in un simbolo in base 8 e, viceversa, ogni simbolo in base 8 nella sequenza di 3 bit in base 2. Nel caso in cui la base è 16 ($16 = 2^4$), si ripete lo stesso procedimento ma con gruppi di 4 (e non 3) bit.

Gli esempi chiariranno questo procedimento.

Esempio: convertire 1110010101000100_2 in ottale

Partendo dalla cifra meno significativa, si considerano le cifre binarie rispettivamente a gruppi di 3:

00 1	110	010	101	000	100
↓	↓	↓	↓	↓	↓
1	6	2	5	0	4

Quindi: $1110010101000100_2 = 162504_8$

Esempio: convertire 111001110101001100_2 in esadecimale

Partendo dalla cifra meno significativa, si considerano le cifre binarie rispettivamente a gruppi di 4:

00 11	1001	1101	0100	1100
↓	↓	↓	↓	↓
3	9	D	4	C

Quindi: $111001110101001100_2 = 39D4C_{16}$

Esempio: convertire 372_8 in binario

Ogni cifra viene codificata nel valore corrispondente in base 2 espresso con 3 bit:

3	7	2
↓	↓	↓
011	111	010

Quindi: $372_8 = 011111010_2$

Esempio: convertire $C2E3_{16}$ in binario

Ogni cifra viene codificata nel valore corrispondente in base 2 espresso con 4 bit:

C	2	E	3
↓	↓	↓	↓
1100	0010	1110	0011

Quindi: $C2E3_{16} = 1100001011100011_2$

Esercizi: conversioni di base

a) da base 10 a base 2

1) 23	[10111]
2) 38	[100110]
3) 100	[1100100]
4) 32	[100000]
5) 70	[1000110]
6) 65	[1000001]
7) 98	[1100010]
8) 255	[11111111]
9) 318	[100111110]
10) 512	[1000000000]

b) da base 10 alla base indicata

11) 34 in base 4	[202 ₄]
12) 128 in base 6	[332 ₆]
13) 7 in base 7	[10 ₇]
14) 100 in base 16	[64 ₁₆]
15) 1024 in base 5	[13044 ₅]
16) 742 in base 9	[1014 ₉]

17) 38 in base 3	[1102 ₃]
18) 42 in base 7	[60 ₇]
19) 78 in base 3	[2220 ₃]
20) 153 in base 11	[12A ₁₁]

c) da base 2 a base 10

21) 1011	[11]
22) 10110	[22]
23) 110011	[51]
24) 111111	[63]
25) 1011100	[92]
26) 11000000	[192]
27) 111010101	[469]
28) 100000000	[256]
29) 10000011	[131]
30) 110011001	[409]

d) da base N a base 10

31) 152 ₈	[106]
32) 123 ₄	[27]
33) 342 ₅	[97]
34) A1F ₁₆	[2591]
35) 351 ₇	[183]
36) 1234 ₄	[impossibile]
37) 100 ₅	[25]
38) 100 ₆	[36]
39) 101 ₇	[50]
40) A19 ₁₂	[1461]

e) da base N a base M

41) 134 ₆ in base 2	[111010 ₂]
42) 615 ₈ in base 2	[110001101 ₂]
43) 326 ₉ in base 16	[10B ₁₆]
44) 714 ₈ in base 7	[1225 ₇]
45) 1110110 ₂ in base 5	[433 ₅]
46) 171 ₈ in base 16	[79 ₁₆]
47) 1110110 ₂ in base 16	[76 ₁₆]
48) 11110110 ₂ in base 8	[366 ₈]
49) 1AB6 ₁₆ in base 2	[1101010110110 ₂]
50) ABC ₁₆ in base 8	[5274 ₈]

Codifica dei caratteri

I numeri rappresentano solo una piccola parte delle informazioni memorizzate ed elaborate da un computer; insieme alla codifica delle informazioni numeriche è sorto il problema della rappresentazione dei dati alfanumerici, ovvero dei documenti testuali e anche in questo caso sono state elaborate leggi di trasformazione da carattere a stringa di bit.

Prima però di entrare nel dettaglio è fondamentale introdurre la definizione di byte. Si definisce BYTE una sequenza di 8 BIT (evidente anche in questo caso la naturale predilezione che si ha in informatica per le potenze del 2 come conseguenza dell'uso della codifica binaria). Essendo codificabili su 8 bit 256 combinazioni distinte di 0/1, si ritenne che tale quantità di bit fosse sufficiente per rappresentare tutti i simboli utilizzabili nella scrittura di messaggi e documenti e da allora il BYTE è divenuta una sorta di unità di misura della codifica (1 BYTE = 1 carattere), tuttora utilizzata per definire la capacità di memoria: sempre privilegiando le potenze del 2 e utilizzando il simbolo “B” per riferirsi al byte (ricorda che “b” è usato per indicare grandezze espresse in bit), si definisce:

sigla	descrizione	Quantità in byte
1kB	1 kilo-byte	1024 bytes
1MB	1 mega-byte	1024^2 bytes = 1.048.576 bytes
1GB	1 giga-byte	1024^3 bytes = 1.073.741.824 bytes
1TB	1 tera-byte	1024^4 bytes = 1.099.511.627.776 bytes

Una unità a disco da 1TB (un tera-byte) è dunque in grado di memorizzare più di mille miliardi di caratteri.

Si è detto che su 1 byte cioè su 8 bit è possibile codificare un carattere: come stabilire allora la corrispondenza tra byte e carattere?

La corrispondenza non può che essere basata su una “convenzione”, cioè su un accordo stabilito da un'apposita commissione composta da tecnici, studiosi, rappresentanti di aziende, istituzioni e governi; delle diverse convenzioni esistenti ne consideriamo due: codice ASCII e codice UNICODE.

Il codice ASCII (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange ovvero *codifica standard americana per lo scambio di informazioni*) è nato come codice a 7 bit: sono sufficienti infatti 7 bit (quindi 128 combinazioni di 0/1) per rappresentare tutti i caratteri

maiuscoli e minuscoli dell'alfabeto inglese, le cifre da 0 a 9, i simboli di punteggiatura, le parentesi, gli operatori matematici (“+”, “-”, “*”, “\”, “%”), gli operatori relazionali (“=”, “>”, “<”) oltre ad alcuni caratteri “non stampabili”, cioè non corrispondenti a simboli, ma testuali utilizzati originariamente come comandi nelle trasmissioni telegrafiche. All'inizio degli anni 1960 con l'affermarsi del byte anche come unità di misura delle memorie, fu aggiunto un bit, passando così dalla codifica ASCII alla codifica ASCII estesa; l'aggiunta di un bit ha permesso di raddoppiare il numero di caratteri codificabili (da 128 a 256), permettendo la codifica dei caratteri nazionali (per esempio le vocali accentate: ora è possibile codificare *età* e non *eta*’).

L'estendersi delle comunicazioni tra paesi che utilizzano alfabeti diversi dall'alfabeto inglese ha comportato la necessità di ampliare ulteriormente l'insieme dei simboli da codificare e, conseguentemente, è stato giudicato inadeguato un solo byte per carattere; da qui il diffondersi del codice UNICODE, che associa ad ogni carattere 16 bit, cioè 2 byte e che quindi permette la codifica di $2^{16} = 65536$ simboli diversi.

Segue la tabella della codifica ASCII standard su 8 bit (i primi 128 caratteri)

Dec	Hx	Oct	Char	Dec	Hx	Oct	Chr	Dec	Hx	Oct	Chr	Dec	Hx	Oct	Chr
0	0	000	NUL (null)	32	20	040	Space	64	40	100	@	96	60	140	`
1	1	001	SOH (start of heading)	33	21	041	!	65	41	101	A	97	61	141	a
2	2	002	STX (start of text)	34	22	042	"	66	42	102	B	98	62	142	b
3	3	003	ETX (end of text)	35	23	043	#	67	43	103	C	99	63	143	c
4	4	004	EOT (end of transmission)	36	24	044	\$	68	44	104	D	100	64	144	d
5	5	005	ENQ (enquiry)	37	25	045	%	69	45	105	E	101	65	145	e
6	6	006	ACK (acknowledge)	38	26	046	&	70	46	106	F	102	66	146	f
7	7	007	BEL (bell)	39	27	047	'	71	47	107	G	103	67	147	g
8	8	010	BS (backspace)	40	28	050	(	72	48	110	H	104	68	150	h
9	9	011	TAB (horizontal tab)	41	29	051	)	73	49	111	I	105	69	151	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	74	4A	112	J	106	6A	152	j
11	B	013	VT (vertical tab)	43	2B	053	+	75	4B	113	K	107	6B	153	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	76	4C	114	L	108	6C	154	l
13	D	015	CR (carriage return)	45	2D	055	-	77	4D	115	M	109	6D	155	m
14	E	016	SO (shift out)	46	2E	056	.	78	4E	116	N	110	6E	156	n
15	F	017	SI (shift in)	47	2F	057	/	79	4F	117	O	111	6F	157	o
16	10	020	DLE (data link escape)	48	30	060	0	80	50	120	P	112	70	160	p
17	11	021	DC1 (device control 1)	49	31	061	1	81	51	121	Q	113	71	161	q
18	12	022	DC2 (device control 2)	50	32	062	2	82	52	122	R	114	72	162	r
19	13	023	DC3 (device control 3)	51	33	063	3	83	53	123	S	115	73	163	s
20	14	024	DC4 (device control 4)	52	34	064	4	84	54	124	T	116	74	164	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	85	55	125	U	117	75	165	u
22	16	026	SYN (synchronous idle)	54	36	066	6	86	56	126	V	118	76	166	v
23	17	027	ETB (end of trans. block)	55	37	067	7	87	57	127	W	119	77	167	w
24	18	030	CAN (cancel)	56	38	070	8	88	58	130	X	120	78	170	x
25	19	031	EM (end of medium)	57	39	071	9	89	59	131	Y	121	79	171	y
26	1A	032	SUB (substitute)	58	3A	072	:	90	5A	132	Z	122	7A	172	z
27	1B	033	ESC (escape)	59	3B	073	;	91	5B	133	[	123	7B	173	{
28	1C	034	FS (file separator)	60	3C	074	<	92	5C	134	\	124	7C	174	
29	1D	035	GS (group separator)	61	3D	075	=	93	5D	135	]	125	7D	175	}
30	1E	036	RS (record separator)	62	3E	076	>	94	5E	136	^	126	7E	176	~
31	1F	037	US (unit separator)	63	3F	077	?	95	5F	137	_	127	7F	177	DEL

Riportiamo per completezza anche i successivi 128 caratteri della codifica ASCII estesa

(solo la codifica decimale)

128	Ç	144	É	160	á	176	░	193	⌚	209	ƒ	225	ß	241	±
129	ù	145	æ	161	í	177	▒	194	⌘	210	π	226	Γ	242	≥
130	é	146	Æ	162	ó	178	▓	195	⌡	211	ℓ	227	π	243	≤
131	â	147	ô	163	ú	179		196	—	212	ℓ	228	Σ	244	ƒ
132	ä	148	ö	164	ñ	180	⌣	197	†	213	ƒ	229	σ	245	∫
133	à	149	ò	165	ñ	181	⌤	198	‡	214	ƒ	230	μ	246	÷
134	â	150	û	166	ª	182	⌥	199	⌢	215	‡	231	τ	247	≈
135	ç	151	ü	167	º	183	⌦	200	ℓ	216	‡	232	Φ	248	°
136	ê	152	—	168	¿	184	⌧	201	ƒ	217	∫	233	⊙	249	÷
137	ë	153	Ö	169	—	185	⌨	202	ℓ	218	∫	234	Ω	250	÷
138	è	154	Û	170	¬	186	〈	203	ƒ	219	■	235	δ	251	√
139	í	156	£	171	½	187	〉	204	⌣	220	■	236	∞	252	—
140	î	157	₣	172	¼	188	⌫	205	=	221	■	237	φ	253	²
141	ï	158	—	173	¡	189	⌬	206	⌣	222	■	238	ε	254	■
142	Ä	159	ƒ	174	«	190	⌭	207	±	223	■	239	∩	255	
143	Å	192	Ł	175	»	191	⌮	208	ℓ	224	α	240	≡		

A titolo di esempio, la sequenza ASCII di codifica del testo “Ciao, mondo!”:

C	i	a	o	,		m	o	n	d	o	!
in esadecimale											
43	69	61	6F	2C	20	6D	6F	6E	64	6F	21
in binario											
01000011	01101001	01100001	01101111	00101100	00010000	01101101	01101111	01101110	01100100	01101111	00100001

E' piuttosto evidente che, se dovessimo codificare in binario un documento senza l'aiuto di un qualche automatismo, ci troveremmo in grave difficoltà così come nella stessa difficoltà si troverebbe chi dovesse decodificarlo nella necessità di ricostruire il documento originale

“Ciao, mondo!” > codificatore > sequenza di bit

sequenza di bit > decodificatore > “Ciao, mondo!”

Nel precedente schema il “codificatore” e il “decodificatore” sono degli strumenti che, conoscendo perfettamente il criterio (cioè le regole) di codifica, sono in grado di effettuare tanto la trasformazione da documento testuale a documento binario quanto la trasformazione inversa.

Codifica delle immagini (fisse)

Così come un testo può essere memorizzato su un supporto elettronico, dopo essere stato codificato in una sequenza di bit grazie a codici come ASCII o UNICODE, anche le immagini, i suoni e i video possono essere memorizzati purché opportunamente “trasformati” in sequenze di 0 e 1. A differenza, però, di un documento testuale, un'immagine non nasce come un oggetto discreto, ovvero costituito da un insieme finito e

ben determinato di simboli (caratteri dell'alfabeto inglese maiuscoli e minuscoli, cifre, simboli aritmetici, segni di punteggiatura, etc.), ma è un insieme di colori, linee, forme, sfumature che combinate tra loro rappresentano appunto un'immagine (una fotografia è un'immagine, un quadro è un'immagine, la stampa di un carattere è un'immagine). Occorre quindi trasformare l'immagine in una opportuna sequenza di simboli che, a loro volta, verranno codificati in sequenze di 0 e 1. Più precisamente occorre DIGITALIZZARE l'immagine, cioè passare dalla sua rappresentazione analogica alla corrispondente rappresentazione digitale.

E' opportuno a questo punto aprire una parentesi e specificare cosa intendiamo con i termini analogico e digitale (a prescindere dalle immagini), per fugare qualsiasi ambiguità di interpretazione rispetto a quanto verrà esposto successivamente.

Una grandezza si definisce **analogica** quando varia in modo continuo nel tempo e/o nello spazio e nei valori assunti, ovvero quando può assumere qualsiasi valore in un determinato intervallo.

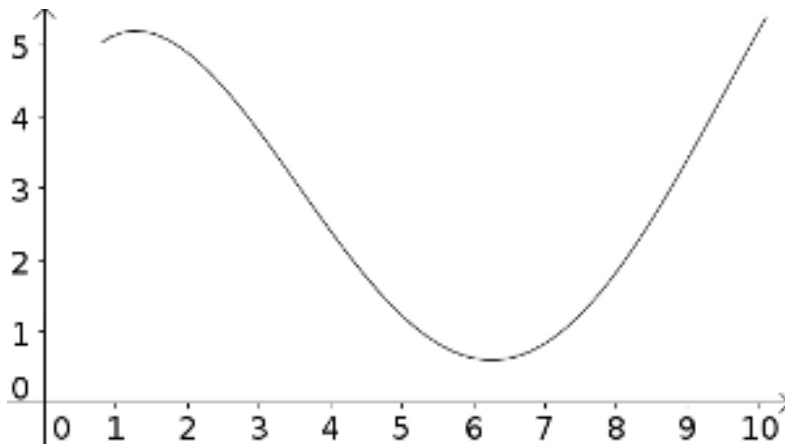
Una grandezza si definisce **digitale** quando NON varia in modo continuo: essa può assumere valori che sono elementi di un insieme discreto e quindi può assumere in un determinato intervallo solo alcuni valori e non altri (si dice che una grandezza digitale assume valori in modo discontinuo o a salti).

Consideriamo due grandezze fisiche con cui interagiamo quotidianamente, tempo e temperatura: siamo abituati a rappresentare il loro valore sia in “analogico” che in “digitale”.

Grandezza (e relativa unità di misura)	Misura analogica	Misura digitale
Tempo (ora, minuti)		
Temperatura (gradi)		

Torniamo al termine digitalizzare con una definizione: “Nel campo dell'informatica e dell'elettronica, con digitalizzazione si intende il processo di trasformazione di un'immagine, di un suono, di un documento in un formato digitale, interpretabile da un computer, dove per formato digitale si intende un codice binario in cui tutto è rappresentato da combinazioni di zero od uno, quindi da stati del tipo acceso/spento.” (Wikipedia)

La sequenza di passi di cui si compone il processo sono il campionamento, la quantizzazione e la codifica.

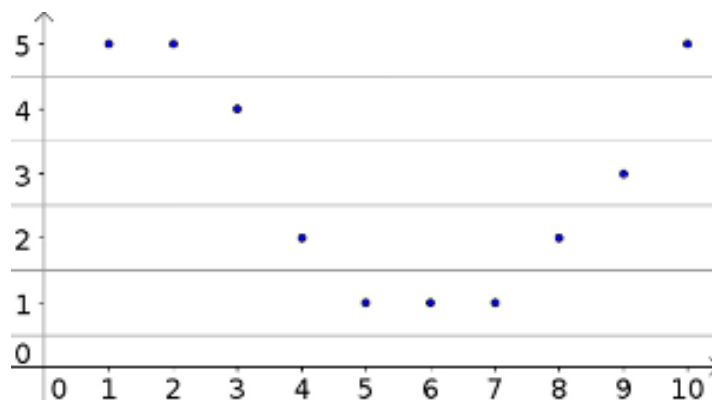
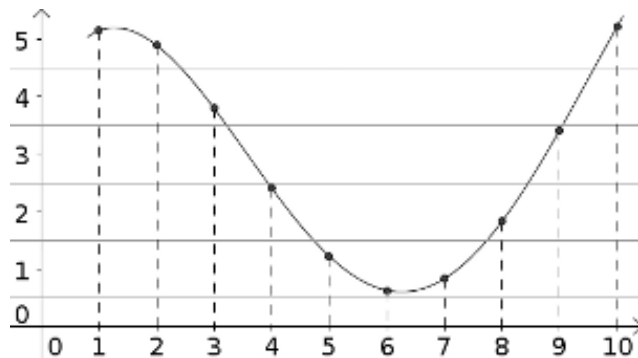


Iniziamo a considerare un segnale analogico qualsiasi: esso può essere rappresentato graficamente tramite una "linea" come nella figura a fianco. Possiamo, ad esempio, pensare che sull'asse delle ascisse rappresentiamo il tempo e sull'asse delle ordinate il valore

che il segnale assume in quell'istante. I valori cambiano con continuità (la funzione è continua) nell'intervallo di tempo considerato.

Eseguiamo i passi necessari per "digitalizzare" questa grandezza.

mpionamento: i valori della grandezza vengono rilevati ad intervalli di tempo distinti (la grandezza è ancora analogica).



Quantizzazione: si definisce l'intervallo di valori all'interno del quale ogni singolo valore campionato viene approssimato (in questa fase si introduce un errore che è inversamente proporzionale al numero di valori che vengono fissati).

difica: ad ogni valore che la grandezza assume nell'intervallo di campionamento viene associato un numero (una sequenza di 0 e 1) in base ad un codice.

Campione	Valore	CODIFICA			
1	5	0	1	0	1
2	5	0	1	0	1
3	4	0	1	0	0
4	2	0	0	1	0
5	1	0	0	0	1
6	1	0	0	0	1
7	1	0	0	0	1
8	2	0	0	1	0
9	3	0	0	1	1
10	5	0	1	0	1

Dopo questa digressione, torniamo all'argomento del paragrafo, la digitalizzazione delle immagini: descriveremo due strategie, la grafica bitmap o raster e la grafica vettoriale.

Grafica bitmap o raster.

Consiste nel sovrapporre all'immagine una griglia di punti o, meglio, di suddividere l'immagine in una serie di celle di dimensioni ridotte (quanto ridotte lo vedremo tra poco): ogni punto o cella prende il nome di pixel (PICture ELeMent); il pixel è quindi il più piccolo elemento costituente un'immagine digitalizzata. Ma in quanti pixel possiamo suddividere un'immagine? Non esiste un valore fisso, esiste però il concetto di “fedeltà” all'immagine iniziale o, meglio, di “qualità” dell'immagine riprodotta. Vediamo di spiegare questo concetto considerando, come esempio, le seguenti immagini: *(N.B. tutte e tre le immagini hanno la stessa dimensione)*

		
Larghezza 25 pixel (*) Altezza 27 pixel	Larghezza 64 pixel (*) Altezza 67 pixel	Larghezza 256 pixel Altezza 270 pixel

(*) valori approssimati

Possiamo quindi affermare che all'aumentare del numero di pixel, mantenendo costanti le dimensioni della figura, riusciamo ad ottenere un'immagine “migliore” cioè sempre più fedele all'originale.

Il numero di pixel in cui si suddivide lo schermo prende il nome di **definizione**, a cui è strettamente connesso, ma non è un sinonimo, il concetto di **risoluzione**, che coincide con la quantità di pixel per unità di misura. La risoluzione è quindi la densità di pixel e da essa dipende la qualità dell'immagine; nell'esempio in figura abbiamo assunto come unità

di misura la dimensione dell'immagine e modificato la quantità di pixel in essa contenuti: aumentando la densità è aumentata la qualità. L'unità di misura più frequentemente usata per quantificare la risoluzione è il **dpi** (Dot Per Inches), numero di pixel per pollice (un pollice= 2,54 cm).

Esiste un terzo parametro che caratterizza un'immagine digitalizzata: la **profondità** dell'immagine (o del colore), anch'essa rappresentata con un certo numero di bit: maggiore è il numero di bit utilizzati maggiore è la qualità del colore riprodotto.

Infatti un'immagine è fatta anche e soprattutto di colori, quindi ad un pixel occorre associare un numero che codifica il colore che ha, nell'immagine da riprodurre, il punto rappresentato. Per semplicità supponiamo di voler rappresentare un carattere, per esempio il carattere 'R', in bianco e nero: basterà un solo bit per rappresentare il colore, per esempio 1 per il nero, 0 per il bianco.

Immagine del carattere 'R'

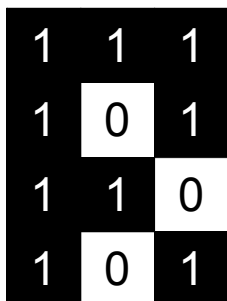


Immagine con sovrapposta una griglia di 4x3 pixel



(le due figure successive sono state ingrandite per permettere una maggiore leggibilità)

Associamo ad ogni pixel un bit in base al colore predominante nel quadratino



Togliendo il colore, rimane la codifica che letta come sequenza di bit è 111101110101

1	1	1
1	0	1
1	1	0
1	0	1

Facendo il procedimento inverso, sostituiamo ai bit il colore (sempre con la stessa regola: 0 → bianco, 1→ nero) e otteniamo



che non coincide esattamente con l'immagine iniziale.

Proviamo a ridurre la grandezza di un pixel ovvero ad aumentare il numero di quadratini della griglia da sovrapporre all'immagine.

Immagine del carattere 'R'



Immagine con sovrapposta una griglia di 8x6 pixel



Ripetiamo gli stessi passi ...

1	1	1	1	1	0
1	1	1	1	1	0
1	1	0	0	1	1
1	1	1	1	1	0
1	1	1	1	0	0
1	1	0	1	1	0
1	1	0	1	1	0
1	1	0	0	1	1

Otteniamo la sequenza 111110
111110 110011 111110 111100
110110 110110 110011

1	1	1	1	1	0
1	1	1	1	1	0
1	1	0	0	1	1
1	1	1	1	1	0
1	1	1	1	0	0
1	1	0	1	1	0
1	1	0	1	1	0
1	1	0	0	1	1

che nuovamente trasformata diventa



più fedele all'originale anche se con una "particolarità". Domanda: sai spiegarne il perché?

Un ultimo raffinamento.

Immagine del carattere 'R'



Immagine con sovrapposta una griglia di 12x16 pixel



Ripetiamo gli stessi passi ...

1	1	1	1	1	1	1	1	0	0	0
1	1	1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	0	0	0	1	1	1	0
1	1	1	1	0	0	0	1	1	1	0
1	1	1	1	0	0	0	1	1	1	0
1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	1	0	0	0
1	1	1	1	0	1	1	1	0	0	0
1	1	1	1	0	1	1	1	0	0	0
1	1	1	1	0	1	1	1	0	0	0
1	1	1	1	0	1	1	1	0	0	0
1	1	1	1	0	0	1	1	1	0	0
1	1	1	1	0	0	1	1	1	0	0
1	1	1	1	0	0	1	1	1	0	0
1	1	1	1	0	0	0	1	1	1	0
1	1	1	1	0	0	0	1	1	1	0
1	1	1	1	0	0	0	1	1	1	0
1	1	1	1	0	0	0	1	1	1	1

Quale sequenza di bit E infine ..
otteniamo?

1	1	1	1	1	1	1	1	1	0	0	0
1	1	1	1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	0	0	0	1	1	1	1	0
1	1	1	1	0	0	0	1	1	1	1	0
1	1	1	1	0	0	0	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	1	1	0	0	0
1	1	1	1	0	1	1	1	1	0	0	0
1	1	1	1	0	1	1	1	1	0	0	0
1	1	1	1	0	1	1	1	1	0	0	0
1	1	1	1	0	1	1	1	1	0	0	0
1	1	1	1	0	1	1	1	1	0	0	0
1	1	1	1	0	0	1	1	1	1	0	0
1	1	1	1	0	0	1	1	1	1	0	0
1	1	1	1	0	0	1	1	1	1	0	0
1	1	1	1	0	0	0	1	1	1	1	0
1	1	1	1	0	0	0	1	1	1	1	0
1	1	1	1	0	0	0	1	1	1	1	1



Non coincide certamente con l'immagine iniziale ma è "più simile" ad essa rispetto alle

precedenti.

La codifica dei soli colori nero e bianco è accettabile se l'immagine è un carattere, non lo è se l'immagine è una fotografia, un disegno, un dipinto. Infatti anche una fotografia in bianco e nero di fatto non contiene solo questi 2 colori ma diverse tonalità di grigio (per le sfumature, i contrasti, le ombre e quant'altro). Immaginiamo allora di utilizzare 4 bit per codificare il colore di un singolo pixel: significa avere a disposizione 16 ($=2^4$) diverse combinazioni di 0 e 1, ognuna delle quali può essere associata ad un pixel e, quindi, ogni pixel può assumere un colore che varia tra i 16 possibili. Più precisamente non si parla di 16 colori ma di 16 tonalità di grigio che vanno da 0000 (bianco) a 1111 (nero) passando per varie sfumature di grigio (alla codifica 0001 corrisponderà un grigio chiarissimo, alla codifica 1110 un grigio molto scuro).

A questo punto è chiaro che, aumentando il numero di bit impiegati per la codifica del colore, aumenta la quantità di toni di grigio che possono essere rappresentati: con 8 bit è possibile rappresentare 256 ($=2^8$) toni di grigio.

Ma, se vogliamo codificare un'immagine con i suoi colori, occorre approfondire alcuni concetti riguardanti la teoria del colore.

La rappresentazione RGB del colore

La "teoria del colore" afferma che una parte piuttosto ampia dei colori percepiti dai nostri occhi può essere immaginata come sovrapposizione di TRE componenti monocromatiche dette PRIMARIE; il lettore avrà certamente avuto modo di sperimentare come, nella pittura, alcuni particolari tipi di ROSSO, BLU e GIALLO si prestino ad essere combinati tra loro per riprodurre gli altri colori: il BIANCO viene usato come una sorta di diluente cioè di dispersore del pigmento colorato con l'obiettivo di ottenere tonalità più chiare, mentre il NERO è utilizzato per scurire i toni; la necessità del "nero" (a dispetto della teoria) scaturisce dal fatto che il miscuglio in parti uguali dei tre colori primari produce il cosiddetto BISTRO, un nero piuttosto imperfetto a causa delle impurità presenti nei colori primari. Anche nella tecnica tipografica si fa uso dello stesso principio adottando il MAGENTA (un rosso violaceo), il CIANO (un celeste) e il GIALLO come colori primari con l'aggiunta del NERO necessario per aggiungere "forza" alle tonalità scure ottenute con la sovrapposizione dei tre colori primari; nella tecnica tipografica si parla di "quadricromia" e si usa l'acronimo "CMYK" intendendo proprio "C" per "cyan" (ciano), "M" per "magenta", "Y" per "yellow" (giallo) e infine "K" che, nonostante in origine indicasse altro ("Key plate", tecnica di allineamento delle lastre CMY), nella prassi corrente si intende per "nero".

Tanto in pittura che in tecnica tipografica, tuttavia, il colore viene riprodotto per SINTESI

SOTTRATTIVA in quanto ciò che i nostri occhi vedono e percepiscono come colore è la parte riflessa della luce, incidente su ciò che osserviamo: la parte non riflessa viene in effetti assorbita, dunque SOTTRATTA, dal pigmento colorato.

In effetti siamo in grado di “vedere” un quadro oppure un'immagine stampata SOLO se siamo in un ambiente luminoso: riuscireste a vedere qualcosa in una stanza perfettamente al buio?

Ben diversa è la situazione quando guardiamo lo schermo del computer oppure un televisore: a differenza di quadri e stampe lo schermo EMETTE la luce. Si parla allora di riproduzione del colore per SINTESI ADDITIVA in quanto ciò che vediamo in effetti è la sovrapposizione di tre sorgenti luminose primarie che in questo caso sono state individuate nella luce ROSSA, nella luce VERDE e nella luce BLU unite nella famosa sigla RGB (“Red”, “Green”, “Blue”).

Nella codifica RGB per ogni pixel occorre distinguere tre distinti valori riferibili, nell'ordine, alle componenti di luci rossa, verde e blu; utilizzando un byte per ogni colore primario ogni pixel è codificato con 3 byte ($3 \times 8 = 24$ bit in tutto). Ogni gruppo di 8 bit è relativo ad un colore primario e può essere pensato come la quantità di quel colore nell'insieme; i primi 8 bit rappresentano quindi la quantità di rosso che c'è in quel pixel, i successivi 8 la quantità di verde e gli ultimi 8 la quantità di blu. Oltre che con la codifica binaria, ogni pixel può essere rappresentato con la corrispondente codifica decimale o esadecimale (in entrambe le codifiche vengono utilizzati 3 numeri, uno per ciascuna delle 3 sequenze di 8 bit), migliorando così la leggibilità della rappresentazione.

Facciamo qualche esempio

COLORE	BINARIO																								DECIMALE			ESADECIMALE		
	R								G								B								R	G	B	R	G	B
rosso 	1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0																								255	0	0	FF	00	00
verde 	0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0																								0	255	0	00	FF	00
blu 	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1																								0	0	255	00	00	FF

Osserviamo che il giallo, colore primario percepito dall'occhio umano, non è presente nella codifica RGB, ma è rappresentabile come combinazione di rosso e verde; facciamo qualche esempio, iniziando con la codifica di una tonalità più luminosa di giallo, passando per una tonalità più scura e finire con una tonalità più chiara:

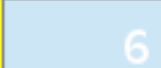
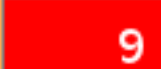
COLORE (giallo)	BINARIO																DECIMALE			ESADECIMALE		
	R					G					B						R	G	B	R	G	B
luminoso 	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
																	255	255	0	FF	FF	00
scurio 	1	1	1	0	0	1	1	0	1	1	1	0	0	1	1	0	0	1	0	0	1	1
																	230	230	102	E6	E6	4C
chiaro 	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	1	1	0
																	255	255	102	FF	FF	66

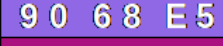
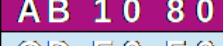
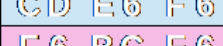
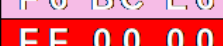
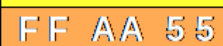
Utilizzando 3 byte, quindi 24 bit, possiamo codificare ben $2^{24} = 16.777.216$ colori distinti: un'enormità, tanto da ipotizzare, nel criterio di codifica, l'uso di un numero minore di bit senza un sensibile degrado della percezione di un'immagine ma con il vantaggio di un minore “peso” in termini di quantità di byte.

Un altro modo per contenere il numero di byte necessari per memorizzare un'immagine a colori è quello di utilizzare un sottoinsieme dei 16.777.216 colori distinti. I colori selezionati vengono memorizzati in una “tavolozza” di colori o “palette”, ogni colore è codificato con lo standard RGB e occupa una posizione nella tavolozza; ad ogni pixel è associato un numero che non rappresenta più il colore del pixel nell'immagine ma la posizione che ha il colore del pixel in esame nella tavolozza del colore.

Prendiamo in considerazione un'immagine composta da 12 pixel, numerati da 1 a 12, in cui sono presenti 8 diversi colori.

Associamo all'immagine la tavolozza che conterrà gli 8 colori: ad ogni pixel è associato una stringa di bit che rappresenta

		
1	2	3
		
4	5	6
		
7	8	9
		
10	11	12

000	
001	
010	
011	
100	
101	
110	
111	

la posizione che il colore del pixel in esame ha nella palette.

Esempio: il pixel 1 è di colore rosso, il rosso occupa nella palette la posizione 101, di conseguenza al pixel 1 verrà associata la codifica 101.

Codifica con PALETTE

pixel	pixel	pixel	pixel	pixel	pixel	pixel	pixel	pixel	pixel	pixel	pixel	pixel
1	2	3	4	5	6	7	8	9	10	11	12	
101	000	100	001	110	011	010	111	101	110	011	001	

Calcoliamo quanta memoria occorre

per l'immagine : 12 (pixel) x 3 bit = 36 bit

per la palette: $8 \times 24 \text{ bit} = 192 \text{ bit}$

dove:

8 = numero di elementi

24 = numero di bit per la codifica in RGB

totale:

228 bit (= 36+192)

In RGB abbiamo la seguente codifica

pixel 1	pixel 2	pixel 3	pixel 4	pixel 5	pixel 6	pixel 7	pixel 8	pixel 9	pixel 10	pixel 11	pixel 12
FF0000	13F345	F6BCE6	9068E5	FFFF33	CDE6F6	AB1080	FFAA55	FF0000	FFFF33	CDE6F6	9068E5

Calcoliamo quanta memoria occorre per l'immagine (in questo caso solo per l'immagine)

$12 (\text{pixel}) \times 24 \text{ bit}(\text{codifica RGB}) = \mathbf{288 \text{ bit}}$

Generalmente, con la tavolozza c'è un risparmio nell'occupazione di memoria.

Domanda: ma se il colore che vogliamo rappresentare non è presente nella tavolozza?

Possiamo scegliere tra due possibili soluzioni: selezioniamo la tonalità, presente nella tavolozza, più vicina a quella da rappresentare oppure cambiamo palette

Grafica vettoriale (cenni)

La codifica delle immagini mediante la grafica bitmap è adatta alla memorizzazioni di fotografie ma non si adatta a memorizzare disegni tecnici, diagrammi, mappe; in questi casi si preferisce utilizzare la grafica vettoriale che si basa sulla considerazione che ogni immagine può essere vista come un insieme di oggetti, i più utilizzati sono i luoghi geometrici come i punti e le rette. Quindi, dopo aver trasformato l'immagine che si intende digitalizzare in un ben determinato insieme di oggetti, si memorizzano le coordinate di tali elementi; occorrerà poi un programma in grado di leggere tali dati e trasformarli in pixel perché l'immagine iniziale venga riprodotta.

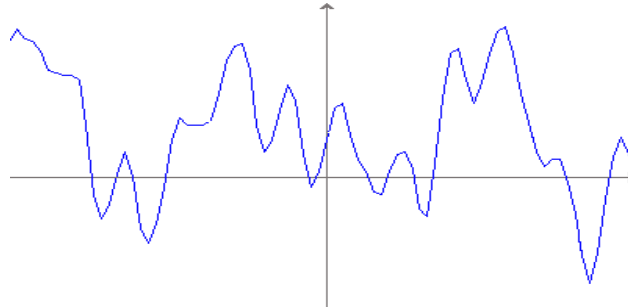
L'utilizzo della grafica vettoriale, oltre a comportare una minore occupazione di memoria rispetto alla memorizzazione della stessa immagine con la grafica bitmap, permette una maggiore manipolazione poiché è possibile modificare la dimensione dell'immagine (ingrandendo o rimpicciolendo) senza perdita di nitidezza; non è un caso che le mappe che è possibile consultare sul Web o sui navigatori GPS sono immagini vettoriali.

D'altra parte la codifica delle immagini con la grafica vettoriale dipende dal software utilizzato, di conseguenza non esiste uno standard universalmente riconosciuto.

Un'ultima nota: anche nel caso di grafica vettoriale c'è il problema di riprodurre il colore, ma la trattazione dell'argomento esula dagli obiettivi di questi appunti.

Codifica dell'audio

Da un punto di vista fisico un suono può essere definito come una sensazione percepita dall'orecchio a causa del movimento di un corpo che mette in oscillazione l'aria. Tale movimento, detto più correttamente vibrazione, ha un andamento ondulatorio nel tempo e perciò un suono può essere rappresentato da un'onda, onda sonora.



Un suono è caratterizzato dalla *frequenza*, misurata in Hertz (Hz), che definisce l'altezza del suono, dall'*ampiezza* dell'onda, misurata in decibel (dB), che determina l'intensità del suono ed infine dalla *durata*.

La memorizzazione dei suoni su un disco di vinile può essere considerata una rappresentazione analogica del suono perché fornisce una rappresentazione continua dell'onda sonora, ma, ovviamente, non può essere memorizzata su un file ed elaborata da un computer. E' necessario quindi convertire il segnale sonoro, analogico, in un segnale digitale; una tecnica utilizzata per la conversione analogico-digitale di un suono è quella di trasformare l'onda (il continuo) in un serie di valori rilevati in istanti di tempo distinti, di mantenere tali valori all'interno di un intervallo e codificare ciascun valore con un numero.

Ti ricorda qualcosa? Ebbene sì, è esattamente quanto già analizzato in dettaglio per la codifica delle immagini con la grafica bitmap; il processo di cui stiamo parlando è quello che, partendo dal campionamento del segnale e passando per la quantizzazione, arriva alla codifica dello stesso.

Non ripetiamo quanto già detto: con le dovute modifiche il procedimento è stato già descritto. Vogliamo solo ricordare alcuni dettagli:

- la griglia che si sovrappone al segnale digitale non ha carattere spaziale ma temporale;
- anche in questo caso il processo di quantizzazione introduce un errore come sempre avviene quando si passa da una grandezza continua ad un insieme discreto di numeri;

- maggiore è il numero di campioni, migliore è la qualità del suono riprodotto ma, ovviamente, maggiore è la quantità di memoria necessaria.

In ultimo, facciamo un po' di conti: nel processo di trasformazione il segnale originale viene trasformato in una sequenza di misure e, dunque, di numeri rappresentabili ovviamente in codifica binaria: 16 bit per ogni singolo campione (quindi 65.536 livelli distinti di ampiezza) sono solitamente sufficienti per una buona qualità di codifica del segnale acustico insieme ad una frequenza di campionamento di 44.100 Hz (cioè 44.100 campioni ogni secondo) utilizzata per una qualità CD (compact-disk).

Passiamo ai calcoli: quanti byte sono indispensabili per codificare quattro minuti di segnale audio (per esempio un brano musicale)? Presto detto:

Tempo (secondi)	N° totale di campioni (=campioni x tempo)	Occupazione (in byte e in Megabyte) (=n° totale campioni x byte occupati da un campione)
240 (=4 minuti * 60)	10.584.000 (= 44.100* 240)	21.168.000 (=10584000*2) = 20,18 MB (circa)

Quindi, un brano musicale di 4 minuti necessita di oltre 20 megabyte di codifica per una qualità audio di tipo CD (compact-disk).

Codifica del video

Alla base della codifica dei “filmati” c'è la riflessione, antica quanto la storia del cinema, che, osservando in rapida sequenza immagini fotografiche (dette “frame”) contenenti piccole variazioni, si ha la percezione della continuità e quindi del movimento.

Una buona velocità in tal senso è 30 frame al secondo.

Se poi aggiungiamo una colonna sonora sincronizzata con le immagini otteniamo un video “completo” (suono+ immagine).

Allo stesso modo, sequenze di immagini digitalizzate producono la stessa percezione: possiamo dunque pensare un video digitale come un oggetto codificato in un file contenente, in successione, i vari fotogrammi che compongono la sequenza video, a cui aggiungere i dati relativi alla digitalizzazione dei suoni (siano essi musica, voce umana, etc.) per completare il prodotto.

La digitalizzazione di un video è quindi una combinazione di tecniche di digitalizzazione di suoni e immagini; tra le inevitabili complicazioni insite in questo processo, vogliamo soffermare la nostra attenzione sul seguente quesito: quanto spazio occupa in termini di byte un filmato?

Facciamo alcune ipotesi e... qualche conto!

Immaginiamo che un filmato (senza sonoro per semplicità di calcolo) di 10 minuti sia composto da sequenze di immagini in alta definizione 1280x720 (per un totale di 921.600 pixel). Ipotizziamo una codifica del colore RGB su 24 bit (3 byte): ogni frame è codificato su 3x921.600 byte cioè 2.764.800 byte.

Se consideriamo 30 frame al secondo, equivalendo 10 minuti a 600 secondi, otteniamo un numero di frame complessivo pari a 18.000 e quindi 49.766.400.000 byte che valgono più di 46 GB (gigabyte): una quantità di memoria enorme per soli 10 minuti di filmato.

In generale, per calcolare l'occupazione di memoria di un file che contiene la digitalizzazione di un video occorre determinare quanto spazio occupa ogni unità elementare in cui è stato scomposto il video (immagini, suoni, caratteri), contare quante unità elementari di ciascun tipo sono presenti, calcolare l'occupazione totale di ciascun tipo e infine sommare i valori ottenuti.

Classificazione dell'informazione

Giunti a questo punto, torniamo sulla prima definizione del capitolo che qui riportiamo arricchendola con l'aggiunta di un solo aggettivo:

“L'informatica si occupa della codifica BINARIA, della memorizzazione, della trasmissione e dell'elaborazione dell'informazione”.

Il riferimento all'uso generalizzato della rappresentazione binaria dell'informazione dovrebbe ora risultare chiaro: ogni genere di informazione trattata da sistemi automatici DEVE essere rappresentabile come sequenza di BIT.

Viceversa per poter riprodurre l'informazione codificata in una sequenza di bit è necessario conoscere i criteri adottati nella codifica. Una sequenza di bit del tipo:

```
101011110111011010001101011110111011001101001100101010010000101001110  
00010101011101000000110101011011111110011011000001101011111110111100110  
010101010100000101010100010101111011011100100101110101000111010101001
```

non è interpretabile fintanto che non viene assegnato un criterio per decifrarla, ossia un criterio di decodifica. Ricordando la distinzione tra dato e informazione possiamo dire che

informazione = dato (ovvero sequenza di bit) + criterio di decodifica

Solo quando è stabilito il criterio di codifica (e dunque di decodifica) ha dunque senso parlare di digitalizzazione dell'informazione secondo il seguente schema:

INFORMAZIONE > CODIFICA > INFORMAZIONE CODIFICATA
 INFORMAZIONE CODIFICATA > DECODIFICA > INFORMAZIONE.

Un'ultima nota: l'informazione codificata viene memorizzata in un “oggetto” chiamato file a cui si assegna un NOME allo scopo di caratterizzare in modo chiaro ciò che rappresenta; nel caso per esempio di un documento di testo contenente il curriculum vitae di un tale “Mario”, codificato in ASCII, si può assegnare un nome del tipo:

curriculum_mario.txt

dove si distinguono due parti, la prima “curriculum_mario” detta propriamente NOME; la seconda “txt” si dice ESTENSIONE e segue il PUNTO che fa da SEPARATORE.

In ambiente WINDOWS l'estensione indica il TIPO di informazione (quindi il criterio utilizzato nella codifica) che con una convenzione piuttosto diffusa nel mondo dell'Informatica indica un contenuto di tipo testuale “semplice” cioè codificato con l'uso del codice ASCII. Il nome “curriculum_mario”, più che avere valore per l'operazione di decodifica, ha valore di “descrittore del contenuto” ed è liberamente scelto dall'autore dell'oggetto come “pro-memoria”: in questo modo, chi si trovasse a leggere nell'elenco dei file:

curriculum_mario.txt
curriculum_cesare.txt
curriculum_rosa.txt

dedurrebbe senza difficoltà il tipo e il contenuto dei file.

Nella seguente tabella si indicano le estensioni più utilizzate insieme al tipo di informazione che generalmente caratterizzano:

Tipologia	Estensioni
Grafica bitmap o raster	bmp, jpg, gif, tiff, png, ico
Grafica vettoriale	eps, ai, wmf
File eseguibili, codice oggetto, librerie dinamiche	exe, com, lib, jar, war, ear, class, dll
Documenti	doc, docx, html, rtf, odt, txt
Audio e musica	wav, wma, au, mp2, mp3, mid
Archiviazione e compressione	zip, 7z, jar, arc, gzip, tar, tar.gz, rar
Linguaggi di programmazione	c, cpp, for, java, lisp, cob, asm, bas, php, asp, jsp, pl, py, rb
Linguaggi di descrizione e stilizzazione pagina	pdf, postscript, xsl-fo, css, xslt/xsl

Approfondimento: aritmetica binaria

Il sistema di numerazione binario è, come sottolineato più volte, un sistema di numerazione posizionale: di conseguenza in esso le quattro operazioni fondamentali seguono le stesse regole applicate nel sistema decimale.

(Per semplicità, supponiamo che non ci siano limiti circa il numero di bit che occorrono per rappresentare il risultato).

Addizione: si sommano le cifre di uguale peso, ricordando che si può avere un riporto (*carry*) sul bit di peso immediatamente superiore.

+	0	1
0	0	1
1	1	0

← con riporto di 1

Sottrazione: si sottraggono le cifre di uguale peso, ricordando che si può avere un prestito (*borrow*) sul bit di peso immediatamente superiore

- (*)	0	1
0	0	1
1	1	0

← con prestito di 1

(*) sulla prima colonna a sinistra le cifre del minuendo, sulla prima riga le cifre del sottraendo

Moltiplicazione: le cifre costituenti il primo numero (moltiplicando) vengono moltiplicate per ciascuna cifra del secondo numero (moltiplicatore): se la cifra del moltiplicatore

*	0	1
0	0	0
1	0	1

è pari a 1 si riscrive il moltiplicando, in caso contrario (cifra=0) il risultato è pari a 0. Si sommano tutti i numeri ottenuti (attenzione ai riporti!)

Divisione: anche in questo caso il procedimento coincide con quello applicato tra numeri in base 10, con la semplificazione che il gruppo di cifre del dividendo esaminato può contenere il divisore 1 o 0 volte.

Esempi:

Addizione: $110100_2 + 111000_2$

$$\begin{array}{r}
 \begin{array}{ccccccc}
 & 1 & & 1 & & & \\
 1 & 1 & 0 & 1 & 0 & 0 & + \\
 1 & 1 & 1 & 0 & 0 & 0 & \\
 \hline
 1 & 1 & 0 & 0 & 1 & 0 & 0
 \end{array}
 = \begin{array}{r}
 110100 \\
 + \\
 111000 \\
 \hline
 1100100_2
 \end{array}
 \end{array}$$

Sottrazione: $10011_2 - 1111_2$

$$\begin{array}{r}
 \begin{array}{ccccccc}
 & & 1 & & & & \\
 0 & 0 & 1 & 0 & 1 & 1 & - \\
 1 & 1 & 1 & 1 & 1 & & \\
 \hline
 0 & 0 & 1 & 0 & 0 & &
 \end{array}
 = 100_2
 \end{array}$$

$10011_2 - 1111_2 = 100_2$

Moltiplicazione: $11101_2 * 1110_2$

$$\begin{array}{r}
 11101^* \\
 1110= \\
 \hline
 1100000 \\
 111001 \\
 111001 \\
 111001 \\
 \hline
 1100010110
 \end{array}$$

11101_2^*
 $1110_2 =$
 11001011
 0_2

Divisione: $110101_2 : 11_2$

$$\begin{array}{r|l}
 110101 & 11 \\
 11 & 10001 \\
 \hline
 000 & \\
 00 & \\
 \hline
 001 & \\
 00 & \\
 \hline
 010 & \\
 00 & \\
 \hline
 101 & \\
 11 & \\
 \hline
 010 &
 \end{array}$$

$110101_2 :$
 $11_2 =$
 10001_2
 (resto =
 10_2)

Interi positivi e negativi

Torniamo sulla sottrazione; non è un caso che tutti gli esempi proposti sono caratterizzati da un valore del minuendo maggiore del valore del sottraendo, poiché finora abbiamo considerato la codifica e, quindi le operazioni, tra numeri naturali (numeri interi senza segno). Il passo successivo è quello esaminare la codifica dei numeri con segno, cioè dei numeri interi (positivi, negativi e 0).

Nota bene: le argomentazioni successive hanno valore solo se si prendono in esame valori numerici codificati con un numero prefissato di cifre.

Tra i diversi metodi per rappresentare i numeri interi in base 2, prendiamo in considerazione

- la rappresentazione in modulo e segno (M&S)
- la rappresentazione in complemento a 2

Rappresentazione in modulo e segno

Dopo aver stabilito il numero N di bit che si vogliono utilizzare per la codifica, si riserva il bit più significativo per il segno, tutti gli altri (N-1) sono utilizzati per codificare il valore assoluto del numero in esame.

La convenzione adottata è di utilizzare il valore 1 per il segno negativo, 0 per quello positivo.

Esempio: con N=4 bit

- numero = -3 → 1011 dove 1 rappresenta il segno -
- numero = +3 → 0011 dove 0 rappresenta il segno +
- i rimanenti 3 bit (011) rappresentano il valore assoluto, ovviamente uguale in entrambe le codifiche

Osservazioni:

- lo 0 ha una doppia rappresentazione, una positiva e una negativa;
- l'intervallo di valori rappresentabili con N bit è $[-2^{N-1}+1, +2^{N-1}-1]$.

Alla facilità di codifica corrispondono però problemi di elaborazione: il fatto che il bit più significativo (il bit del segno) ha un significato diverso dagli altri implica che deve essere trattato in modo diverso nelle operazioni.

Esempio: Calcoliamo $-20 + 2$ (supponiamo di utilizzare codifiche su 8 bit)

$$\begin{array}{rcl} -20 & = & 10010100 \\ + 2 & = & 00000010 \\ \hline & & 10010110 \end{array} \quad \text{quindi} \quad \begin{array}{r} 10010100 + \\ 00000010 = \\ 10010110 \end{array}$$

che corrisponde a -22 (non a -18 che è il risultato corretto)

Il problema deriva dal fatto che per il bit del segno non vale più la regola del peso associato alla posizione.

Si preferisce utilizzare, invece, la notazione in complemento a 2, che ci permette di trasformare la sottrazione in addizione.

Rappresentazione in complemento a 2

Definiamo il significato di complemento ad una base:

in generale si definisce complemento ad una base M di un numero X rappresentato con N cifre, quel numero Y che sommato a X dà come risultato la potenza N-esima della base. Difficile? No, se rileggiamo quanto detto attraverso degli esempi:

esempio 1: se $X=6$, $M=10$ e $N=1$, la potenza 10^1 è 10;

quanto vale Y, ovvero quale è quel numero che sommato a 6 dà come risultato 10?

la risposta, ovvia, è $Y=4$

esempio 2: se $X=38$, $M=10$ e $N=2$, la potenza 10^2 è 100;

quanto vale Y?

la risposta è $Y=62$ ($62+38=100$).

E se la base è 2? Il procedimento è il medesimo; esaminiamolo attraverso un esempio:

se $X=11001$ e $N=5$, la potenza 2^5 è 100000, quindi il numero che sommato a X dà come risultato 100000 è $Y=00111$. Infatti

$$\begin{array}{r} 11001+ \\ 00111 = \\ 100000 \end{array}$$

Leggiamo quanto svolto in base 10: $11001_2 = 25$, la potenza 2^5 è $32 = 100000$ (ricordati che devi ragionare in base 2).

Quale è quel numero che sommato a 25 dà 32?

Risposta: 7 che in base 2 è proprio 111.

Introduciamo ora la rappresentazione dei numeri interi in complemento a 2 (con N bit).

Definizione:

- la rappresentazione di un numero positivo coincide con la rappresentazione in modulo e segno (entrambe con N bit)
- la rappresentazione di un numero negativo è costituita dal complemento a 2 del corrispondente numero positivo.

Osservazioni:

- l'intervallo di valori rappresentabili con N bit è $[-2^{N-1}, +2^{N-1}-1]$, quindi è possibile rappresentare un valore in più rispetto alla rappresentazione in modulo e segno.
- i numeri positivi iniziano tutti con un bit a 0
- i numeri negativi iniziano tutti con un bit a 1
- esiste una sola rappresentazione dello 0: (0 ... 0).

Riportiamo nella successiva tabella la codifica in modulo e segno e in complemento a 2 dei numeri interi rappresentabile con $N=4$ bit.

in base 10	modulo e segno	complemento a 2
-8	-----	1000
-7	1111	1001
-6	1110	1010
-5	1101	1011
-4	1100	1100
-3	1011	1101
-2	1010	1110
-1	1001	1111
-0	1000	-----
+0	0000	0000
+1	0001	0001
+2	0010	0010
+3	0011	0011
+4	0100	0100
+5	0101	0101
+6	0110	0110
+7	0111	0111

Esiste una regola di facile applicazione che permette di scrivere la rappresentazione di un numero negativo in complemento a 2 con N bit:

1. si scrive la codifica con N bit del numero positivo corrispondente
2. si lascia inalterata, partendo dal bit meno significativo, la sequenza di bit fino al primo 1 che si incontra, che rimane anch'esso invariato; si modificano tutti i altri bit: 0 diventa 1, 1 diventa 0.

Esempio: vogliamo codificare in base 2 con 6 bit il corrispondente valore di -22 in base 10

1° passo: $+22_{10} = 010110_2$

2° passo: si complementa bit a bit, applicando la regola appena esposta

0	1	0	1	1	0	
↓	↓	↓	↓	↓	↓	
1	0	1	0	1	0	quindi $-22_{10} = 101010_2$ in complemento a 2

Affrontiamo ora il problema posto all'inizio del paragrafo:

la sottrazione tra numeri relativi.

Partiamo dalla considerazione che la differenza tra due numeri equivale alla somma algebrica tra il primo operando e l'opposto del secondo.

Infatti

$$\begin{array}{r} 23 - \\ \underline{12} = \\ +11 \end{array} \rightarrow \text{equivale} \quad 23 + (-12) = +11$$

È sufficiente rappresentare il sottraendo in complemento a 2, trasformando così una sottrazione in un'addizione.

Enunciamo la regola per eseguire la somma algebrica tra 2 numeri interi codificati in complemento a 2 su N bit:

- si esprimono i numeri in complemento a 2 su N bit (la codifica differisce solo per i numeri negativi)
- si esegue la somma
- si trascura l'eventuale overflow (ovvero il bit di posizione N+1)

Applichiamo la regola (negli esempi successivi i numeri sono codificati su 6 bit)

Esempio: Operandi di segno opposto

in questo caso il valore ottenuto è sempre corretto (il numero di bit è sufficiente per memorizzare il risultato)

$-22 = 101010$	$101010 +$
$+2 = 000010$	$\underline{000010} =$
	101100

Nel risultato il bit del segno è 1 → valore negativo, quindi occorre complementare 101100 diventa 010100 = 20

risultato finale [-22 + 2 = -20] → risultato corretto

Esempio: Operandi di segno concorde + **è sempre senza overflow** (perché?)

in questo caso il valore ottenuto non sempre è corretto

risultato corretto
(senza overflow)

$$+17 + 8 = +25$$

$$\begin{array}{r} +17 = 010001 \\ +8 = 001000 \\ \hline 010001 + 001000 = 011001 \end{array}$$

$$\text{poiché } [011001_2 = 25_{10}]$$

risultato corretto

risultato errato
(senza overflow)

$$+17 + 16 = +33$$

$$\begin{array}{r} +17 = 010001 \\ +16 = 010000 \\ \hline 010001 + 010000 = 100001 \end{array}$$

Nel risultato il bit del segno è 1 → valore negativo

risultato errato

Esempio: Operandi di segno concorde - **è sempre con overflow** (perché?)

in questo caso il valore ottenuto non sempre è corretto

risultato corretto
(con overflow)

$$-24 + (-7) = -31$$

$$\begin{array}{r} -24 = 101000 \\ -7 = 111001 \\ \hline 101000 + 111001 = 110001 \end{array}$$

Il bit più significativo è il bit di overflow → si trascura (è il 7° bit), il bit immediatamente successivo è il bit del segno: valore 1 quindi il risultato si complementa

$$\text{poiché } [110001_2 = -31_{10}]$$

risultato corretto

risultato errato
(con overflow)

$$-24 + (-12) = -36$$

$$\begin{array}{r} -24 = 101000 \\ -12 = 110100 \\ \hline 101000 + 110100 = 1011100 \end{array}$$

Il bit più significativo è il bit di overflow → si trascura (è il 7° bit), il bit immediatamente successivo è il bit del segno: valore 0 quindi il risultato è positivo

risultato errato

Esercizi: eseguire le seguenti operazioni in aritmetica binaria, codificare i numeri in decimale e controllare i risultati ottenuti

addizione

- | | |
|-----------------------------|----------------------|
| 1) $111_2 + 1_2$ | [8 ₁₀] |
| 2) $1111_2 + 1101_2$ | [28 ₁₀] |
| 3) $1001001_2 + 1110_2$ | [87 ₁₀] |
| 4) $1111101_2 + 1111111_2$ | [380 ₁₀] |
| 5) $1000000_2 + 10000000_2$ | [192 ₁₀] |
| 6) $1100100_2 + 11011_2$ | [127 ₁₀] |

$$7) \quad 1000000_2 + 11111_2 \quad [95_{10}]$$

sottrazione(trai numeri naturali)

$$\begin{aligned} 8) \quad & 100000_2 - 10000_2 & [16_{10}] \\ 9) \quad & 100110_2 - 110_2 & [32_{10}] \\ 10) \quad & 1111000_2 - 10100_2 & [100_{10}] \\ 11) \quad & 1001000_2 - 1000001_2 & [7_{10}] \\ 12) \quad & 1000110_2 - 110010_2 & [20_{10}] \\ 13) \quad & 1011011_2 - 100110_2 & [53_{10}] \\ 14) \quad & 1011000_2 - 11010_2 & [62_{10}] \end{aligned}$$

moltiplicazione

$$\begin{aligned} 15) \quad & 101_2 * 10_2 & [10_{10}] \\ 16) \quad & 101_2 * 110_2 & [30_{10}] \\ 17) \quad & 1010_2 * 111_2 & [70_{10}] \\ 18) \quad & 11011_2 * 1100_2 & [324_{10}] \\ 19) \quad & 11111_2 * 111_2 & [217_{10}] \\ 20) \quad & 11111_2 * 1111_2 & [465_{10}] \\ 21) \quad & 11111_2 * 11111_2 & [961_{10}] \end{aligned}$$

divisione

$$\begin{aligned} 22) \quad & 1100_2 : 11_2 & [Q= 4_{10} ; R= 0_{10}] \\ 23) \quad & 101111_2 : 101_2 & [Q= 9_{10} ; R= 2_{10}] \\ 24) \quad & 101111_2 : 111_2 & [Q= 6_{10}; R= 5_{10}] \\ 25) \quad & 1010000_2 : 1011_2 & [Q=7_{10} ; R= 3_{10}] \\ 26) \quad & 1111101_2 : 110_2 & [Q= 20_{10}; R= 5_{10}] \\ 27) \quad & 10001000_2 : 1111_2 & [Q= 9_{10}; R= 1_{10}] \\ 28) \quad & 100000000_2 : 10000_2 & [Q= 16_{10}; R= 0_{10}] \end{aligned}$$

somma algebrica (in complemento a 2 su 7 bit)

$$\begin{aligned} 29) \quad & 0100011_2 + 0101101_2 & [-48_{10} \text{ errore}] \\ 30) \quad & 0100011_2 + 0001100_2 & [47_{10}] \\ 31) \quad & 1010011_2 + 0010111_2 & [-22_{10}] \\ 32) \quad & 0110011_2 + 1000100_2 & [-9_{10}] \\ 33) \quad & 0011011_2 + 1100100_2 & [-1_{10}] \\ 34) \quad & 1000100_2 + 1001110_2 & [-46_{10} \text{ errore}] \\ 35) \quad & 1110001_2 + 1001111_2 & [-64_{10}] \\ 36) \quad & 1101110_2 + 1110010_2 & [-32_{10}] \end{aligned}$$

LABORATORIO

Conversione dei numeri da un sistema numerico ad un altro

Ora proviamo a costruire un foglio di calcolo che mi permetta di convertire i numeri nei vari sistemi numerici.

Come ipotesi di partenza per tutti gli esercizi che realizzeremo, consideriamo sempre un numero con massimo 6 cifre, ma tale foglio di calcolo potrà essere ampliato a vostro piacimento utilizzando un numero diverso di cifre.

CONVERSIONE BINARIO → DECIMALE

Innanzitutto apriamo il foglio di calcolo e scriviamo le seguenti informazioni:

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO							
2	da:							a:
3	BINARIO							DECIMALE
4	0	1	1	1	0	1		
5	5	4	3	2	1	0	POSIZIONE	
6	2	2	2	2	2	2	BASE	
7							POTENZA	
8							PRODOTTO	
9								

Con lo schema impostato come figura riusciremo a convertire il numero binario "011101" in un numero decimale.

Per prima cosa inserire tutte le informazioni così come scritte in figura.

Nella cella A7 scrivere la formula $=A6^A5$ e trascinare fino alla cella F7

Nella cella A8 scrivere la formula $=A7*A4$ e trascinare fino alla cella F8

Nella cella H4 scrivere la formula = A8+B8+C8+D8+E8+F8 e premere INVIO

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO							
2	da:							a:
3	BINARIO							DECIMALE
4	0	1	1	1	0	1		29
5	5	4	3	2	1	0	POSIZIONE	
6	2	2	2	2	2	2	BASE	
7	32	16	8	4	2	1	POTENZA	
8	0	16	8	4	0	1	PRODOTTO	

Ora sappiamo che “011101” corrisponde a “29” nel sistema decimale!

CONVERSIONE OTTALE → DECIMALE

Proviamo ora ad impostare un foglio di calcolo per effettuare la conversione da un numero ottaale in decimale:

Per esempio, vogliamo trovare il corrispondente numero decimale del numero “256” in base 8.

Partiamo sempre costruendo una tabella come la figura seguente:

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO							
2	da:							a:
3	OTTALE							DECIMALE
4				2	5	6		
5	5	4	3	2	1	0	POSIZIONE	
6	8	8	8	8	8	8	BASE	
7							POTENZA	
8							PRODOTTO	

Successivamente inserite le seguenti informazioni:

Nella cella A7 scrivere la formula =A6^A5 e trascinare fino alla cella F7

Nella cella A8 scrivere la formula =A7*A4 e trascinare fino alla cella F8

Nella cella H4 scrivere la formula = A8+B8+C8+D8+E8+F8 e premere INVIO.

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO							
2	da:							a:
3	OTTALE							DECIMALE
4				2	5	6		174
5	5	4	3	2	1	0	POSIZIONE	
6	8	8	8	8	8	8	BASE	
7	32768	4096	512	64	8	1	POTENZA	
8	0	0	0	128	40	6	PRODOTTO	

Ora sappiamo che il numero "256" in base 8 corrisponde al numero 174 nel sistema decimale.

CONVERSIONE ESADECIMALE → DECIMALE

Per ultimo, proviamo a convertire un numero in sistema esadecimale in decimale.

Ora le cose si complicano un poco perchè le cifre del sistema esadecimale non sono tutte numeriche, ma dopo la cifra 9 dobbiamo considerare A, B, C, D, E, F che rispettivamente corrispondono a 10, 11, 12, 13, 14, 15.

Procediamo quindi nel seguente modo:

Per prima cosa scriviamo la seguente tabella come figura:

	J	K	L
	0	0	
	1	1	
	2	2	
	3	3	
	4	4	
	5	5	
	6	6	
	7	7	
	8	8	
	9	9	
	A	10	
	B	11	
	C	12	
	D	13	
	E	14	
	F	15	

Successivamente impostiamo la seguente tabella:

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO							
2	da:							a:
3	ESADECIMALE							DECIMALE
4				1	4	A		
5							NUMERO CORRISPONDENTE	
6	5	4	3	2	1	0	POSIZIONE	
7	16	16	16	16	16	16	BASE	
8							POTENZA	
9							PRODOTTO	
10								
11								

Nelle celle A5:F5 dobbiamo impostare una Funzione che mi permetta di trovare la cifra numerica corrispondente alla cifra esadecimale inserita.

La funzione in questione è =CERCA.VERT().

Come funziona? Questa funzione, in generale, permette di visualizzare un dato che si trova sullo stesso rigo di un valore dato in input.

La sintassi della funzione è la seguente:

=CERCA.VERT(valore da cercare; tabella dei valori; colonna in cui si trova il valore che si vuole visualizzare).

In particolare, il valore da cercare lo abbiamo inserito nelle caselle gialle; la tabella dei valori si trova nelle celle J1:K16 e l'informazione che ci serve si trova nella seconda colonna della nostra tabella.

Nella cella A5, quindi, proviamo ad inserire la funzione: =CERCA.VERT(A4;\$J\$1:\$K\$16;2) e trascinatela fino alla cella F5.

Nella cella A8 scrivere la formula =A7^A6 e trascinare fino alla cella F8

Nella cella A9 scrivere la formula =A8*A5 e trascinare fino alla cella F9

Nella cella H4 scrivere la formula = A8+B8+C8+D8+E8+F8 e premere INVIO.

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO							
2	da:							a:
3	ESADECIMALE							DECIMALE
4				1	4A			330
5							NUMERO	
	0	0	0	1	4	10	CORRISPONDENTE	
6	5	4	3	2	1	0	POSIZIONE	
7	16	16	16	16	16	16	BASE	
8	1048576	65536	4096	256	16	1	POTENZA	
9	0	0	0	256	64	10	PRODOTTO	
10								

Ora sappiamo che il numero in base esadecimale "14A" corrisponde al numero decimale "330".

CONVERSIONE DECIMALE → BINARIO

Ora proviamo a fare il contrario. Conoscendo il valore in decimale vogliamo calcolare il corrispondente numero in binario.

Visto che abbiamo già utilizzato il numero 29 nel primo esempio, proviamo a prendere in considerazione questo numero per effettuare una controprova.

Impostare il foglio secondo la seguente figura:

	A	B	C	D	E
1	CONVERSIONE DI UN NUMERO				
2					
3	DECIMALE		BINARIO		
4	29				
5					
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO
7		2			
8					
9					
10					
11					
12					
13					

Nella cella A7 scriviamo =A4

Nella cella C7 scriviamo =A7/B7

Nella cella D7 scriviamo =INT(C7)

Nella cella E7 scriviamo =RESTO(A7;B7)

se avete fatto tutto bene, otterrete il risultato come in figura:

	A	B	C	D	E
1	CONVERSIONE DI UN NUMERO				
2					
3	DECIMALE		BINARIO		
4	29				
5					
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO
7	29	2	14,5	14	1
8					
9					
10					
11					

Ora nella cella A8 scriviamo =D8

Nella cella C8 scriviamo =A8/B8

Nella cella D8 scriviamo =INT(C8)

Nella cella E8 scriviamo =RESTO(A8;B8)

Ripetiamo la stessa operazione fino a quando la cella che contiene la “Parte Intera” raggiungerà il valore “0”, come la seguente figura:

	A	B	C	D	E
1	CONVERSIONE DI UN NUMERO				
2					
3	DECIMALE		BINARIO		
4	29				
5					
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO
7	29	2	14,5	14	1
8	14	2	7	7	0
9	7	2	3,5	3	1
10	3	2	1,5	1	1
11	1	2	0,5	0	1
12					

Ora bisogna fare in modo che i resti delle varie divisioni vengano letti a partire dall'ultimo.

Possiamo usare per esempio la funzione =CONCATENA() che permette di unire in un'unica cella informazioni che si trovano in più celle.

Proviamo a scrivere nella cella C4 la seguente funzione =CONCATENA(E11;E10;E9;E8;E7).

Otterremo la seguente figura:

	A	B	C	D	E
1	CONVERSIONE DI UN NUMERO				
2					
3	DECIMALE		BINARIO		
4	29		11101		
5					
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO
7	29	2	14,5	14	1
8	14	2	7	7	0
9	7	2	3,5	3	1
10	3	2	1,5	1	1
11	1	2	0,5	0	1
12					

Abbiamo, in questo modo, verificato che effettivamente il numero decimale “29” corrisponde al numero binario “11101”.

CONVERSIONE DECIMALE → OTTALE

Per quanto riguarda l'impostazione del foglio per eseguire questa conversione procediamo come nell'esercizio precedente, solo che questa volta nella colonna DIVISORE dobbiamo digitare "8".

Otterrete il seguente risultato:

	A	B	C	D	E
1	CONVERSIONE DI UN NUMERO				
2					
3	DECIMALE		BINARIO		
4	174		00256		
5					
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO
7	174	8	21,75	21	6
8	21	8	2,63	2	5
9	2	8	0,25	0	2
10	0	8	0	0	0
11	0	8	0	0	0
12					

CONVERSIONE DECIMALE → ESADECIMALE

Anche in questo caso procedete sempre allo stesso modo, modificando semplicemente la colonna relativa a divisore, inserendo questa volta il valore "16".

Otterrete il seguente risultato:

	A	B	C	D	E
1	CONVERSIONE DI UN NUMERO				
2					
3	DECIMALE		BINARIO		
4	330				
5					
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO
7	330	16	20,63	20	10
8	20	16	1,25	1	4
9	1	16	0,06	0	1
10	0	16	0	0	0
11	0	16	0	0	0

Per poter assegnare la lettera per i numeri maggiori di 9, bisogna utilizzare sempre il comando =CERCA.VERT() impostando a parte una tabella simile a quella utilizzata nell'esercizio di conversione da esadecimale a decimale.

Nella cella C4 bisognerà utilizzare la funzione =CONCATENA(E7;G1:H16;2). Otterremo, in questo modo la seguente figura:

	A	B	C	D	E	F	G	H
1	CONVERSIONE DI UN NUMERO						0	0
2							1	1
3	DECIMALE		BINARIO				2	2
4	330						3	3
5							4	4
6	DIVIDENDO	DIVISORE	RISULTATO	PARTE INTERA	RESTO		5	5
7	330	16	20,63	20	10	A	6	6
8	20	16	1,25	1	4	4	7	7
9	1	16	0,06	0	1	1	8	8
10	0	16	0	0	0		9	9
11	0	16	0	0	0		10	A
12							11	B
13							12	C
14							13	D
15							14	E
16							15	F

BOOK IN PROGRESS

I SISTEMI INFORMATIVI

A cura del prof Giuseppe Cavallo
adattamenti a cura di Salvatore Madaro

INDICE DEI CONTENUTI

I SISTEMI INFORMATIVI.....	3	Sistema di reporting aziendale MIS	13
Il Ciclo di vita di un sistema informativo .	6	Sistema di supporto alle decisioni DSS .	13
Componenti del sistema informativo.	8	Le tipologie di sistemi informativi.	14
Centro del controllo ed attività critiche. ...	9	Sistemi di gestione dei flussi di lavoro. ..	14
Storia	9	Sistemi gestionali	14
Sistema Informativo Automatizzato	10	Sistemi per l'automazione d'ufficio.....	14
Il Sistema Informativo dal punto di vista strategico	11	Sistemi di comunicazione.....	15
Evoluzione dei sistemi informativi	12	Sistemi per il governo dell'azienda.	15
Sistema di elaborazione dati EDP	13	Sistemi statistici.....	15
		Sistemi territoriali.	15

I SISTEMI INFORMATIVI

Definizione

Un Sistema Informativo è un insieme di apparecchiature, procedure e persone che hanno il compito di:

- **Raccogliere**
- **Organizzare**
- **Selezionare**
- **Archiviare**
- **Comunicare i dati riguardanti l'attività di un'organizzazione.**

Con l'obiettivo di :

mettere a disposizione dei responsabili delle decisioni operative tutte le informazioni necessarie per effettuare le migliori scelte possibili .

Il sistema informativo è una componente fondamentale di ogni organizzazione.

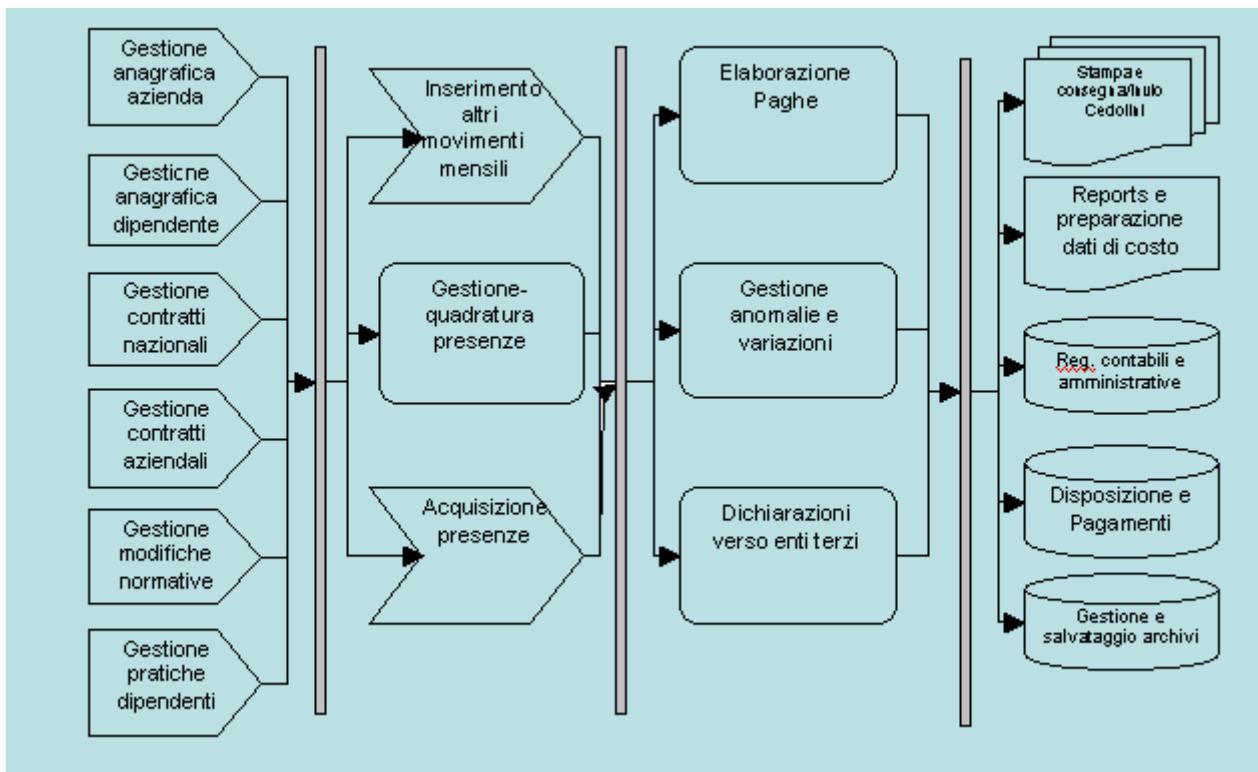
La definizione del sistema informativo di una organizzazione richiede l'identificazione:

- ✓ della missione
- ✓ degli obiettivi
- ✓ delle risorse disponibili
- ✓ dei processi di gestione delle risorse.

Il processo è considerato l'elemento chiave di un'organizzazione e rappresenta l'insieme delle attività tra loro correlate e finalizzate alla realizzazione di un risultato definito misurabile, che coinvolge più risorse ed attraversa più ruoli e strutture.

I sistemi informativi devono essere coerenti con i processi identificati nell'organizzazione.

Nel seguente grafico è rappresentato l'insieme dei processi coinvolti nell'attuazione di un sistema informativo contabile che ha come obiettivo la produzione dei cedolini paga dei dipendenti di una azienda



Secondo un'ottica tradizionale i processi si distinguono in :

- Processi operativi
- Processi di controllo
- Processi direzionali

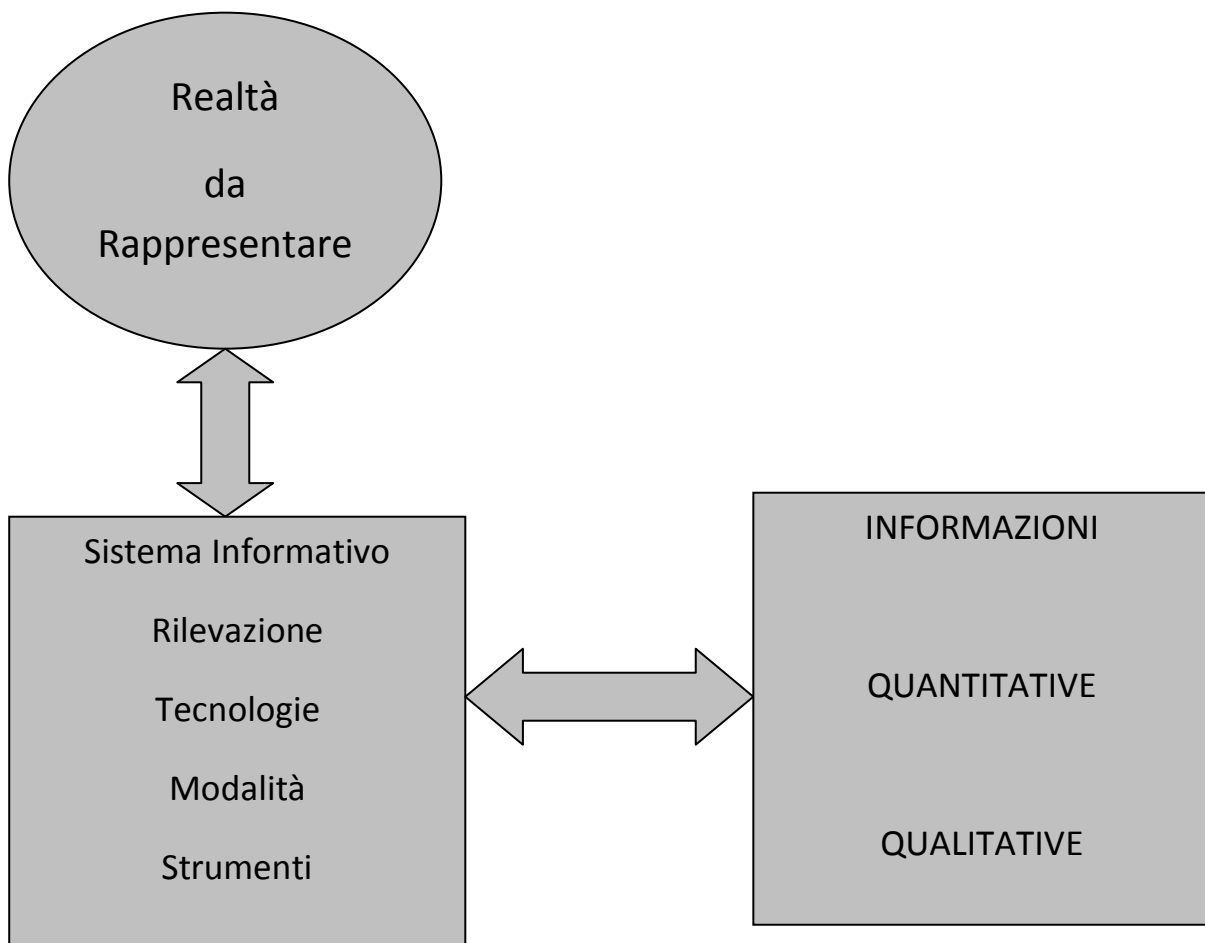
Di conseguenza anche i sistemi informativi che li supportano e realizzano risentono di questa classificazione, richiedono perciò diverse modalità di realizzazione .

L'evoluzione dei sistemi informativi ha portato ad affrontare in ordine consecutivo :

- l'automazione dei processi operativi
- l'automazione delle attività di controllo
- l'automazione delle attività direzionali.

Il percorso di evoluzione dei sistemi informativi ha comportato una loro graduale integrazione, corrispondentemente l'informatica si è evoluta da tecnologia di produzione, con obiettivi principalmente di efficienza , a strumento di governo delle aziende.

In un sistema informativo è necessario stabilire quali tipologie di dati e di attività devono essere trattati: la loro identificazione guida alla scelta delle opportune tecnologie e metodologie da impiegare per lo sviluppo.



Il Ciclo di vita di un sistema informativo

Le fasi di realizzazione di un sistema informativo sono generalmente le seguenti:

✓ **nascita dell'esigenza**

- Data una specifica realtà si tende a migliorare la situazione esistente tramite il perseguimento di determinati obiettivi.

✓ **studio di fattibilità**

- serve a definire in maniera per quanto possibile precisa:
 - quali potrebbero essere i vari componenti del sistema ;
 - i costi di realizzazione
 - le varie alternative possibili,
 - l'analisi costi/benefici per singola alternativa
 - stabilire le priorità della realizzazione delle varie componenti del sistema.

✓ **raccolta e analisi dei requisiti:**

- consiste:
 - nell'individuazione e nello studio delle proprietà e delle funzionalità che il sistema informativo dovrà avere;
 - nel coinvolgimento di tutti coloro che dovranno utilizzarlo effettivamente al termine della realizzazione;
 - nel produrre una descrizione completa ma generalmente informale di tutti i dati coinvolti e delle operazioni su di essi;
 - nella descrizione , in maniera il più possibile precisa , degli input da fornire al sistema e degli output attesi;
 - nel definire i requisiti software e hardware del sistema informativo.

✓ **progettazione:**

- riguarda :
 - la progettazione dei dati : definizione delle strutture dati che conterranno in seguito le informazioni elementari
 - la progettazione delle applicazioni: le caratteristiche dei progetti applicativi
 - la progettazione dell'architettura tecnica di sistema : rappresenta l'infrastruttura individuandone le caratteristiche in termini di sistemi (server), connettività, sicurezza fisica e logica.

✓ **sviluppo:**

- consiste:
 - nella realizzazione del sistema informativo secondo la struttura e le caratteristiche definite nella fase di progettazione.
 - nella creazione e caricamento di una base di dati
 - nella produzione del codice dei programmi.

✓ **validazione e collaudo:**

- consiste :
 - nella verifica del corretto funzionamento di tutte le applicazioni (il sistema risponde correttamente a tutti i requisiti?)
 - nella verifica della qualità del sistema informativo.
 - nella attivazione di una fase sperimentale che possa coinvolgere tutte le condizioni operative anche in situazione di stress del sistema

✓ **avviamento:**

- consiste :
 - nella messa in funzione del sistema
 - nell'affiancamento per un periodo transitorio all'utente finale
 - nell'individuazione di possibili affinamenti delle procedure e di possibili bug di analisi scaturiti dalla messa in funzione e dall'utilizzo delle nuove procedure
 - nell'erogazione di corsi di formazione al personale addetto all'utilizzo del sistema
 - nel recupero e travaso di dati strategici provenienti da applicazioni preesistenti
 - nell'attivazione dei collegamenti tra tutte le procedure e le applicazioni
 - nell'avvio della operatività reale.

✓ **funzionamento:**

- consiste nell'operatività a regime del sistema che svolge tutti i compiti per il quale e' stato progettato. Se non si verificano malfunzionamenti o revisioni delle funzionalità del sistema, questa attività richiede solo operazioni di gestione e manutenzione

✓ **manutenzione:**

- consiste :
 - nella manutenzione correttiva nella quale si consolida e si correggono eventuali anomalie non rilevate nella fase di collaudo e di avviamento
 - nella manutenzione evolutiva dove il sistema cresce arricchendosi con altre funzionalità scaturite in un secondo momento dopo la fase di analisi e di sviluppo dovute all'insorgere di nuove esigenze applicative .

Il processo su indicato non è detto che sia sempre strettamente sequenziale, in quanto spesso durante l'esecuzione di una attività citata bisogna rivedere decisioni prese nell'attività precedente. Quello che si ottiene è un ciclo di operazioni di tipo ripetitivo che termina con il corretto funzionamento del sistema .

Molto spesso , in accordo con le disponibilità finanziarie , si aggiunge una ulteriore attività accessoria che prevede :

- realizzazione rapida di un prototipo semplificato del sistema mediante l'utilizzo di sistemi software
- sperimentazione delle funzionalità del sistema mediante il prototipo

In genere tale fase di verifica porta, di conseguenza, un affinamento dei requisiti del progetto e quindi revisioni successive con l'inevitabile riprogrammazione di tutte le attività coinvolte.

Componenti del sistema informativo.

Il sistema informativo è da interpretarsi come un sistema aperto interagente con le altre parti costituenti l'azienda, nonché l'ambiente esterno all'azienda.

Il sistema informativo si configura come un insieme ordinato di elementi , anche molto diversi tra loro, che raccolgono, elaborano, scambiano e archiviano dati con lo scopo di produrre e distribuire le informazioni alle persone che ne hanno bisogno, nel momento e nel luogo adatto.

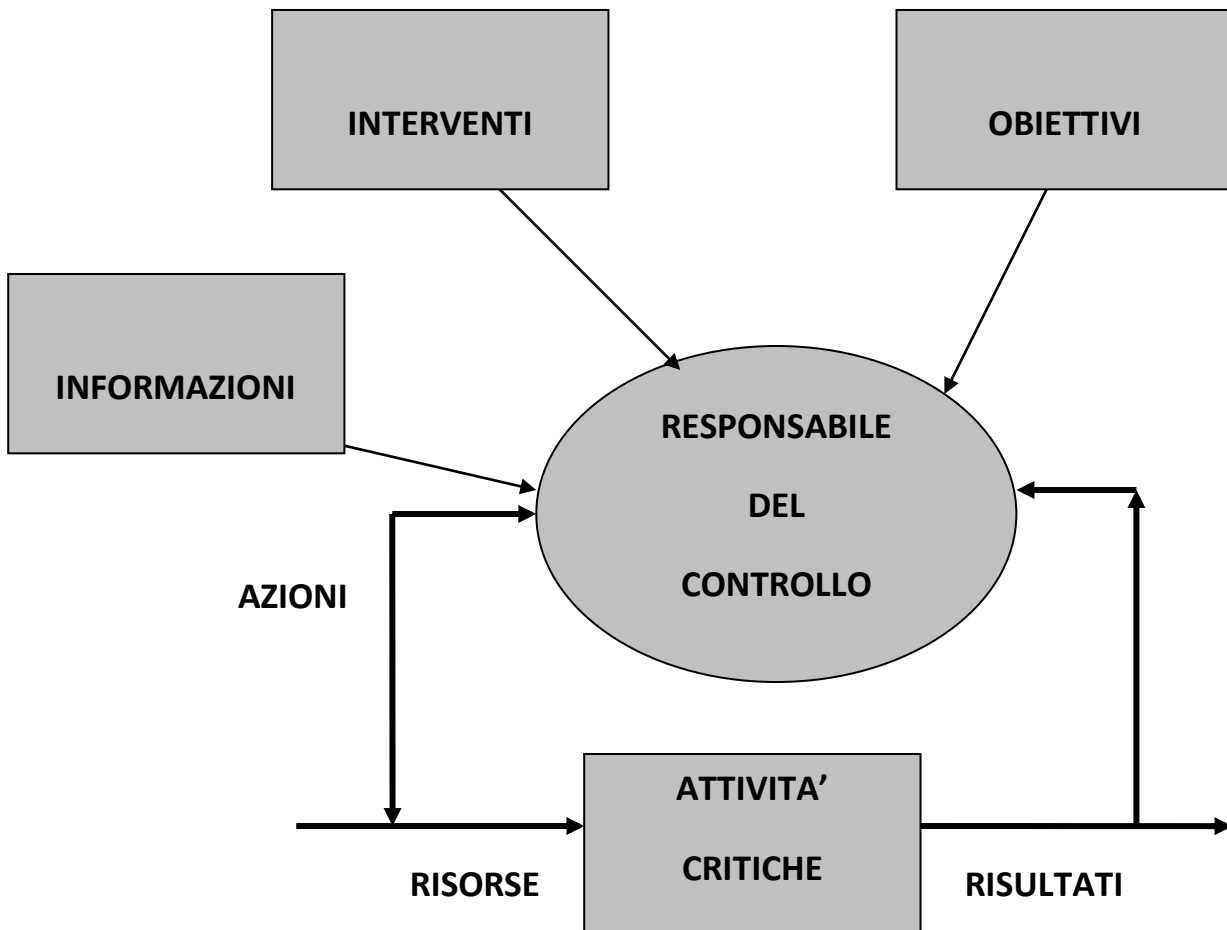
Il sistema informativo in una impresa deve quindi essere visto come quel complesso di elementi in grado di fornire le informazioni necessarie alle persone che lavorano a tutti i livelli della struttura .

Storia

Il primo tentativo di identificare il concetto di sistema informativo è stato fatto da Forrester intorno al 1960. La sua intuizione deriva dall'osservazione che i sistemi sociali, e quindi le aziende, presentano al loro interno attività che necessitano di un elevato grado di attenzione da parte di organi di controllo:

si tratta di attività critiche per il conseguimento degli obiettivi del sistema.

Centro del controllo ed attività critiche.



Il centro del controllo è un punto di decisione: interviene quando le informazioni sull'andamento delle attività dell'organizzazione indicano la necessità di azioni correttive.

In questa ottica il sistema informativo può quindi essere visto come un sottosistema del sistema di controllo aziendale e deve svolgere un ruolo di standardizzazione nella rilevazione e misurazione dei fenomeni aziendali.

Sistema informativo si può avvalere di un sistema informatico.

Con la denominazione di Sistema Informativo Automatizzato si intende quella parte del sistema informativo che è realizzata con tecnologie informatiche.

Le informazioni vengono :

- ✓ raccolte,
- ✓ elaborate,
- ✓ archiviate,
- ✓ scambiate mediante l'utilizzo delle tecnologie dell'informazione e della comunicazione.

Nelle varie istituzioni , prima dell'introduzione delle macchine d'ufficio e degli elaboratori elettronici, le informazioni venivano gestite con metodi manuali.

Si sono così instaurate complesse e lunghe procedure finalizzate a:

- ✓ registrare e gestire i dati su supporti cartacei
- ✓ archiviare i documenti
- ✓ ricercare negli schedari

La manualità di queste operazioni comportavano una notevole lentezza nei riscontri nonché una elevata generazione di errori .

Il tasso di errore elevato dovuto ai motivi su indicati costringe ad introdurre cicli di controllo delle informazioni: determinando la bassa efficienza del sistema.

Queste procedure si basano sulla gestione di processi anche complessi attraverso meccanismi che richiedono attività non strettamente necessarie. Si pensi ad esempio al documento o fascicolo che arriva sulla scrivania di chi deve poi effettuare una lavorazione:

l'oggetto sul quale si basa il funzionamento del processo è il documento cartaceo.

Per il funzionamento del processo in questo caso è necessario:

fotocopiare il documento in numero quantomeno pari alle lavorazioni in parallelo che deve subire!

Le tecnologie informatiche e di comunicazione incidono proprio sull'esecuzione delle varie operazioni accelerando le funzioni di registrazione, elaborazione, archiviazione, ricerca dei dati.

Il Sistema Informativo dal punto di vista strategico .

Un caso molto particolare di risorsa su cui operano tutte le aziende è l'informazione.

L'informazione è infatti una risorsa che implica tutte le altre risorse.

I processi, per poter essere eseguiti, si basano sulla conoscenza delle risorse sulle quali agiscono, in altri termini hanno bisogno di informazioni: l'insieme delle informazioni gestite, - generate, utilizzate, elaborate dai processi costituisce il sistema informativo.

Per esempio, i processi aziendali che riguardano la gestione del personale hanno bisogno di informazioni riguardanti :

- ✓ i dati sulla disponibilità, sul mercato del lavoro, di personale con specifiche caratteristiche professionali
- ✓ le informazioni relative alle persone contattate
- ✓ la modalità di valutazione del personale
- ✓ l'individuazione del tipo di formazione idoneo
- ✓ la raccolta dei dati relativi alle singole persone presenti nell'azienda in organico

Quella parte del sistema informativo in cui le informazioni sono raccolte, elaborate, archiviate, scambiate mediante l'uso delle tecnologie della informazione e della comunicazione costituisce il Sistema Informatico.

Evoluzione dei sistemi informativi: dal supporto all'operatività alle decisioni.

Inizialmente le imprese sono partite dalle applicazioni di natura operativa. Gli investimenti richiesti dall'automazione erano giustificati dai volumi di dati trattati e dalla ripetitività delle procedure (tipicamente l'elaborazione di stipendi e fatture). Si tratta in sostanza delle applicazioni basilari che storicamente rappresentano il risultato tangibile e positivo dell'introduzione di calcolatori nelle aziende.

L'insieme di queste applicazioni è stato denominato sistema di elaborazione dati, nella letteratura anglosassone EDP System (Electronic Data Processing System).

In un secondo tempo sono stati costruiti sistemi informativi aventi lo scopo di supportare le attività di controllo. Il fine è rappresentato dal controllo costante sull'andamento delle attività operative. Sono applicazioni che forniscono alla direzione informazioni standardizzate in modo ciclico . Si tratta di applicazioni di controllo di gestione e reporting statistico sulle attività aziendali.

L'insieme di queste applicazioni è stato denominato sistema per la produzione di informazioni per il controllo direzionale, nella letteratura anglosassone MIS (Management Information System).

Infine troviamo i sistemi per il supporto alle attività strategiche. Si tratta di sistemi che supportano le attività direzionali non predefinite e poco strutturate.

Sono stati denominati sistemi per il supporto alle decisioni, nella letteratura anglosassone DSS (Decision Support System).

Questi tre aspetti del sistema informativo automatizzato si differenziano in modo sensibile, nella tabella che segue si presentano in sintesi le principali caratteristiche di questi tre aspetti.

Sistema di elaborazione dati EDP (Electronic Data Processing System)

- ✓ Sostituisce e integra le attività impiegate manuali
- ✓ È utilizzato passivamente
- ✓ Ha come obiettivo l'efficienza nel risparmio del tempo e nella qualità
- ✓ Gestisce abilmente attività concorrenti effettuandone la sincronizzazione
- ✓ Punta sulla precisione e cura i risultati
- ✓ Utilizza standard per la produzione
- ✓ Rimane costante nel tempo

Sistema di reporting aziendale MIS (Management Information System).

- ✓ Fornisce periodicamente dati di sintesi per favorire il controllo periodico aziendale
- ✓ Richiede una attenta analisi interpretativa dei dati da parte dell'utente
- ✓ È orientato all'efficacia e all'efficienza
- ✓ Mantiene il riferimento ai dati relativi a situazioni pregresse
- ✓ Usa modelli standard per la comunicazione delle informazioni
- ✓ Mette in risalto in maniera evidente le situazioni rilevate fuori norma
- ✓ Rimane costante nel tempo

Sistema di supporto alle decisioni DSS (Decision Support System).

- ✓ È di supporto agli staff manageriali

- ✓ Deve essere utilizzato continuamente
- ✓ Punta all'efficacia
- ✓ Ipotizza , prevede possibili scenari futuri in base alla analisi delle situazioni passate già consolidate
- ✓ Accetta l'approssimazione se è attendibile e giustificata
- ✓ Estremamente flessibile

Le tipologie di sistemi informativi.

I sistemi informativi possono essere rappresentati in diverse categorie in dipendenza di diversi fattori:

- ✓ la natura delle informazioni trattate
- ✓ i domini applicativi interessati dal sistema
- ✓ il contesto organizzativo di riferimento
- ✓ la tecnologia utilizzata

Si possono quindi distinguere sistemi informativi con diverse caratteristiche.

Sistemi gestionali (o transazionali).

Si tratta dei sistemi più tradizionali, perseguono l'obiettivo dell'informatizzazione delle attività strutturate e ripetitive. Riguardano prevalentemente i processi ed i settori operativi.

Ad esempio si tratta di: sistema di bilancio, gestione degli stipendi, gestione di un'anagrafe comunale, magazzino , fatturazione ecc.

Sistemi di gestione dei flussi di lavoro.

Sono i sistemi che normalmente vanno ad integrazione di quelli gestionali. L'obiettivo che perseguono è la formalizzazione dei flussi di lavoro che coinvolgono più persone e settori, per consentirne la massima automazione.

Sistemi per l'automazione d'ufficio.

Sono i sistemi che si sono diffusi a seguito dell'introduzione dei personal computer e dei relativi software per lo svolgimento di attività tipiche d'ufficio (elaboratori di testo, gestione di tabelle, grafici e disegni).

Lo sviluppo delle reti, in particolare di quelle locali, e l'integrazione con i sistemi di gestione dei flussi di lavoro ha portato all'evoluzione di questi sistemi dall'utilizzo individuale al lavoro di gruppo.

Consentono quindi la condivisione di dati, funzioni e risorse tecnologiche (stampanti, scanner, ecc.).

Sistemi di comunicazione.

Sono quei sistemi che facilitano lo scambio di informazioni tra operatori coinvolti in diverse attività, consentono comunicazioni informali. La modalità più utilizzata è la posta elettronica.

Sistemi statistici.

L'obiettivo di questi sistemi è la rilevazione di dati dalle fonti di origine, la loro organizzazione, analisi, omogeneizzazione, aggregazione statistica, per fini conoscitivi e di pianificazione.

Sistemi per il governo dell'azienda.

Sono rivolti agli alti livelli aziendali per le decisioni di tipo gestionale ed operativo:

- ✓ DSS – Decision Support System, per la valutazione di ipotesi alternative di supporto a decisioni poco strutturate
- ✓ MIS – Management Information System, per il supporto all'analisi e valutazione di decisioni strutturate tipiche del livello di controllo dell'azienda
- ✓ EIS – Executive Information System, per prospettare al top management i dati di sintesi con elevata interattività ed alta flessibilità di presentazione (cruscotti aziendali)

Sistemi territoriali.

Sono i sistemi che trattano le informazioni in modo georeferenziato, tramite carte, mappe e sovrapposizione di dati tematici.

BOOK IN PROGRESS

LA STRUTTURA DI UN ELABORATORE

A cura del prof Giuseppe Cavallo

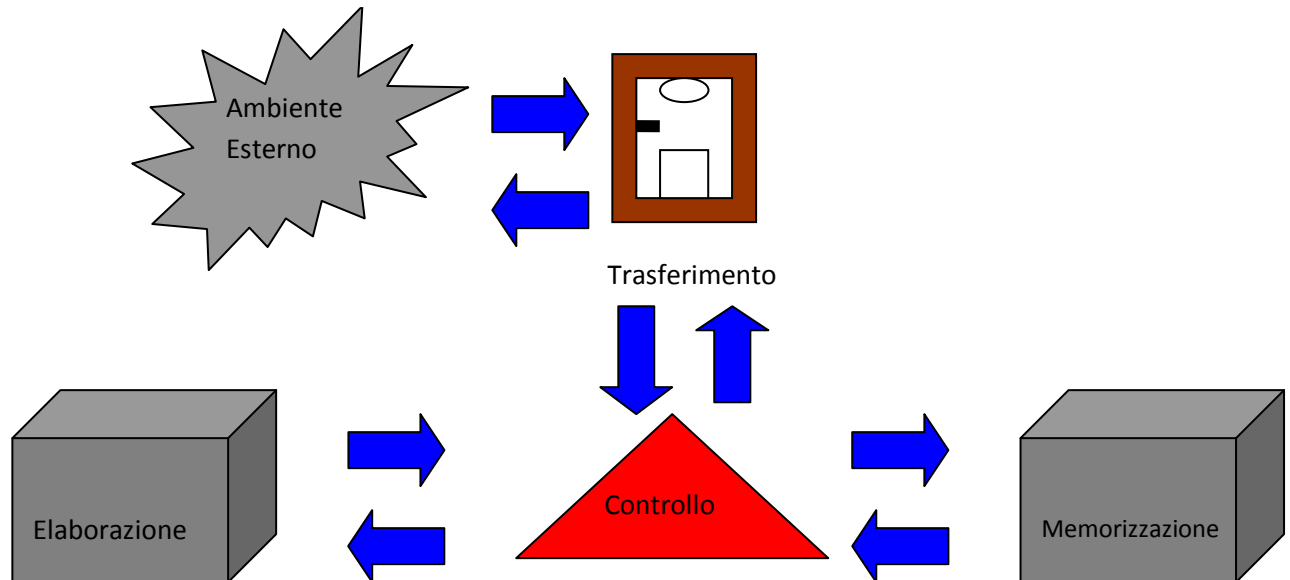
Competenze	Abilità	Conoscenze
Utilizzare, con autonomia e responsabilità, le reti e gli strumenti informatici nelle attività di studio, ricerca e approfondimento disciplinare.	Riconoscere le caratteristiche logico-funzionali di un computer e il ruolo strumentale svolto nei vari ambiti.	Architettura e componenti di un computer
		Interfacce e periferiche di un computer

INDICE DEGLI ARGOMENTI

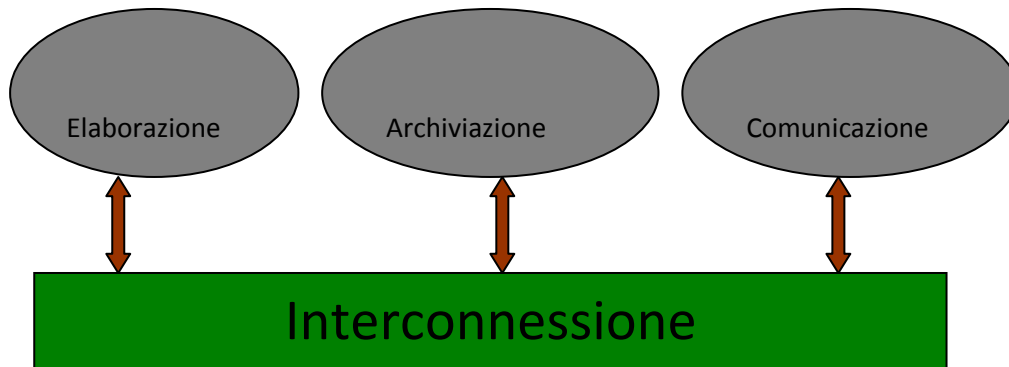
Schema funzionale di un calcolatore	Pag 3
Schema concettuale di un calcolatore	Pag 4
Le istruzioni	Pag 5
Componenti strutturali di un calcolatore	Pag 6
Macchina di Von Neumann	Pag 8
Rappresentazione delle informazioni nella memoria centrale	Pag 10
Architettura dei calcolatori elettronici	Pag 14
Funzionamento dell'Unità di Elaborazione	Pag 16

La struttura di un calcolatore

Schema Funzionale di un Calcolatore



Schema concettuale di un Calcolatore



Il calcolatore è capace di:

a) Contenere Informazioni

b) Trasferire Informazioni (da un punto all'altro al suo interno, da dentro a fuori e viceversa)

c) Fare Calcoli con le informazioni che possiede

b) e c) significa ELABORARE, MANIPOLARE Informazioni in modo "preciso" e veloce.

Le istruzioni per eseguire queste manipolazioni sono scritte nei **programmi**.

Ci chiediamo adesso che cosa sono le **Informazioni**:

per comprendere ciò consideriamo un **esempio di informazioni**

Quali dati sono coinvolti nel calcolo dell'area di un triangolo ? :

a) il valore della base di un triangolo in cm.

b) il valore della altezza in cm.

c) il fatto che dobbiamo moltiplicare base per altezza.

d) il valore 2.

e) il fatto che dividendo base x altezza per 2 si ottiene l'area del triangolo.

DATI e ISTRUZIONI rappresentano quindi l'informazione che da ora in avanti indicheremo con il termine INFORMAZIONI.

Il comportamento umano per la soluzione del su detto problema e' il seguente :

- sapere quanto è la base e scriverlo su un foglio di lavoro $b=5$
- sapere quanto è l' altezza e scriverlo su un foglio di lavoro $h=6$
- fare il prodotto $b \times h=30$ scritto sempre sul foglio
- dividere il prodotto ottenuto per 2, il risultato scritto sul foglio
- dire che l'area del triangolo è uguale a 15.

Il calcolatore lavora allo stesso modo , ma il foglio su cui normalmente appuntiamo le nostre operazioni coincide con la memoria centrale, che e' rappresentabile come una matrice in cui le celle sono le locazioni dove sono memorizzate le informazioni .

Esempio di locazioni di memoria centrale dove sono coinvolte le informazioni precedenti

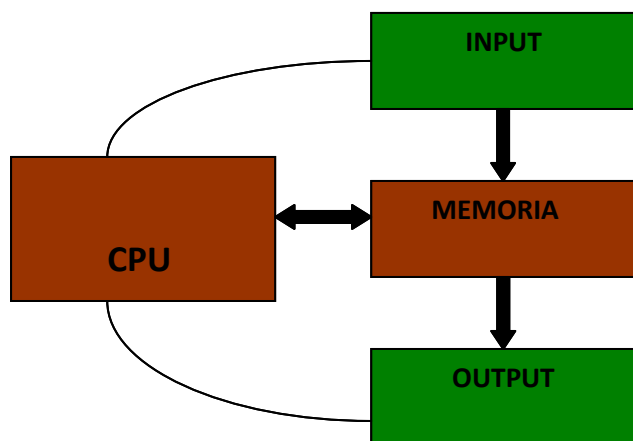
5	6	2
	30	
		15

Le Istruzioni

calcolare $b \times h$	dividere per 2	stampare l'area
------------------------	----------------	-----------------

Normalmente le istruzioni, cioè particolari tipi di informazioni, sono azioni che una macchina deve eseguire e sono memorizzate in locazioni di memoria consecutive

Le informazioni nel nostro caso sono i passi del problema, cioè la **base** è da sapere, **l'altezza** è da sapere, la **moltiplicazione**, la divisione per **2**, il problema trovare **l'area** del triangolo è da assegnare.



Possiamo immaginare che nella memoria sia già contenuto il programma da far eseguire al calcolatore.

Quello che avviene può essere schematizzato così:

- ✓ i dati vengono trasferiti dall'unità di INPUT nella memoria; infatti c'è bisogno di far transitare da "fuori" a "dentro" il calcolatore i dati che dovranno servire alla soluzione.

Questi dati vengono memorizzati in due locazioni (quelle con 5 e 6), che chiameremo base e altezza .

Eseguendo l'istruzione: "calcolare base per altezza", il calcolatore esegue 6×5 e memorizza il risultato in un'altra locazione che chiameremo prodotto. Eseguendo l'istruzione calcolare $\text{prodotto}/2$, il risultato viene memorizzato in una locazione che chiameremo area. Poi, eseguendo l'ultima istruzione, stampare area (cioè il valore 15) viene trasferito all'esterno del calcolatore, verso l'unità di OUTPUT.

Componenti strutturali di un calcolatore

Nel calcolatore possiamo riconoscere le seguenti componenti strutturali fondamentali:

La **MEMORIA** è la parte del calcolatore per contenere le INFORMAZIONI. Per essere più precisi in un sistema di elaborazione la memoria ha lo scopo di memorizzare programmi, dati di ingresso, risultati parziali e finali.

La **CPU** è la parte che elabora INFORMAZIONI ed esegue **calcoli e trasferimenti**. Infatti noi possiamo pensare la CPU come costituita da due parti :

- ✓ la **Unità Logico-Aritmetica (ALU)** avente la funzione di eseguire calcoli logici ed aritmetici
- ✓ la **Unità di Controllo (CU)** che governa il funzionamento complessivo dell'elaboratore.

L' INPUT è la parte (Unità) che è il canale attraverso cui le informazione vanno **in memoria** (dall'esterno del calcolatore al suo interno).

L' OUTPUT è il canale di trasferimento delle informazioni **dalla memoria all'esterno**

Quindi l' elaboratore esegue istruzioni di :

- ✓ Input (ingresso)
- ✓ Calcolo Trasferimento
- ✓ Output (uscita)

controllate/eseguite dalla CPU (Central Processing Unit) cioè il processore del computer.

Le unità di ingresso e di uscita

Ci limitiamo a presentare solo alcune delle unità che sono attualmente in uso.

UNITA' DI INPUT

La tastiera, il mouse, lo joystick, il microfono, il telefono (modem), il touch screen, gli strumenti di misura digitali connessi con il calcolatore.

UNITA' DI OUPUT

Il video (monitor), la stampante, lo speaker, la cuffia, gli strumenti medici, plotter.

Attraverso la tastiera l'utente invia al computer comandi e dati mentre attraverso il video l'utente riceve sia l' output dei programmi sia la visualizzazione delle informazioni da lui inserite.

La memoria secondaria

La memoria centrale che abbiamo visto è destinata ad archiviare istruzioni e dati che siano immediatamente disponibili per essere trattati dalla CPU.

Se si vuole (e si deve) memorizzare le informazioni in modo **permanente** serve la memoria secondaria o ausiliaria chiamata anche memoria di massa .

Queste informazioni vanno trasferite in memoria centrale ogni qualvolta vanno elaborate. La memoria secondaria (MS) è strutturalmente diversa dalla memoria centrale (MC) la

quale ha la caratteristica di cancellare tutte le informazioni ivi contenute ogni qualvolta non sia alimentata elettricamente; innanzitutto è molto più **grande** in termini sia fisici che di capacità di immagazzinamento di INFORMAZIONI, poi è più **lenta** nel senso che per prelevare qualsiasi informazione da essa l'elaboratore impiega un tempo maggiore per l'accesso ai dati. Esempi di memoria secondaria sono: i floppy disk, i cd-rom, l'hard disk, i nastri magnetici. L'hard disk può avere una dimensione da 100 a 1000 volte maggiore della MC, ma queste INFORMAZIONI contenute nello hard disk devono essere trasferite sempre in memoria centrale.

Quanto detto fin qua coincide con l'Architettura di Von Neumann.

Macchina di Von Neumann

Lo scienziato John Von Neumann nel 1946 pubblicò un articolo sul progetto di una macchina per il calcolo automatico che introduceva il concetto di programma registrato in memoria.

Il calcolatore digitale moderno nasce all'inizio degli anni '40 per merito di John von Neumann, J. Presper Eckert e John William Mauchly. Questi concettualizzarono l'idea di macchina ad accesso casuale (RAM Random Access Machine).

Tale nomenclatura sta a indicare che il calcolatore ha una memoria organizzata come sequenza di celle di informazione, in cui il tempo per accedere a una cella non dipende dalla cella stessa (non si devono scorrere le celle sino a trovare quella cercata, la si accede immediatamente).

Tale macchina è anche detta macchina di von Neumann.

La caratteristica fondamentale della macchina di von Neumann è che la memoria contiene sia i programmi che i dati. Il programma è:

- ✓ la sequenza di istruzioni che devono essere eseguite
- ✓ i dati le informazioni che il programma elabora per produrre il risultato (queste informazioni includono:
 - i dati di ingresso
 - parte dei dati intermedi (che possono anche essere memorizzati nei registri interni della CPU per poi essere cancellati)
 - i risultati della computazione.

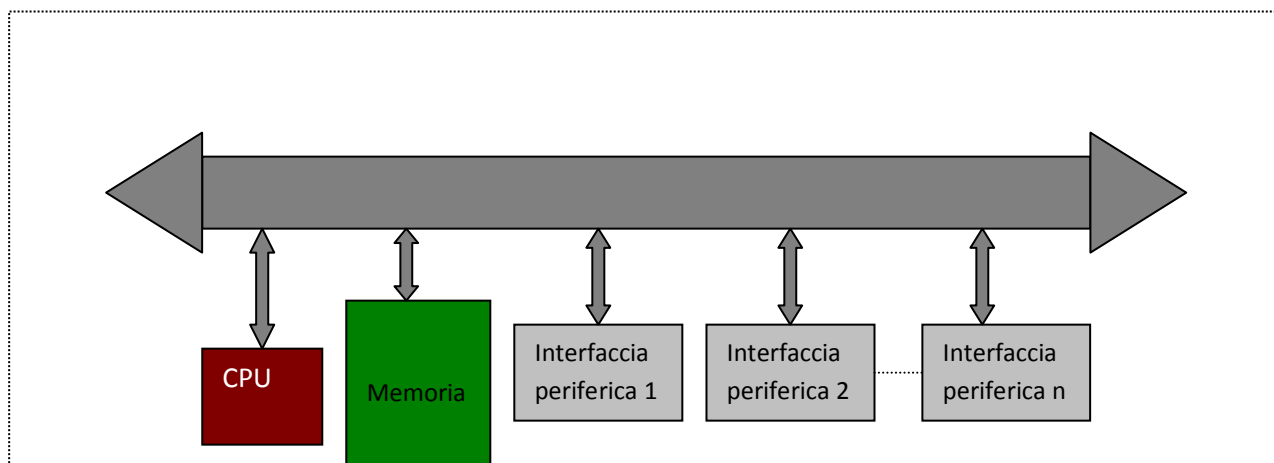
Questa rappresenta la fondamentale differenza tra le macchine costruite o pensate sino a quel momento e i moderni calcolatori digitali:

il programma memorizzato nella stessa memoria dei dati e la possibilità che un programma modifichi un altro programma nel senso che possa scrivere nell'area di memoria dove risiede un altro programma .

Schema della macchina von Neumann

Gli elementi funzionali della Macchina sono quattro :

- ✓ **Unità' di elaborazione (CPU , Central Processor Unit)**
- ✓ **Memoria centrale**
- ✓ **Periferiche**
- ✓ **Bus di sistema**



L'Elaboratore Elettronico è una macchina costituita da vari organi, dette UNITÀ, che possono essere così suddivise:

- Unità centrale di Elaborazione (Microprocessore) composta dall'UNITÀ di CONTROLLO, che ha la funzione di governare tutte le attività di sistema, e dall'UNITÀ ARITMETICO-LOGICA (ALU), che realizza le operazioni di tipo aritmetico (addizioni, sottrazioni, moltiplicazioni, ecc.) e logico decisionale (AND, OR, NOT, confronti, ecc.).
- Unità di INPUT e OUTPUT, in pratica tutti quei dispositivi che consentono di immettere informazioni nel sistema e di leggerne le risposte.
- Unità di Memoria Centrale (ROM e RAM) e di Memoria di Massa (Nastri e Dischi), in altre parole tutti i dispositivi atti a contenere informazioni binarie, con metodi

differenti secondo la tecnologia usata; la scrittura e la lettura di queste informazioni avvengono sotto il controllo dell'Unità Centrale.

Questa struttura di massima (schema di Von Neumann) è comune sia ad un grande elaboratore sia un microcomputer; tutti i componenti fisici che costituiscono la base dell'elaboratore sono indicati con il termine **HARDWARE**, mentre l'insieme dei programmi che consentono alla macchina di funzionare formano il **SOFTWARE**.

Il tempo necessario per l'esecuzione di ogni istruzione elementare è variabile secondo l'operazione e secondo il microprocessore, ma è misurabile nell'ordine dei nanosecondi (miliardesimi di secondo); il dispositivo che scandisce questo tempo è un oscillatore al quarzo (detto **CLOCK**) la cui frequenza è misurata in MegaHertz.

Le informazioni si misurano in termini di celle elementari binarie:

- ✓ **BIT** la singola cifra binaria che può assumere il valore 0 o 1.
- ✓ **BYTE** un gruppo di 8 bit, generalmente identificato con una cella di memoria.
- ✓ **KILOBYTE Kb** pari a $1024 \text{ byte} = 2^{10}$
- ✓ **MEGABYTE Mb** pari a $1048575 \text{ byte} = 2^{20}$
- ✓ **GIGABYTE Gb** pari a $1073740800 \text{ byte} = 2^{30}$
- ✓ **TERABYTE Tb** pari a $1024 \text{ Gb} = 2^{40}$

Rappresentazione delle informazioni nella Memoria Centrale del calcolatore

Precisazioni

I : acquisizione di nuovi dati che vanno in memoria centrale(**Input**).

O : dati da memoria centrale all'esterno (**Output**)

MS : memorizzazione permanente dei dati (**Memoria di Massa**)

MC : è l'area di lavoro che contiene dati e istruzioni da usare (**Memoria Centrale**).

CPU : usa quel che c'e' in memoria centrale per eseguire operazioni di
trasferimento e di calcolo (Central Processor Unit) .

L'informazione elementare prende il nome di **bit** (binary digit) e può assumere il valore zero od il valore uno. Dal punto di vista fisico tale informazione è realizzata in memoria con dispositivi a due stati (**FLIP-FLOP**) , es. stato spento 0, stato acceso 1.

Possiamo schematizzare ciò dicendo che :

il bit vale 0 se è **NO, FALSO, SPENTO, NON PASSA CORRENTE.**

Il bit vale 1 se è **SI, VERO, ACCESO, PASSA CORRENTE.**

La memoria è una struttura mono-dimensionale suddivisa in celle di dimensione fissa. Ogni cella può essere individuata univocamente mediante un' indirizzo (numerico) ed è costituita da una sequenza di N valori binari (valori 0 e 1).

Quindi abbiamo detto che un bit o meglio il circuito fisico che lo realizza è già sufficiente a rappresentare due informazioni numeriche **0** oppure **1**.

Tanti bit implicano tante informazioni distinte.

Esempio se abbiamo 2 bit possiamo rappresentare i numeri da 0 fino a 3 , non di più! ; con 2 bit rappresento 00, 01, 10, 11 rispettivamente: 0, 1, 2, 3. Il numero 4 con 2 bit non lo si può rappresentare.

Nel caso che la MC avesse una capacità di 3 bit potrei rappresentare 8 informazioni diverse ad esempio i numeri da 0 fino a 7.

000, 001, 010, 011, 100, 101, 110, 111 che corrispondono ai valori 0,1,2,3,4,5,6,7

Con 4 bit si rappresentano (2^4) informazioni diverse valori da 0 a 15.

Ad esempio tutti i numeri decimali che vanno da 0 fino a 15. In generale con n bit si rappresentano (2^n) informazioni diverse. Se le informazioni sono numeri, si riescono a rappresentare tutti i numeri compresi fra 0 e ($2^n - 1$) estremi inclusi.

Codifica binaria dei numeri naturali tra 0 e 31 con 5 bit

0	00000	8	01000	16	10000	24	11000
1	00001	9	01001	17	10001	25	11001
2	00010	10	01010	18	10010	26	11010
3	00011	11	01011	19	10011	27	11011
4	00100	12	01100	20	10100	28	11100
5	00101	13	01101	21	10101	29	11101
6	00110	14	01110	22	10110	30	11110
7	00111	15	01111	23	10111	31	11111

La codifica di ogni dato avviene con sequenze di bit.

In realtà la memoria centrale contiene centinaia di milioni di bit, questo implica che si possono rappresentare molte informazioni. Per comodità si indicano con il termine di :

- ✓ byte gruppi di 8 bit
- ✓ word gruppi di 16/32 bit.

Quindi abbiamo detto che la MC è immaginabile come un insieme di celle indirizzabili (ogni **cella** o **locazione** di memoria è distinta dalle altre da un numero detto **indirizzo**) in cui l'indirizzo (ADDRESS) è un numero progressivo.

Ad ogni cella corrisponde la coppia < indirizzo, valore >

cella n. 100	< indirizzo , valore >	"
" 101	"	"
" 102	"	"
" 103	"	"

Specificando un indirizzo la CPU può **accedere** alla cella corrispondente per leggere l'informazione contenuta o **memorizzare** una informazione (numerica) nella cella.

Accedere ad un dato (informazione) significa quindi :

- ✓ selezionare mediante l'indirizzo la cella in cui esso è memorizzato
- ✓ prelevarne il valore.

Il valore della cella è sempre una stringa binaria (sequenza di bit).

Per la memorizzazione di un dato, bisogna stabilire l'indirizzo della cella in cui si vuole introdurlo.

La MC è costituita da **RAM** (random access memory) memoria ad accesso diretto od uniforme, dato che il tempo per accedervi è costante, cioè non dipende dalla posizione della cella nella RAM.

Il tempo di accesso, definito nel periodo di tempo che passa tra quando si chiede una informazione e quando tale informazione viene resa effettivamente disponibile, nelle Ram dell'ultima generazione è inferiore ai 10 ns.

Anche le **ROM (read only memory)** fanno parte della MC, ma non vi si possono memorizzare nuovi valori; la RAM (detta anche memoria volatile) quando si spegne il computer si smemorizza, mentre la ROM conserva le informazioni ivi contenute.

Una breve parentesi : la definizione del concetto di algoritmo.

Un **algoritmo** è un procedimento risolutivo del problema, cioè un insieme di regole di calcolo che, eseguite ordinatamente, permettono di calcolare i risultati del problema a partire dalle informazioni a disposizione.

Perchè un insieme di istruzioni possa considerarsi un algoritmo deve rispettare le seguenti proprietà:

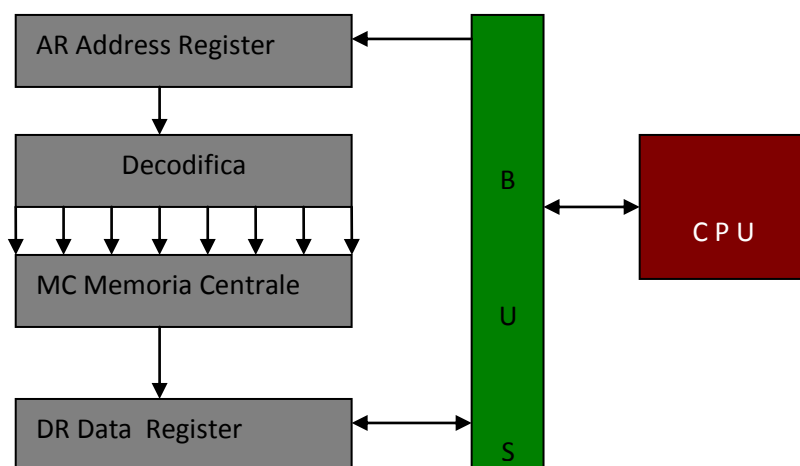
- **Non ambiguità** :
 - le istruzioni devono essere univocamente interpretate dall'esecutore dell'algoritmo (macchina o uomo).
- **Eseguibilità** :
 - l'esecutore deve essere in grado, con le risorse a disposizione, di eseguire ogni istruzione ed in un tempo finito.
- **Finitezza** :
 - l'esecuzione dell'algoritmo deve terminare in un tempo finito per ogni insieme di valori di ingresso.

Per delegare ad un calcolatore la soluzione di un problema, cioè l'esecuzione di una serie di calcoli si deve:

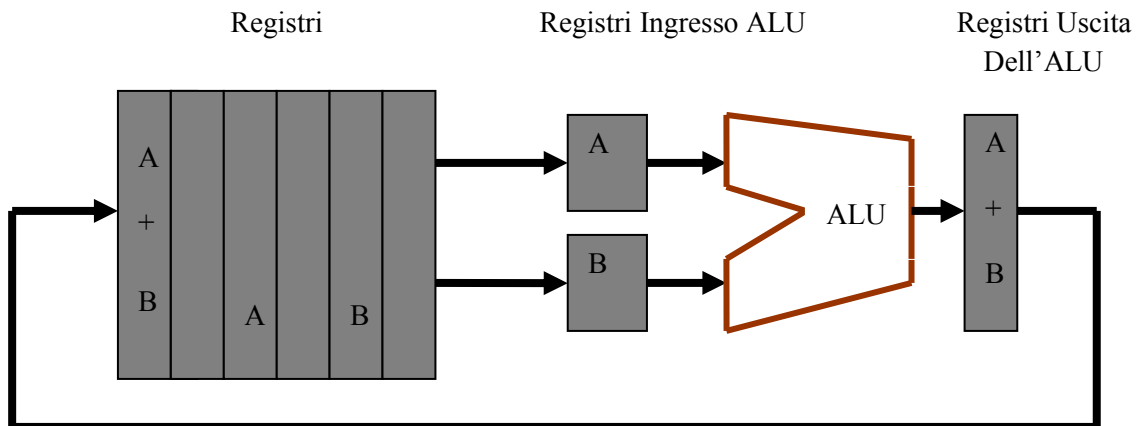
- individuare **almeno un algoritmo** adatto alla risoluzione del problema,
- individuare una **rappresentazione** dell'algoritmo, dalle informazioni a disposizione e di quelle utilizzate dall'algoritmo per mezzo di un linguaggio detto: **linguaggio di programmazione**.

Il programma è la rappresentazione / traduzione dell'algoritmo in un linguaggio di programmazione.

Schema dell' Unità di Memoria Centrale



Struttura del Data Path (percorso dei dati)



La comunicazione fra la memoria ed il BUS avviene attraverso i **Registri**, cioè aree di memorizzazione aventi tale compito. Precisamente attraverso il registro degli indirizzi **AR** (ADDRESS REGISTER) ed il registro dati **DR** (DATA REGISTER).

E' appunto attraverso questi due registri che avviene:

- ✓ il **prelievo** del contenuto di una cella di memoria (la lettura, che non distrugge il contenuto della cella)
- ✓ la **scrittura** in una cella (che cancella il contenuto precedente della cella).

Il registro AR e la memoria hanno una ovvia correlazione: dato che il registro è formato da 32 bit la memoria non può avere un numero di celle maggiore di (2^{32}) , perchè il massimo numero di indirizzi che si possono rappresentare con 32 bit è appunto (2^{32}) . Il blocco DECODIFICA. "Decodificatore di indirizzi", ha il compito di selezionare tra tutte le celle della memoria, quella cella che ha l'indirizzo contenuto nel registro AR.

Più è grande la MC, più grandi e complessi saranno i programmi e la quantità di dati che essa potrà contenere. Le Ram degli home computer di ultima generazione hanno dimensioni che vanno ben oltre il Gb.

Architettura dei calcolatori elettronici

Abbiamo visto che la CPU (processore) è il cervello del computer ossia è un insieme di circuiti integrati che gestiscono la Esecuzione di istruzioni macchina di **calcolo** e di **trasferimento** .

Con trasferimento intendiamo:

- ✓ ricevere un dato in MC da Input
- ✓ emettere un dato da MC verso Output
- ✓ copiare un dato da MS a MC.

Col termine calcolo intendiamo: accedere ad una cella di memoria e ad esempio sommare il suo contenuto con qualcosa. Quindi l'istruzione macchina è una istruzione eseguibile (attivazione di un circuito) dalla CPU. Un **programma eseguibile** coincide con una sequenza di istruzioni macchina. Ciascuna istruzione macchina è **CODIFICATA <---- > RAPPRESENTATA** in una cella. Es. se vogliamo caricare nel registro R1 il contenuto della cella di indirizzo 144 (144 che in binario si rappresenta con 10010000), allora "caricare" (**load**) è una operazione identificata da un certo codice operativo es. 0000, per cui nella cella della memoria centrale avremo questa istruzione:

0000	10010000
codice operativo	indirizzo operando
(OC Operative Code)	(OA Operand Adress)

La sequenza di istruzioni macchina, memorizzate in MC è accessibile dalla CPU che così può vedere ed eseguire le istruzioni.

Per la CPU possiamo pensare ad una divisione di comodo $CPU = CU + ALU + REGISTRI$ dove Unit Control (CU) , ha il compito di:

- ✓ prelevare dalla memoria le istruzioni di un programma
- ✓ interpretarle attraverso un organo di decodifica

- ✓ farle eseguire regolando le azioni delle varie unità
- ✓ del trasferimento dei dati .

Per queste azioni necessita di una funzione di temporizzazione il **clock** che coordini le varie azioni.

L'ALU si occupa del calcolo, infatti essa riceve dalla CU il comando da eseguire, insieme con l'operando o gli operandi, su cui l'operazione deve essere effettuata.

Per quanto riguarda i REGISTRI si può asserire che : la maggior parte degli elaboratori possiede:

- ✓ il **Program Counter** (contatore di programma **PC**), che contiene in ogni istante l'indirizzo della prossima istruzione che deve essere eseguita
- ✓ l'**Instruction Register** (registro istruzione **IR**) in cui è contenuta l'istruzione che deve essere interpretata e poi eseguita "istruzione corrente",
- ✓ gli **Accumulatori** (**ACC**) in cui sono contenuti gli operandi di una data istruzione ed al termine dell'esecuzione della istruzione contengono il risultato,
- ✓ il registro **Program Status Word** (parola di stato **PSW**) che fornisce con il suo contenuto particolari informazioni sull' ultima istruzione eseguita - ad esempio se il risultato dell'operazione aritmetica sia zero od il segno del risultato od il riporto del risultato ecc.

Quindi nel PC c'è l'indirizzo della prossima istruzione da eseguire, (cioè quella dopo, in memoria centrale, di quella copiata in IR)

ACC è memoria (poche celle) ausiliaria per operazioni.

PSW contiene info nell' ultima esecuzione (risultato= 0,pos.,neg.; riporto; overflow ecc).

Funzionamento dell' unità di elaborazione; Elaborazione delle istruzioni

Immaginiamo che un programma eseguibile cioè una sequenza di istruzioni macchina sia stata **caricata** in MC a partire da un certo indirizzo es. 100.

La CPU accede a queste istruzioni una alla volta e le esegue:

1. Fase di Fetch (Alimentazione) :

- l'istruzione che deve essere eseguita viene prelevata dalla cella di memoria 100 e trasferita nel registro PC
- il contenuto della cella di indirizzo PC viene trasferito nel registro istruzioni IR
- il Program Counter viene incrementato di 1 cioè $PC = PC + 1$; adesso dentro PC c'è l'indirizzo della prossima istruzione da eseguire .

2. Fase di Decodifica (Decode) :

- In base al contenuto dell'IR l'istruzione viene "decodificata" (che istruzione è) .

3. Fase di Execute (Esecuzione) :

- In base al codice operativo OC, si attiva il circuito della CPU corrispondente, che userà OA ed eseguirà l'istruzione.

4. Torna al punto 1.

BOOK IN PROGRESS

IL SISTEMA OPERATIVO

Competenze	Abilità	Conoscenze
Utilizzare, con autonomia e responsabilità, le reti e gli strumenti informatici nelle attività di studio, ricerca e approfondimento disciplinare.	Riconoscere le caratteristiche logico-funzionali di un computer e il ruolo strumentale svolto nei vari ambiti. Tipologie ed evoluzioni di un Sistema Operativo.	Struttura e funzioni di un sistema operativo Interfacce e periferiche di un computer
	Tipologie ed evoluzioni di un Sistema Operativo.	

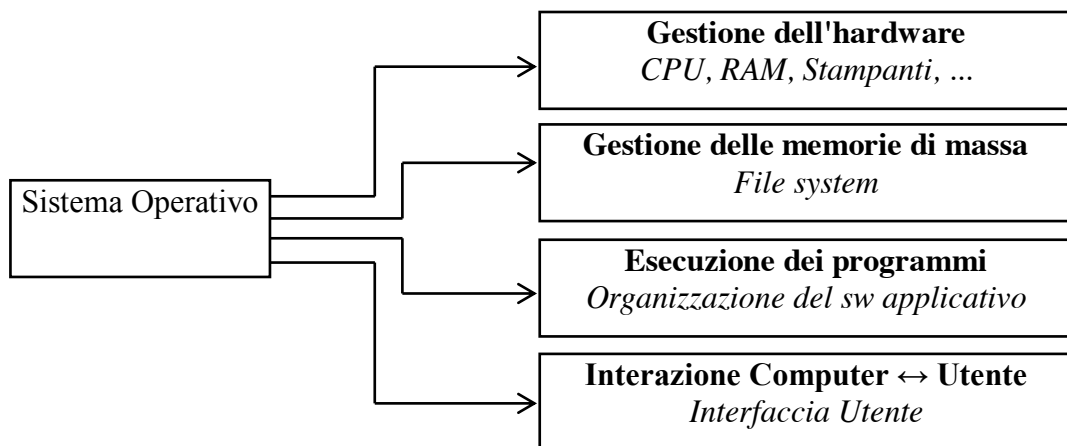
A cura del prof. Luca Peresson

INDICE DEGLI ARGOMENTI

I diversi sistemi operativi	3
La struttura di un sistema operativo	5
L'utilizzo dell'interfaccia grafica	10
Il mouse	12
Il Desktop Environment	15
La finestra	17
File system e pathname	19
Configurazione e manutenzione di un sistema operativo	25
Coesistenza di sistemi operativi: multiboot e virtualizzazione	30
Approfondimento: il concetto di <i>processo</i>	33

Il Sistema Operativo

Con il termine *Sistema Operativo* (o talvolta *software di base*) si indica quell'insieme di programmi destinato -genericamente- alla gestione del computer. Tra le principali funzioni da esso svolte troviamo la gestione dell'hardware, la memorizzazione dei dati nelle memorie di massa, l'esecuzione dei programmi applicativi e l'interazione fra il computer e l'utente.



I diversi sistemi operativi

Nel mercato mondiale esistono numerosi sistemi operativi che possono essere classificati secondo diversi criteri:

- per *famiglia*

seguendo questa classificazione possiamo individuare i sistemi della famiglia **Unix** e **Microsoft**. Alla prima famiglia appartengono, fra gli altri, i sistemi operativi di tipo *Linux* (nelle diverse "distribuzioni" quali *Slackware*, *Fedora*, *Ubuntu*, *Debian*, *Suse*, *Gentoo*, *Mandriva*,...), *BSD* (tra i quali è d'obbligo citare *FreeBSD*, *OpenBSD* e *DarwinOS*) e gli stessi sistemi *Unix*. Alla famiglia Microsoft appartengono i sistemi operativi *XENIX*, *MS-DOS*, *OS/2* e la serie di sistemi operativi di tipo *Windows* (tra i quali le recenti *XP*, *Vista*,

Server 2008, Mobile, 7, ...);

- per tipo di *licenza*

seguendo questa classificazione si individuano i cosiddetti sistemi *proprietary* e i sistemi *open source*;

- per *dispositivo*

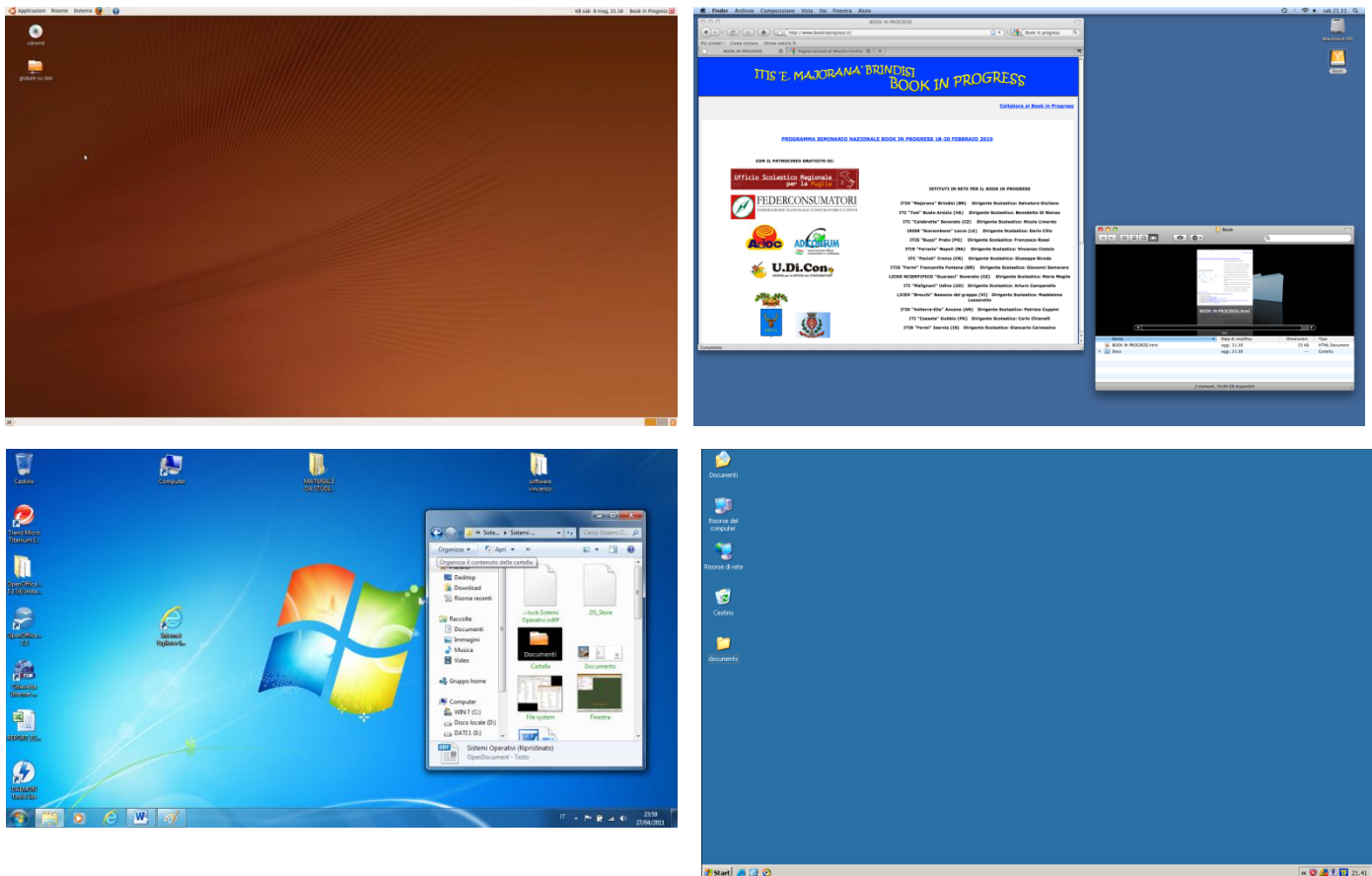
è una classificazione che tiene conto del tipo di elaboratore sul quale è destinato a funzionare il sistema operativo. In questo caso potremo individuare sistemi operativi orientati alla gestione di *smartphone* (computer integrati a dispositivi telefonici), di *lettori di eBook* (dispositivi portatili orientati alla lettura di documenti in formato elettronico), di *personal computer* (computer da tavolo o portatili) fino alla gestione di *mainframe* (detti anche *sistemi centrali*, gli elaboratori considerati di alta fascia e destinati alla gestione di applicazioni particolarmente delicate che necessitano di alta affidabilità);

- per *applicativo*;

è una classificazione che tiene conto dei programmi che verranno utilizzati dal sistema. In questo caso potremo individuare sistemi operativi in grado di eseguire particolari software applicativi orientati a specifici settori quali la *grafica*, l'*office automation*, l'*amministrazione/gestione* aziendale, il *controllo automatico* di processi produttivi, l'*intrattenimento*, ecc..

La struttura di un sistema operativo

Nell'immaginario legato all'uso del computer, un sistema operativo viene individuato dal tipo di gestione del monitor e del mouse, di quel programma che tecnicamente viene indicato con il termine *interfaccia grafica*. Siamo quindi abituati ad associare un dato sistema operativo a come esso si presenta graficamente al suo avvio.



Questa semplificazione è perfettamente giustificata dal fatto che l'innovazione rappresentata dall'uso del mouse e delle finestre ha permesso la straordinaria diffusione dei computer in questi ultimi decenni.

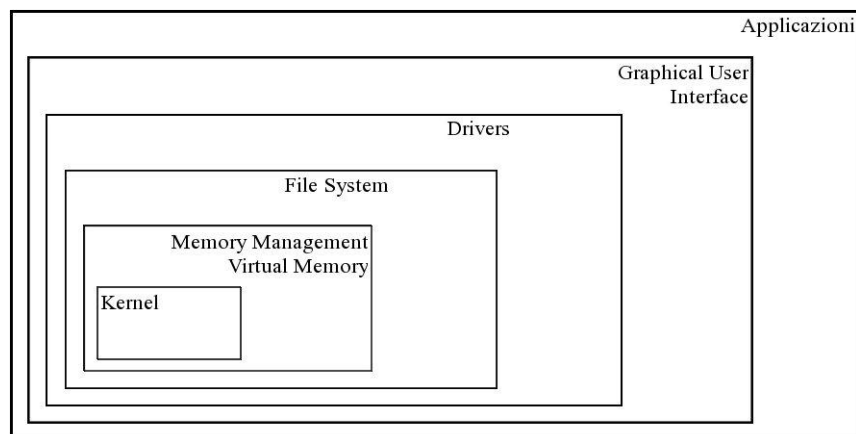
Poter “aprire” delle *finestre*, essere in grado di visualizzare il contenuto di una memoria di massa attraverso la sua rappresentazione in *icone* e *cartelle* ha semplificato l'organizzazione dei contenuti nei supporti di memorizzazione; poter avviare dei programmi attraverso un *doppio clic* del tasto di un mouse o selezionare aree di un documento attraverso l'azione di *trascinamento* ha consentito di velocizzare l'accesso e l'utilizzo dei programmi applicativi.

Tuttavia, il software di interfaccia grafica non rappresenta nemmeno la millesima parte dei moduli che vengono controllati e gestiti da un sistema operativo. È solo la componente del sistema operativo più “vicina” all'uomo, non l'unica e certamente non la più importante per il suo funzionamento.

Volendo schematizzare il sistema operativo nelle sue componenti essenziali possiamo individuare:

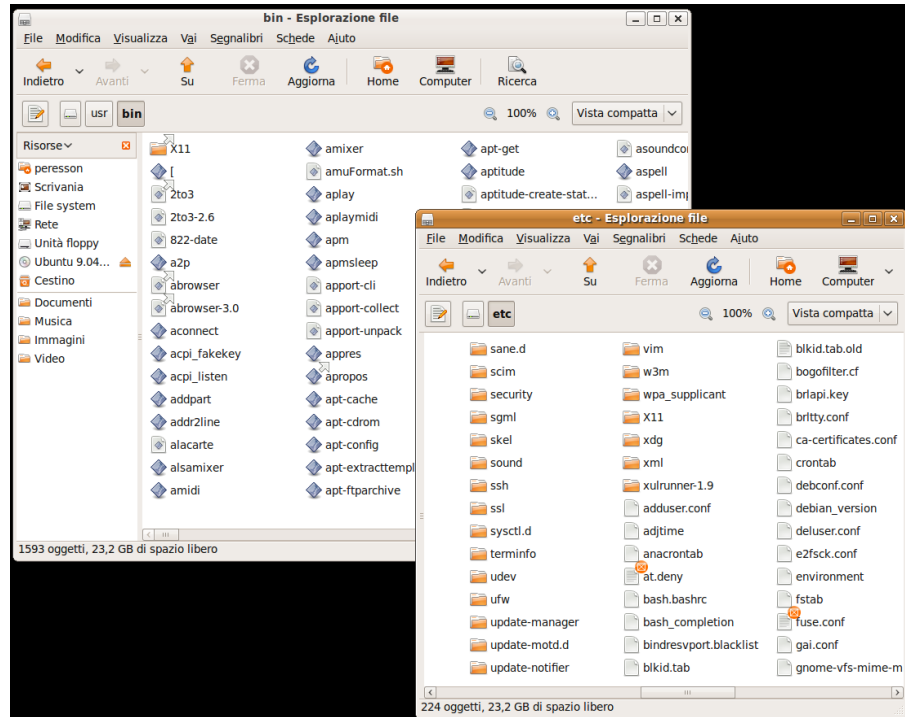
- il *kernel*, o *nucleo*

che è l'insieme dei programmi destinati a gestire l'accesso di tutti gli altri programmi all'hardware della macchina. L'efficienza di questo modulo determina la velocità con cui i programmi vengono eseguiti considerato che è questo modulo ad assegnare ai programmi l'accesso a risorse quali il processore e la memoria centrale;



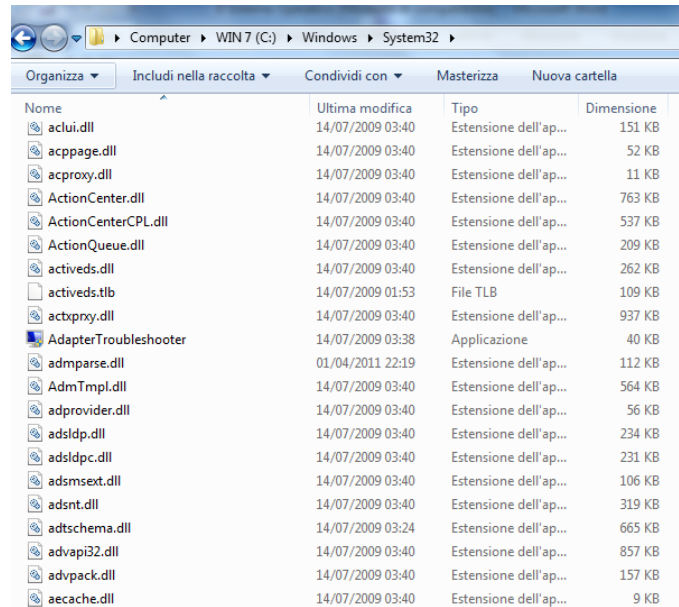
- il *file system*

che è l'insieme di regole che stabiliscono come viene identificato un *file* all'interno dei dispositivi di memorizzazione di un computer. Il concetto di *file system* (e, in conseguenza di ciò, di *file*) è uno dei concetti più importanti e delicati dell'informatica, basti pensare che dagli stessi dispositivi periferici (tastiera, stampante, ecc.) alle risorse di rete (sistemi remoti, pagine html, ...) viene associata una rappresentazione basata sul modello del file system;



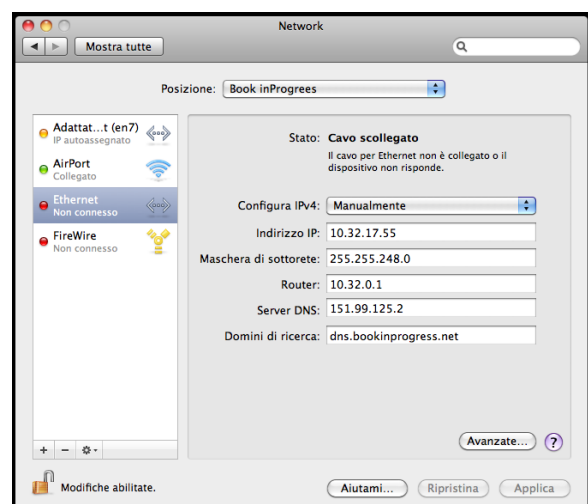
- il *memory management*,
il gestore della memoria è quella parte di sistema operativo destinato a gestire la *memoria centrale*. Lo scopo di questo modulo è quello di assegnare ad ogni processo una porzione di memoria centrale facendo attenzione a che il funzionamento di un dato programma non vada a occupare zone di memorie già utilizzate da altri programmi
- la *memoria virtuale*
è una complessa tecnica di gestione della memoria che riguarda l'estensione dello spazio disponibile nella RAM attraverso l'utilizzo di una parte dello spazio di memoria dell'hard disk. Considerata la significativa differenza di capacità e di velocità fra i due dispositivi il migliore bilanciamento fra gestione dello scambio dei dati e utilizzo delle risorse permette di “simulare” una memoria centrale significativamente maggiore di quella disponibile senza una sensibile diminuzione delle prestazioni della macchina;
- i *driver* o -più correttamente- i *device driver*
sono quei moduli software che consentono al sistema operativo di gestire un determinato dispositivo fisico attraverso l'uso delle interfacce standard.

L'utilizzo di questi moduli è di estrema importanza nel funzionamento di un elaboratore perché permette da un lato la semplificazione della gestione delle periferiche e dall'altro un notevole aumento della flessibilità nella configurazione e nell'aggiornamento della configurazione hardware del sistema;

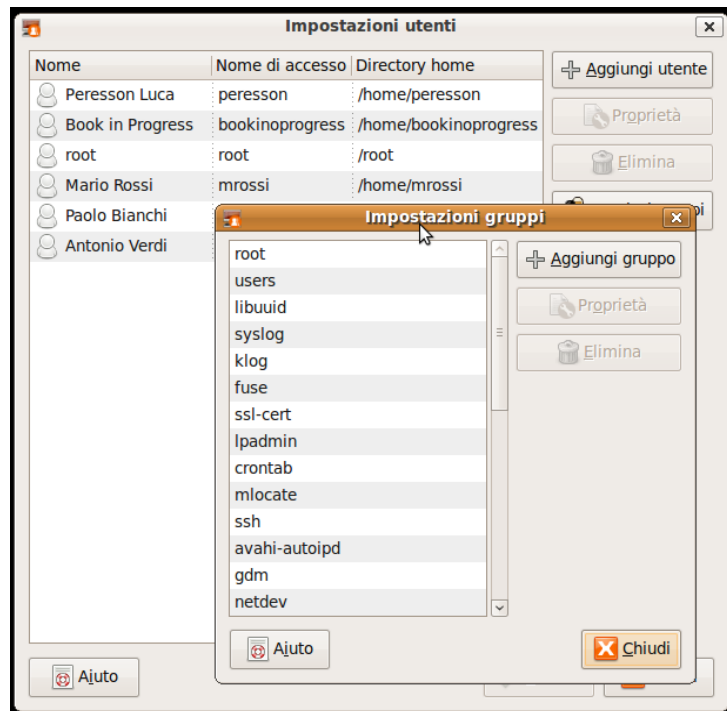


Nome	Ultima modifica	Tipo	Dimensione
aclui.dll	14/07/2009 03:40	Estensione dell'ap...	151 KB
acppage.dll	14/07/2009 03:40	Estensione dell'ap...	52 KB
acproxy.dll	14/07/2009 03:40	Estensione dell'ap...	11 KB
ActionCenter.dll	14/07/2009 03:40	Estensione dell'ap...	763 KB
ActionCenterCPL.dll	14/07/2009 03:40	Estensione dell'ap...	537 KB
ActionQueue.dll	14/07/2009 03:40	Estensione dell'ap...	209 KB
activeds.dll	14/07/2009 03:40	Estensione dell'ap...	262 KB
activeds.tlb	14/07/2009 01:53	File TLB	109 KB
actxprxy.dll	14/07/2009 03:40	Estensione dell'ap...	937 KB
AdapterTroubleshooter	14/07/2009 03:38	Applicazione	40 KB
admparse.dll	01/04/2011 22:19	Estensione dell'ap...	112 KB
AdmTmpl.dll	14/07/2009 03:40	Estensione dell'ap...	564 KB
adprovider.dll	14/07/2009 03:40	Estensione dell'ap...	56 KB
adslap.dll	14/07/2009 03:40	Estensione dell'ap...	234 KB
adslapc.dll	14/07/2009 03:40	Estensione dell'ap...	231 KB
adsmsex.dll	14/07/2009 03:40	Estensione dell'ap...	106 KB
adsnt.dll	14/07/2009 03:40	Estensione dell'ap...	319 KB
adtschema.dll	14/07/2009 03:24	Estensione dell'ap...	665 KB
advapi32.dll	14/07/2009 03:40	Estensione dell'ap...	857 KB
advpack.dll	14/07/2009 03:40	Estensione dell'ap...	157 KB
aecache.dll	14/07/2009 03:40	Estensione dell'ap...	9 KB

- il *multitasking*
concetto ormai acquisito da molte generazioni di elaboratori, è la capacità di un sistema di eseguire contemporaneamente diversi programmi assegnando ad ognuno di questi l'utilizzo delle stesse risorse (CPU, RAM, hard disk, monitor, ecc.);
- Il *gestore delle connessioni di rete*
conferisce al sistema la capacità di effettuare uno scambio di dati attraverso le infrastrutture di rete con altri sistemi. Tale funzione è legata all'utilizzo di diversi *protocolli di rete* ed alla capacità di supportare i programmi applicativi e i processi ad essi correlati;



- la *gestione di utenti* è la caratteristica di un sistema di creare, regi-strare e organizzare di-versi metodi di accesso che garantiscano una graduale scala di priorità e un controllato utilizzo delle proprie risorse. A questa caratteristica è fortemente legato il tema della sicurezza e della vulnerabilità del sistema ad attacchi esterni;



- l'*interfaccia grafica o GUI* (Graphical User Interface)

è quel software che consente all'utente di inviare dei comandi all'elaboratore facendo uso di dispositivi fisici (mouse e tastiera) che gestiscono oggetti grafici bidimensionali (puntatore, icone, finestre). In tal modo l'utente viene liberato dall'obbligo di utilizzare la sola tastiera per la scrittura e l'invio di comandi testuali (la più antica forma di utilizzo del sistema: la CUI, Character User Interface ossia interfaccia utente basata sul carattere).

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Versione 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Tutti i diritti riservati.

C:\Users\SALVATORE MADARO>dir
Il volume nell'unità C: è WIN 7
Numero di serie del volume: E369-8C80

Directory di C:\Users\SALVATORE MADARO
01/04/2011 20:40 <DIR> .
01/04/2011 20:40 <DIR> ..
11/04/2011 21:30 <DIR> Contacts
27/04/2011 20:10 <DIR> Desktop
11/04/2011 21:30 <DIR> Documents
27/04/2011 21:02 <DIR> Downloads
11/04/2011 21:30 <DIR> Favorites
11/04/2011 21:30 <DIR> Links
11/04/2011 21:30 <DIR> Music
11/04/2011 21:30 <DIR> Pictures
11/04/2011 21:30 <DIR> Saved Games
11/04/2011 21:30 <DIR> Searches
11/04/2011 21:30 <DIR> Videos
0 File 0 byte
13 Directory 34.847.920.128 byte disponibili

C:\Users\SALVATORE MADARO>*.doc
"*.doc" non è riconosciuto come comando interno o esterno,
un programma eseguibile o un file batch.

C:\Users\SALVATORE MADARO>time
Ora corrente: 23:38:27,72
Innettere nuova ora:

C:\Users\SALVATORE MADARO>ipconfig

Configurazione IP di Windows

Scheda LAN wireless Connessione rete wireless:
Suffisso DNS specifico per connessione: homenet.telecomitalia.it
Indirizzo IPv6 locale rispetto al collegamento . : fe80::742c:5e03:334d:5660%13
Indirizzo IPv4. . . . . : 192.168.1.238
Subnet mask . . . . . : 255.255.255.0
Gateway predefinito . . . . . : 192.168.1.1

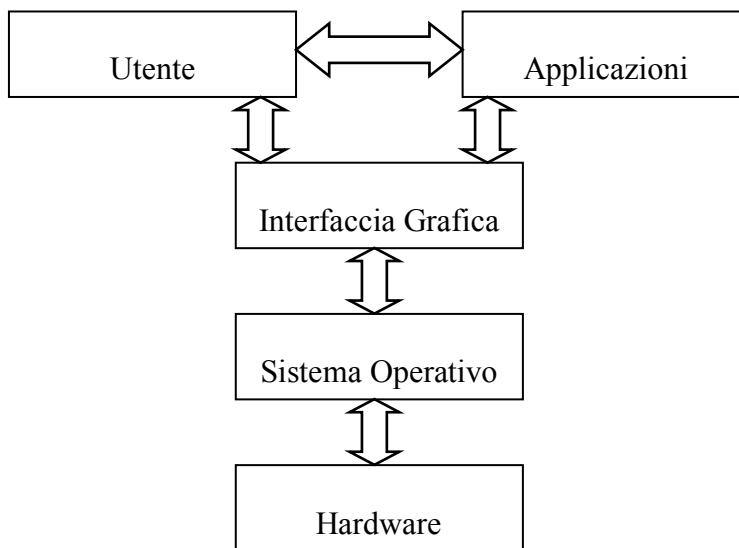
Scheda Ethernet Connessione alla rete locale (LAN):
Stato supporto. . . . . : Supporto disconnesso
Suffisso DNS specifico per connessione: homenet.telecomitalia.it

Scheda Tunnel Connessione alla rete locale (LAN)* 2:
Suffisso DNS specifico per connessione:
Indirizzo IPv6 . . . . . : 2001:0:4137:9e76:3458:194d:a011:ecd2
```

L'utilizzo dell'interfaccia grafica

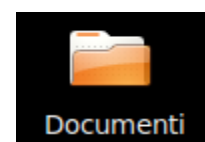
Il funzionamento dell'interfaccia grafica si deve principalmente all'invenzione di un piccolo dispositivo di puntamento chiamato *mouse*, che consente, attraverso il proprio spostamento, di muovere un *puntatore* (un'immagine, generalmente una freccia, talvolta indicata con il termine *cursore*) sullo schermo e -attraverso la pressione su di uno o più pulsanti- di eseguire particolari operazioni.

Questo semplice e rivoluzionario dispositivo viene ideato dall'inventore statunitense Douglas Carl Engelbart e realizzato come prototipo nel 1963. Solo dopo vent'anni, agli inizi degli anni '80 il dispositivo inizia a diffondersi e acquisisce notorietà nel 1984 con il computer *Apple Macintosh*, che presenta una semplice interfaccia grafica il cui funzionamento si basava su finestre e icone. Attualmente ogni sistema che preveda una frequente interazione di tipo *uomo-macchina* (tipicamente i sistemi installati su personal computer) rende disponibile uno o più tipi di interfaccia grafica.



Le più diffuse interfacce grafiche sono, come si è già accennato, la *Microsoft Windows*, la *Mac Os X* e la *X Windows System*. Ognuna di queste presenta uno, stilisticamente diverso, *ambiente desktop* che simula un piano di lavoro (definito *scrivania*) sul quale vengono disposti alcuni oggetti grafici:

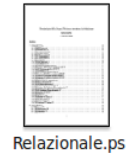
- la *cartella*
rappresentazione grafica della
“directory”, l'area di memoria destinata
all'archiviazione di documenti e



programmi;

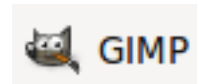
– i *documenti*

rappresentazione grafica dei “file di dati”, la struttura di dati destinata alla memorizzazione dei risultati di un'elaborazione;



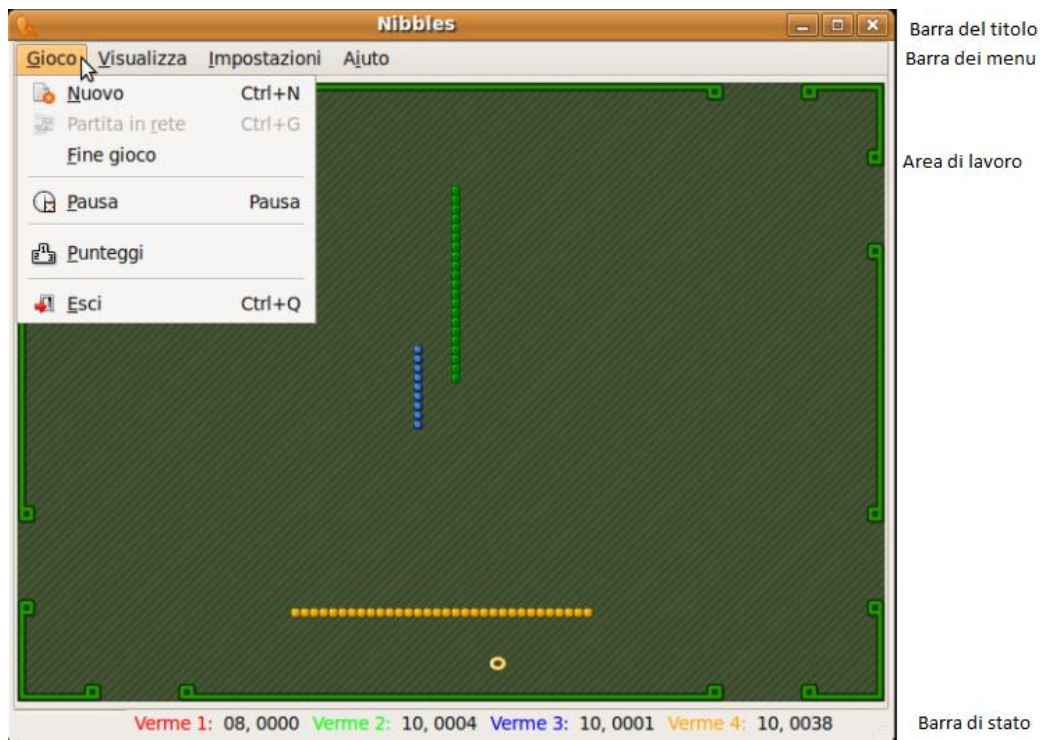
– i *programmi*

rappresentazione grafica dei “file eseguibili”, la struttura di dati destinata alla memorizzazione dei programmi;



– le *finestre*

rappresentazione grafica dei programmi in esecuzione. All'interno di questo elemento grafico si distinguono diversi altri oggetti fra i quali citiamo le *barre*, i *menu*, i *pulsanti*, ecc.;



Il mouse

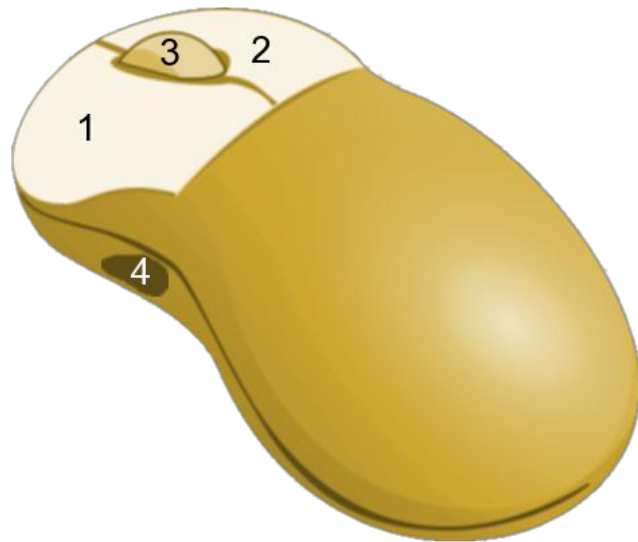
Il mouse è un dispositivo tipicamente composto da un lettore di movimento, da un certo numero di pulsanti e da una rotellina.

Il lettore di movimento può essere di natura ottica (un laser) o meccanica (una piccola sfera) . In entrambi i casi l'obiettivo del lettore di movimento è quello di spostare un indicatore (detto "puntatore") sullo schermo facendo in modo che tale movimento coincida con quello del dispositivo fisico.

La rotellina ha la funzione di attivare lo *scroll* ovvero lo scorrimento verticale (vero l'alto o verso il basso a seconda della rotazione impressa)

Il numero dei pulsanti (o tasti) di un mouse è comunemente tre anche se può variare da uno a cinque. Più specificamente si parla di tasto destro, tasto sinistro e tasto centrale. Nella figura la rotellina di scorrimento ha anche la funzione di tasto centrale.

L'utilizzo più comune di un mouse si basa su alcune semplici operazioni con i tasti. Tali operazioni sono standardizzate e praticamente identiche per ogni sistema operativo ed è piuttosto raro veder corrispondere ad ognuna di esse un comportamento diverso.



– il *clic*

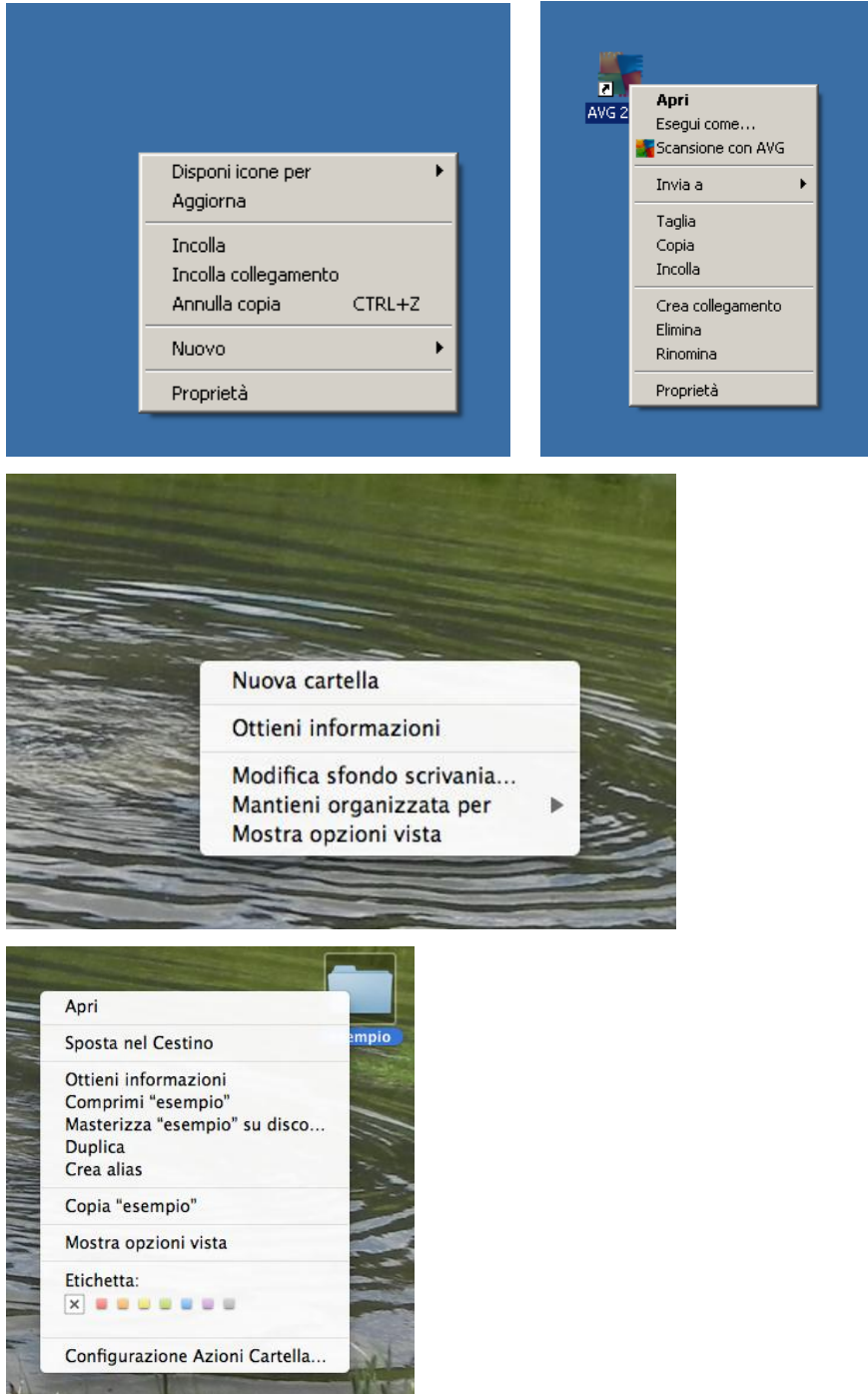
indica una pressione del pulsante seguita dall'immediato rilascio. Il clic con il tasto sinistro ha lo scopo di selezionare un **oggetto** e –a seconda del caso– può determinare diversi risultati quali *selezionare un'icona*, *aprire un menu*, *selezionare la voce di un menu*, *premere un pulsante* (in questo caso il

termine "pulsante" indica un oggetto grafico quali i pulsanti
, *attivare un link ipertestuale*,...

Calcola

Cancella

Il clic con il tasto destro su un oggetto grafico ha solitamente la funzione di far apparire il *menu contestuale* all'oggetto stesso. Per esempio il clic sul tasto destro effettuato su una porzione libera della scrivania aprirà un menu contestuale avente come opzioni voci quali “Modifica sfondo scrivania” o “Disponi icone per...”, la medesima operazione fatta su un'icona aprirà un menu contestuale avente come opzioni voci quali “Apri” o “Esegui come...” o ancora “Crea alias”.



Il clic con il tasto centrale non ha un significato predefinito e dipende dal

software applicativo.

- il *doppio clic*

indica una serie di due clic eseguiti a mouse fermo in successione abbastanza rapida. Il significato del doppio clic con il tasto sinistro su un'icona è –a seconda del caso– quello di *avviare un programma* (se l'icona è quella di un software applicativo o se è quella di un documento di tipo associato ad un dato programma) o quella di *visualizzare il contenuto di una directory*. Sebbene il doppio clic sia ovviamente possibile con ogni tasto del mouse il suo utilizzo riguarda il solo tasto sinistro.

- il *trascinamento*

indica il mantenimento della pressione del dito sul pulsante ed il contemporaneo spostamento del mouse. In sostanza viene *prolungato* il clic ed allo stesso tempo viene effettuato un movimento del mouse. Il significato di tale operazione è principalmente quello di *selezione di oggetti* anche se su alcuni software rappresenta l'attivazione di strumenti a *mano libera* (cancellazione, disegno, applicazione di effetti,...). Un'importante variante del trascinamento è l'azione detta *drag and drop* (letteralmente “trascina e fai cadere”) che ha come significato lo spostamento di alcuni oggetti grafici e l'attivazione di un'azione (*spostamento/copia* di uno o più file da una directory ad un'altra, *avvio* di un programma con relativa apertura di un documento,...)

- altre operazioni con i tasti

Vi sono diverse altre operazioni con i tasti del mouse, generalmente non consolidate o standardizzate. Fra quelle di maggior frequenza ricordiamo il *triplo clic*, il *doppio clic con trascinamento*, il contemporaneo clic dei tasti sinistro e destro.

- Le *mouse gestures*

Rappresentano combinazioni di movimenti del mouse e pressioni di tasti che vengono interpretate come particolari comandi. Sono tipicamente legate a specifici software (per esempio lo zoom applicato attraverso la combinazione del tasto “CTRL” e la contemporanea rotazione della rotellina o il movimento

di un oggetto tridimensionale attraverso il trascinamento e la pressione contemporanea dei due tasti sinistro e destro del mouse, ...). Le *mouse gestures* rivestono un particolare ruolo nella gestione di particolari software ed anche nel supporto verso persone con difficoltà nell'utilizzo della tastiera.

Il Desktop Environment

Con il termine *desktop environment* (in italiano “ambiente di scrivania”) o più semplicemente *desktop* (o *scrivania*) si indica quel software che utilizza l'interfaccia grafica di un sistema operativo creando un particolare ambiente che consente all'utente di utilizzare il computer attraverso l'uso di oggetti grafici quali *finestre* (che tratteremo in seguito), *barre* e *icone*. A parziale approfondimento di quanto già introdotto nelle precedenti pagine è opportuno fornire una spiegazione dei due più comuni oggetti presenti in una scrivania:

- la *barra delle applicazioni* (o *taskbar*), la *dockbar* e il *launcher*

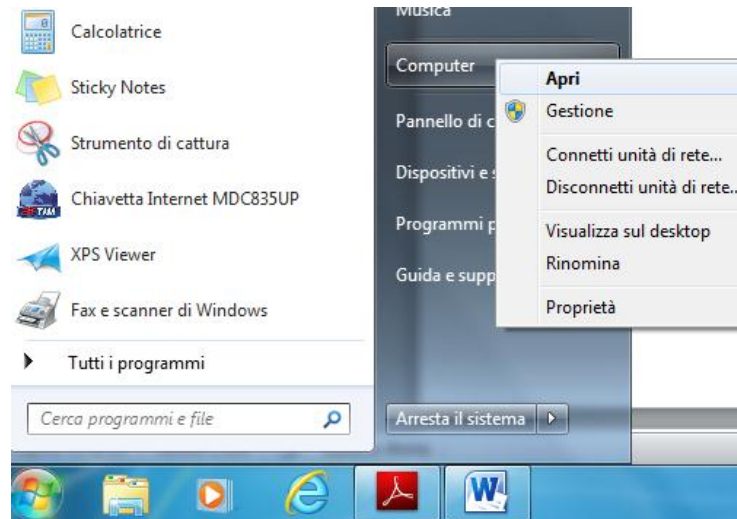
La barra delle applicazioni è una barra solitamente posizionata sul lato inferiore dello schermo (spesso riposizionabile a piacere dall'utente) che contiene uno più tasti per l'apertura di menu orientati all'elencazione ed all'avvio delle applicazioni installate. Diverse evoluzioni di natura grafica a questa componente sono state apportate a seconda dei sistemi operativi. Le più conosciute fra queste sono la dockbar del sistema operativo Mac OS X e il launcher del sistema Ubuntu nelle più recenti versioni



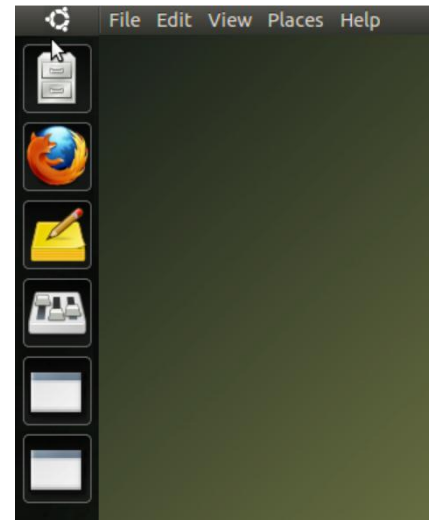
Barra delle applicazioni in ambiente Microsoft Windows XP



Dock bar in ambiente Macintosh OS X



Barra delle applicazioni in ambiente Microsoft Windows 7 con relativo menu principale



Launcher su Linux Ubuntu 11

– i *widget*

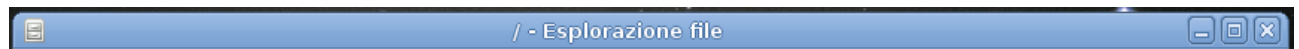
Una delle caratteristiche che ha avuto maggiore evoluzione nei desktop environment specialmente con la nascita dei sistemi operativi orientati alla *mobilità* (quei sistemi operativi orientati alla gestione di smartphone, tablet e netbook) sono i *widget* (contrazione di *window gadget*) tanto da creare una nuova tipologia di GUI detta *desktop widget*, una GUI specializzata, di dimensioni ridotte, volta a facilitare l'avvio di applicazioni di uso frequente (blocco note, calendario, calcolatrice, elenco contatti, ...). Una certa confusione sull'uso del termine è causata dal fatto che con la stessa parola vengono indicati sia i cosiddetti *GUI widget*, quegli elementi grafici che hanno lo scopo di semplificare la configurazione di un'applicazione (radio button, checkbox, menu a discesa, pulsanti, slider, barre di scorrimento, progress bar, ...) che i cosiddetti *application widget*. Questi ultimi sono applicativi –normalmente di dimensioni contenute– creati allo scopo di semplificare la gestione o di un particolare programma o di specifiche componenti del sistema. I *desktop widget*, widget orientati all'ambiente di desktop, arricchiscono gli strumenti di scrivania consentendo di accedere a informazioni organizzate senza la necessità di lanciare specifici software (dalle ultime notizie alle previsioni meteo, dalla rapida consultazione di un dizionario alla scrittura di una nota, dall'accesso all'agenda/calendario alla comunicazione via chat,...).

Considerata la notevole proliferazione di questi elementi (dovuta solo parzialmente alla semplificazione che essi portano nell'utilizzo del sistema) segnaliamo soltanto le due famiglie di widget più diffuse –i *mobile widget* e i *web widget*– sui quali rimandiamo alla curiosità dello studente un eventuale approfondimento.

La finestra

La finestra è, come abbiamo avuto modo di dire, la rappresentazione grafica di un programma in esecuzione. Normalmente di forma rettangolare, presenta un'*area di lavoro*, ove vengono presentati i risultati dell'elaborazione e delle *barre* contenenti oggetti necessari per l'utilizzo del programma stesso:

– la *barra del titolo*



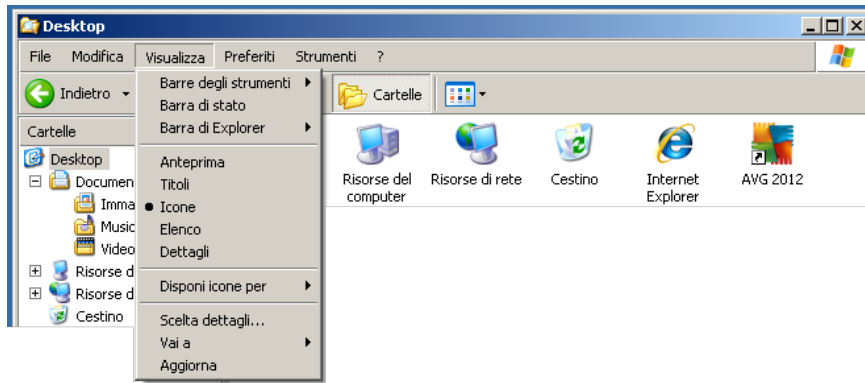
è una barra solitamente posizionata sul lato superiore della finestra. Contiene il *titolo*, un codice mnemonico formato solitamente dal nome del file e del programma utilizzato, ed alcuni *pulsanti*:



1. il pulsante di *riduzione a icona*, che provoca la scomparsa grafica della finestra dalla scrivania senza la terminazione dei processi ad essa associati (la finestra potrà essere ripristinata con un clic sull'opportuno pulsante posto nella *task bar*);
2. il pulsante di *chiusura*, che termina i processi associati alla gestione della finestra;
3. il pulsante di *ingrandimento/ripristino* che consente una rapida espansione della finestra alla massima dimensione possibile (per una finestra le dimensioni saranno quelle del desktop, per una sottofinestra quelle della finestra principale).

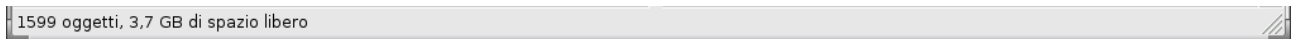
– la *barra dei menu*

è una barra tipicamente posizionata al di sotto della barra del titolo. Contiene una serie di elementi, solitamente *titoli di menu* e/o piccoli elementi grafici, che una volta selezionati portano all'apertura di un menu a tendina contenente le opzioni selezionabili (le operazioni eseguibili)



– la barra di stato

è una barra normalmente posizionata alla base della finestra. Viene solitamente utilizzata per visualizzare delle informazioni aggiuntive che variano a seconda del programma utilizzato: nel caso di un elaboratore di testo conterrà il numero di pagina, riga e colonna corrente, nel caso di un editor di immagini le coordinate di posizionamento, nel caso di un software per la navigazione nel file system il numero di file presenti in una data directory, la dimensione di un dato file, lo spazio occupato e disponibile in una data memoria di massa, ...



File system e pathname

Abbandoniamo l'aspetto più *user oriented* del sistema operativo per affrontare, nelle prossime pagine, alcune tematiche di estrema importanza nello studio delle funzioni di un sistema operativo. Come abbiamo avuto modo di osservare il sistema operativo deve gestire tutte le risorse hardware ed allo stesso tempo occuparsi di eseguire le indicazioni impartite dall'utente. Se la necessità di corrispondere al meglio alle esigenze di uso comporta un accurato studio dell'interfaccia verso l'utente, la vera sfida è quella di sfruttare al meglio le risorse della macchina. L'utente è il *dispositivo più lento* fra quelli che si interfacciano all'elaboratore, il tempo di reazione di un essere umano ad uno stimolo è –nell'ipotesi più veloce– di qualche centesimo di secondo (un tempo di reazione inferiore ai 10/100 di secondo ai blocchi di partenza in una gara di atletica comporta una “falsa partenza”); il sistema operativo deve invece gestire dispositivi che reagiscono in tempi decisamente più contenuti: il processore si sincronizza su una frequenza di clock superiore a 1 Gigahertz (significa che il *tempo di reazione* di una CPU è inferiore ad un milionesimo di secondo).

Per gestire al meglio le risorse hardware, il sistema operativo deve operare a diversi livelli, uno di questi, sicuramente tra i più importanti, riguarda l'organizzazione e la rappresentazione logica delle risorse di un computer. Questo compito viene realizzato, tra i diversi metodi, attraverso l'uso combinato del *file system* (sistema per l'organizzazione dei file) e del *pathname* (letteralmente *nome del percorso*, il sistema standard per l'individuazione di un file o di una risorsa) .

Il file system può essere sinteticamente descritto come quel insieme di regole che consente di organizzare i *dati* all'interno di un computer. Questa definizione ha spesso portato a pensare al file system come qualcosa relativo esclusivamente alle memorie di massa (hard disk, usb memory, DVD, ...); la realtà è un po' diversa (forse un po' meno “facile” ma, ci auguriamo, un po' più interessante). Il sistema operativo è, come abbiamo visto nelle prime pagine, il primo livello logico (software) che “avvolge” la macchina fisica (hardware) e si pone tra l'aspetto più puramente astratto (le applicazioni orientate all'utente, i documenti da questi creati e/o manipolati, le immagini salvate in un server remoto...) e quello più evidentemente fisico (l'hard disk che memorizzerà un brano musicale, la rete locale a cui è collegato il computer, la stampante che produrrà la versione cartacea di un documento, lo

schermo che visualizzerà una fotografia,..). Di conseguenza ogni dispositivo che partecipi all'elaborazione di dati dovrà avere una sua rappresentazione logica ed ogni rappresentazione dovrà essere organizzata in modo da consentire lo scambio di dati più efficiente possibile: il monitor e la stampante saranno quindi, agli "occhi del sistema operativo", della stessa natura della videoregistrazione di una vacanza o del foglio di elettronico per la soluzione di un'equazione di secondo grado.

Il file system ed il pathname sono il perno di questa importante strategia: il sistema di regole che organizzano la più efficace rappresentazione delle risorse all'interno di un computer (e, quando necessario, di una rete di computer).

– Il *file*

Il concetto di *file* è uno dei più interessanti concetti dell'informatica. L'immagine più utilizzata per rappresentarlo è quella di un *archivio* o più generalmente quella di un *contenitore di dati*¹. A prescindere dalla metafora preferita, un file può essere ragionevolmente visto come un *insieme logico* (non fisico!) *di dati strutturati* (ovvero organizzati secondo regole ben definite) a cui viene associato un *nome* e, talvolta, un'*estensione*. Sarà a questa definizione che il presente capitolo farà riferimento. Ogni informazione complessa presente in un computer viene memorizzata attraverso un file: ogni oggetto che venga elaborato o registrato digitalmente (un testo, un'immagine, un programma, un disegno o un brano musicale) viene rappresentato attraverso file².

– Il *file system* di una memoria di massa

Quando un hard disk viene connesso ad un computer si presenta la necessità di una fase preparatoria al suo utilizzo: l'hard disk deve essere *predisposto* a ricevere e memorizzare i dati. A questo scopo si effettuano tre operazioni:

1. la scelta del file system;
2. l'eventuale suddivisione logica dello spazio disponibile in diverse *partizioni*;
3. la *formattazione* (la preparazione dello spazio disponibile alla registrazione dei file).

La scelta del file system è di estrema importanza poiché ogni sistema operativo è

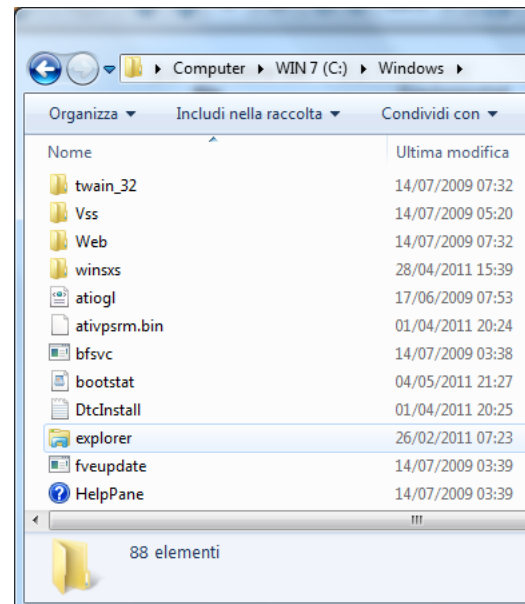
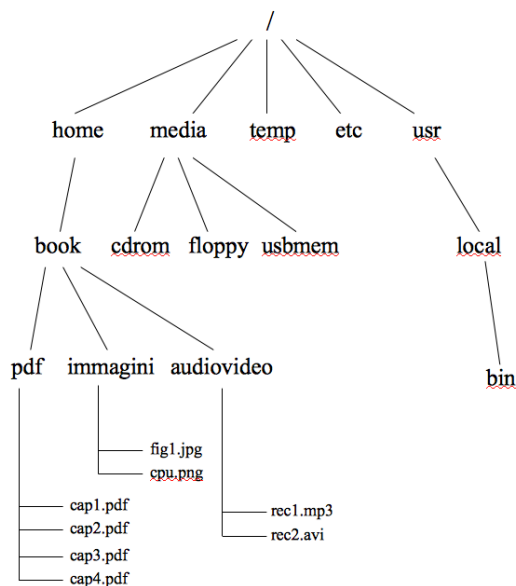
¹ Esistono altre formalizzazioni che utilizzano immagini quali *tubo* o *flusso* di dati organizzati.

² A seconda dei sistemi operativi, il concetto di file si può estendere in maniera "imbarazzante": per i sistemi operativi basati su Unix la stampante, il monitor, la tastiera, il mouse ed ogni altro dispositivo fisico hanno una propria rappresentazione in forma di file.

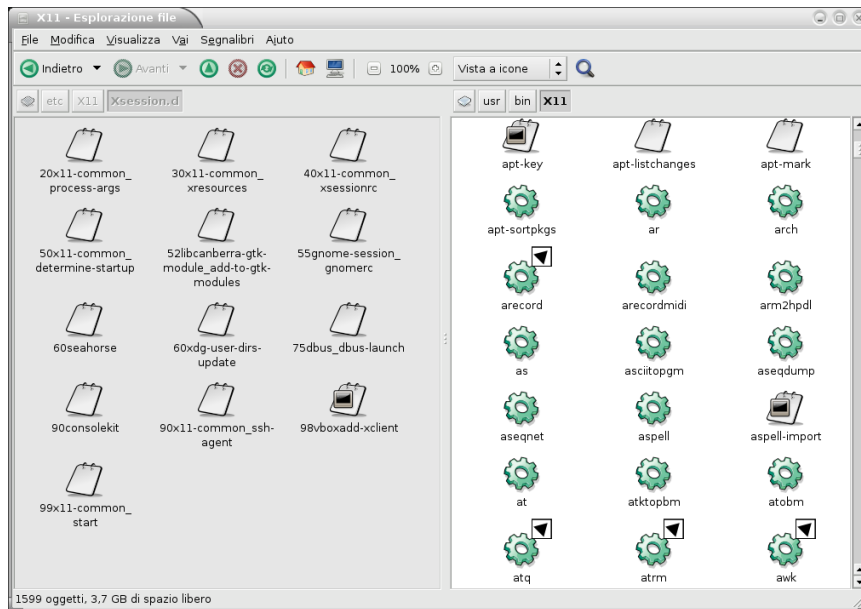
ottimizzato per utilizzare solo alcuni tipi di file system di disco: i sistemi Microsoft organizzano le memorie di massa secondo i file system di tipo NTFS, ExFAT e FAT32, i sistemi Macintosh HFS e HFS+, i sistemi Linux Ext3, Ext4, ecc.. Ogni sistema operativo viene studiato per organizzare i dati secondo determinate regole e l'applicazione di queste regole fornisce le garanzie per il raggiungimento della più alta affidabilità e delle migliori prestazioni possibili.

– Il *pathname*

Come precedentemente affermato, attraverso il *pathname*, è possibile individuare un generico file all'interno di una memoria di massa. All'interno delle memorie di massa si possono individuare due elementi (solo astrattamente) diversi: i *file* e le *directory*³. Attraverso questi elementi ogni documento e programma viene archiviato in modo organizzato seguendo *percorsi* di directory e sottodirectory che permettono ad ogni informazione di essere memorizzata secondo precise regole.



³ Dal punto di vista tecnico la *directory* non è altro che un particolare tipo di file contenente l'elenco dei riferimenti ai file ad esso riferiti.



A seconda del sistema operativo le regole per la scrittura di questo percorso possono cambiare:

1. nei sistemi operativi della famiglia Microsoft ogni percorso interno ad una memoria di massa locale inizia con la sigla del disco (composta da una lettera seguita dal simbolo “due punti”) seguita da un *backslash* e continua con la sequenza dei nomi di ogni directory, separato ognuno dall'altro con un *backslash*, fino all'ultimo elemento. Per esempio, con il pathname

C:\Documents\Proprietario\Documenti\miodoc.odt

si dice che il file *miodoc.odt* è contenuto nella directory *Documenti*, a sua volta contenuta nella directory *Proprietario*, a sua volta contenuta nella directory *Documents* che è una sottodirectory della *radice* (la directory principale) del *disco C*.

Nei sistemi operativi Microsoft, quindi, ad ogni disco è associato un indipendente albero delle directory

2. nel caso dei sistemi operativi di tipo Unix ogni percorso ha origine dalla directory principale (chiamata *root* e indicata con il simbolo *slash*) a partire dalla quale si dipana ogni possibile pathname. In questo caso quindi l'intero sistema dei dati è racchiuso in un singolo *albero delle directory*: ogni memoria di massa, hard disk, CD o altro, viene vista semplicemente come una sottodirectory. Il pathname in un sistema di tipo Unix segue regole simili a quanto visto in precedenza: il simbolo iniziale (e separatore) sarà lo slash. Per indicare, ad esempio, il file *rec2.avi* si dovrà scrivere il seguente

pathname:

/home/book/audiovideo/rec2.avi

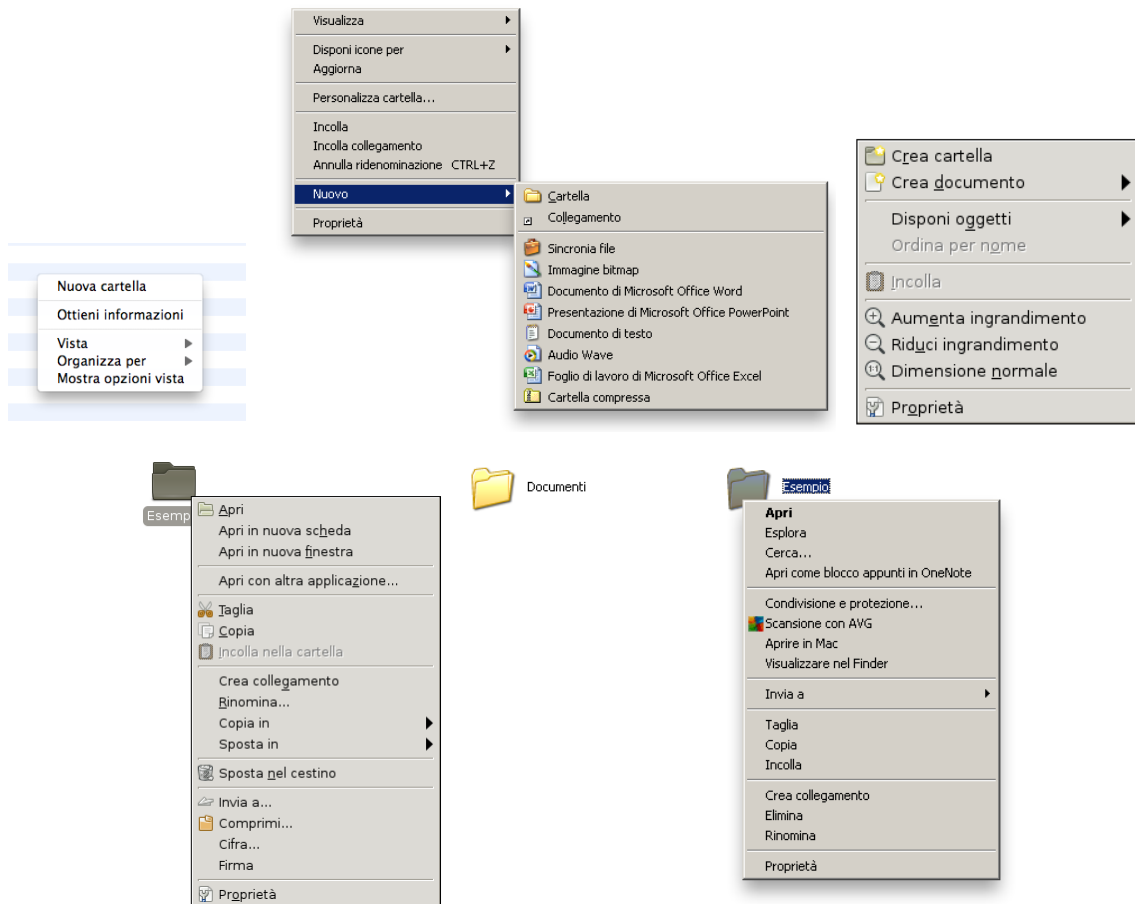
– Le *azioni* su un file system

L'organizzazione dei file all'interno di una memoria di massa richiede la possibilità di collocare i file all'interno di directory a seconda di un'organizzazione che può essere definita da diversi fattori che vanno dalla configurazione del software di installazione di un'applicazione alla necessità di collocare i propri documenti in archivi suddivisi e collocati secondo criteri definiti dall'utente. A tal fine esistono diverse azioni le più comuni delle quali sono:

1. *creare* una directory;
2. *copiare* uno o più file;
3. *cancellare* uno o più file;
4. *spostare* uno o più file;
5. *rinominare* un file.

Ognuna delle azioni può essere eseguita attraverso movimenti del mouse o attraverso istruzioni in interfaccia carattere. La copia di file è solitamente legata alla creazione di archivi di backup o alla memorizzazione dei file su supporti trasportabili mentre lo spostamento (operazione che si compone delle due azioni di copia in una diversa locazione e successiva cancellazione dell'originale) è spesso legato alla necessità di posizionare il file nella directory (nel percorso) più corretto in relazione ai criteri di archiviazione adottati. Le immagini a seguire mostrano le azioni di *creazione* di directory e le azioni di copia, sposta, cancellazione e cambiamento del nome attraverso mouse e successivamente le istruzioni di cp (copia), rm (cancellazione) e mv (spostamento) in linea di comando Linux.

Capitolo D: IL Sistema Operativo



```
root@cit:~# cp /etc/X11/XvMCConfig /root/XvMCConfig.old
root@cit:~# rm /root/XvMCConfig.old
root@cit:~# mv /home/laura/Docs/carta.doc /root/carta.doc
```

Configurazione e manutenzione di un sistema operativo

Volendo rappresentare la configurazione di un sistema operativo su un computer come un percorso è abbastanza naturale affermare che il primo “passo” sia rappresentato dalla fase di *installazione* del sistema operativo stesso nelle memorie di massa dell’elaboratore. Per poter avviare tale fase è necessario verificare che ogni componente hardware del computer sia gestibile (tecnicamente si utilizza il termine *supportato*) dal sistema operativo candidato all’installazione. L’importanza di tale verifica è evidente: essendo compito del sistema operativo quello di gestire ogni dispositivo presente nel computer è essenziale verificare l’adeguatezza dell’hardware alle specifiche tecniche del sistema operativo pena un funzionamento lento o problematico (nel migliore dei casi) o semplicemente l’impossibilità di *avviare* il computer. Per esempio:

- se l’obiettivo è quello di installare un sistema Linux su un “vecchio” computer con un processore Pentium III dovrà essere esclusa, tra i possibili candidati, la distribuzione *Debian 6.0 “Squeeze”* a 64 bit perché l’architettura hardware (il Pentium III è una CPU a 32 bit) non è compatibile con le risorse richieste dal sistema operativo;
- se invece la scelta fosse quella di installare la distribuzione *Debian 6.0 “Squeeze”* a 32 bit su un elaboratore dotato di processore Intel® Core™ i5 (a 64 bit) sarebbe possibile raggiungere l’obiettivo (poiché l’architettura hardware potrebbe riconoscere le istruzioni rivolte ad una versione meno recente) anche se il prezzo da pagare sarebbe quello di un inadeguato sfruttamento delle risorse del sistema e quindi un’inferiore velocità nell’esecuzione delle azioni richieste.

Una volta effettuati i necessari controlli si avvia la procedura di installazione vera e propria che generalmente procede attraverso una prima fase di *riconoscimento automatico* delle risorse presenti nel sistema fisico:

- tipo e quantità di memoria centrale;
- modello di CPU;
- tipologia di bus e presenza di slot (PCI, AGP, ATA, SATA, ...)
- tipo di scheda video, caratteristiche e compatibilità;

- memorie di massa;
- numero e tipologia delle porte seriali;
- connessioni alla rete;
- presenza di alcune periferiche essenziali (tastiera, mouse, touch screen,...)
- ...

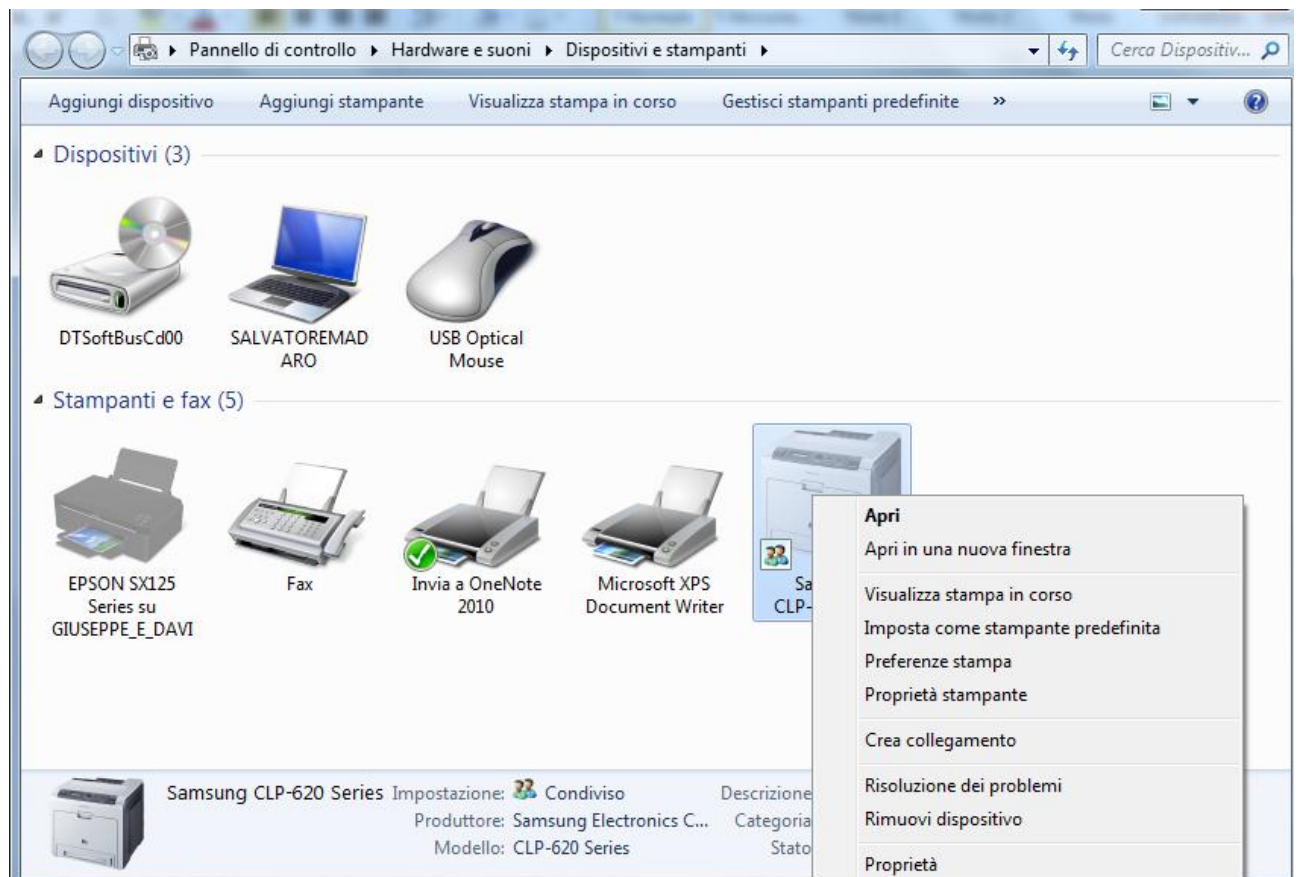
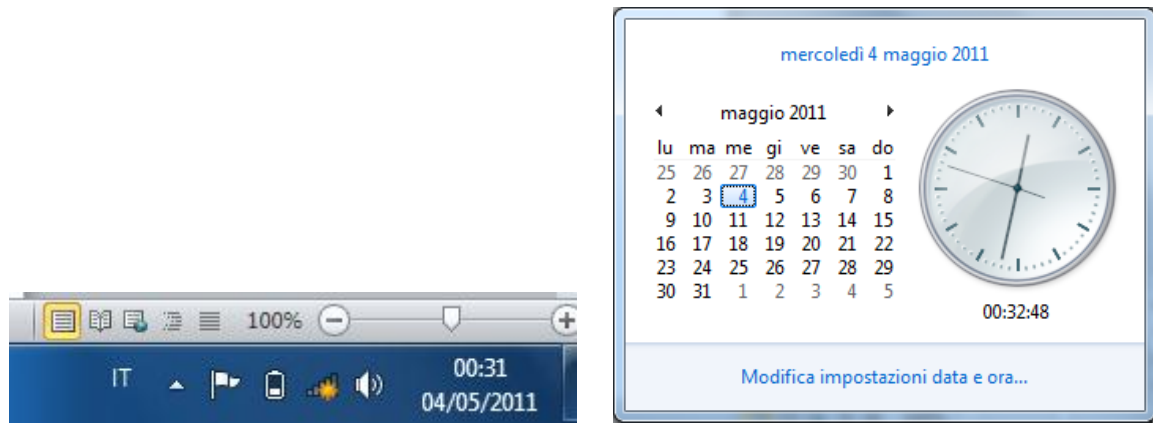
L'installazione procede con la preparazione del disco principale e la successiva installazione del kernel del sistema e dei device driver.

Successivamente vengono installati gli strumenti per l'utilizzo del sistema (i cosiddetti *software di utilità*), le interfacce utente e un certo numero di semplici software applicativi.

Le fasi finali dell'installazione riguardano:

- la *configurazione* dei dispositivi periferici (tipologia di tastiera, lingua e localizzazione, orologio di sistema, scheda/e di rete, stampante, videocamera, scheda bluetooth, ...);
- la configurazione dei moduli necessari all'aggiornamento del sistema (repository, antivirus,...);
- l'installazione di software proprietari orientati alla gestione di specifici dispositivi (un caso abbastanza frequente riguarda la scheda video o periferiche quali stampante o scanner).

È bene sottolineare che la differenza fra un'installazione di un sistema operativo eseguita con competenza ed una stessa installazione eseguita con superficialità può portare nel migliore dei casi a differenze di prestazione di diversi ordini di grandezza nell'esecuzione di un medesimo compito (il computer potrebbe risultare fino a *centinaia* di volte più lento, potrebbe risultare impossibile –o molto difficile– visualizzare o elaborare dati complessi come immagini, rendering tridimensionali, collegamenti in videoconferenza, filmati,...), mentre –nel peggiore dei casi– a blocchi del sistema, o a improvvise interruzioni del suo funzionamento (causati da eventi male controllati quali aumenti di temperatura, perdita di dati, difficoltà di sincronismo,...) o –infine– alla cancellazione parziale/totale del contenuto di alcune memorie di massa.



A conclusione delle fasi di installazione di un sistema operativo e di configurazione dei dispositivi periferici seguono:

1. la configurazione dell'accesso al sistema;
2. l'installazione e la configurazione dei software di protezione;
3. la predisposizione per la protezione dei dati contenuti.

– L'accesso al sistema

Ogni sistema operativo prevede delle *politiche* (in gergo tecnico "policy") per la

gestione dell'accesso. In generale è sempre possibile:

- creare utenti;
- fornire/privare gli utenti dei privilegi di amministratore;
- creare gruppi di utenti;
- creare aree di dati (o singoli documenti) condivise fra più utenti.

È buona pratica evitare assolutamente di usare un account di tipo amministrativo per operazioni “generalì” (che cioè non siano legate ad azioni che richiedono l'uso dei privilegi di amministratore); in modo particolare è assolutamente sconsigliato l'accesso come amministratore per utilizzare software di posta elettronica, web browser e in generale software orientati alla rete. È buona prassi *creare sempre almeno due utenze* (amministratore e utente generico) avendo cura di utilizzare l'utenza generica come la propria utenza.

La ragione di questa regola è che ogni programma (più correttamente dovremmo dire ogni “processo”) lanciato in esecuzione agisce all'interno del sistema ereditando i privilegi dell'utenza che lo ha avviato: se un programma lanciato da un utente generico cercasse di cancellare o modificare dati non propri l'azione fallirebbe per la banale assenza dei privilegi, se la medesima azione venisse tentata da un programma con i privilegi da amministratore avrebbe un sicuro “successo”.

– I *malware*

Il termine *virus informatico* è entrato da anni nel lessico generale. Capita molto spesso sentire dire «il computer non mi funziona più bene... avrà preso un virus!»⁴ ma la realtà è che il cosiddetto “virus” non è che una delle molte forme dell'aspetto meno nobile dell'informatica, quello relativo alle applicazioni create allo scopo di danneggiare un sistema informatico: i cosiddetti *malware*. Con questo termine si indica un variegato insieme di software fra i quali evidenziamo:

- i *virus*;
- i *worm*;
- gli *spyware*;

⁴ È bene sottolineare come il linguaggio tecnico spesso prenda a prestito termini e metafore proprie del linguaggio comune. È normale quindi utilizzare frasi che partono da “uccidere un processo” o “ibernare un'applicazione” fino a metafore quali “navigare nella rete” o addirittura “entrare in una cartella”; il caso dei “virus” è forse l'esempio più straordinario di metafora del mondo reale. Per la necessità di sintesi non potremo trattare la questione dei virus informatici in modo approfondito e invitiamo quindi lo studente interessato a “difendere” in modo efficace il proprio sistema a leggere le molte pagine presenti nei siti specializzati nella produzione di software antivirus.

- i *cavalli di Troia* (in inglese *trojan horse*);
- i *keylogger*,

e la lista potrebbe continuare a lungo con altre tipologie (per esempio i cosiddetti *backdoor* o i *virus polimorfi*) ma per gli scopi di questo capitolo non ha senso uno sterile elenco di dati.

A cosa serve un malware? Tra gli scopi principali di un malware si trova:

- la *sottrazione dei dati personali* presenti nel computer;
- l'*utilizzo del sistema* (all'insaputa del proprietario) per fini illeciti;
- la *distruzione dei dati* al fine di causare malfunzionamenti o perdita permanente di informazioni.

Per limitare la possibilità di incorrere nei danni derivanti dall'*infezione* di un virus è necessario:

- evitare di usare in modo improprio l'account di amministratore;
- mantenere un *costante aggiornamento* del sistema;
- mantenersi *informati*;
- evitare di utilizzare file provenienti da altri sistemi senza averli precedentemente controllati;
- configurare il proprio *firewall* ed evitare scambi di dati incontrollati con la rete;
- installare (e aggiornare frequentemente) un software antivirus;
- scegliere –se possibile– un sistema operativo *resistente* ai malware.

Per quanto possa essere efficace un malware è molto difficile che un computer si “prenda un virus” da solo... è molto più probabile che sia un nostro comportamento a causare l'infezione: il peggior virus informatico è un amministratore di sistema incapace.

- La *protezione dei dati*

All'interno di ogni sistema informativo, oltre ai documenti realizzati dall'utente, sono presenti dati personali quali:

- account di posta elettronica;
- password connesse a diversi servizi (servizi VOIP, messaggistica istantanea,...);
- archivi di dati (i messaggi di posta inviata e ricevuta, bookmark di siti internet);

- rubriche (indirizzi di posta elettronica, numeri di telefono, ID di conoscenti,...).

Maggiore è l'utilizzo del sistema maggiore sarà la mole di dati importanti e riservati contenuti in un computer. Evitare la perdita di questi dati è il compito dei *software di backup*, applicativi composti da due componenti disgiunte: la prima orientata al *salvataggio* dei dati (denominata –appunto– di “backup”), e la seconda orientata al ripristino della condizione precedente al salvataggio (detta di “restore”).

Esistono diverse tipologie di backup⁵:

- immagine del disco (disk image): consiste nella copia completa del contenuto di un disco, bit a bit, in un unico file;
- backup completo (full backup): serve per salvare tutti i dati presenti in una o più partizioni e/o memorie di massa;
- backup differenziale: esegue il salvataggio delle sole differenze rispetto al backup completo precedente;
- backup incrementale: simile al precedente ma il salvataggio riguarda le differenze fra l'ultimo backup (sia questo completo o differenziale);
- ...

È importante sottolineare che la gran parte dei software di backup sono strettamente legati al sistema operativo utilizzato: non è garantito poter eseguire il ripristino di una configurazione salvata su un sistema diverso da quello originale.

Altre modalità di salvataggio e protezione dei dati, generalmente orientata alla *trasportabilità* dei dati (legata spesso all'uso di usb memory) consistono nell'utilizzo di algoritmi di *compressione* e *crittografia*⁶.

⁵ Cfr. it.wikipedia.org/wiki/Bakup

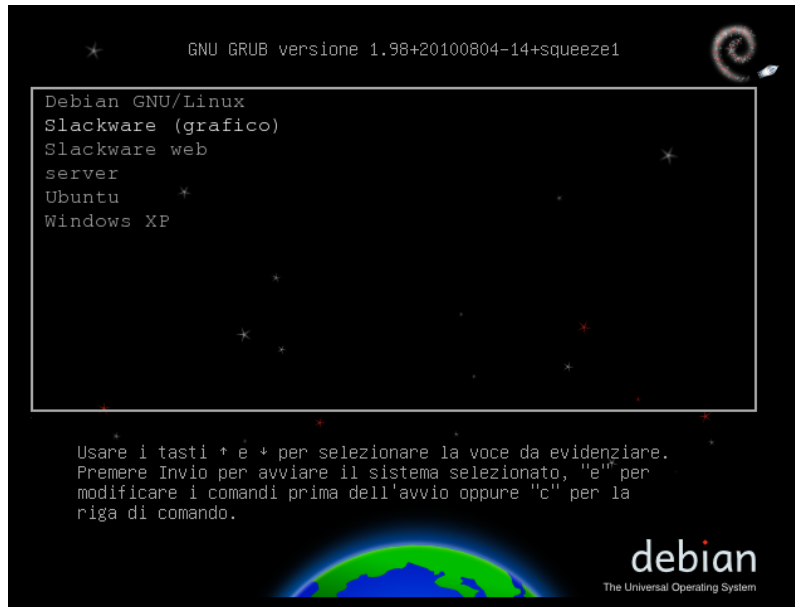
⁶ Si trovano diversi software per la cifratura e il salvataggio dei dati su memoria USB quali, ad esempio, il software open source utilizzabile sia su sistemi Microsoft che Macintosh e Linux *TrueCrypt*. Non essendo questo argomento collegato strettamente al contenuto del presente capitolo si rimanda allo studente un eventuale approfondimento personale.

Coesistenza di sistemi operativi: multiboot e virtualizzazione

Per quanto detto nelle pagine precedenti potrebbe risultare scontata una rigida relazione “un sistema operativo per un computer”, in pratica una relazione di unicità che vieti l’installazione di due o più sistemi operativi su uno stesso computer. La considerazione potrebbe sembrare assolutamente ragionevole soprattutto in considerazione del fatto che il sistema operativo ha tra i suoi scopi principali la gestione delle risorse hardware di un elaboratore (e quindi verrebbe da domandarsi a cosa mai potrebbe servire installare due diversi sistemi operativi su uno stesso hardware). Invece una convinzione di tale natura è del tutto priva di senso. In primo luogo ogni sistema operativo ha le proprie peculiarità e quindi non tutti i sistemi operativi sono equivalenti: esistono sistemi operativi studiati appositamente per ospitare processi *server* (condivisione di dati, fornitura di servizi, condivisione di risorse hardware,...), esistono sistemi operativi progettati per la gestione delle problematiche dell’utente (ambienti desktop evoluti, facilità di utilizzo,...) e infine esistono software sviluppati per un solo specifico sistema operativo. In secondo luogo l’esistenza di diverse tipologie di sistemi operativi comporta una competizione e una naturale ricerca di soluzioni volte al miglioramento del funzionamento. In altre parole conoscere e saper utilizzare diversi sistemi operativi consente di saper utilizzare più efficacemente e con maggior competenza ogni strumento informatico, che sia uno smartphone basato su “Android”, che sia un tablet basato su Mac OS o che sia un personal computer o un portatile basato su Linux, Microsoft o Free BSD.

- il *multiboot*

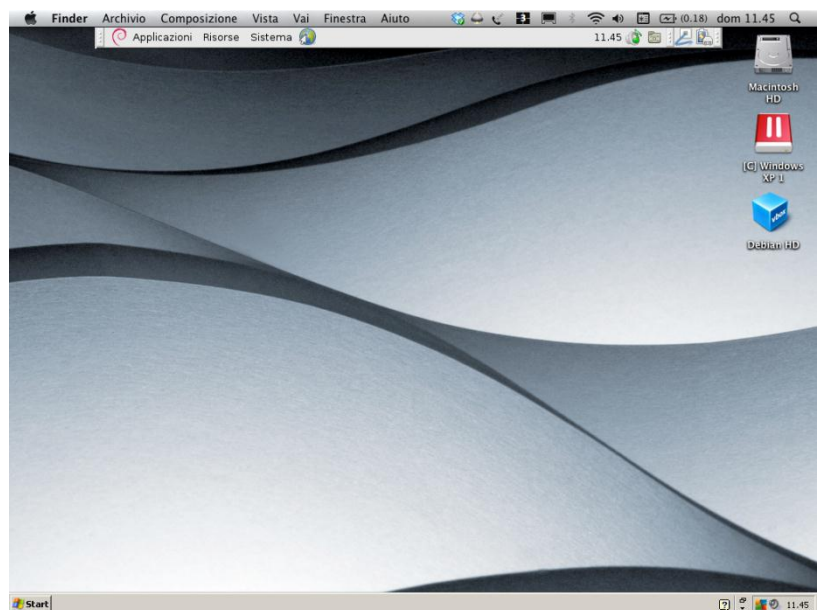
è una tecnica che consente all’utente di scegliere –durante la fase di avvio del computer– con quale sistema operativo utilizzare la macchina. Normalmente si tratta di un software che si avvia prima della fase di installazione del sistema operativo (la cosiddetta fase di *boot*) e presenta un menu (in versione grafica o testuale) che consente la scelta del sistema operativo da avviare fra quelli installati;



– la *virtualizzazione*

È quella tecnica che consente la coesistenza contemporanea di più sistemi operativi su una stessa macchina. Solitamente si realizza avviando un sistema operativo “di base” e lanciando in esecuzione un software applicativo volto alla virtualizzazione di un elaboratore fisico (sul quale vengono installati e avviati i sistemi operativi prescelti). A differenza della precedente tecnica, la virtualizzazione richiede un utilizzo piuttosto importante di risorse, specialmente

memoria centrale e processore, in quanto i sistemi operativi *concorrono* nello sfruttamento della CPU essendo simultaneamente in esecuzione. Nella figura si può vedere una configurazione che gestisce



simultaneamente un sistema MAC OS, un sistema Microsoft ed un sistema Linux nella medesima macchina.

Approfondimento: il concetto di *processo*

La letteratura e la necessità didattica consiglia di osservare un sistema operativo come un ambiente statico, composto da diversi moduli che vengono attivati a richiesta o a seguito dell'esecuzione di alcuni programmi applicativi. Questo punto di vista è abbastanza semplicistico e non spiega alcuni cruciali aspetti del funzionamento di un elaboratore. Per esempio:

- come mai quando viene inserito un dispositivo USB si attiva una procedura di riconoscimento?
- come fa un antivirus a rilevare un tentativo di intrusione?
- com'è possibile lanciare contemporaneamente in esecuzione decine programmi se il mio computer ha solo una CPU?
- ...

La risposta a queste e a molte altre domande di questo tipo può essere data solo se si comprende il concetto di *processo* e di *accesso allo sfruttamento di risorse in regime di concorrenza*. Un elaboratore in funzione presenta al suo interno migliaia di entità attive (dette appunto *processi*) contemporaneamente occupate a gestire le risorse fisiche e logiche (la memoria, la scheda video, il mouse, le connessioni di rete, la tastiera, la stampante, la CPU, le interfacce seriali, i documenti aperti, le utenze, le richieste di accesso al sistema, ...) per garantire il corretto (evitare *crash* del sistema) ed equilibrato (evitare rallentamenti o blocchi dell'elaborazione) funzionamento del computer. La gestione di queste migliaia di entità è un elemento chiave nella progettazione di un sistema operativo.

Ma cos'è un processo?

Un processo può essere definito come un'*istanza* di un *programma in esecuzione*.

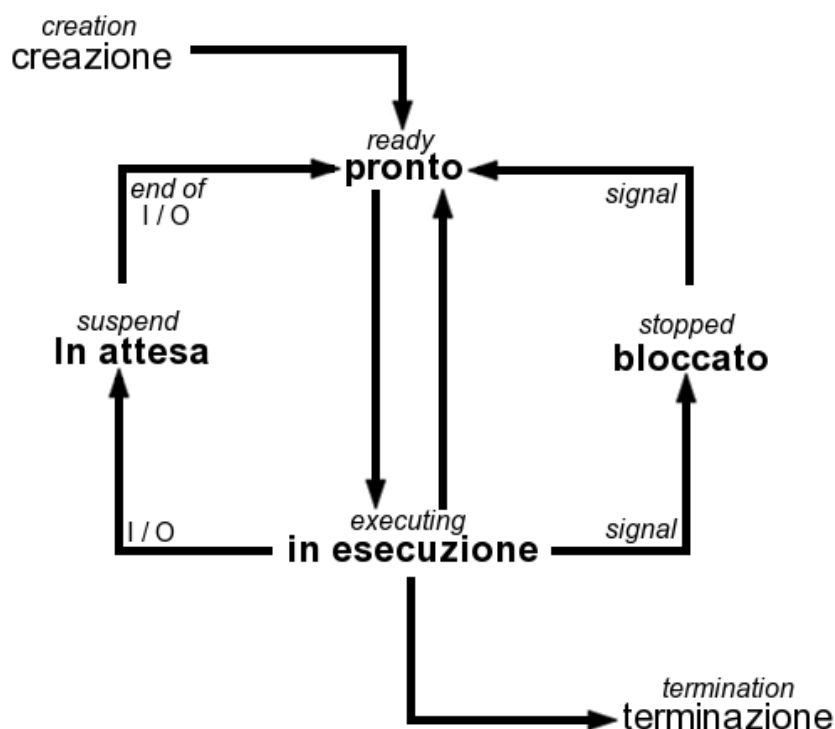
La definizione non è molto semplice, cerchiamo di darne una spiegazione comprensibile senza incorrere in un'eccessiva banalizzazione: un programma eseguibile è un'entità statica (un file) che il sistema può lanciare in esecuzione anche diverse volte contemporaneamente (per esempio un sistema può lanciare in stampa diversi documenti contemporaneamente, un web browser può effettuare chiamate contemporaneamente a diversi server e un web server può ricevere allo stesso istante centinaia di richieste da diversi web browser), ogni esecuzione di un programma ne rappresenta un'*istanza* e, come affermato in precedenza, *ogni istanza di un programma in esecuzione è un processo*. Un processo è quindi un'entità assolutamente *logica* (software) e *dinamica*

(riguarda, e in qualche modo concorre a definire, il concetto stesso di *esecuzione*).

L'operazione che coincide con l'inizio dell'attività di un processo viene chiamata *creazione* mentre la fine dell'attività di un processo viene chiamata *terminazione*. A seconda della disponibilità delle risorse di cui ha bisogno un processo può trovarsi in tre diversi *stati*:

- pronto;
- in attesa;
- bloccato;
- in esecuzione.

Il passaggio da uno stato all'altro può essere rappresentato con il seguente diagramma:



La transizione fra uno stato e l'altro è determinata:

- dalla *disponibilità/indisponibilità della risorsa* (transizione da *ready* a *executing* e viceversa);
- da una *chiamata* di I/O (da *executing* a *suspend* a *ready*);
- da un *segnale* che provoca l'interruzione dell'esecuzione del processo (da *executing* a *stopped* a *ready*).

BOOK IN PROGRESS

Le reti strutturate e la rete Internet

A cura dei proff. Massimo Mancino
e Carmelo Russo

Premessa

Questo *book* è stato pensato e realizzato per *introdurre* i molteplici aspetti delle moderne reti di calcolatori agli allievi che iniziano il loro percorso nella scuola media superiore. Suddiviso in due moduli, “*Le reti strutturate*” e “*La rete Internet*”, propone diversi contenuti: dai dispositivi di rete più comuni ai servizi che oggi Internet offre, al fine di far acquisire nelle nuove tecnologie legate all'informazione le *competenze* richieste al termine del biennio.

Conoscenze

- 1) Funzioni e caratteristiche della rete Internet e della posta elettronica;
- 2) Normativa sulla privacy e sul diritto d'autore;
- 3) Forme di comunicazione commerciale e di pubblicità.

Abilità

- 1) Utilizzare gli strumenti informatici nelle applicazioni d'interesse;
- 2) Utilizzare la rete Internet per ricercare fonti e dati;
- 3) Utilizzare la rete per attività di comunicazione personale;
- 4) Riconoscere i limiti e i rischi dell'uso della rete, con particolare riferimento alla tutela della privacy;
- 5) Individuare i principali strumenti di gestione per la diffusione e commercializzazione di un prodotto industriale o artigianale.

Indice dei contenuti

I) Parte prima - Le reti strutturate

1. Introduzione
2. Vantaggi e svantaggi
3. Classificazioni
4. Dispositivi di rete
5. Le reti locali
6. Esercizio 1

II) Parte seconda - La rete Internet

1. Introduzione
2. Cenni sulla struttura
3. Il modello Client-Server
4. Il modello TCP/IP
5. Funzionalità e servizi
6. La posta elettronica
7. Gli ipertesti e il Web
8. Il commercio elettronico
9. Cenni sulla sicurezza, le normative sulla privacy e il diritto d'autore
10. Esercizio 2

III) Parte terza - Approfondimenti ed appendici

1. Approfondimento A: I motori di ricerca
2. Approfondimento B: Classificazione degli indirizzi IP
3. Appendice A: Elenco protocolli

Parte prima – Le reti strutturate

Introduzione

Più computer connessi tra di loro e opportunamente configurati permettono di aggiungere importanti vantaggi all'utilizzo di un singolo calcolatore, come lo scambio veloce di dati, l'utilizzo di risorse condivise, la possibilità di gestire copie di sicurezza su computer diversi. Quando ogni dispositivo collegato alla rete (computer, stampante, modem, ecc.) ha un compito specifico e riesce a scambiare informazioni, allora si parla di *rete strutturata*. L'esempio più classico di rete strutturata è la rete Internet. Un computer connesso ad una rete viene anche chiamato *host* (ospite).

Vantaggi e svantaggi

Numerosi sono i *vantaggi* che una rete strutturata può offrire, ad esempio:

- la condivisione di una o più risorse (un file, una stampante, ...);
- l'utilizzo condiviso di un servizio grazie ad un server (posta elettronica, pagine web, ...);
- lo scambio di informazioni, agevolando la comunicazione tra persone che sono lontane.

Mentre tra gli *svantaggi* abbiamo:

- la manutenzione di una rete è un'attività complessa e costosa, richiede spesso l'intervento di personale specializzato;
- problemi di sicurezza e di privacy dei dati sensibili.

Le classificazioni

Le reti strutturate possono essere classificate secondo vari criteri.

In base alla *dimensione*:

- le LAN (*Local Area Network*) sono reti solitamente contenute all'interno di un edificio, come i computer di un laboratorio;
- le MAN (*Metropolitan Area Network*) sono reti più estese delle LAN, ma solitamente contenute in un'area urbana come una città;
- le WAN (*Wide Area Network*) sono reti geografiche molto estese, possono coprire una intera nazione, un continente o l'intero pianeta.

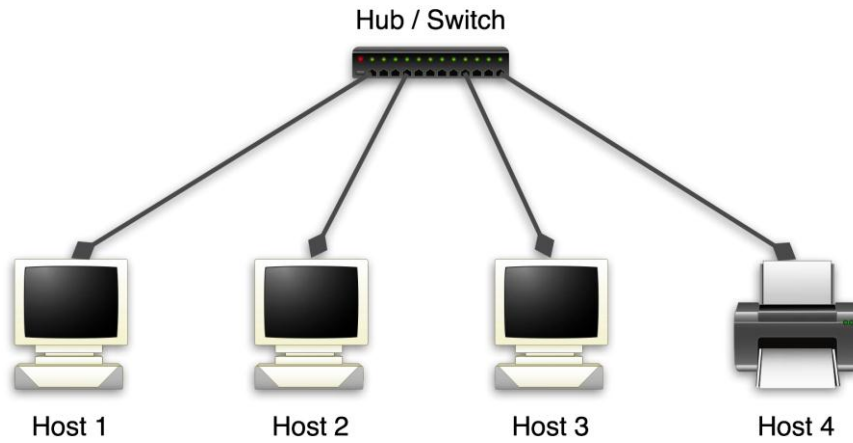


Figura 1 - Rete LAN

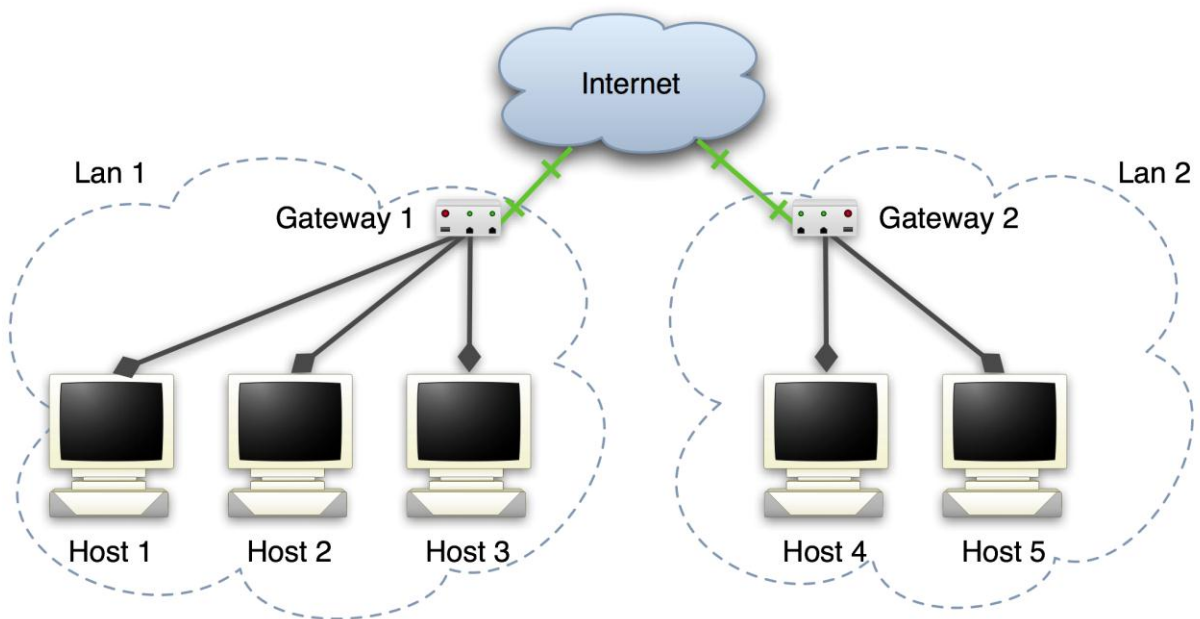


Figura 2 - Rete WAN

In base alla *forma* (topologia):

- le reti a *bus*;
- le reti a *stella*;
- le reti ad *anello*.

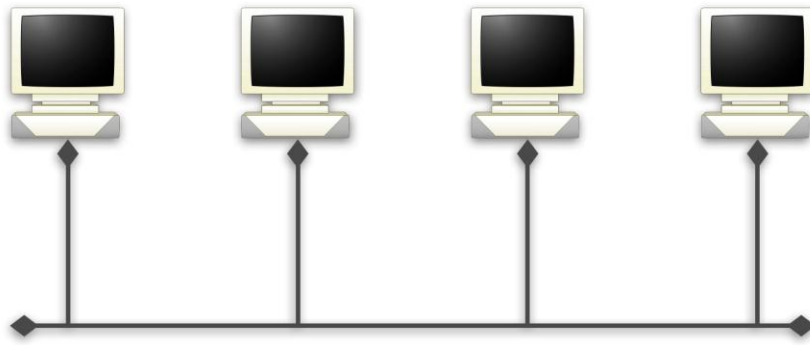


Figura 3 - Rete a bus

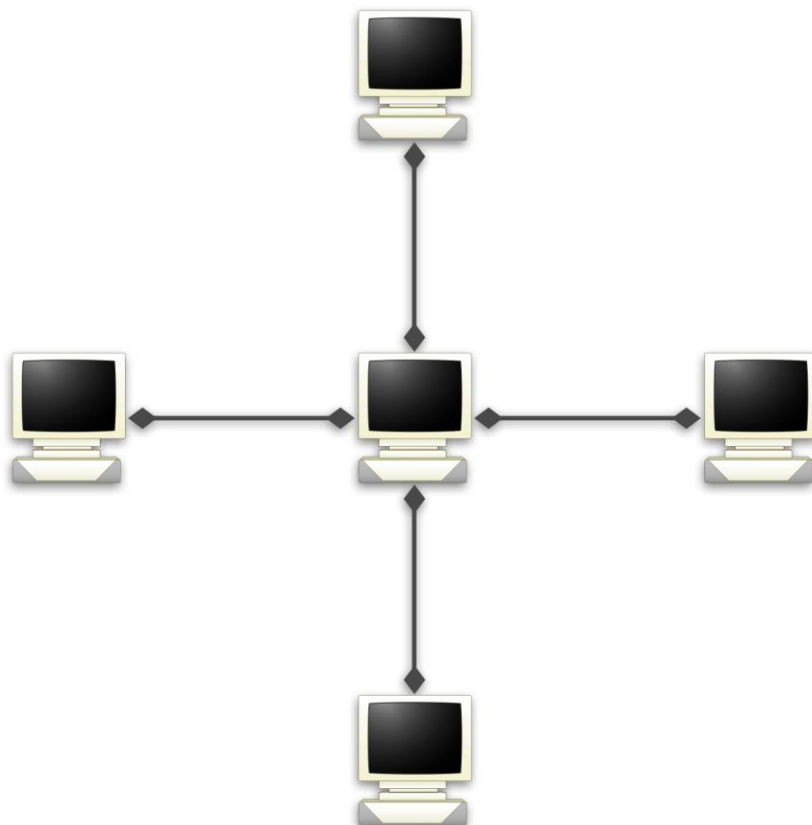


Figura 4 - Rete a stella

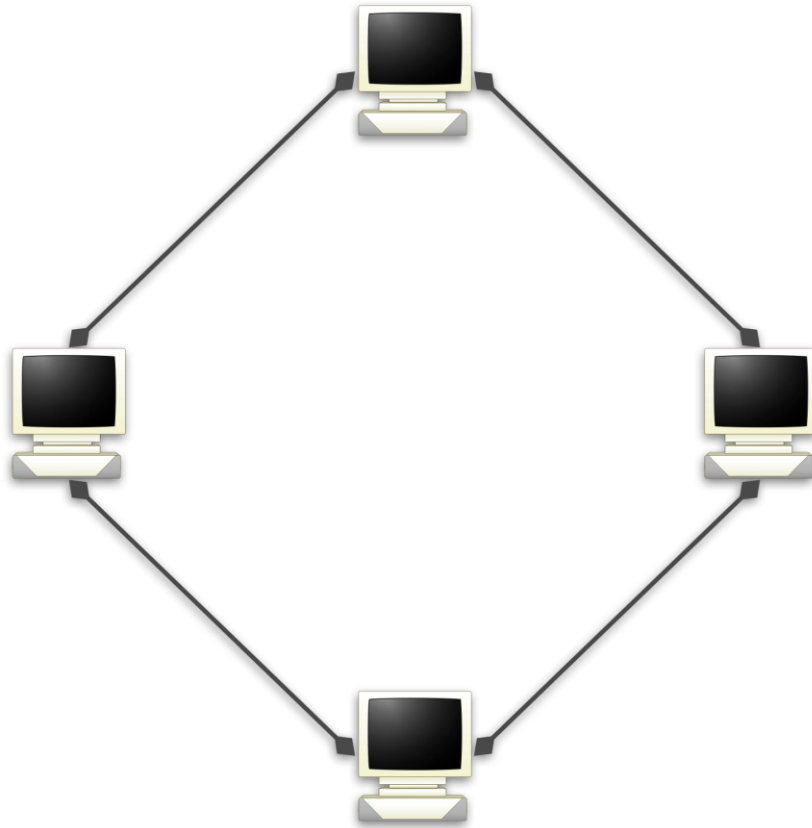


Figura 5 – Rete ad anello

Dispositivi di rete

Sono diversi i dispositivi che possono essere utilizzati in una rete, ciascuno con una particolare funzione:

- cavi e scheda di rete;
- hub/switch;
- bridge;
- router/gateway;
- modem.

Cavi e scheda di rete

I cavi vengono utilizzati come mezzo di trasmissione tra i vari dispositivi, ne esistono di vario tipo: il doppino telefonico, il cavo coassiale, il cavo in fibra ottica ed altri. E' possibile tuttavia trasmettere le informazioni attraverso l'etere utilizzando ad esempio le onde radio come le moderne stazioni *wireless* (WiFi). Uno dei cavi più utilizzati è quello per le reti Ethernet (10BaseT). Questo sistema utilizza un cavo intrecciato (derivato dal doppino telefonico) a quattro fili e un connettore RJ45 come in figura.

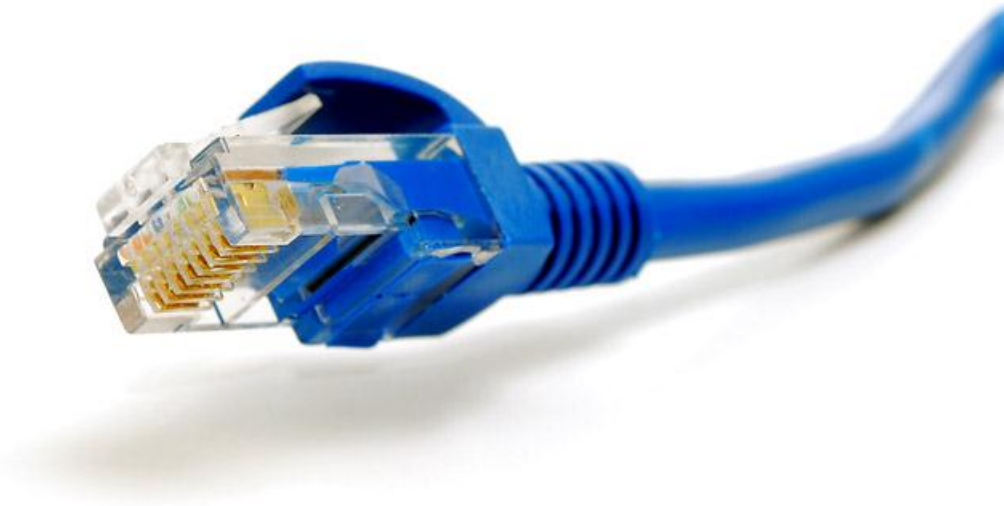


Figura 6 – Connettore RJ45

Questa configurazione è molto utilizzata: tutti i PC hanno una scheda di rete Ethernet predisposta per il connettore RJ45. Ad ogni scheda di rete è assegnato un indirizzo fisico univoco MAC (*Media Access Control*) affinché possa essere identificata all'interno di una rete locale.



Figura 7 – Scheda di rete

Hub / Switch

Questi dispositivi permettono di collegare più computer tra di loro, facendo così da nodi di unione in una rete locale. Il comportamento di un hub o di uno switch è detto *trasparente*, ovvero i computer possono comunicare come se questo non ci fosse. Uno *switch* è un dispositivo che riesce ad associare ad ogni porta l'host ad essa collegato, così i dati possono raggiungere un preciso destinatario, contrariamente all'*hub* che inoltra i dati ricevuti su tutte le porte, facendo così da ripetitore. Uno switch può connettere più postazioni (computer, stampanti di rete, ...) e questo dipende dal numero delle sue porte (4, 8, 16, 24, 48, ...).



Figura 8 – Switch a 8 porte

Bridge

Il *bridge* connette due LAN distinte, inoltrando i dati da una rete all'altra e viceversa. E' quindi utile quando è necessario connettere reti di diversa natura oppure dividere una rete, sovraffollata da host, in due sottoreti.

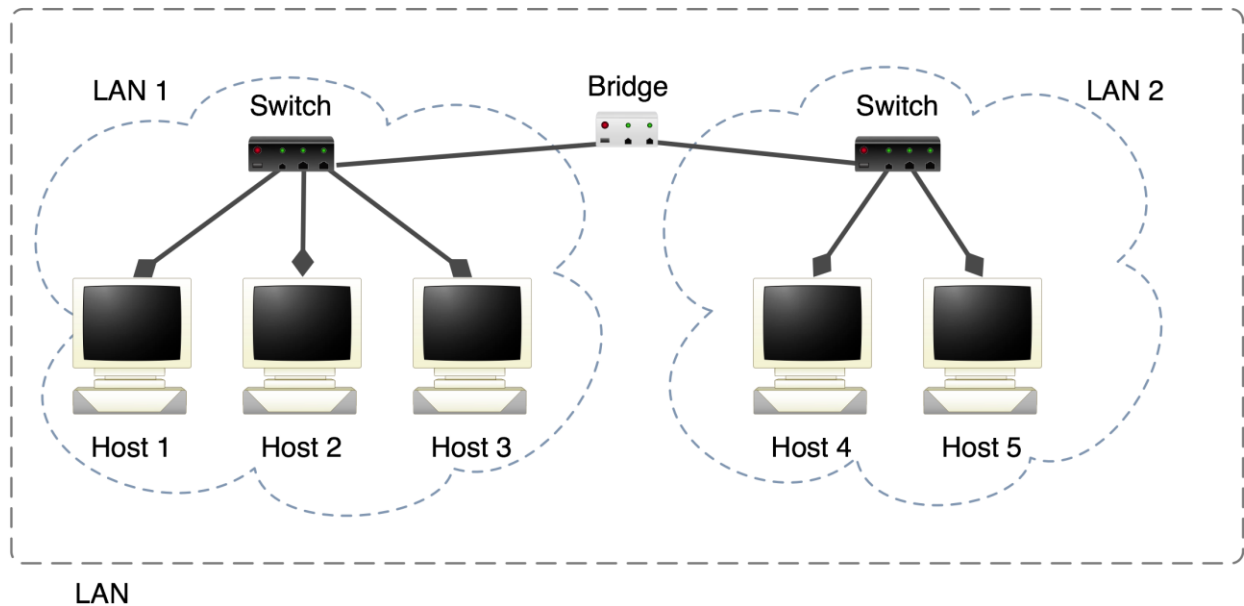


Figura 9 – Ruolo del Bridge

Router / Gateway

Contrariamente allo switch e al bridge che smistano le informazioni, all'interno di reti locali, facendo riferimento agli indirizzi MAC delle schede di rete dei computer connessi, il *router* smista (instrada) i dati avendo come indirizzi nella tabella di instradamento (*table routing*) intere reti. Questo dispositivo permette di far comunicare computer che stanno su reti diverse, esso è specializzato per l'indirizzamento delle informazione attraverso le varie reti e sottoreti di Internet fino ad arrivare all'host destinazione.

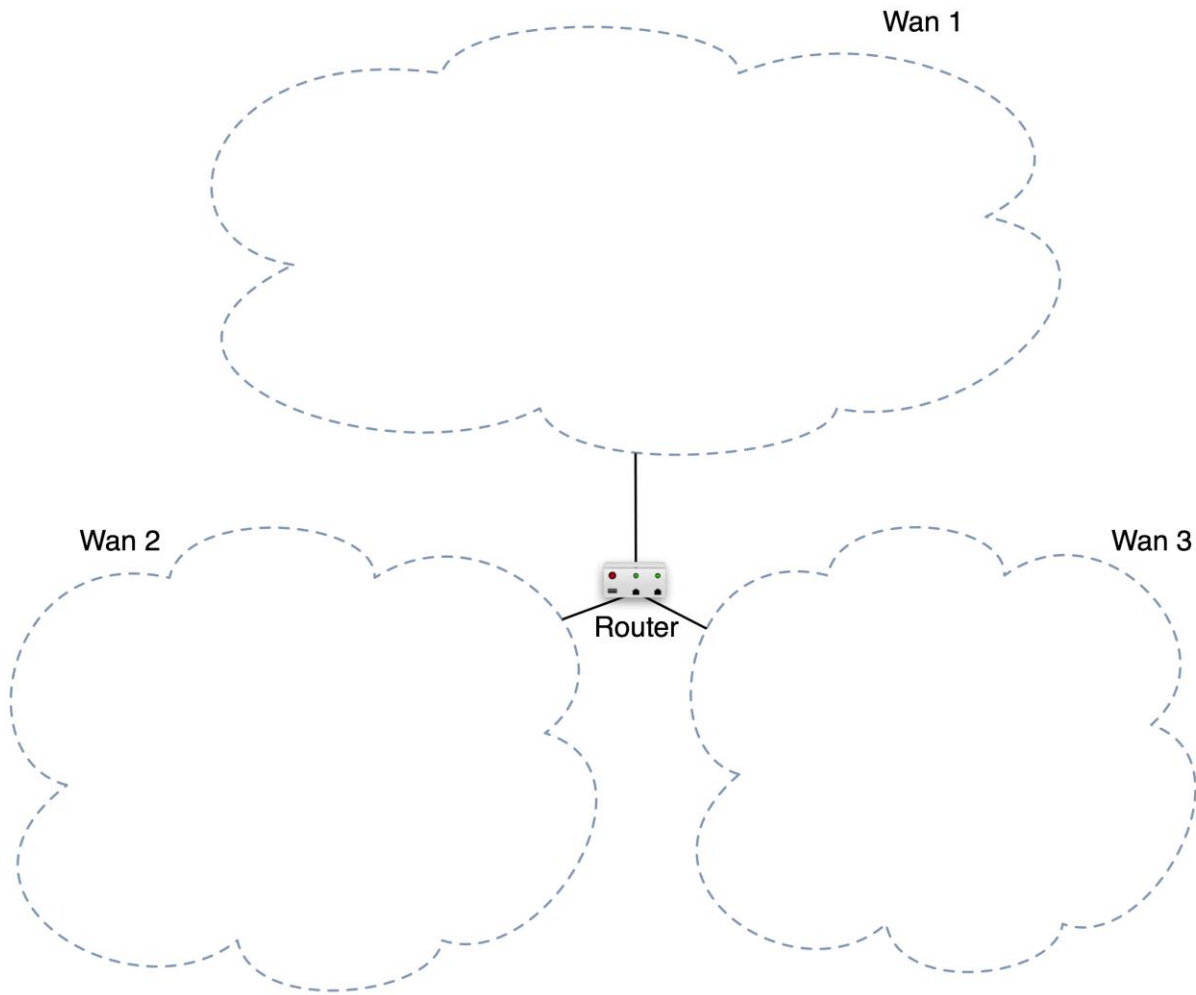


Figura 10 - Internetworking tramite un router

Il router è un dispositivo molto utilizzato nella rete Internet, dove ogni host ha associato un indirizzo IP (Internet Protocol) che identifica anche la rete dove esso si trova.

Un *gateway* è un dispositivo che oltre ad integrare le funzionalità di un router, fornisce altri servizi, come ad esempio:

- il *DNS* (Domain Name System), è un sistema per associare ad un nome di un sito un indirizzo numerico IP (ad es. `www.google.it` <-> `74.125.232.112`);
- il *Proxy* è un software che si interpone tra due computer della rete (il client e il server), ad esempio un proxy http permette di monitorare le pagine visitate e limitare l'accesso a determinati siti;
- il *Firewall* è un sistema di sicurezza che divide sostanzialmente una rete in due parti (ad esempio la rete locale di un'azienda dal di Internet), controllando, ed eventualmente bloccando, i dati in entrata.

Il modem

Questo dispositivo (modulatore-demodulatore) è probabilmente quello più conosciuto visto che da quando Internet “è arrivata” nelle nostre case i primi *modem* hanno permesso di avere una connessione attraverso la normale linea telefonica. Il modem è posizionato tra la rete telefonica e il nostro PC, esso riesce a trasformare una serie di bit (flusso) in un segnale telefonico utilizzando una tra le seguenti rete telefoniche:

- *PSTN* (Public Switched Telephone Network), la rete telefonica pubblica;
- *ISDN* (Integrated Services Digital Network), una rete di telecomunicazione digitale;
- *DSL* (Digital Subscriber Line), una rete a banda larga che sfrutta come mezzo di trasmissione il normale doppino telefonico;
- *GSM* (Global System for Mobile Communications), standard nel mondo per la telefonica mobile (senza fili);
- *UMTS*, rete mobile di terza generazione che migliora le prestazioni della rete GSM.

Le reti locali

Come già visto nelle classificazioni le reti locali (LAN) sono reti di estensione contenuta che non attraversano il suolo pubblico. Ma come si realizza una rete LAN? In realtà molto dipende dai servizi che la LAN deve offrire per soddisfare le necessità dei suoi utenti. Nel nostro caso supporremo di configurare una rete domestica per avere il collegamento tra un desktop, un laptop e una stampante; inoltre ogni computer deve poter connettersi alla rete Internet. Con queste ipotesi dobbiamo avere a disposizione uno switch che permetta di collegare tra di loro tutti e tre gli host e un gateway con un modem che gestisca la connessione ADSL e la condivida tra i vari computer. Se non vogliamo installare due dispositivi differenti possiamo acquistare un unico dispositivo gateway che integri le funzionalità dello switch (4 porte RJ45) e del modem-router per la connessione ad internet. Lo schema che si vuole seguire è quindi il seguente:

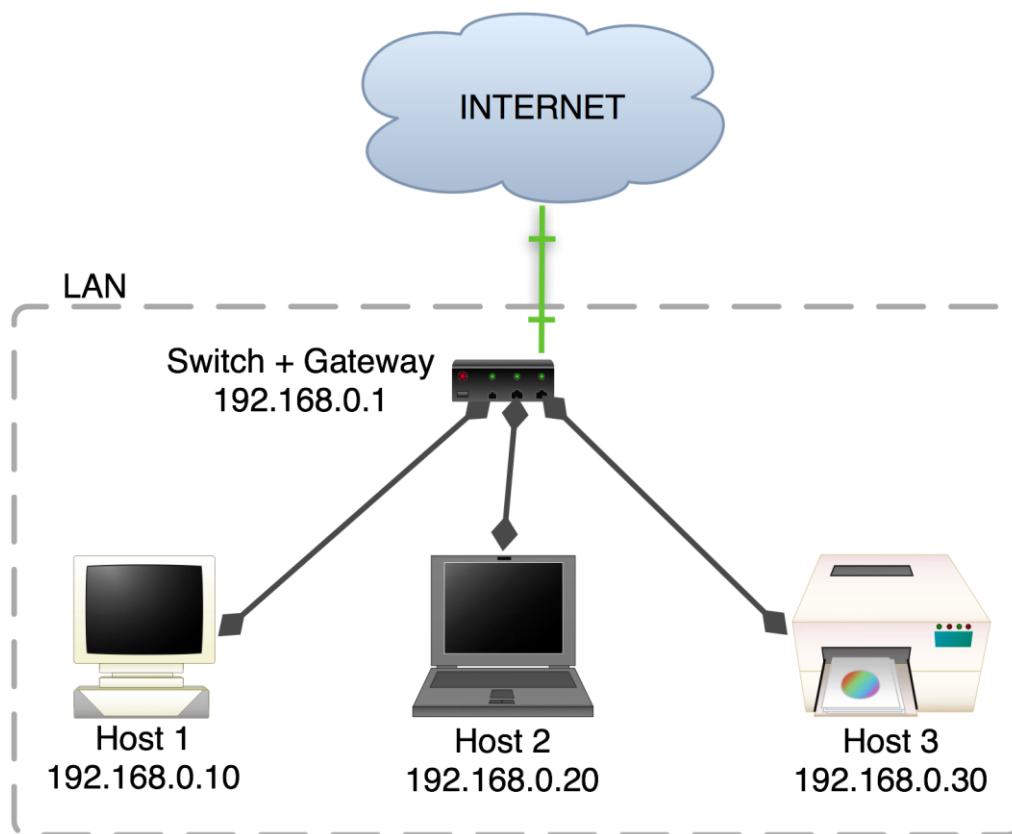


Figura 11 – Schema di una LAN domestica

Per prima cosa dobbiamo configurare il gateway, in questo caso sarà necessario collegare inizialmente uno solo degli host e ricavare dal libretto delle istruzioni il suo indirizzo IP privato, nel nostro caso sarà 192.168.0.1. A questo punto, normalmente, si utilizza il browser per avviare una connessione http verso tale indirizzo:



Figura 12 – Indirizzo privato del gateway

Il gateway chiederà una username e una password, anche questi riportati nel manuale.

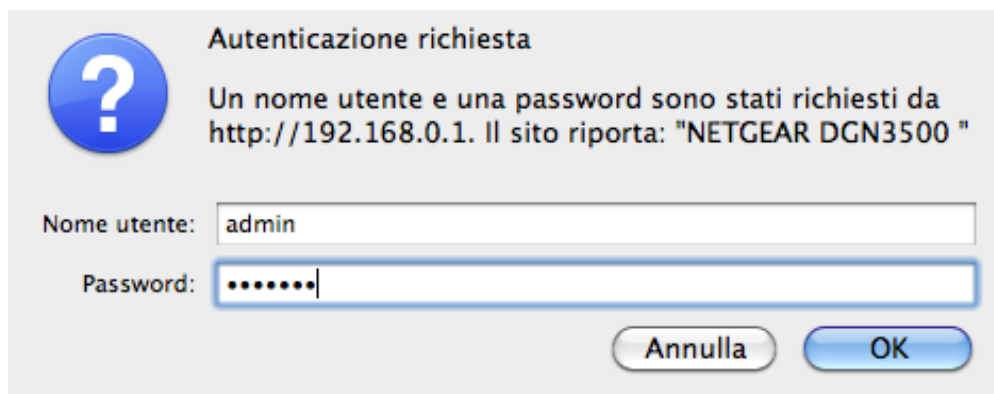


Figura 13 – Accesso al gateway

Ad autenticazione avvenuta avremo davanti svariati parametri di configurazione, come per esempio le impostazioni per la connessione ADSL al provider con il quale abbiamo il contratto:

Impostazioni di base

La connessione Internet richiede un account di accesso?

- ☒ Sì
☐ No

Incapsulamento

PPPoE (PPP over Ethernet)

Accesso

aliceadsl

Password

.....

Nome servizio (se necessario)

Tempo di inattività (in minuti)

0

Indirizzo IP Internet

- ☒ Caricamento dinamico da ISP
☐ Utilizza indirizzo IP statico

Indirizzo IP

. . .

Indirizzo Domain Name Server (DNS)

- ☒ Caricamento automatico da ISP
☐ Utilizza questi server DNS

DNS primario

. . .

DNS secondario

. . .

NAT (Network Address Translation)

☒ Attiva ☐ Disattiva ☐ Disattiva firewall

Applica

Annulla

Verifica

Figura 14 – Dati della connessione ADSL

e le impostazioni della rete LAN che il gateway deve gestire:

Impostazione IP della LAN

Impostazione TCP/IP LAN

Indirizzo IP . . .

Subnet mask IP . . .

Versione RIP

Direzione RIP

☐ Accedi all'interfaccia di gestione del router da una porta aggiuntiva
(Solo con NAT disattivato)

☒ Utilizza router come server DHCP

Indirizzo IP iniziale . . .

Indirizzo IP finale . . .

Prenotazione indirizzi

	#	Indirizzo IP	Nome periferica	Indirizzo MAC
Aggiungi Modifica Elimina				
Applica Annulla				

Figura 15 – Dati della LAN

Se lo si desidera l'indirizzo IP del gateway può essere cambiato, inoltre è possibile attivare il servizio *DHCP* (Dynamic Host Configuration Protocol), delegando al gateway l'assegnazione degli indirizzi IP locali ai vari host ad esso collegati.

Un'apposita schermata ci indicherà "lo status" del gateway, mostrando i dati della connessione ADSL (indirizzo IP pubblico, velocità in download, velocità in upload ed altro).

A questo punto possiamo passare alla configurazione dei singoli host.

Indipendentemente dal sistema operativo utilizzato, i parametri della connessione di rete, se scegliamo di impostarli manualmente, sono riportati nella figura successiva.

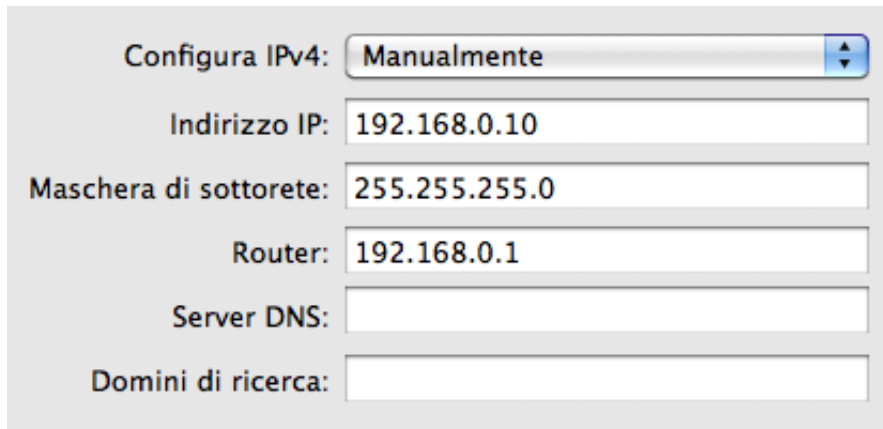
The image shows a network configuration window with a light gray background. At the top, there is a label 'Configura IPv4:' followed by a dropdown menu set to 'Manualmente'. Below this are several input fields: 'Indirizzo IP:' with the value '192.168.0.10', 'Maschera di sottorete:' with the value '255.255.255.0', 'Router:' with the value '192.168.0.1', 'Server DNS:' which is empty, and 'Domini di ricerca:' which is also empty. Each input field has a thin border and a small blue arrow on the right side.

Figura 16 – Configurazione del primo host

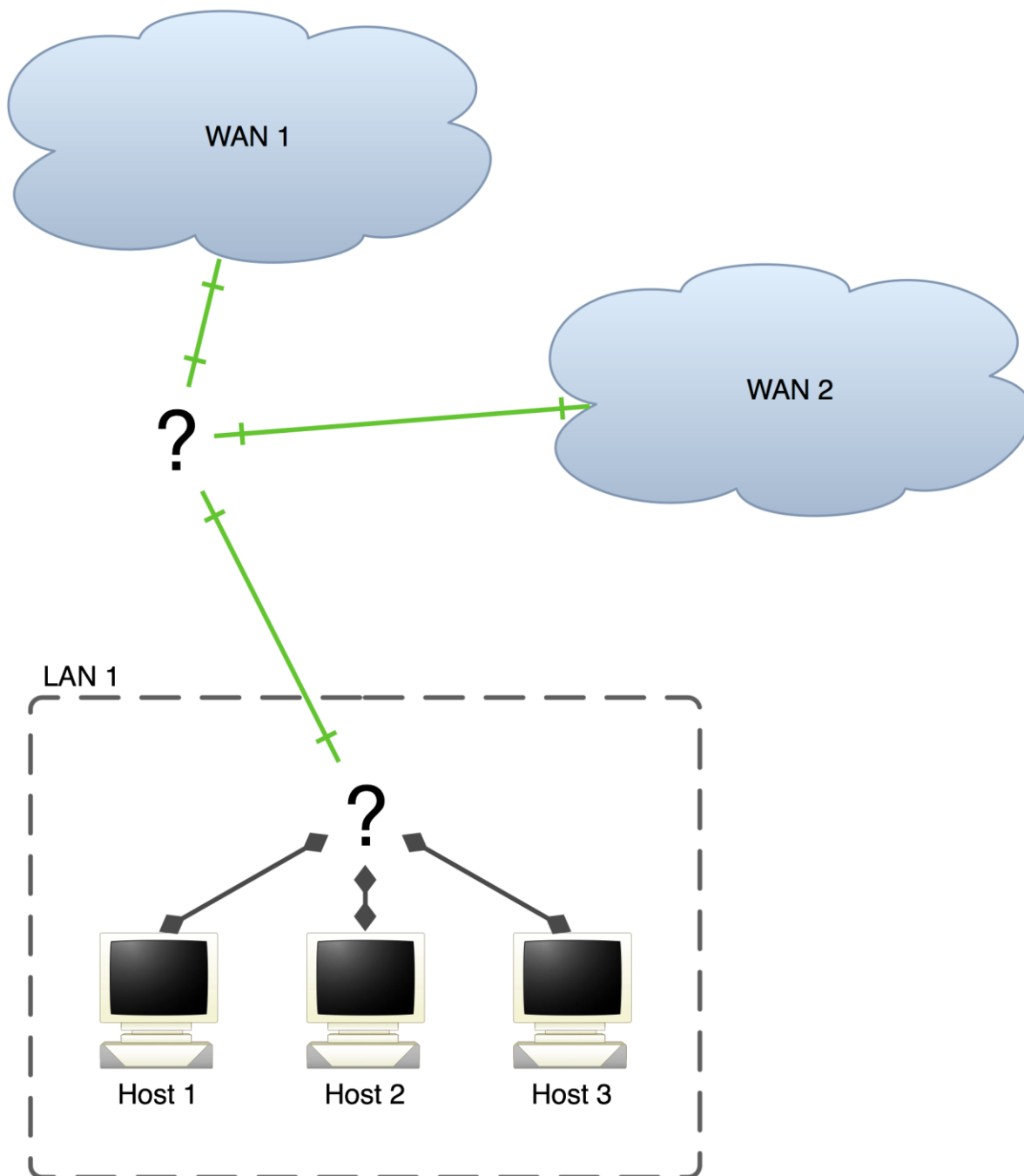
Come IP imposteremo 192.168.0.10 per il desktop, 192.168.0.20 per il laptop e 192.168.0.30 per la stampante di rete; mentre gli altri dati saranno uguali per tutti e tre gli host. Il server DNS verrà fornito dal provider, spesso viene conservato automaticamente dal gateway, in questo caso possiamo impostare come indirizzo IP del Server DNS lo stesso del router/gateway.

Finita la fase di configurazione della rete sarà possibile installare i driver della stampante sui entrambi i computer. A questo punto la LAN potrà offrire i seguenti servizi:

- gli utenti del desktop che del laptop potranno collegarsi ad Internet;
- gli utenti del desktop che del laptop potranno accedere alla stampante in modo indipendente;
- documenti e cartelle condivisi dagli utenti del desktop potranno essere visibili dagli utenti del laptop (e viceversa) ed eventualmente da altri utenti della rete Internet.

Esercizio 1

Completa il seguente schema sostituendo il '?' con il dispositivo di rete corretto.



Parte seconda – La rete Internet

Introduzione

Conosciuta come “la reti delle reti” si è via via sviluppata partendo dalla rete ARPANET (fine anni '60), la rete inizialmente utilizzata dal dipartimento della difesa degli Stati Uniti ed in seguito ampliata con la rete NSFNET (National Science Foundation) dalle università americane. La rete ebbe un successo strepitoso e dal primo gennaio 1983 il protocollo TCP/IP divenne l'unico ufficiale e da allora la connessione di numerose reti secondarie divenne più agevole; a metà degli anni ottanta la rete venne vista come un insieme di reti e sottoreti connesse tra di loro chiamate appunto Internet.

Cenni sulla struttura

Possiamo dire che la struttura di Internet è molto complessa, milioni di host sono connessi utilizzando svariati mezzi trasmissivi (doppini telefonici, cavi coassiali, fibre ottiche e onde radio), dispositivi di rete (switch, hub, bridge, router, gateway e modem) e protocolli di comunicazione (TCP/IP, UDP, HTTP, FTP, ...). Tale struttura oltre ad essere complessa non è omogenea, infatti interconnette reti di natura diversa sia per estensione (LAN, MAN e WAN) sia per topologia (a stella, a bus e ad anello).

Il modello Client – Server

La stragrande maggioranza dei software che funzionano sulle reti di computer sono stati sviluppati secondo il *modello client-server*. Questo modello presuppone che una parte del software, detto appunto *client*, venga installato su uno dei due computer e serva essenzialmente per inviare delle richieste, tramite messaggi, all'altra parte dell'applicazione, il server, installata sul secondo computer. Il *server*, invece, è la parte più complessa, infatti deve gestire le funzionalità dell'intero software, ad esempio un server di posta elettronica funziona esattamente come un ufficio postale. Il server, dopo l'elaborazione, comunica con il client tramite un *protocollo* stabilito, inviando la risposta.

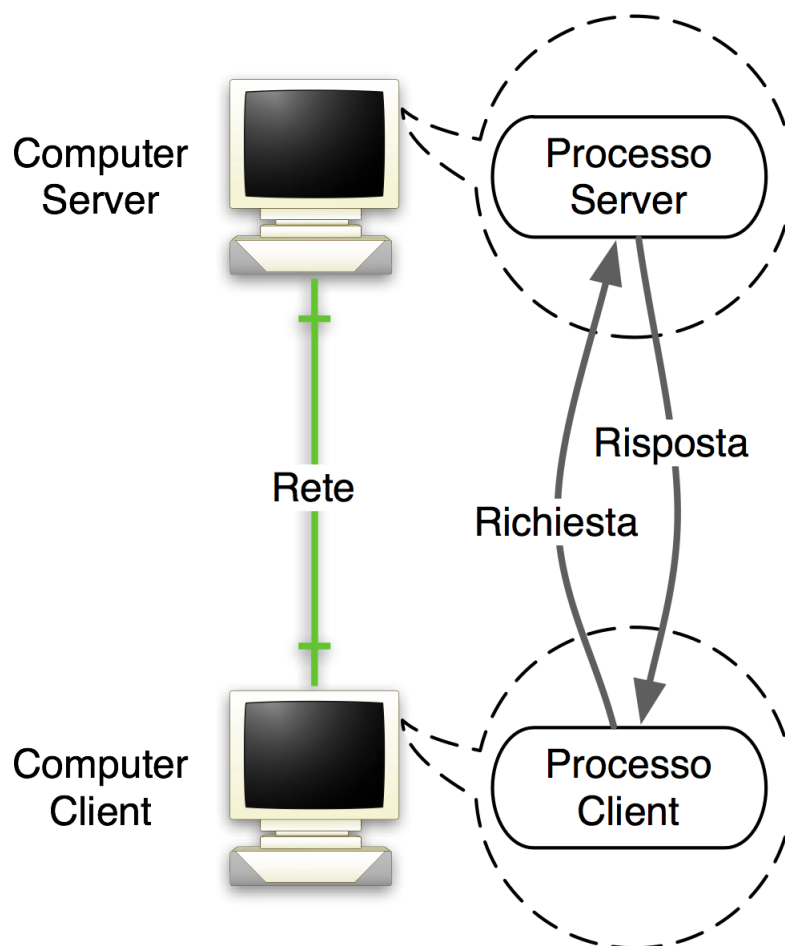


Figura 17 – Modello Client – Server

Il modello TCP/IP

Il modello di riferimento TCP/IP è quello utilizzato dalla rete Internet, esso si basa su due importanti protocolli, *TCP* e *IP*. Il modello è realizzato tramite una “pila” di protocolli di trasmissione posta tra un mittente ed un ricevente così da garantirne la loro comunicazione. Questa è la struttura:

Applicazione
Trasporto
Internet
Host-Rete

Il livello host-rete

Il modello TCP/IP non definisce questo livello, eccetto il fatto che l’host debba essere connesso alla rete tramite una connessione Ethernet, WiFi, PPP, Token ring, HSDPA, ecc.

Il livello internet

Questo è il perno di tutta l’architettura, garantisce tramite protocolli come l’IP versione 4 e versione 6, il trasferimento di pacchetti da un host ad un altro, anche appartenenti a reti di differenti nazioni. Ogni pacchetto, contrassegnato da numero identificativo, viaggia verso la destinazione percorrendo “instradamenti” diversi, tutto in modo *trasparente* sia per il mittente sia per il destinatario.

Il livello trasporto

In questo livello protocolli come TCP e UDP garantiscono la continuità della comunicazione. Il primo è un protocollo *affidabile*: garantisce che una comunicazione espressa come sequenza di byte venga consegnata, senza errori, al destinatario. Un esempio è un messaggio di posta elettronica. In alternativa, il protocollo UDP è *non affidabile*: non garantisce che tutti i pacchetti siano arrivati a destinazione e senza errori, ma in compenso è molto veloce. Un esempio è lo streaming video.

Il livello applicazione

Questo è il livello più alto, troviamo tutti i protocolli utilizzati direttamente dai nostri software per comunicare sulla rete: HTTP, Telnet, FTP, SMTP, DNS, ecc.

Funzionalità e servizi

Le reti di calcolatori, come Internet, offrono innumerevoli servizi che possono essere fruiti purché si abbia installato e configurato il software adeguato, vediamo alcuni:

1. il *terminale virtuale*, permette di gestire un altro computer della rete, di cui si abbia l'accesso, dalla postazione in cui ci si trova;
2. la *posta elettronica*, realizza uno scambio di messaggi tra utenti che abbiano un'apposita casella *e-mail*, i messaggi vengono inviati e ricevuti tramite dei computer chiamati server di posta elettronica, i quali gestiscono quotidianamente milioni di messaggi, la comunicazione in questo caso è *asincrona*;
3. la *chat*, un mezzo di comunicazione con il quale due o più utenti possono comunicare in contemporanea, come avviene con le conversazioni al telefono (comunicazione *sincrona*);
4. la *teleconferenza*, con la quale si possono organizzare degli incontri virtuali con un collegamento audio/video tra i partecipanti distanti tra loro;
5. il *Web* (WWW, World Wide Web), è un ambito della rete molto ampio in cui grazie a un programma detto browser (Firefox, MS Explorer, Opera, ...), ormai in dotazione su tutti i computer, si naviga in un sito web (commercio elettronico, informazione, meteo, ...) seguendo i collegamenti di un ipertesto. Il Web ha fatto sì che la rete sia divenuta un potente mezzo di comunicazione di massa.

La posta elettronica

Questo servizio, che implementa un modello client-server, permette di inviare e ricevere messaggi tra i vari utenti. Il provider (Gmail, Hotmail, Yahoo, ...) assegna ad ogni utente che lo richieda un indirizzo e-mail che individua in modo univoco la casella di posta elettronica. L'indirizzo è composto da due parti separate dal simbolo '@' (si legge at), a sinistra abbiamo il nome che si sceglie durante la fase di registrazione della casella, invece a destra abbiamo il dominio del provider.

paolo@gmail.com

Durante la registrazione di un nuovo indirizzo e-mail il *provider* ci richiede una password oltre ad un nome utente (*username*). La coppia username e password costituisce il *login* per poter accedere successivamente alla propria casella e-mail.

Per consultare e gestire la propria casella di posta elettronica, sostanzialmente, esistono due vie: installare un programma detto *client* di posta elettronica; oppure, consultare con il browser il sito del proprio provider che normalmente darà accesso alla e-mail.

Accesso con ThunderBird

Anche se i client di posta elettronica sono tanti (basti citare MS Outlook, Thunderbird, Apple Mail, ...), essi funzionano più o meno allo stesso modo. Con le figure successive vedremo come creare un nuovo *account* di posta con il programma *Thunderbird*, ovviamente è necessario avere già registrata la propria casella di posta su un provider a nostra scelta.

- 1) Scegliamo di creare un nuovo account di posta come in figura:

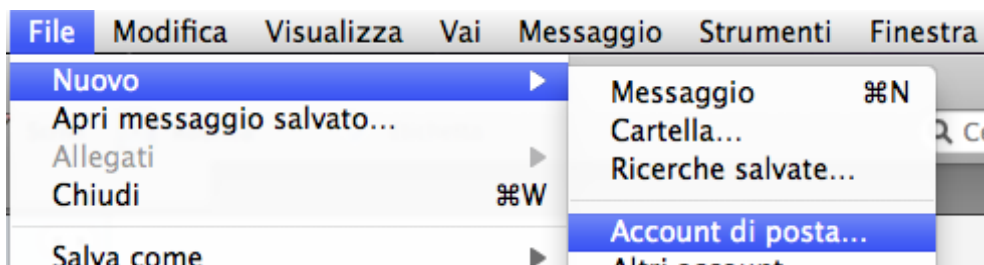


Figura 18 – Creazione di un account di posta

- 2) Nella fase successiva il software richiede i dati essenziali per poter accedere alla casella di posta elettronica:

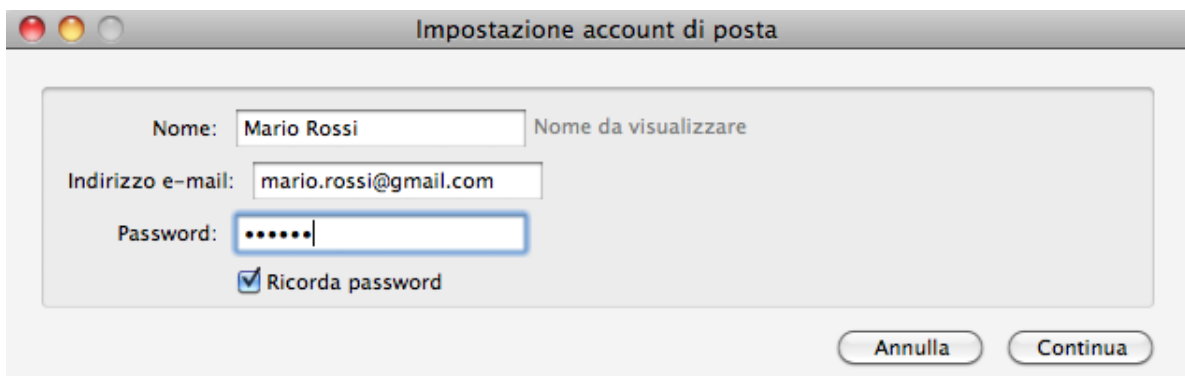


Figura 19 – Impostazioni account di posta

- 3) Infine, Thunderbird, recupera automaticamente le impostazioni dei server di posta in uscita ed in entrata mostrando il seguente risultato:

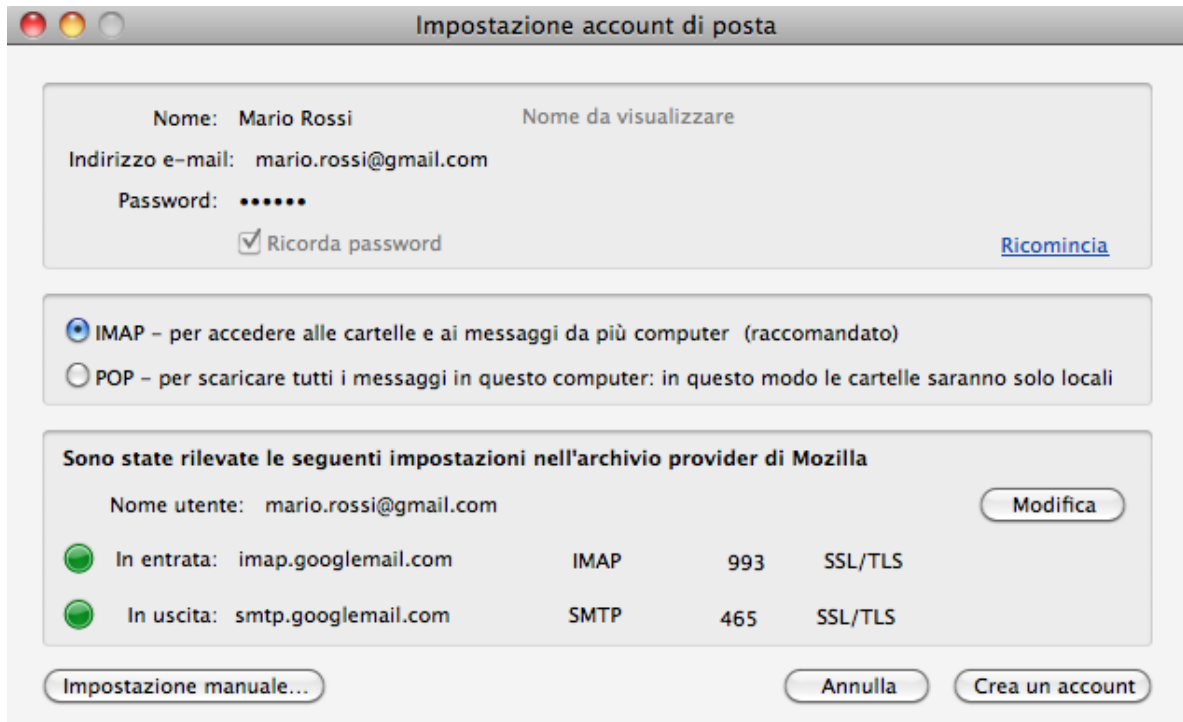


Figura 20 – Impostazioni account di posta

Nella figura 20 sono presenti due scelte *IMAP* o *POP*, la prima è raccomandata quando si vuole accedere alla propria e-mail da diversi computer pur mantenendo una visione sincronizzata dei messaggi in ingresso e in uscita; la seconda, invece, è consigliata se la gestione avviene da un solo computer, dove verranno scaricati i messaggi in delle cartelle locali.

Non sempre Thunderbird riesce a rilevare autonomamente le impostazioni dei server (SMTP, POP o IMAP), in questi casi è possibile scegliere l'impostazione manuale per inserire manualmente gli indirizzi del proprio provider.

Gli ipertesti e il Web

Un ipertesto è una raccolta di documenti messi in relazione tramite delle parole chiave dette *hot words*. Ogni documento costituisce un *nodo* dell'ipertesto, mentre la parola chiave un *link*. Un ipertesto che contiene nei suoi nodi delle immagini, dei suoni o dei filmati prende il nome di *ipertesto multimediale*.

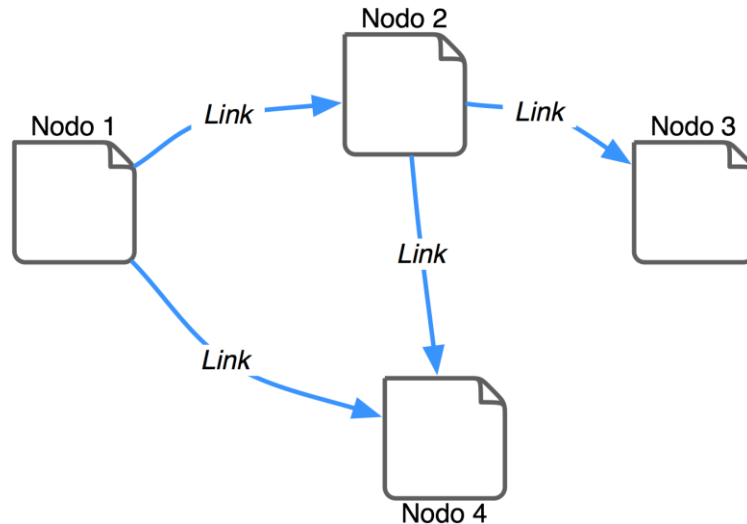


Figura 21 – Ipertesto con nodi e link

Un po' di storia

Gli ipertesti più utilizzati sono sicuramente quelli informatici. Nel 1987 un software chiamato *HyperCard* e sviluppato dalla *Apple Computer*, in anticipo sul linguaggio HTML, permetteva, grazie ad una interfaccia grafica intuitiva, la creazione di schede multimediali (composte da testo, immagini e suoni) e il loro collegamento attivando delle parole chiave. Questo software, sebbene molto flessibile, non era adatto allo sviluppo di un ipertesto in rete, esigenza nata in seguito con l'avvento delle reti di calcolatori.

Nel 1989 un ambiente informatico conosciuto come *WWW* nasceva al *CERN* (Centro europeo per la ricerca sulla fisica nucleare), dove gruppi di ricercatori riuscivano così a condividere documenti, foto ed appunti per il loro lavoro. Un programma (il primo browser!), chiamato *Mosaic*, permetteva la consultazione e la navigazione attraverso questi documenti seguendo i vari link.

Il Web (WWW) e i suoi elementi

Il *World Wide Web* è l'architettura software per utilizzare gli ipertesti multimediali sulla rete Internet. Questa architettura che rispecchia il modello client-server è realizzata grazie ad alcuni software:

- un linguaggio per la creazione degli ipertesti, chiamato *HTML* (HyperText Markup Language);
- un *Web Server* per pubblicare le pagine ipertestuali di un computer (Apache, TomCat, IIS, ...);
- un *browser* per la navigazione, installato sul client (Firefox, Internet Explorer, Opera, Safari, ...);
- un protocollo di comunicazione, chiamato *HTTP* (HyperText Transfer Protocol).

Quando un utente richiede una pagina Web, tramite la rete Internet, utilizza un browser per comporre il relativo indirizzo *URL* (Uniform Resource Locator)

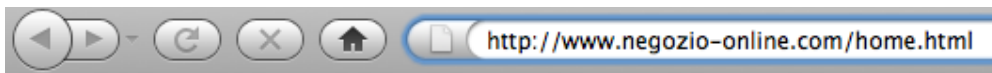


Figura 22 – Esempio di URL

specificando:

- il protocollo utilizzato, in questo caso `http`;
- l'indirizzo del sito, `www.negozio-online-com`;
- e la pagina alla quale si intende accedere, `home.html`.

A questo punto il Web Server risponde alla richiesta ricevuta e sempre utilizzando lo stesso protocollo invia la pagina *home.html* al browser, il quale ne interpreta il codice html contenuto, mostrando il risultato.

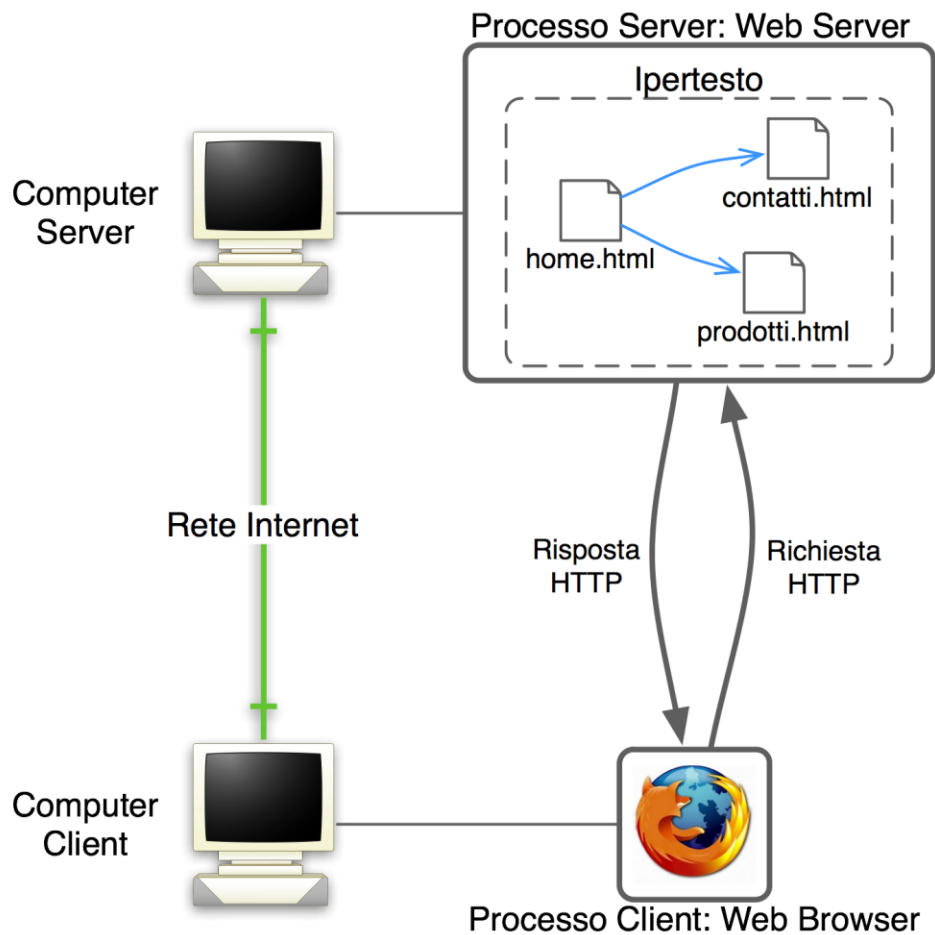


Figura 23 – L'architettura Web

Il commercio elettronico

L'utilizzo del Web e di altri strumenti come i database e i web server hanno dato la possibilità a milioni di persone di effettuare i propri acquisti comodamente da casa, dando così vita al commercio elettronico (E-Commerce). Annualmente, in Italia, milioni di ordini passano attraverso le connessioni di Internet e vengono gestiti dal sistema software e hardware dei negozi on-line.

La figura seguente mostra l'aspetto tecnico-informatico di un'attività on-line.

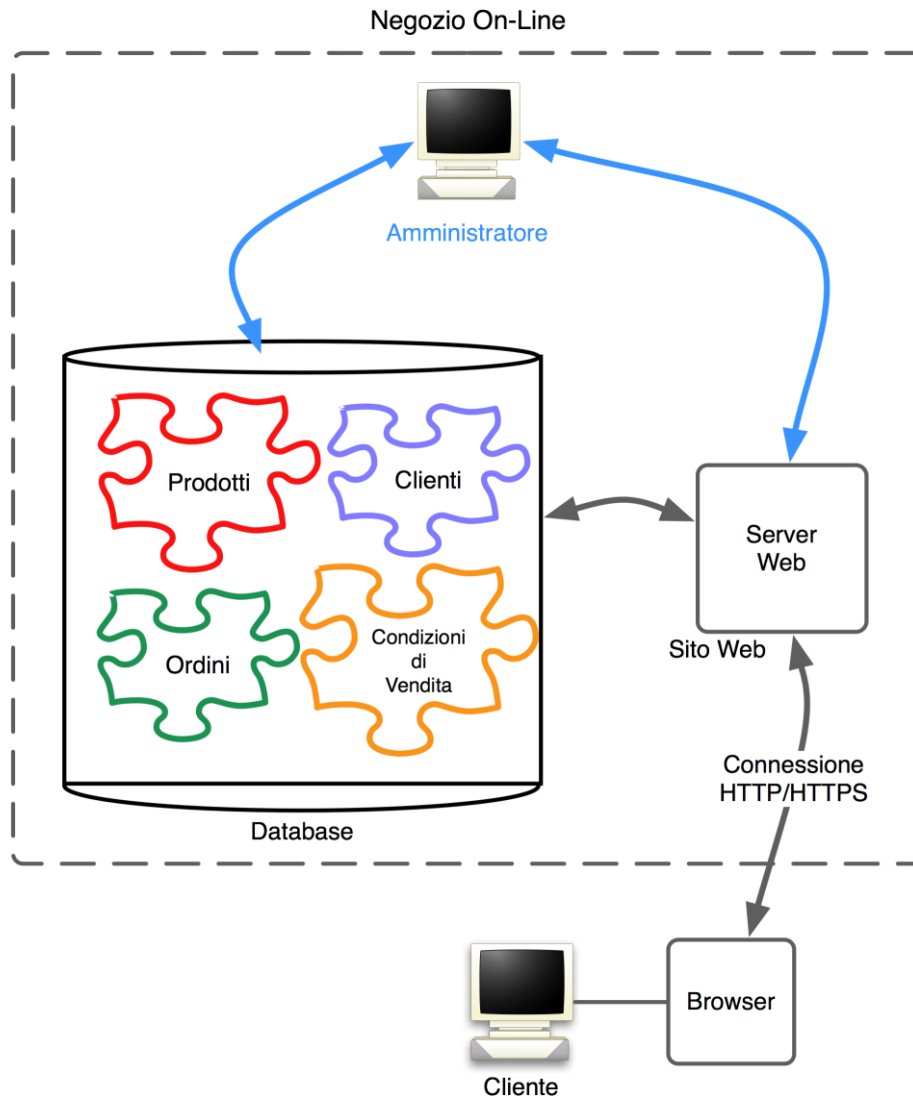


Figura 24- Commercio elettronico

L'amministratore

Si occupa di gestire il sito Web promuovendo i prodotti e i servizi attraverso delle schede dettagliate ipertestuali. L'amministratore deve mantenere aggiornata la base di dati (database), inserendo nuovi prodotti, eliminando quelli non più trattati ed eseguendo

periodicamente delle copie di sicurezza (back-up).

Il database

Il database è quel modulo software, realizzato con un DBMS (Database Management System), che consente di archiviare e gestire i dati che riguardano i prodotti (o i servizi), i clienti, gli ordini e le condizioni di vendita. Anche se “trasparente” ai clienti che visitano il sito Web è il cuore di questo sistema.

Il sito web

Costituisce, oltre che una vetrina virtuale del negozio, un'interfaccia tra il database e i clienti che intendono effettuare degli acquisti. Infatti, il modulo Web si interfaccia con il database per raccogliere, ad esempio, i dati di un prodotto, oppure, come spesso succede, richiedere una ricerca in base alla marca, al modello o al prezzo. Il sito Web si presenta come un *ipertesto multimediale* dove l'utente può navigare tra i vari reparti del negozio e visionare i prodotti per lui più interessanti. Chiaramente il sito Web deve essere accattivante e ben realizzato, mescolando accuratamente disegni grafici, foto, caratteri tipografici e animazioni.

Il cliente

L'utente che vuole finalizzare un acquisto deve registrarsi con una *username* e una *password* che verranno conservati nel database e diventare così un cliente del negozio on-line. I prodotti che si desiderano acquistare verranno raccolti in un carrello virtuale che potrà essere portato alla cassa completando l'ordine di acquisto. La conferma dell'ordine costituisce una formale accettazione, e include la tipologia di pagamento e il tipo di trasporto. Come si vede dallo schema l'unico software di cui deve disporre il cliente è un browser Web.

Cenni sulla sicurezza, le normative sulla privacy e il diritto d'autore

Sicurezza

“Per persone a casa, vagabondare per Internet è molto divertente. Per gli esperti di sicurezza di società private è un incubo” [A. Tanunbaum – “Reti di Computer”]. I dati intesi come informazione hanno avuto da sempre un ruolo rilevante e garantirne la loro segretezza non è affatto facile. Pensiamo ai dati aziendali, ai segreti commerciali, agli schemi progettuali, all’elenco dei clienti e dei fornitori, sono tutti dati che un’azienda non vorrebbe mai rivelare ai propri rivali. Se poi prendiamo in considerazione i dati di natura militare, allora il ruolo della sicurezza diventa cruciale.

In campo informatico la sicurezza riguarda vari ambiti, noi ne vedremo due. Cominciamo con la *segretezza*. Questa può essere raggiunta o quanto meno migliorato il suo livello, in vari modi. Il *firewall* è uno strumento che agisce a livello di rete, è sostanzialmente un router che ha la capacità di controllare i singoli pacchetti ispezionandone il contenuto. I pacchetti che violano anche una sola delle regole stabilite vengono scartati. Le regole vengono raccolte in delle tabelle accessibili soltanto dagli amministratori del network; grazie a questi filtri è possibile specificare gli indirizzi IP dei mittenti ammessi e/o dei destinatari e le porte abilitate per i servizi.

A livello di applicazione un altro strumento viene utilizzato per garantire un buon livello di sicurezza: la *crittografia*. I *cifrari* specificano le regole per trasformare un messaggio in chiaro in un messaggio cifrato (*cifratura*) e viceversa (*decifratura*). E’ necessario l’utilizzo di una chiave k , conosciuta solo ai due interlocutori, ma segreta per tutti gli altri (*cifratura simmetrica*). Uno esempio è il *cifrario di Cesare*.

Nella realtà non è semplice scambiare a priori una chiave segreta tra il mittente e il destinatario attraverso un canale sicuro. Questo problema è stato risolto con i *cifrari asimmetrici* in cui la chiave per cifrare e quella per decifrare sono differenti. Un utente A che vuole iniziare una comunicazione cifrata con B , dovrà generare due chiavi, tramite appositi algoritmi, una per cifrare C_A (*chiave pubblica*) ed una per decifrare D_A (*chiave privata*). L’utente A rende pubblica la chiave di cifratura, in questo modo chi vuole inviare un messaggio cifrato ad A , per esempio l’utente B , può farlo utilizzando C_A . Una volta ricevuto il messaggio l’utente A può decifrarlo facendo uso della chiave D_A che è in suo possesso. Una delle condizioni necessarie affinché i cifrari asimmetrici funzionino è che sia estremamente difficile dedurre D_A da C_A .

Il secondo ambito della sicurezza è l’*affidabilità dei documenti*. Il problema presenta due aspetti:

- dato un documento deve essere sempre possibile conoscerne il suo autore che non potrà disconoscerlo;
- non sia possibile modificare il documento di un'altra persona senza lasciare traccia o creare un documento attribuendolo a qualcun altro.

L'affidabilità dei documenti viene raggiunta oggi tramite la firma digitale, un meccanismo che utilizza la chiave privata per cifrare e quella pubblica per decifrare. In pratica, una persona o un ente che vuole apporre la propria firma digitale su un documento in formato elettronico deve richiedere il rilascio ad un'autorità competente di un certificato digitale valido. Una volta acquistata la firma digitale (che avrà un periodo di validità) può essere utilizzata per cifrare il documento, ottenendo nuovo file cifrato che conterrà le informazioni del documento e la firma dell'autore. Chi riceve il file può controllare la paternità leggendo la firma digitale, attualmente il formato PDF supporta queste tecnologie. In realtà, grazie alle firme digitali, viene anche affrontato il secondo aspetto: se una terza persona volesse contraffare un documento cifrato questo verrebbe automaticamente reso incoerente rispetto alla firma apposta e quindi segnalata la manomissione in fase di lettura dal destinatario. Allo stesso modo sarebbe difficoltoso poter produrre un nuovo documento attribuendolo ad un'altra persona senza aver prima trafugato la sua firma digitale.

Legge sulla privacy (codice sulla sicurezza dei dati)

Quando un utente vuole accedere ad una rete di computer (LAN, MAN, WAN) deve possedere un account che può ottenere previa compilazione di un questionario in cui inserire i propri dati personali. Successivamente il *net manager* (gestore della rete) o *provider* invia l'username (nome utente) e la password (parola chiave) personalizzabile e modificabile periodicamente, per l'accesso al servizio. A partire da questa semplice operazione, si pone la necessità della protezione dei dati individuali. I soggetti che custodiscono questi dati (chiamati titolari), istituti bancari, amministrazioni pubbliche, professionisti, fornitori di servizi internet (web, e-mail,...), sono obbligati ad osservare la legge sulla privacy (codice in materia di protezione dei dati personali, Dlgs 196/2003) che regola la registrazione, la custodia, l'elaborazione e la trasmissione dei dati individuali contenuti nei database (archivi). In passato, quando gli archivi erano in forma cartacea, potevano essere protetti più facilmente, perché custoditi in luoghi delimitati, accessibili a pochi operatori e riproducibili solo manualmente. Attualmente gli archivi possono essere trattati (raccolti, registrati, organizzati, conservati, consultati, elaborati, diffusi) con strumenti elettronici (reti di computer) e software specifici per database: i documenti

cartacei (hard copy) sono stati sostituiti da quelli informatici (soft copy) e la loro elaborazione viene eseguita quasi istantaneamente (real time). Se da un lato questo è un vantaggio, in termini di velocità e precisione, anche in presenza di archivi composti da numerosi elementi (record), da un altro lato aumenta il rischio che tali dati siano usati dal responsabile dell'archivio per usi personali oppure consultati illecitamente da altri individui per fini commerciali (pubblicità, statistiche, ...), nel caso meno grave, o per fini criminali (persecuzioni, truffe, ...). Il codice impone, a chi gestisce un archivio, di usare determinate procedure di protezione e limitare al minimo la quantità di dati personali, identificativi e sensibili (come la razza, la religione, l'orientamento politico e sessuale,...). Ogni addetto alla gestione dell'archivio è obbligato all'inserimento di un numero di autenticazione (username) e una parola chiave personale (password), che costituiscono le credenziali di autenticazione, segrete, per entrare nel sistema (procedura di identificazione). La password, composta di almeno otto caratteri, senza riferimenti riconducibili all'operatore, deve essere modificata al primo utilizzo e almeno ogni sei mesi. Le credenziali di autenticazione devono essere disattivate per mancato uso nell'arco di sei mesi o in caso di smarrimento. L'interessato (individuo presente in un database) ha infine il diritto di accedere ai propri dati per conoscerne le finalità, modificarli, cancellarli, o trasformarli in forma anonima e può opporsi al loro trattamento sia che vengano usati solamente allo scopo della raccolta sia che vengano usati per fini commerciali, pubblicitari e statistici .

Diritti d'autore

La diffusione del computer e di internet ha reso più pressante la questione della tutela del *copyright* (diritto d'autore). Infatti è noto come sia facile copiare un programma per videogame con il computer o scaricare, da internet, files di qualsiasi genere (immagini, musica,...) e software (programmi per elaboratore). La legislazione italiana sulla protezione del diritto d'autore e di altri diritti connessi al suo esercizio, risalente al 1941 (legge 633/1941), è stata aggiornata per comprendere anche il software e i database, equiparati ad opere dell'ingegno di carattere creativo. Gli autori di opere letterarie, composizioni musicali, disegni, opere cinematografiche e fotografiche erano tutelati nella prima emanazione della legge. La digitalizzazione (conversione in formato digitale) di tali opere ha semplificato e velocizzato la riproduzione e la trasmissione mediante computer e rete internet. Costituiscono software i sistemi operativi (windows, linux, mac,...), i pacchetti applicativi per casa e ufficio (Microsoft office, Lotus synfony, iwork Apple,...) che permettono il word processing (videoscrittura), gli spread-sheet (fogli elettronici), i programmi per videogame (videogiochi). L'autore di una di queste opere è il titolare dei

diritti esclusivi di riprodurre, distribuire, noleggiare o dare in prestito l'opera o esemplari di essa. La SIAE detiene il registro pubblico in cui è trascritto il nome del software e dell'autore che così ne può provare la paternità. Chi abusivamente utilizza, duplica, riproduce le opere munite del contrassegno della SIAE oppure acquista o noleggia attrezzature per eluderne le protezioni è punito con una multa e la confisca del materiale. Oltre la multa, rischia il carcere chi abusivamente duplica, distribuisce, vende e detiene, per trarne profitto, software o banche dati, di cui non è l'autore, oppure ne rimuove le chiavi di protezione. Dal punto di vista dei diritti d'autore il software diffuso nel Web si suddivide in vari tipi.

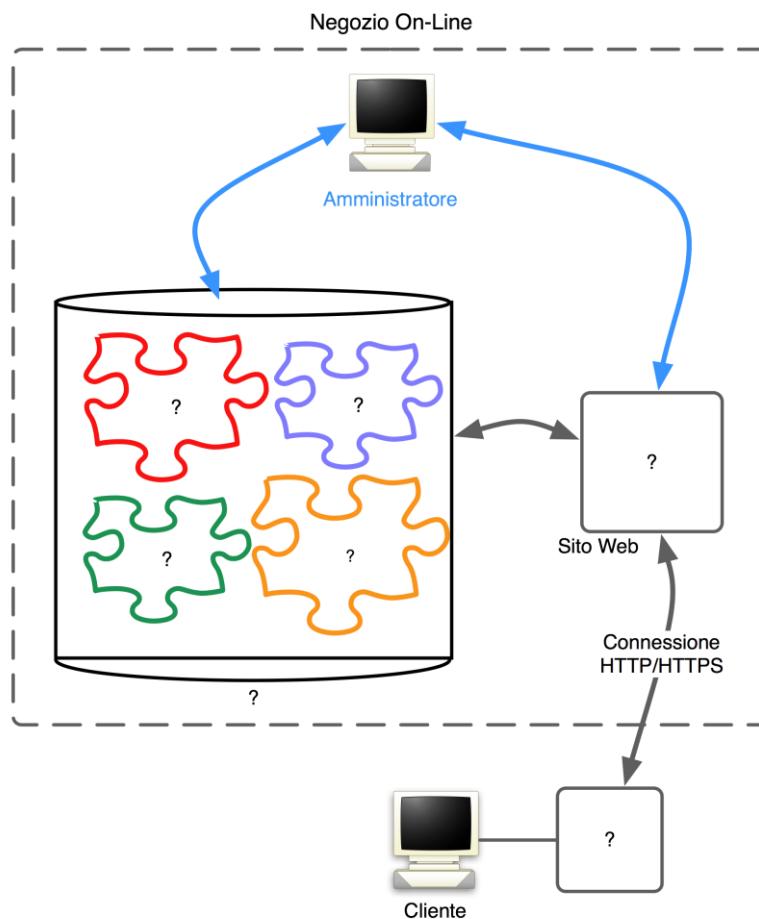
- 1) *open source* (codice aperto): distribuito e copiabile gratuitamente, il cui codice sorgente è accessibile e modificabile, senza violare il diritto d'autore.
- 2) *freeware* (libero): distribuito e duplicabile gratuitamente, il cui codice non è però accessibile.
- 3) *shareware* (condiviso): circola liberamente sulla rete e può essere copiato ed utilizzato gratuitamente nei limiti indicati dalla licenza. Possono essere previsti l'utilizzo:
 - a. temporaneo, entro un certo termine;
 - b. parziale, con alcune funzioni disattivate.
- 4) *public domain* (dominio pubblico): chi lo ha creato non esercita il diritto d'autore, per cui è gratuitamente utilizzabile e duplicabile.

Esercizio 2

1) Utilizzando Google vogliamo ricercare le pagine Web che parlino di `spiderman` ma non di `superman`. Volendo restringere la ricerca al sito `www.fumetti.org`, quale di queste è la scrittura corretta:

- a. `"spiderman superman" site:www.fumetti.org`
- b. `spiderman +superman site:fumetti.org`
- c. `spiderman -superman site:fumetti.org`
- d. Nessuna delle precedenti.

2) Relativamente al commercio elettronico completa il seguente schema:



3) Fai le giuste associazioni:

Shareware	<i>Il cui codice sorgente è accessibile e modificabile, senza violare il diritto d'autore</i>
Open source	<i>Distribuito e duplicabile gratuitamente, il cui codice non è però accessibile</i>
Freeware	<i>Chi lo ha creato non esercita il diritto d'autore</i>
Public domain	<i>Circola liberamente sulla rete e può essere copiato e utilizzato nei limiti indicati nella licenza (entro un certo termine o con alcune funzioni disabilitate)</i>

4) Si vuole realizzare nella pagina `home.html` un collegamento alla pagina `prodotti.html`, utilizzando la parola chiave `prodotti`. Quale delle seguenti scelte è quella corretta?

- Inserire nel codice della pagina `home.html`
`<a href = "home.html">Prodotti</a>;`
- Inserire nel codice della pagina `home.html`
`<a href = "prodotti.html">Prodotti</a>;`
- Inserire nel codice della pagina `prodotti.html`
`<a href = "prodotti.html">Prodotti</a>;`
- Nessuna delle precedenti.

Parte terza – Approfondimenti ed appendici

Approfondimento A: I motori di ricerca

I motori di ricerca sono oggi i siti Web più consultati, questi veri e propri *sistemi automatici* si occupano di raccogliere dati sul Web e costruire dei cataloghi in modo tale da rendere disponibile un elenco di risposte ad ogni possibile ricerca di una o più parole chiave.

Gli algoritmi di ricerca e di classificazione/catalogazione sono generalmente segreti.

Le fasi che realizzano la ricerca sul Web sono:

1. la raccolta e l'analisi;
2. la catalogazione;
3. la risposta.

La raccolta e l'analisi

Degli speciali programmi (chiamati *crawler* o *spider*), visitano quotidianamente i siti Internet seguendo i vari link ipertestuali, le pagine vengono analizzate ed una quantità enorme di dati, soprattutto di natura testuale, viene immagazzinata nel *database* del motore di ricerca.

La catalogazione

I “dati grezzi” così raccolti vengono organizzati dentro la base dati, sono così costituiti i cataloghi del motore di ricerca che verranno consultati nella fase successiva.

La risposta

Questo è il modulo di un motore di ricerca che noi tutti utilizziamo: in questa fase, il motore di ricerca propone un'interfaccia grafica, tipicamente una pagina web, che permette di sottoporre una richiesta di ricerca, inserendo una parola chiave, più parole oppure un'intera frase. Alla richiesta il motore di ricerca risponde proponendo un elenco di siti web, archiviati nel database dalle fasi precedenti, che contengono il testo cercato.

Visto l'enormità di informazioni contenute sul Web è importante formulare in modo opportuno la nostra richiesta (*query*),



Figura 25 – Il motore di ricerca Google

Ricerche avanzate

La pagina iniziale di Google riesce normalmente a soddisfare le nostre richieste purché le parole che inseriamo siano descrittive e nel minor numero possibile. Ad esempio, come suggerito dalla stessa guida di Google, piuttosto che utilizzare la ricerca [mi fa male la testa] è più redditizio impostare la ricerca con [mal di testa].

Sempre utilizzando la pagina iniziale è possibile rendere le ricerche più precise facendo uso di alcuni simboli speciali:

- le virgolette " ", per ricercare una frase esatta ad esempio ["le reti strutturate"];
- la direttiva `site`, permette di ricercare informazioni all'interno di un sito specifico, ad esempio ["Galileo Galilei" `site:wikipedia.org`];
- il simbolo `-` prima di una parola ci permetta di escludere le pagine contenenti la parola in questione, ad esempio [`software -commerciale`] dare come esito le pagine che contengono la parola `software` ma non la parola `commerciale`;
- l'operatore `OR`, nel caso ci siano due o più parole, inserirà nella risposte le pagine che contengano uno o l'altra parola, ad esempio [`nero d'avola 2004 OR 2005`].

Le opzioni che Google riesce a gestire permettono molto spesso di affinare le nostre ricerche, soprattutto in quei casi in cui ad un primo tentativo vengano visualizzate troppe pagine. Il motore di ricerca mette inoltre a disposizione una seconda pagina di ricerca, la quale riesce a gestire ricerche molto avanzate; ad essa si accede seguendo il link [Ricerca avanzata](#).

Google **Ricerca avanzata** [Suggerimenti per la ricerca](#) | [Tutto su Google](#)

Trova risultati

- che contengano **tutte** le seguenti parole
- che contengano la **seguente frase**
- che contengano **una qualunque** delle seguenti parole
- che **non contengano** le seguenti parole

Numero per pagina Visualizzazione in ogni pagina

Lingua Visualizza solo le pagine scritte in

Paese Cerca pagine situate in

Formato file i risultati in

Data Restituisci pagine web visitate

Cerca in Visualizza i risultati quando i termini selezionati appaiono

Domaino i risultati contenuti nel seguente sito o dominio Esempi: .org, google.com [Ulteriori Informazioni](#)

Diritti di utilizzo Trova risultati che

SafeSearch ☒ Nessun filtro ☐ Filtra con [SafeSearch](#)

Cerca pagina specifica

Pagine simili Cerca pagine simili alla pagina Esempio: www.google.com/help.html

Collegamenti Cerca pagine che contengano collegamenti alla seguente pagina

Ricerche su un argomento specifico

Novità! [Google Code Search](#) - Ricerca codici sorgente pubblici

©2011 Google

Figura 26 – La ricerca avanzata di Google

Approfondimento B: Classificazione degli indirizzi IP

Come abbiamo visto nell'ambito delle reti locali, ad ogni scheda di rete viene assegnato dal produttore un indirizzo *MAC* univoco, in questo modo possiamo identificare ogni scheda (e quindi ogni computer) presente in una rete locale. Nel caso di Internet, un insieme di reti e sottoreti interconnesse, possiamo pensare di identificare una macchina e la relativa sottorete facendo uso dell'indirizzo *MAC*? La risposta è no. L'indirizzamento delle schede di rete è detto "piatto", quindi non dà alcune informazioni sulla struttura della rete a cui l'host appartiene: tutti gli host sono posti allo *stesso livello*. Per risolvere questa situazione esistono gli *indirizzamenti gerarchici*, che permettono di identificare sia l'host e sia la rete o sottorete di appartenenza. I router sono i dispositivi delegati a "risolvere" gli indirizzi IP per instradare la comunicazione sotto forma di pacchetti.

Gli indirizzi IP sono convenzionalmente espressi in notazione decimale, composti da 4 gruppi di numeri compresi tra 0 e 255 (8 bit). Occorrono 32 bit per ogni indirizzo. Come vedremo questa suddivisione in gruppi, cosa che non avviene negli "indirizzamenti piatti", permetterà di identificare ogni host in una rete estesa come Internet.

Ecco il formato:

xxx	.	xxx	.	xxx	.	xxx
I gruppo		II gruppo		III gruppo		IV gruppo
8 bit		8 bit		8 bit		8 bit

Adesso che conosciamo il formato IP, studieremo le varie classi di indirizzi.

Classe A

Rete		Host			
xxx	.	xxx	.	xxx	xxx
I gruppo		II gruppo		III gruppo	IV gruppo
8 bit		8 bit		8 bit	8 bit

Il primo gruppo viene utilizzato per la rete, gli altri tre gruppi per l'host. In teoria avremmo 256 reti di classe A ed ognuna di esse con 16.777.216 host! In realtà il primo bit è sempre 0 e le reti possibili sono 128, con le seguenti eccezioni:

- gli indirizzi che iniziano con 0 (da 0.0.0.0 a 0.255.255.255);

- gli indirizzi che iniziano con 10 (da 10.0.0.0 a 10.255.255.255), IP privati;
- gli indirizzi che iniziano con 127 (da 127.0.0.0 a 127.255.255.255), IP di *loopback*.

Anche per gli host dobbiamo ricordare come non sia possibile usare indirizzi con tutti 0 o tutti 1, quindi sono 16.777.214 per ogni rete di classe A.

Classe B

Rete				Host			
xxx	.	xxx	.	xxx	.	xxx	
I gruppo		II gruppo		III gruppo		IV gruppo	
8 bit		8 bit		8 bit		8 bit	

Il primo e il secondo gruppo vengono utilizzati per la rete, gli altri due gruppi per l'host. In questa situazione avremo 14 bit per la rete, i primi due sono sempre 1 e 0, per un totale di 16.384 reti, con le seguenti eccezioni:

- gli indirizzi che iniziano con 169.254 (da 169.254.0.0 a 169.254.255.255);
- alcuni indirizzi privati (da 172.16.0.0 a 172.31.255.255).

Gli host per una rete di classe B scendono a 65.534.

Classe C

Rete						Host	
xxx	.	xxx	.	xxx	.	xxx	
I gruppo		II gruppo		III gruppo		IV gruppo	
8 bit		8 bit		8 bit		8 bit	

Il primi tre gruppi vengono utilizzati per la rete e l'ultimo per l'host. Con questa classe abbiamo 24 bit per la rete, di cui i primi tre sono sempre 1, 1 e 0, per un totale di 2.097.152 reti, con le seguenti eccezioni:

- gli indirizzi da 192.0.2.0 a 192.0.2.255;
- gli indirizzi da 192.88.99.0 a 192.88.99.255;
- gli indirizzi privati da 192.168.0.0 a 192.0.255.255;
- gli indirizzi da 198.18.0.0 a 198.19.255.255.

Gli host per una rete di classe C sono 254.

Concludiamo questo approfondimento con la tabella degli intervalli

Classe	Bit di intestazione	Inizio intervallo	Fine intervallo
Classe A	0	0.0.0.0	127.255.255.255
Classe B	10	128.0.0.0	191.255.255.255
Classe C	110	192.0.0.0	223.255.255.255
Classe D (Multicast)	1110	224.0.0.0	239.255.255.255
Classe E (Usi futuri)		240.0.0.0	255.255.255.255

Appendice A: Elenco protocolli

Elenco dei protocolli più utilizzati, a vari livelli, nelle reti di calcolatori.

Protocollo	Descrizione
TCP/IP	<i>Insieme di protocolli che permettono il funzionamento di Internet e lo scambio di dati tra i vari utenti. Le informazioni vengono segmentate in pacchetti, il protocollo cura la ritrasmissione dei pacchetti persi e il loro riordino.</i>
UDP/IP	<i>Contrariamente al protocollo TCP non gestisce i pacchetti persi o il loro riordino, per la sua rapidità viene utilizzato nello streaming video ed audio.</i>
HTTP	<i>Protocollo utilizzato per il trasferimento di un ipertesto</i>
HTTPS	Un protocollo crittografato utilizzato sul Web per trasferire i dati sensibili e riservati.
FTP	<i>Protocollo utilizzato per il trasferimento di file</i>
Telnet	<i>Protocollo utilizzato per il login remoto</i>
SMTP, POP e IMAP	<i>Protocolli utilizzati per i servizi di posta elettronica</i>

BOOK IN PROGRESS

“Il linguaggio HTML”

A cura del prof Massimo Mancino

Indice dei contenuti

1. Introduzione
2. Struttura e prime informazioni sul linguaggio
3. Evoluzione del linguaggio
4. Quali strumenti per sviluppare in HTML?
5. Un primo esempio
6. Struttura globale di una pagina
7. Formattazione del testo
8. Inseriamo le immagini
9. I collegamenti ipertestuali
10. Le tabelle, indispensabile strumento
11. Indicazioni generali per la realizzazione di un ipertesto
12. Approfondimento A: I moduli
13. Approfondimento B: I fogli di stile a cascata (CCS)
14. Esercizi
15. Appendice A: Tabella Html dei colori

Introduzione

Il linguaggio HTML (HyperText Mark-up Language) è lo strumento che ha permesso la nascita del Web (o WWW) e il suo sviluppo a livello mondiale, facendo diventare questo ambiente della rete Internet un potente mezzo di comunicazione.

Questo linguaggio, dotato di una specifica sintassi ed inizialmente creato a scopi di ricerca, consente la creazione di ipertesti multimediali ponendo in relazione, tramite dei collegamenti, informazioni riportate in documenti diversi ed archiviati in computer sparsi per il mondo.

Struttura e prime informazioni sul linguaggio

Il linguaggio è costituito da elementi sintattici di base detti *tag*. In generale possiamo dire che i tag permettono di strutturare un documento (o pagina html) definendone le varie sezioni e i relativi contenuti. Precisamente ogni tag ha un identificativo, per esempio `html`, `head`, `body`, `table`, ..., che viene racchiuso tra le parentesi `<` e `>`, ottenendo così `<html>`, `<head>`, `<body>`, `<table>`, ... Inoltre ogni sezione della pagina viene aperta dal relativo tag, esempio `<body>`, e viene terminata dal *tag di chiusura*, `</body>`.

Il codice html viene conservato in un file con estensione `htm` o `html`, come per esempio `home.html`. Ma come si osserva il risultato di un codice scritto in html? Semplicemente aprendo il relativo file con uno dei tanti browser oggi utilizzati (Chrome, MS Explorer, Firefox, Safari, ...); infatti, il *browser* si occuperà di interpretare il codice scritto nel file e visualizzare il risultato in una finestra (*rendering*). Se il file si trova in un altro computer della rete, come accade durante la navigazione Web in Internet, il browser lavorerà in modalità *HTTP*, un protocollo di comunicazione per il trasferimento degli ipertesti HTML che sfrutta l'*architettura client-server*.

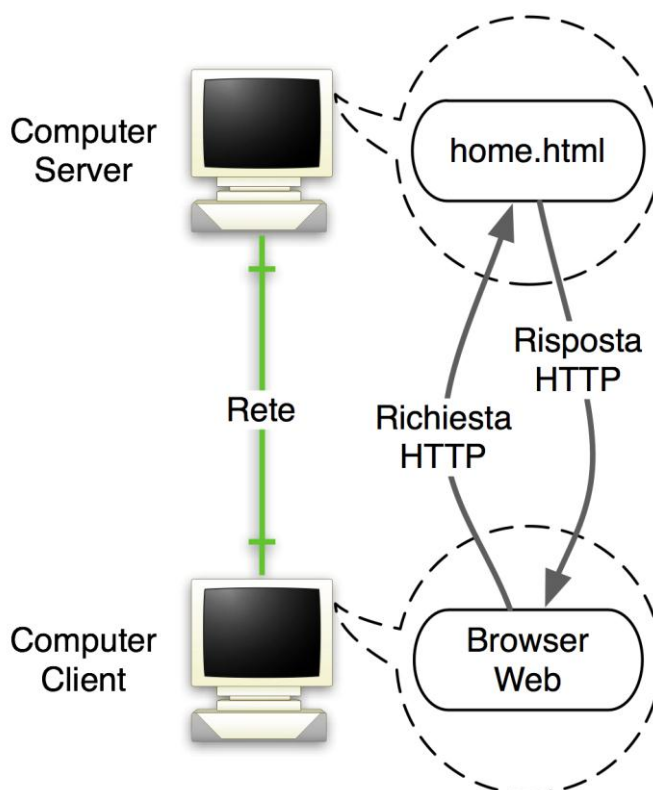


Figura 1 - Architettura client-server per il protocollo HTTP

Evoluzione del linguaggio

L'ideatore di HTML (Tim Berners-Lee, ricercatore del CERN) progettò nel 1989 questo linguaggio per collegare più documenti descrivendone allo stesso tempo sia l'aspetto grafico, come colori, caratteri tipografici, ecc., sia la sua struttura, intestazione, corpo, ecc.; successivamente, il consorzio W3C (www.w3c.org), l'ente che si occupa di mantenere lo standard, ha operato una scelta molto significativa: separare la struttura di una pagina dalla sua veste grafica. Così al linguaggio HTML è rimasto il compito di strutturare i contenuti mentre ad un nuovo linguaggio, il CSS (*Cascade Style Sheet*), si è dato il compito di gestire l'aspetto grafico. Con questa scelta la stessa pagina HTML apparirà completamente diversa, pur preservando gli stessi contenuti e la stessa struttura, se abbinata a fogli di stile diversi.

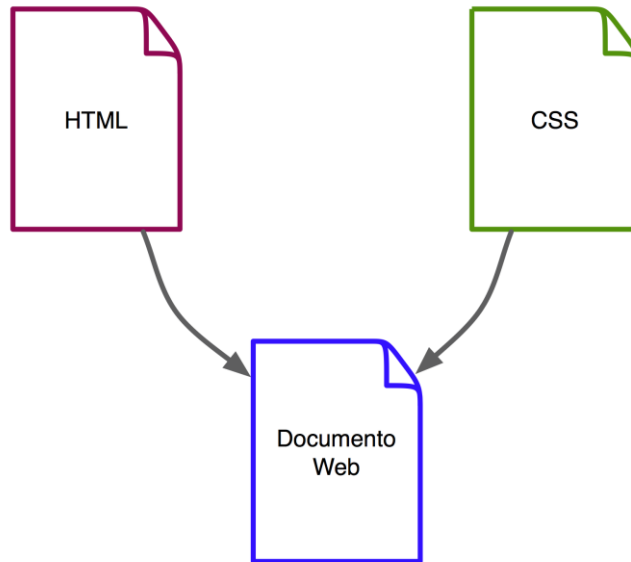


Figura 2 - HTML e CSS

Nei nostri appunti utilizzeremo un unico file dove scrivere il codice html specificando all'interno dei tag quale sarà lo stile che si vuole applicare (*metodo inline*), così ad esempio con la sintassi

```
<body style = "background-color: red";>
```

stiamo aprendo la sezione riguardante il *corpo* della pagina, ma stiamo anche specificando il *colore di sfondo*, ovvero rosso, una caratteristica che fa parte dello stile.

Quali strumenti per sviluppare in HTML?

Chi vuole oggi cimentarsi nella creazione di siti Web tramite questo linguaggio ha sostanzialmente due strade:

1. utilizzare un *editor di testo* e scrivere per ogni pagina il relativo codice HTML;
2. oppure, fare affidamento su dei *software di sviluppo* che utilizzino un'*interfaccia grafica*.

Entrambe le scelte hanno dei vantaggi e degli svantaggi, vediamo quali.

L'utilizzo di un editor di testo comporta una buona conoscenza dei vari tag HTML, quindi è necessario dedicare inizialmente parecchio tempo allo studio del linguaggio affidandosi ad un buon manuale. Inoltre, l'occorrenza di errori nel codice provocherà risultati non desiderati. D'altra parte avremo un controllo completo sul codice, che oggi è più facile realizzare e mantenere grazie agli editor di testo che riconoscono ed evidenziano gli elementi sintattici sfruttando una sintassi colorata

(*colouring syntax*).

Con la seconda scelta avremo dalla nostra la capacità di realizzare il nostro sito senza apprendere prima il linguaggio, ma utilizzando direttamente un'interfaccia grafica di tipo WYSIWYG (*What You See Is What You Get*), controllando i risultati finali nel momento stesso in cui si opera la realizzazione. Lo svantaggio sarà quello di perdere il controllo diretto sul codice, in quanto ci affideremo a quello che il software genererà per noi, situazione che a volte causerà la creazione di un HTML non sempre ottimizzato e mantenibile.

Quindi quale delle due strade scegliere? Dipende. Se, ad esempio, un titolare di una ditta di manifatture vuole realizzare il sito per la propria azienda, probabilmente non avrà il tempo né la voglia di imparare un linguaggio informatico ed opterà quindi per un software che gli semplifichi il lavoro e magari pubblichi automaticamente il sito. Viceversa, uno specialista informatico che vuole progettare e realizzare un sito può sì appoggiarsi ad un apposito programma di sviluppo, ma deve in ogni caso conoscere le specifiche del linguaggio per operare direttamente sul codice in quei casi in cui il software non riesce a generare i risultati attesi. In questi appunti faremo uso di un semplice editor di testo!

Un esempio di ipertesto in html

Completiamo questa introduzione fornendo un esempio costituito da due pagine html collegate fra di loro tramite un link. Supporremo di realizzare le pagine `home.html` e `contatti.html` rappresentate nell'ipertesto delle Figure 3 e 4. Per favorire la comprensione dei vari tag sono stati inclusi nel codice vari commenti.

Codice della pagina `home.html`:

```
<!-- Codice della pagine home.html -->

<!-- Tag di apertura -->
<html>

    <!-- Intestazione della pagina contenente il titolo -->
    <head>
        <title> Home page del negozio online </title>
    </head>

    <!-- Corpo della pagina -->
    <body>

        <!-- Viene realizzato un titolo di benvenuto centrato -->
        <div style="text-align: center;">
            <h2>Benvenuto nel sito del Negozio Online</h2>
        </div>

        <!-- Viene visualizzata un'immagine centrata -->
        <div style="text-align: center;">
            
        </div>

        <!-- Viene realizzato il link alla pagina contatti.html -->
        <div style="text-align: center;">
            <a href="contatti.html">Contatti</a>
        </div>

    </body>

</html>
```

Il risultato è questo:



Figura 3 - Visualizzazione della pagina home.html

Codice della pagina contatti.html:

```
<!-- Codice della pagine contatti.html -->

<!-- Tag di apertura -->
<html>

    <!-- Intestazione della pagina contenente il titolo -->
    <head>
        <title> Contatta il nostro Negozio Online </title>
    </head>

    <!-- Corpo della pagina -->
    <body>

        <!-- Vengono elencati i contatti -->
        <h3>I nostri contatti:</h3>
        Tel.: 05/1234567<br>
        Cel.: 355/1234567<br>

        <!-- Realizziamo un link per la casella di posta elettronica -->
        E-mail: <a href="mailto:info@negozio-online.com">
            info@negozio-online.com</a><br>

        <!-- Viene realizzato il link per ritornare alla pagina home -->
        <div style="text-align: center;">
            <a href="home.html">Home</a>
        </div>

    </body>

</html>
```

Il risultato è questo:



Figura 4 - Visualizzazione della pagina contatti.html

Struttura globale di una pagina

Con il primo esempio studieremo la struttura di una pagina Web, passando in rassegna i principali tag.

```
1. <html>
2.
3. <head>
4.   <title>Esempio 1 - Struttura della pagina</title>
5. </head>
6.
7. <body>
8.
9.   <div style="text-align: center;">
10.    <h2>Book in Progress</h2>
11.  </div>
12.
13.  <br>
14.
15.  <p>Benvenuto nel modulo HTML, questo è il primo esempio!</p>
16.
17. </body>
18.
19. </html>
```

Tabella 1 - Codice Esempio 1

Ecco il risultato:

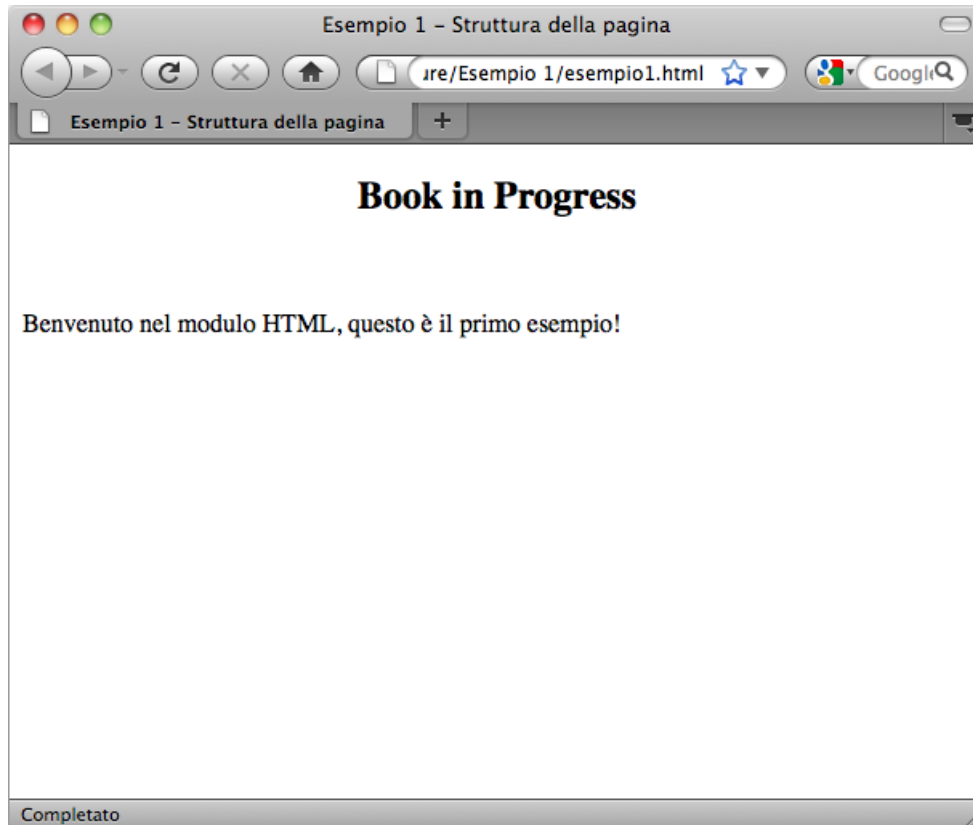


Figura 5 - Risultato Esempio 1

Il primo tag che incontriamo è il tag `<html>`, questo tag specifica l'inizio della pagina; il rispettivo tag di chiusura, `</html>`, terminerà il codice del nostro primo esempio (riga 19).

Una pagina viene divisa in due sezioni: *intestazione* e *corpo*. L'intestazione viene specificata con il tag `<head>` (riga 3), mentre il corpo con `<body>` (riga 7). Nell'intestazione viene messo il tag `<title>` (riga 4) per scegliere un titolo che comparirà nella barra del titolo della finestra del browser. Sempre nell'intestazione è possibile inserire alcuni programmi detti *script*, in *Javascript*, che arricchiscono di funzionalità il nostro documento.

Nel corpo ci sarà il vero e proprio contenuto della pagina. Con il tag `<body>` è possibile specificare, oltre che l'apertura del corpo della pagina, anche alcune caratteristiche visive utilizzando gli stili (attributo *style*). Ecco alcuni degli attributi utilizzabili con questo tag.

Attributi del tag <code><body></code>	Descrizione
---	-------------

<code>background-color: white</code>	Colore di sfondo
<code>background-image: url("sfondo.jpg")</code>	Immagine di sfondo
<code>color: black</code>	Colore del testo
<code>a.link: blue</code>	Colore dei collegamenti
<code>a.visited: purple</code>	Colore dei collegamenti già visitati

Continuando con l'esamina del codice troviamo alla riga 9 `<div style="text-align: center;">`; il tag `<div>` permette di specificare una o più caratteristiche di una sezione come ad esempio l'allineamento degli elementi (testo, immagini e tabelle), in questo caso il testo "Book in Progress" risulta essere centrato.

La scritta "Book in Progress" è specificata come *intestazione*, infatti è contenuta nel tag `<h2>`. I tag per creare un'intestazione sono 6: da `<h1>` ad `<h6>`. Anche per questi tag è previsto il tag `style` che potrebbe essere usato, per esempio, per avere un'intestazione colorata in blu:

```
<h2 style = "color: blue;">Book in Progress</h2>
```

Il documento termina con il testo "Benvenuto nel modulo HTML, questo è il primo esempio!" che è gestito come un paragrafo con il tag `<p>`. Come vedremo nel prossimo paragrafo questo tag è molto utile per gestire la formattazione del testo.

Formattazione del testo

Diversi sono i tag che HTML dedica alla formattazione del testo, il secondo esempio ne chiarirà l'utilizzo.

```

1.  <html>
2.
3.  <head>
4.    <title>Esempio 2 - Formattazione del testo</title>
5.  </head>
6.
7.  <body>
8.
9.    <p style="text-align: left;">Testo allineato a sinistra</p>
10.   <p style="text-align: right;">Testo allineato a destra</p>
11.   <p style="text-align: center;">Testo centrato</p>
12.   <p style="text-align: justify;">Testo giustificato</p>
13.
14.   <p>
15.     Puoi scrivere parole in
16.     <span style="font-weight: bold;">grassetto</span>,
17.     <span style="font-style: italic;">corsivo</span>
18.     e
19.     <span style="text-decoration: underline;">sottolineate</span>.
20.   </p>
21.
22.   <span style="color: red;">Puoi</span>
23.   <span style="color: blue;">anche</span>
24.   <span style="color: purple;">utilizzare</span>
25.   <span style="color: gray;">diversi</span>
26.   <span style="color: green;">colori</span>.
27.
28. </body>
29.
30. </html>

```

Tabella 2 - Codice Esempio 2

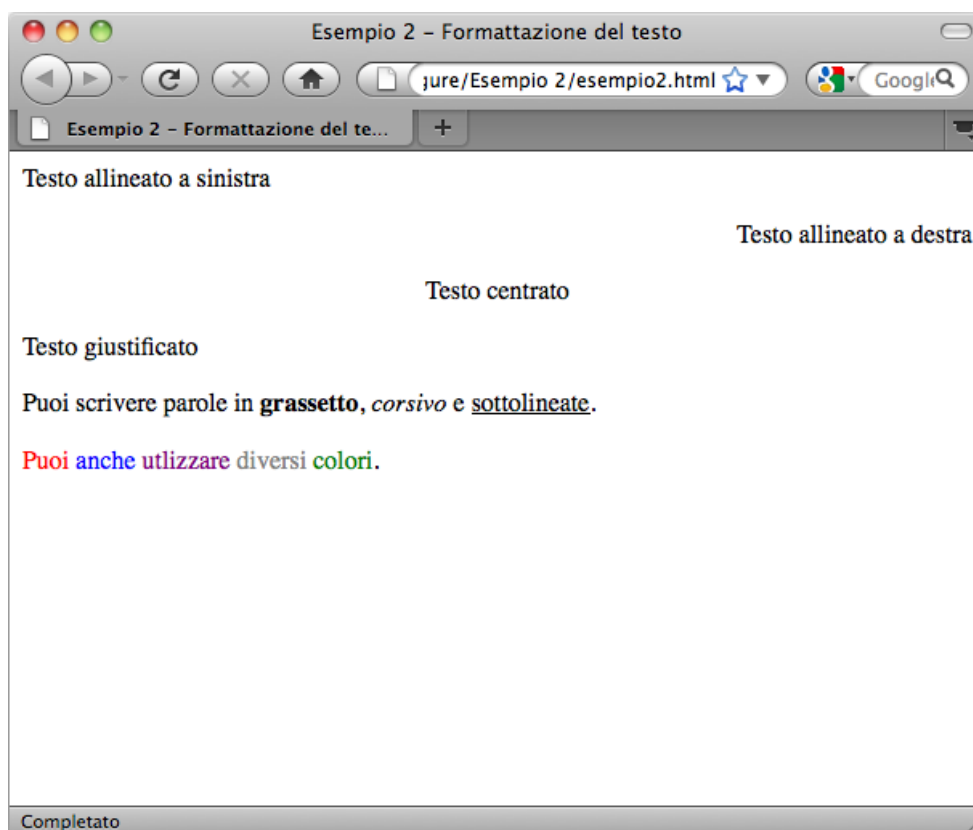


Figura 6 - Risultato Esempio 2

Come con i programmi di videoscrittura anche HTML mette in campo diversi metodi per formattare il testo, ancora una volta utilizzando i tag e le loro proprietà. Nella Fig. 6 osserviamo il risultato finale dell'esempio 2, dove:

- con le righe da 9 a 12, il testo di un paragrafo è stato allineato in diversi modi (a sinistra, a destra, al centro e giustificato);
- con le righe da 14 a 20, il tag `<span>` e gli attributi di stile `font-weight`, `font-style` e `text-decoration` hanno attribuito a determinate parole di un paragrafo gli stili grassetto, corsivo e sottolineato;
- ed infine con le righe da 22 a 26, le parole appaiono di diverso colore.

Continuando con la formattazione del testo prenderemo in esame un altro utile elemento: gli elenchi puntati (liste non ordinate) e numerati (liste ordinate).

```
1. <html>
2.
3. <head>
4.   <title>Esempio 3 - Elenchi puntati e numerati</title>
5. </head>
6.
7. <body>
8.
9.   Elenco puntato:<br>
10.  <ul>
11.    <li>voce 1;</li>
12.    <li>voce 2;</li>
13.    <li>voce 3.</li>
14.  </ul>
15.  <br>
16.
17.   Elenco numerato:<br>
18.  <ol>
19.    <li>voce 1;</li>
20.    <li>voce 2;</li>
21.    <li>voce 3.</li>
22.  </ol>
23.  <br>
24.
25. </body>
26.
27. </html>
```

Tabella 3 - Codice Esempio 3

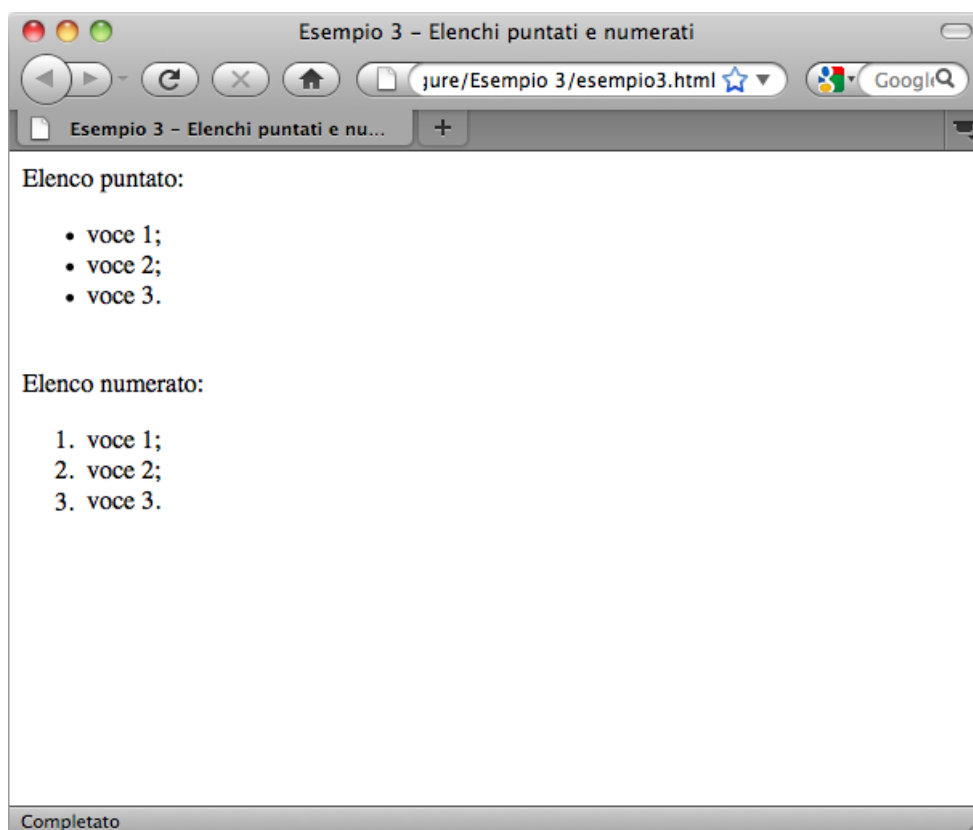


Figura 7 - Risultato Esempio 3

Gli elenchi puntati vengono definiti con il tag `<ul>` (*unordered list*) e le voci in elenco con il tag `<li>` (*list item*). Nella seconda parte dell'esempio con le righe 18-22 viene definito un elenco numerato con `<ol>` (*ordered list*).

Negli esempi fin qui presentati vi sarete accorti dell'occorrenza del tag `<br>`, ma a cosa serve? Andare a capo durante la scrittura del codice HTML non implica che nel documento visualizzato il testo o l'elemento successiva venga riportato in una nuova linea, per ottenere ciò è necessario specificare una nuova linea (andata a capo) con `<br>`.

Inseriamo le immagini

In questo paragrafo vedremo come far diventare il nostro ipertesto un ipertesto multimediale, imparando ad inserire le immagini. Ecco di seguito l'esempio 4.

```
1. <html>
2.
3. <head>
4.   <title>Esempio 4 - Le immagini</title>
5. </head>
6.
7. <body>
8.
9.   <div style="text-align: center;">
10.    <h2>Leonardo da Vinci (1452 - 1519)</h2>
11.    <img src = "leonardo.jpg" width = "50%" title = "Autoritratto
12.      1510-1515 circa" alt = "Autoritratto di Leonardo da Vinci">
13.
14.   </div>
15.
16. </body>
17.
18. </html>
```

Tabella 4 - Codice Esempio 4

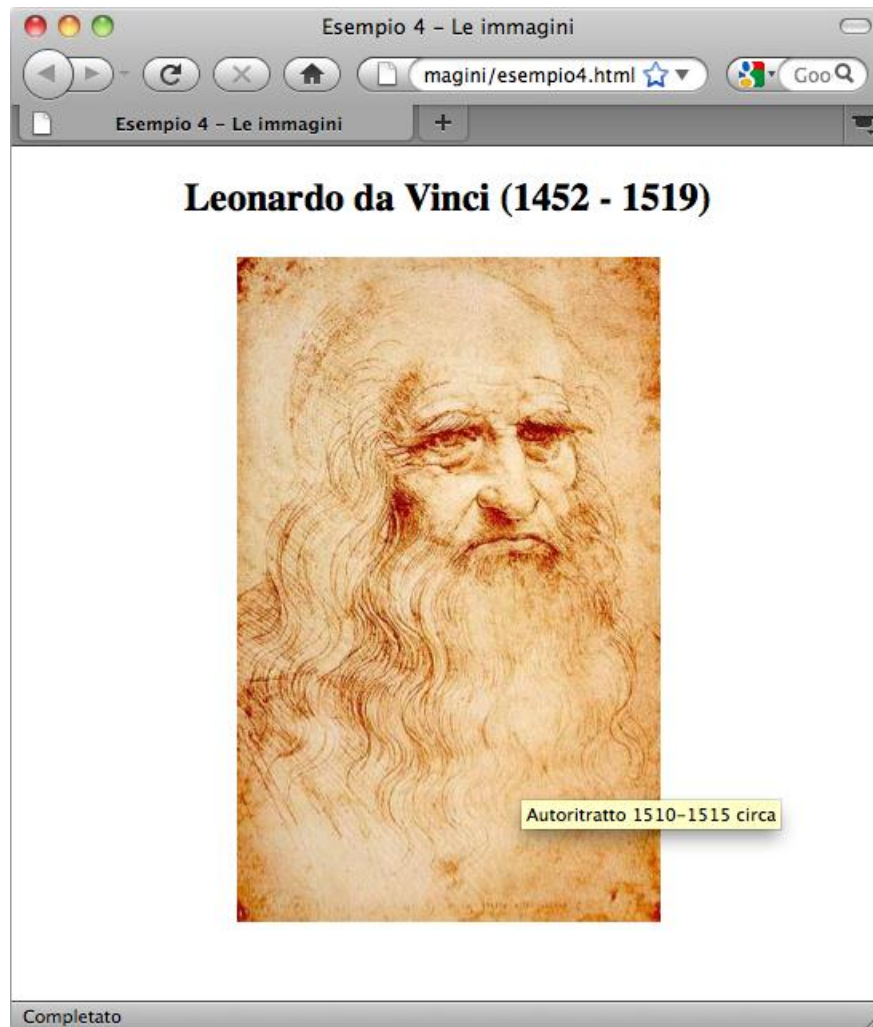


Figura 8 - Risultato Esempio 4

La riga 11 della Fig. 9 mostra il tag `<img>` (*image*) per l'inserimento dell'immagine "leonardo.jpg", gli attributi impostati sono i seguenti:

- `src = "leonardo.jpg"`, è il percorso del file che contiene l'immagine;
- `width = "50%"`, l'immagine viene caricata con una larghezza pari al 50% della larghezza finestra del browser;
- `title = "Autoritratto 1510-1515 circa"`, è il titolo dato all'immagine, come si può vedere nella Fig. 10 esso compare non appena il puntatore del mouse sosta sull'immagine;
- `alt = "Autoritratto di Leonardo da Vinci"`, è il testo alternativo che il browser presenterà in luogo dell'immagine qualora il relativo file fosse danneggiato o non più presente.

Formati riconosciuti per le immagini sono: JPEG, GIF e PNG.

I collegamenti ipertestuali

Le pagine realizzate fino a questo punto non possono essere definite ipertestuali, poiché manca l'elemento fondamentale, il *collegamento* o *link*. La fruizione di un documento ipertestuale passa attraverso i collegamenti tra una pagina ed un'altra.

L'esempio 5 illustra uno stesso collegamento (tra la nostra pagina e la home page di Google) realizzato in tre modi differenti.

```
1. <html>
2.
3. <head>
4.   <title>Esempio 5 - I Collegamenti</title>
5. </head>
6.
7. <body>
8.
9.   <a href = "http://www.google.it/">Clicca per Google</a>
10.  <br>
11.
12.   <a href = "http://www.google.it/" target = "_blank">
13.     Clicca per Google in una nuova finestra
14.   </a>
15.  <br>
16.
17.   <a href = "http://www.google.it/">
18.     <img src = "Google_logo.gif">
19.   </a>
20.
21. </body>
22.
23. </html>
```

Tabella 5 - Codice Esempio 5

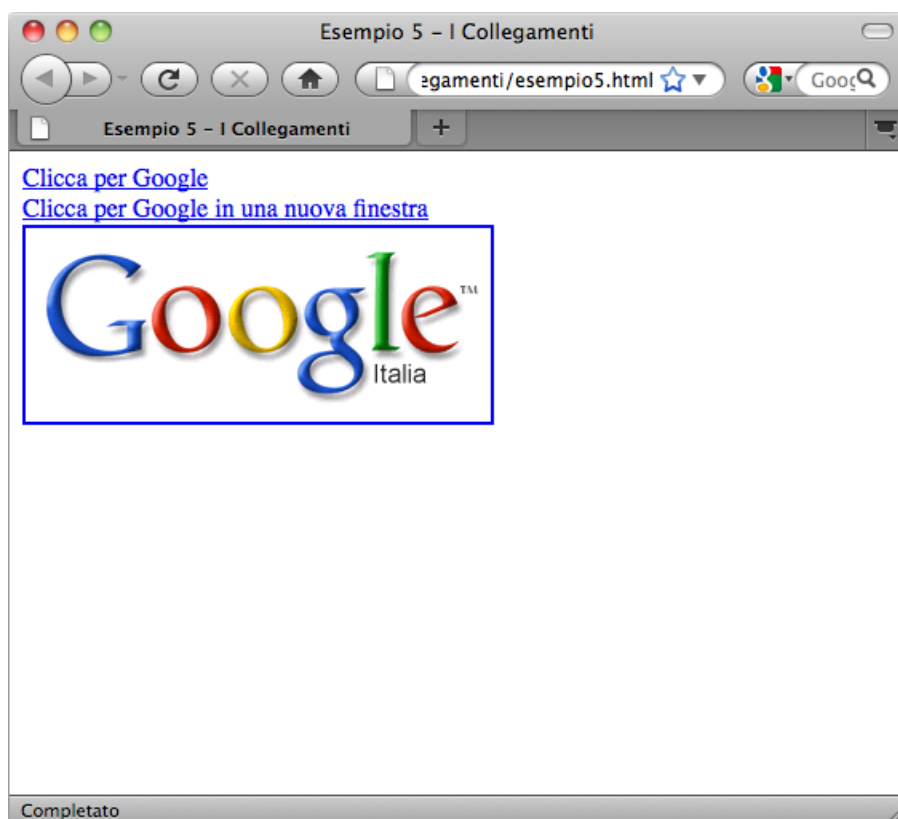


Figura 9 - Risultato Esempio 5

Le sintassi utilizzate per specificare il collegamento richiesto sono queste

```
9. <a href = "http://www.google.it/">Clicca per Google</a>
```

per ottenere un collegamento che “salti” dalla pagina dell’esempio 5 a Google cliccando sul testo “Clicca per Google”;

```
12. <a href = "http://www.google.it/" target = "_blank">
13.   Clicca per Google in una nuova finestra
14. </a>
```

per attivare un collegamento come nel caso precedente, con l’unica differenza che cliccando sul testo la pagina di Google verrà caricata su una nuova finestra e resterà, quindi, ancora disponibile la finestra con la pagina dell’esempio 5;

```
17. <a href = "http://www.google.it/">
18.   <img src = "Google_logo.gif">
```

19. `</a>`

In questo ultimo caso il collegamento viene eseguito cliccando sull'immagine di Google.

Le tabelle, un indispensabile strumento

Come una griglia ordinata formata da righe e colonne, allo stesso modo le tabelle HTML hanno il compito di facilitare la disposizione degli oggetti all'interno di caselle chiamate celle. Una tabella viene specificata indicando le righe con il tag `<tr>` (*table row*) e per ogni riga le appartenenti celle con `<td>` (*table data*). Le tabelle sono un potentissimo mezzo di disposizione (*layout*) in quanto permettono di definire innumerevoli caratteristiche: bordi, dimensioni, allineamenti, distanza tra le celle, colori di sfondo, immagini di sfondo e molto altro ancora. In questo paragrafo grazie agli esempi 6 e 7 impareremo ad utilizzarle.

```

1.  <html>
2.
3.  <head>
4.    <title>Esempio 6 - Le tabelle</title>
5.  </head>
6.
7.  <body>
8.
9.    <table border = "1" width = "100">
10.   <tr>
11.     <td>
12.       <h1>1</h1>
13.     </td>
14.     <td>
15.       <h1>2</h1>
16.     </td>
17.   </tr>
18.
19.   <tr>
20.     <td>
21.       <h1>3</h1>
22.     </td>
23.     <td>
24.       <h1>4</h1>
25.     </td>
26.   </tr>
27. </table>
28.
29.
30. </body>
31.
32. </html>

```

Tabella 6 - Codice Esempio 6

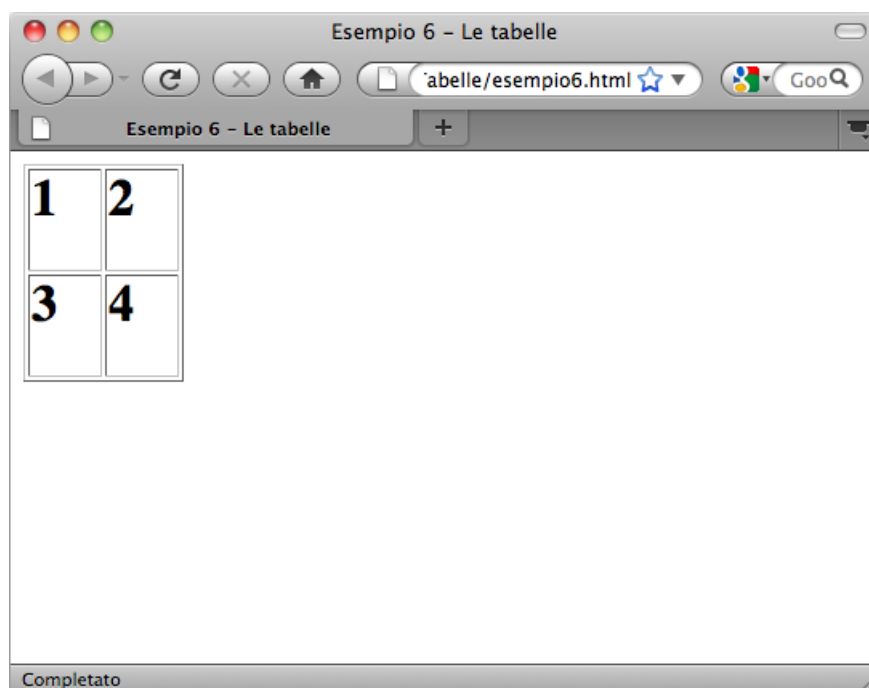


Figura 10 - Risultato Esempio 6

Con la riga 9 del codice inizia la dichiarazione di una tabella con bordi di dimensione 1 pixel e larghezza 100 pixel; le dimensioni, intese come numero di righe e numero di colonne, vengono specificate in seguito. Le righe sono due: la prima viene specificata con le righe 10-17, mentre la seconda con le righe 19-27. All'interno di ciascuna riga vengono indicate due celle con i tag `<td>`. Quindi in definitiva avremo una tabella 2x2.

```

1. <html>
2.
3. <head>
4.   <title>Esempio 7 - Le tabelle</title>
5. </head>
6.
7. <body>
8.
9.   <table border = "1" cellpadding = "5">
10.  <tr>
11.    <td align = "center" valign = "middle">
12.      <img src = "monnalisa.jpg" width = "50%">

```

```

13.     </td>
14.     <td>
15.         <span style="font-weight: bold;">Titolo</span>: La Gioconda (Monna Lisa)<br>
16.         <span style="font-weight: bold;">Autore</span>: Leonardo da Vinci<br>
17.         <span style="font-weight: bold;">Data</span>: 1503-1514 circa<br>
18.         <span style="font-weight: bold;">Tecnica</span>: Olio su tavola<br>
19.         <span style="font-weight: bold;">Dimensioni</span>: 77x53 cm<br>
20.         <span style="font-weight: bold;">Ubicazione</span>: Parigi, Museo del Louvre
21.     </td>
22. </tr>
23.
24. <tr>
25.     <td align = "center" valign = "middle">
26.         <img src = "vitruviano.jpg" width = "50%">
27.     </td>
28.     <td>
29.         <span style="font-weight: bold;">Titolo</span>: L'uomo vitruviano<br>
30.         <span style="font-weight: bold;">Autore</span>: Leonardo da Vinci<br>
31.         <span style="font-weight: bold;">Data</span>: 1490 circa<br>
32.         <span style="font-weight: bold;">Tecnica</span>: Matita e inchiostro su carta<br>
33.         <span style="font-weight: bold;">Dimensioni</span>: 34x24 cm<br>
34.         <span style="font-weight: bold;">Ubicazione</span>: Venezia, Galleria
dell'Accademia
35.     </td>
36. </tr>
37. </table>
38.
39. </body>
40.
41. </html>

```

Tabella 7 - Codice Esempio 7

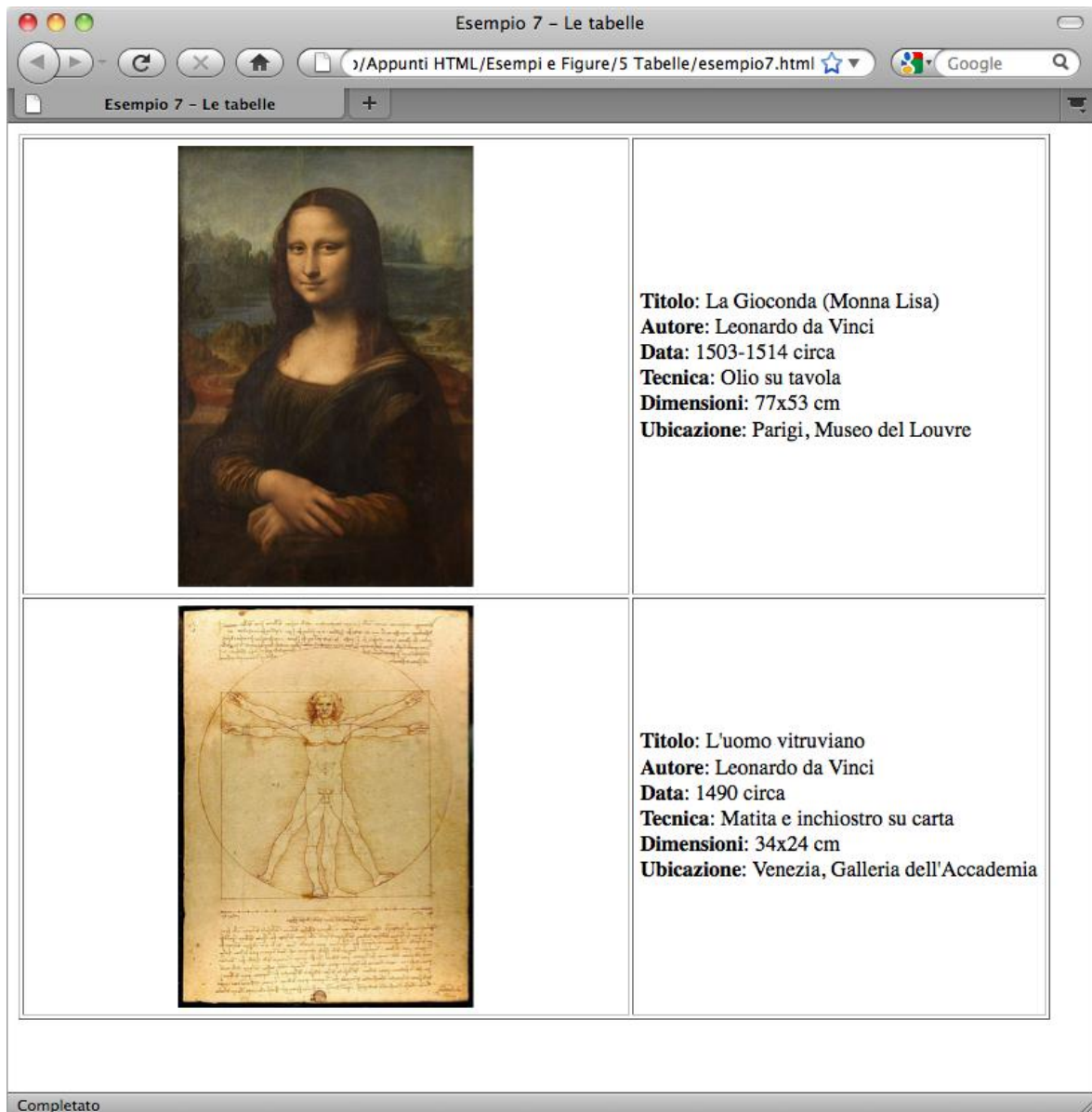


Figura 11 - Risultato Esempio 7

L'esempio 7 presenta una situazione più interessante nella quale una tabella, della stessa struttura dell'esempio precedente, è stata utilizzata per collocare due immagini e le relative descrizioni.

Indicazioni generali per la realizzazione di un ipertesto

Concludiamo questo modulo con dei consigli pratici che possono aiutare nella *realizzazione* di un buon ipertesto e facilitarne la *manutenzione*.

Progettazione

In quei casi in cui il documento ipertestuale è costituito da diverse pagine è opportuno creare prima uno schema su un foglio di carta per avere le idee chiare prima di cominciare la successiva fase di realizzazione.

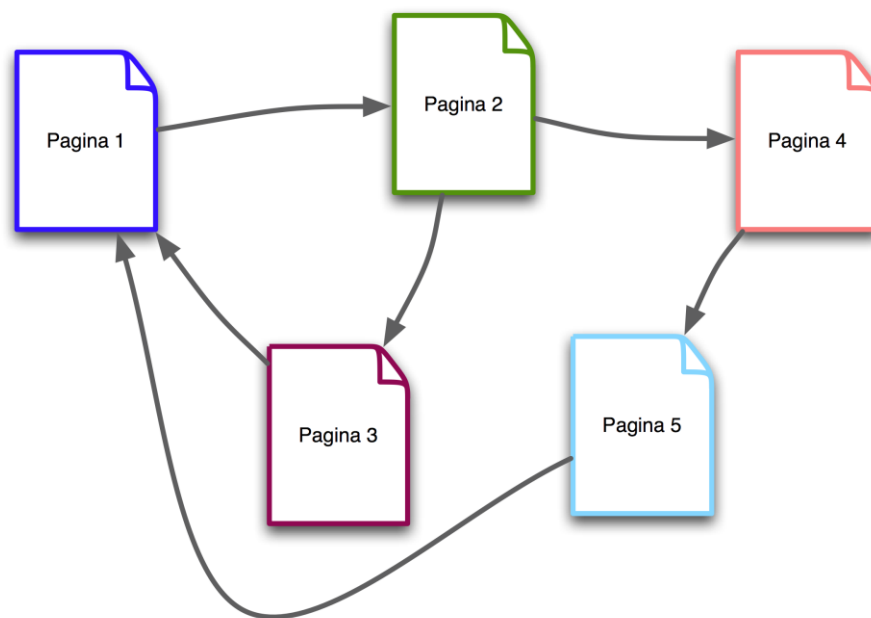


Figura 12 - Struttura di navigazione di un ipertesto

Editor di testo

Nel caso si decida di ricorrere ad un editor di testo è opportuno scegliere accuratamente quello più adatto al nostro caso. Caratteristiche da tenere presente sono:

- numerazione automatica progressiva delle linee di codice scritte;
- sintassi colorate dei tag;
- indentazione dei blocchi e sotto-blocchi per migliorare la struttura logica del codice.

I commenti

Il linguaggio HTML, come altri linguaggi informatici, permette di inserire dei *commenti* che verranno ignorati dal browser in fase di *rendering* della pagina, ma che sono utilissimi per rendere il

codice più leggibile. Non sono presenti nei nostri esempi solo per non distrarre il lettore dall'apprendimento degli altri tag in questione, ma in realtà andrebbero sempre usati.

```
<!-- Questo è un commento HTML -->
```

Utilizzo dei font

Se a prima vista l'utilizzo di font diversi può sembrare un'arma vincente per “abbellire” il nostro testo, bisogna tenere in considerazione che: 1) i font menu diffusi andrebbero evitati perché non presenti in alcuni sistemi operativi e quindi non riproducibili dal browser del client; 2) è importante preservare la leggibilità del testo, scegliendo font che permettano una lettura scorrevole.

Utilizzo dei colori

Come per i font anche per i colori valgono le stesse indicazioni, essi devono essere utilizzati in modo appropriato:

- non cambiare continuamente lo sfondo passando da una pagina all'altra;
- scegliere dei colori non troppo accesi che non stanchino la vista;
- scegliere dei colori che facilitino la lettura e stacchino lo sfondo dagli elementi del documento.

In appendice è presente una tabella con i colori più utilizzati: è possibile specificare il nome del colore o in alternativa il suo codice.

Struttura delle cartelle

La raccolta dei dati multimediali, come immagini, audio e filmati, deve avvenire tramite apposite cartelle (img, audio, movie) e la struttura delle cartelle deve essere ben chiara fin dall'inizio. Particolare attenzione bisogna dare alle specifiche dei link, in particolare all'attributo `href`: infatti se i nostri collegamenti funzionano benissimo quando proviamo l'ipertesto sul nostro computer, ciò potrebbe non accadere quando lo stesso ipertesto viene pubblicato sul Web oppure quando spostiamo la struttura in un'altra cartella. Una causa è molto spesso dovuta all'utilizzo di percorsi assoluti piuttosto che relativi, quindi i percorsi (*path*) dei file delle risorse come le immagini e i link dovrebbero essere espressi utilizzando percorsi relativi alla pagina in questione in modo tale da garantire il buon funzionamento dell'ipertesto a prescindere dalla sua posizione.

Il testo alternativo

Il tag `<img>` ha una proprietà denominata `alt`, la sua funzione è stata già descritta, ma è importante sapere che dichiarare un testo alternativo per ogni immagine aumenta il livello di accessibilità. Infatti, esistono software che lo riconoscono e tramite un sintetizzatore vocale facilitano la “lettura” della nostra pagina alle persone ipovedenti o con difficoltà di decodifica visiva.

Approfondimento A: I moduli

Quante volte ci è capitato di effettuare un accesso ad un sito come utente registrato? Pensiamo ad esempio al nostro provider di posta elettronica, al nostro conto in banca, al nostro social network preferito, ecc. Ed ancora, quante volte ci è capitato di compilare una scheda per effettuare un ordine on-line, oppure una registrazione con i nostri dati anagrafici? Tante. Alcune di queste azioni, come il login, vengono da noi eseguite quasi quotidianamente. Ma come si realizzano? Che strumenti mette a disposizione HTML?

In HTML un insieme di elementi che danno all'utente la possibilità di inserire dei dati in input, prende il nome di *modulo*; c'è un tag per raccogliere tutti questi elementi, ed è `<form>`. In questa tabella sono elencati gli attributi del tag:

Attributo	Descrizione
<code>name</code>	Il nome simbolico del modulo
<code>action</code>	L'azione che viene intrapresa quando il modulo viene inviato
<code>method</code>	Il metodo con cui è possibile sottoporre i dati (GET oppure POST)
<code>enctype</code>	Il tipo di codifica da utilizzare per inviare i dati

Prima di entrare nel vivo con il primo esempio, evidenziamo fin da subito alcuni aspetti dei moduli. La sola specifica HTML `<form>` non ci consente di implementare per intero il processo di acquisizione, invio ed elaborazione dei dati, infatti bisogna tenere conto di alcune cose:

- il tag `<form>` è sostanzialmente un raccoglitore “invisibile” ed è quindi necessario specificare altri tag, uno per ogni elemento di input desiderato;
- il controllo dei dati immessi in un modulo (come ad esempio la presenza dei dati obbligatori) non può essere fatto in HTML, ma in un linguaggio di *scripting* come *Javascript*;
- i dati inviati verso il server devono essere gestiti da un ulteriore script, tipicamente in *Php*, *Asp* o *Jsp*.

Esempio 1 – “Login”

Con il primo esempio simuleremo un modulo per l'autenticazione di un utente.

1.	<code><html></code>
2.	
3.	<code><head></code>

```

4.    <meta http-equiv = "content-type" content = "text/html;
      charset = utf-8">
5.    <meta http-equiv = "Content-Language" content = "it">

6.    <script type = "text/javascript">
7.        function data_control() {

8.            if (window.document.frmLogin.txtUsername.value == "") {
9.                alert("Il campo username è obbligatorio!");
10.               document.frmLogin.txtUsername.focus();
11.               return;
12.            }

13.
14.            document.frmLogin.submit();
15.
16.        }
17.    </script>
18.
19.    <title>Login</title>
20. </head>
21.
22. <body>
23.     <div style="text-align: center;">Login</div>
24.     <table style="text-align: left; margin-left: auto;
      margin-right: auto;"
      border = "1" cellpadding = "10" cellspacing = "0">
25.     <tbody>
26.     <tr>
27.         <td><form action = "login_control.php" method = "post"
      name = "frmLogin">
28.             <table border = "0">
29.             <tbody>
30.             <tr>
31.                 <td style="text-align: right;">Username:</td>
32.                 <td><input type = "text" name = "txtUsername"
      size = "24"></td>
33.             </tr>
34.             <tr>
35.                 <td style="text-align: center;">Password:</td>
36.                 <td><input type = "password" name = "txtPassword"
      size = "24"></td>
37.             </tr>
38.             <tr>
39.                 <td style="text-align: center;">Language:</td>
40.                 <td><select size = "1" name = "menuLang">

```

```

41.         <option value = "en" selected = "selected">
42.             english
43.         </option>
44.         <option value = "it">italiano</option>
45.     </select></td>
46. </tr>
47. </tbody>
48. </table>
49. </td>
50. </tr>
51. </tbody>
52. </table>
53.
54. <table style="text-align: left; margin-left: auto;
55.     margin-right: auto;" border = "0" cellpadding = "5"
56.     cellspacing = "0">
57. <tbody>
58. <tr>
59.     <td style = "text-align: center;">
60.         <input type = "reset" value = "Reset">
61.     </td>
62.     <td style = "text-align: center;">
63.         <input type = "button" value = "Login"
64.             onclick = "javascript:data_control();">
65.     </td>
66. </tr>
67. </tbody>
68. </table>
69. </form>
70. </body>
</html>

```

Figura 13 - Codice modulo "Login"

Ecco il risultato:

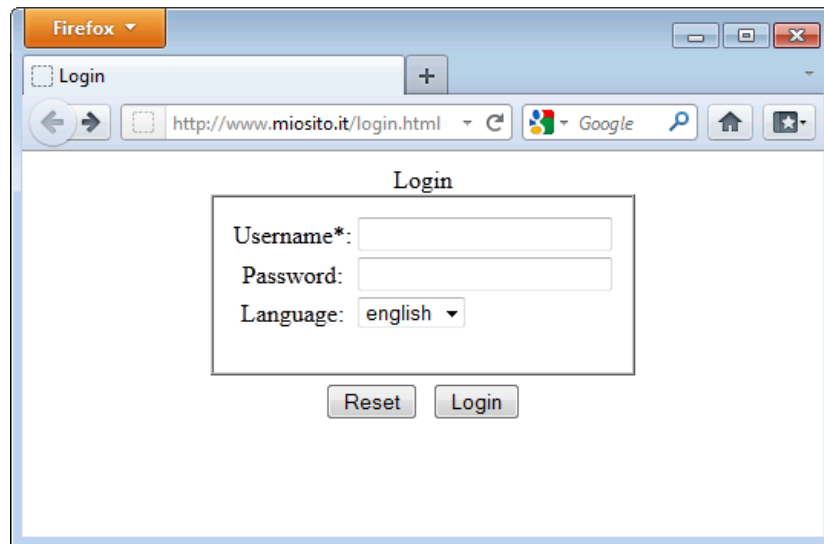


Figura 14 - Modulo "Login"

Analizziamo in dettaglio il codice, cominciando proprio con il tag `<form>` della riga 27

```
27.      <td><form action = "login_control.php" method = "post"
           name = "frmLogin">
```

L'attributo `action` definisce quale script sul server elaborerà i dati inviati dal client (in questo caso lo script `login_control.php` riceverà la username, la password, e la lingua scelta). Il secondo attributo, `method`, associa un metodo per il trasferimento dei dati, in questo caso `POST`. Le alternative sono sostanzialmente due, `GET` e `POST`:

Metodo	Caratteristiche
GET	– pochi parametri; – parametri di piccole dimensioni; – parametri solo di tipo testuale; – i parametri vengono inviati con l'indirizzo <code>url</code>
POST	– molti parametri; – parametri di grandi dimensioni; – parametri anche di tipo non testuale (files, immagini, ...); – i parametri non vengono inviati con l'indirizzo <code>url</code>

Uno degli attributi fondamentali da utilizzare, non solo per il tag `<form>`, ma anche per gli elementi in esso contenuti è `name`, questo perché è necessario avere successivamente un riferimento agli elementi del modulo; in questo caso sono cinque, tre per l'input e due pulsanti.

Ecco in dettaglio il codice HTML.

Un campo per la username

```
32.      <td><input type = "text" name = "txtUsername"
           size = "24"></td>
```

identificato dal tipo `text`, ha larghezza 24 e nome `txtUsername`.

Un campo per la password

```
36.      <td><input type = "password" name = "txtPassword"
           size = "24"></td>
```

identificato dal tipo `password`, ha larghezza 24 e nome `txtPassword`. Ogni carattere digitato in questo caso viene nascosto da * o •.

Per questi due tipi di elementi è possibile stabilire una dimensione massima di input tramite l'attributo `maxlength`.

Il *popup menu* per la scelta della lingua

```
40.      <td><select size = "1" name = "menuLang">
41.          <option value = "en" selected = "selected">
42.              english
43.          </option>
44.          <option value = "it">italiano</option>
45.      </select></td>
```

dove sono presenti due voci "english" e "italiano" realizzate tramite il tag `<option>`, la prima è quella selezionata inizialmente.

Il pulsante di reset per azzerare il modulo

```
58.      <input type = "reset" value = "Reset">
```

Il pulsante "Login" attiva lo script `data_control`

```
61.      <input type = "button" value = "Login"
           onclick = "javascript:data_control();">
```

Questo script definito nell'intestazione della pagina utilizza una funzione (le righe dalla 7 alla 16) per controllare che almeno sia inserita una username prima della conferma del modulo. Il modulo viene inviato al server con il comando della riga 14; è importante notare come questo script non potrebbe interagire con il modulo senza un nome identificativo per ogni elemento.

Esempio 2 – “Tutti in palestra!”

```

1. <html>
2.
3. <head>
4.
5.     <meta http-equiv="content-type" content="text/html;
        charset = utf-8">
6.     <meta http-equiv="Content-Language" content="it">
7.
8.     <script type="text/javascript">
9.         function data_control() {
10.
11.             if (document.frmRegistrazione.txtNome.value == "") {
12.                 alert("Il campo nome è obbligatorio!");
13.                 document.frmRegistrazione.txtNome.focus();
14.                 return;
15.             }
16.
17.             if (document.frmRegistrazione.txtCognome.value == "") {
18.                 alert("Il campo cognome è obbligatorio!");
19.                 document.frmRegistrazione.txtCognome.focus();
20.                 return;
21.             }
22.
23.             if (document.frmRegistrazione.txtCodFiscale.value == "")
24.             {
25.                 alert("Il campo codice fiscale è obbligatorio!");
26.                 document.frmRegistrazione.txtCodFiscale.focus();
27.                 return;
28.             }
29.
30.             if (document.frmRegistrazione.cBtnAerobica.checked == false &&
31.                 document.frmRegistrazione.cBtnBodyBuilding.checked==false &&
32.                 document.frmRegistrazione.cBtnSpinByke.checked == false &&
33.                 document.frmRegistrazione.cBtnStep.checked == false &&
34.                 document.frmRegistrazione.cBtnCorpoLibero.checked==false) {

```

```

35.         alert("Devi scegliere almeno un'attività!");
36.         return;
37.     }
38.
39.     document.frmRegistrazione.submit();
40.
41. }
42. </script>
43.
44. <title>Registrazione</title>
45. </head>
46.
47. <body>
48.
49.     <div style="text-align: center;">Scheda di registrazione</div>
50.
51.     <form action="login_control.php" method="post"
52.         name="frmRegistrazione">
53.         <table style="text-align: left; margin-left: auto;
54.             margin-right: auto;" border="1" cellpadding="10"
55.             cellspacing="0">
56.             <tbody>
57.             <tr>
58.                 <td>
59.                     <table border="0">
60.                         <tbody>
61.                         <tr>
62.                             <td style="text-align: right;">Cognome*:</td>
63.                             <td><input maxlength="30" name="txtNome" size="24"></td>
64.                         </tr>
65.                         <tr>
66.                             <td style="text-align: right;">Nome*:</td>
67.                             <td><input maxlength="30" name="txtCognome"
68.                                 size="24"></td>
69.                         </tr>
70.                         <tr>
71.                             <td style="text-align: right;">Cod. Fiscale*:</td>
72.                             <td><input maxlength="16" name="txtCodFiscale"
73.                                 size="24"></td>
74.                         </tr>
75.                         <tr>
76.                             <td style="vertical-align: top;
77.                                 text-align: right;">Abbonamento:</td>
78.                             <td style="vertical-align: top;">
79.                                 <input name="rBtnAbbonamento" checked="checked"

```

```

value="1" type="radio">Mensile<br>
75.         <input name="rBtnAbbonamento" value="2"
            type="radio">Trimestrale<br>
76.         <input name="rBtnAbbonamento" value="3"
            type="radio">Semestrale<br>
77.         <input name="rBtnAbbonamento" value="4"
            type="radio">Annuale
78.     </td>
79. </tr>
80. <tr>
81.     <td style="vertical-align: top;
            text-align: right;">Attività:</td>
82.     <td style="vertical-align: top;">
83.         <input value="1" name="cBtnAerobica"
            type="checkbox">Aerobica<br>
84.         <input value="1" name="cBtnBodyBuilding"
            type="checkbox">Body building<br>
85.         <input value="1" name="cBtnSpinByke"
            type="checkbox">Spin byke<br>
86.         <input value="1" name="cBtnStep"
            type="checkbox">Step<br>
87.         <input value="1" name="cBtnCorpoLibero"
            type="checkbox">Corpo libero
88.     </td>
89. </tr>
90. </tbody>
91. </table>
92.
93. </td>
94. </tr>
95. </tbody>
96. </table>
97.
98. <table style="text-align: left; margin-left: auto;
            margin-right: auto;" border="0" cellpadding="5"
            cellspacing="0">
99.     <tbody>
100.    <tr>
101.        <td style="text-align: center;"><input name="azzer"
            value="Reset" type="reset"></td>
102.        <td style="text-align: center;">
103.            <input name="registra" value="Registra"
            onclick="javascript:data_control();"
            type="button"></td>
104.    </tr>

```

```

105.         </tbody>
106.     </table>
107. </form>
108.
109. </body>
110. </html>

```

Tabella 8 - Codice esempio "Tutti in palestra!"

Figura 15 - Risultato esempio "Tutti in palestra!"

Vediamo in dettaglio i nuovi elementi di input introdotti in questo secondo esempio. Il primo è il *radio button*, un controllo che realizza una scelta esclusiva tra più opzioni. La condizione necessaria per questo comportamento è la scelta del nome: deve essere uguale per tutti gli elementi che appartengono allo stesso gruppo di scelta. Nel nostro esempio l'utente potrà scegliere, in fase di registrazione, la durata dell'abbonamento: mensile, trimestrale, semestrale o annuale.

```

74.         <input name="rBtnAbbonamento" checked="checked"
75.             value="1" type="radio">Mensile<br>
76.         <input name="rBtnAbbonamento" value="2"
77.             type="radio">Trimestrale<br>
78.         <input name="rBtnAbbonamento" value="3"
79.             type="radio">Semestrale<br>
80.         <input name="rBtnAbbonamento" value="4"
81.             type="radio">Annuale<br>

```

```
type="radio">Annuale
```

Il secondo controllo da approfondire è il *checkbox*, un controllo per la scelta di più opzioni, questa volta in modo non esclusivo. Ecco il codice:

```
83. <input value="1" name="cBtnAerobica"
    type="checkbox">Aerobica<br>
84. <input value="1" name="cBtnBodyBuilding"
    type="checkbox">Body building<br>
85. <input value="1" name="cBtnSpinByke"
    type="checkbox">Spin byke<br>
86. <input value="1" name="cBtnStep"
    type="checkbox">Step<br>
87. <input value="1" name="cBtnCorpoLibero"
    type="checkbox">Corpo libero
```

Si può notare come non sia necessario avere un unico nome per le caselle di spunta in questione, dato che un cliente può eventualmente scegliere più di un'attività da svolgere in palestra. Infine, essendo necessario scegliere almeno un'attività per completare la registrazione, lo script `data_control` gestisce questo controllo con le righe 30-37.

Elementi di un modulo

Completiamo questo approfondimento elencando gli elementi più utilizzati in un modulo.

Elemento	Descrizione	Attributi principali
text	Casella di testo	name, size, maxlength e value
password	Casella di testo che nasconde l'input	name, size, maxlength e value
checkbox	Casella di selezione	name, checked e value
radio	Gruppo di pulsanti per una selezione esclusiva	name, checked e value
file	Casella di input per la selezione di un file	name
hidden	Campo nascosto di ausilio per gli script	name, value

submit	Pulsante per confermare il modulo e inviarlo	name e value
reset	Pulsante per cancellare i valori inseriti nel modulo	name e value

Approfondimento B: I fogli di stile a cascata (CSS)

La figura 2 riguardante il paragrafo “*Evoluzione del linguaggio*”, mostra come sia possibile dividere i contenuti dalla veste grafica del documento Web. In questo approfondimento introdurremo il CSS (*Cascade Style Sheet*), un potente strumento che permette di definire dettagliatamente ogni proprietà grafica degli elementi html.

Per “agganciare” le specifiche CSS alla nostra pagina Html abbiamo sostanzialmente tre modi:

1. Fogli collegati (esterni);
2. Fogli incorporati (interni);
3. Fogli in linea.

Diciamo subito che con tutti e tre gli approcci si possono ottenere gli stessi risultati. Ma allora, quale scegliere? E’ necessario pianificare prima su carta il nostro sito; infatti immaginando di voler realizzare un sito con molte pagine, è utile isolare le caratteristiche grafiche comuni, definendole una volta sola in un file esterno CSS. Invece, per le proprietà specifiche di un pagina, o persino per un solo elemento Html, vanno specificate con i fogli incorporati o quelli in linea. Anche in questo caso studieremo la situazione con un esempio realizzato con tre file: `style.css`, `dipinti.html` e `ultima_cena.html`.

```
1.  .background {
2.      background-color: lightyellow;
3.  }
4.
5.  .testointestazione {
6.      font: 20px arial;
7.      color: black;
8.      text-align: center;
9.  }
10.
11. .testobase {
12.     font: 14px arial;
13.     color: gray;
14. }
15.
16. .box {
17.     border-color: lightgray;
18.     margin-top: 50px;
19.     margin-right: auto;
20.     margin-left: auto;
```

```

21.     margin-bottom: auto;
22. }

```

Tabella 9 - Codice file "style.css"

```

1.  <html>
2.
3.  </head>
4.    <meta http-equiv = "content-type" content = "text/html;
      charset = utf-8">
5.    <meta http-equiv = "Content-Language" content = "it">
6.    <link rel = "stylesheet" type = "text/css" href = "style.css">
7.    <title>Dipinti</title>
8.  </head>
9.
10. <body class = "background">
11.   <p class = "testointestazione">Seleziona un dipinto</p>
12.
13.   <table class = "box" border = "1" cellpadding = "10"
      cellspacing = "0">
14.     <tbody>
15.       <tr>
16.         <td>
17.           <a href = "ultima_cena.html">
18.             <img src = "The_Last_Supper.jpg" width = "250px">
19.           </a>
20.           <p class = "testobase" style = "text-align: center;">
              L'ultima cena</p>
21.         </td>
22.         <td>
23.           <a href = "madonna_dei_fusi.html">
24.             <img src = "Madonna_of_the_Yarnwinder.jpg" width = "150px">
25.           </a>
26.           <p class = "testobase" style = "text-align: center;">
              Madonna dei Fusi</p>
27.         </td>
28.       </tr>
29.     </tbody>
30.   </table>
31. </body>
32.
33. </html>

```

Tabella 10 - Codice file "dipinti.html"

```

1. <html>
2.
3. </head>
4.   <meta http-equiv = "content-type" content = "text/html;
      charset = utf-8">
5.   <meta http-equiv = "Content-Language" content = "it">
6.   <link rel = "stylesheet" type = "text/css" href = "style.css">
7.
8.   <style type = "text/css">
9.     .bg_black {
10.       background: black;
11.     }
12.   </style>
13.
14.   <title>L'ultima cena</title>
15. </head>
16.
17. <body class = "background">
18.   <p class = "testointestazione">L'ultima cena</p>
19.
20.   <table class = "box" border = "1" cellpadding = "10"
      cellspacing = "0">
21.     <tbody>
22.       <tr>
23.         <td class = "bg_black">
24.           <img src = "The_Last_Supper.jpg" width = "400px">
25.         </td>
26.       </tr>
27.       <tr>
28.         <td>
29.           <p class = "testobase" style = "text-align: center;">
              L'ultima cena</p>
30.           <p class = "testobase">
              Autore: Leonardo da Vinci<br>
31.              Anno: 1945 - 1948<br>
32.              Ubicazione: Santa Maria delle Grazie, Milano (Italia)
33.            </p>
34.          </td>
35.        </tr>
36.      </tbody>
37.    </table>
38.
39.
40. </body>

```

```
41.
42. </html>
```

Tabella 11 - Codice file "ultima_cena.html"

Questo il risultato delle due pagine:



Figura 16 - Risultato della pagina "dipinti.html"



Figura 17 - Risultato della pagina "ultima_cena.html"

La prima cosa da tenere in considerazione è l'insieme delle proprietà grafiche comuni alle due pagine:

- Il colore di sfondo;
- Il formato del testo per l'intestazione della pagina;
- Il formato del testo per la didascalia;
- La posizione delle tabelle rispetto all'intera pagina.

Come anticipato queste proprietà sono state definite una volta sola nel file `style.css`, più precisamente:

Proprietà grafica	Linee nel file <code>style.css</code>
Colore di sfondo	1-3
Formato testo intestazione della pagina	5-9
Formato del testo per la didascalia	11-14
Posizione delle tabelle	16-22

Una volta definite, queste caratteristiche possono essere utilizzate all'interno di ogni pagina (vedi riga 6 nei due file `.html`). Fatto questo è possibile richiamare queste definizioni tramite il selettore `class`.

Per il file `dipinti.html` abbiamo:

- nella riga 10 la direttiva `class = "background"`, fa sì che il colore di sfondo della pagina sia quello definito nel foglio di stile;
- nella riga 11, `class = "testointestazione"` associa al paragrafo la formattazione del testo con `font: 20px arial, color: black e text-align: center;`
- nella riga 13, `class = "box"` permette di posizionare la tabella al centro e con distanza 50 pixel dall'elemento che sta sopra di essa;
- infine nella riga 20, `class = "testobase"` definisce `font: 14px, arial e color: gray` per il testo generico della pagina.

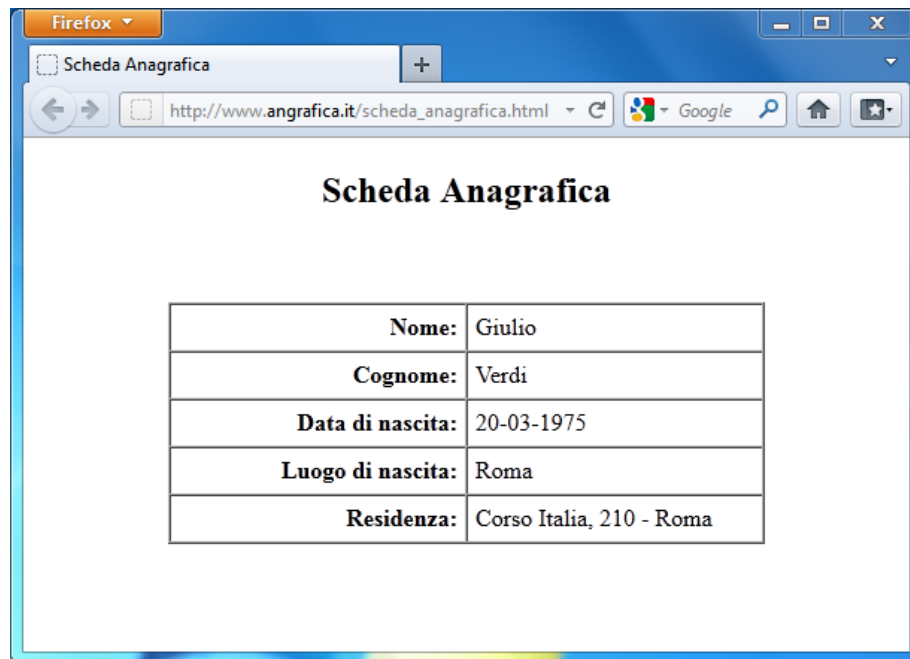
La peculiarità dei `css` è quella di poterli utilizzare "in cascata", se osserviamo la riga 20 del file possiamo notare come le direttive `css` siano due: la prima è quella relativa al `testobase` del foglio esterno, la quale però non dà alcuna indicazione sull'allineamento del testo; la seconda, completa la formattazione dando un allineamento centrato. Quale sarebbe stato il comportamento del browser se anche nella classe `testobase` del foglio di stile esterno fosse definito un allineamento, ad esempio a sinistra? Il risultato sarebbe rimasto invariato; questo perché le direttive "in line" hanno la precedenza su quelle dei fogli di stile incorporati, che a loro volta hanno la precedenza sui fogli di stile esterni.

Resta da esaminare il secondo file HTML: `ultima_cena.html`. La parte più interessante è contenuta nelle righe 8-11, che illustrano come definire un foglio di stile incorporato o interno alla pagina. A differenza da quello esterno le specifiche contenute in questo tipo di foglio sono utilizzabili solo nella stessa pagina. Con la riga 23 la classe `bg_black` viene richiamata e la cella contenente l'immagine del dipinto di Leonardo da Vinci avrà uno sfondo nero.

Esercizi

Esercizio 1

Utilizzando una tabella e il corretto allineamento del testo scrivi il codice HTML per la seguente pagina:



Esercizio 2

Si vogliono realizzare due pagine Html, con le seguenti caratteristiche

File HTML	Descrizione
home.html	– Titolo: "Home Page"; – Contenuto: Una breve presentazione di un'azienda locale con una immagine rappresentativa; – Un collegamento "Dove siamo" al file dove_siamo.html.
dove_siamo.html	– Titolo: "Dove siamo"; – Contenuto: Una scheda sintetica realizzata tramite tabella con i indirizzo,

	numero civico, cap, comune e provincia; – Un collegamento “Home” al file home.html.
--	---

Esercizio 3

In molti siti Web aziendali è presente un menù di scelta con le seguenti voci: “la nostra azienda”, “dove siamo”, “contatti”, “catalogo”. Dopo aver scelto un’immagine per ognuna di queste voci, utilizza un tabella per organizzare gli elementi ed attiva i rispettivi link.

Esercizio 4

Nel sito di una libreria si vuole avere a disposizione un modulo per inserire un nuovo libro. Tenendo in considerazione i seguenti generi: Avventura, Biografico, Fantasy, Giallo, Romantico, Horror, Poliziesco e Storico, scrivi il codice Html.

The screenshot shows a Firefox browser window with the address bar displaying `http://www.libreria.it/inserimento_libro.html`. The page title is "Inserimento nuovo libro". The main content area is titled "Scheda di inserimento nuovo libro" and contains a form with the following fields:

Codice:	
Titolo:	
Autore:	
Genere:	Avventura
Prezzo:	
Editore:	
Già in magazzino:	☒ Sì ☐ No

At the bottom of the form are two buttons: "Azzera" and "Inserisci".

Progettare e realizzare un secondo modulo per la ricerca in archivio di un libro.

Appendice A: Tabella Html dei colori

	#A0CE00	ALICEBLUE		#FAEBD7	ANTIQUEWHITE
	#00FFFF	AQUA		#7FFFD4	AQUAMARINE
	#F0FFFF	AZURE		#F5F5DC	BEIGE
	#FFE4C4	BISQUE		#000000	BLACK
	#FFEB3D	BLANCHEDALMOND		#0000FF	BLUE
	#8A2BE2	BLUEVIOLET		#A52A2A	BROWN
	#DEB887	BURLYWOOD		#5F9EA0	CADETBLEUE
	#7FFF00	CHARTREUSE		#D2691E	CHOCOLATE
	#FF7F50	CORAL		#6495ED	CORNFLOWERBLUE
	#FFF8DC	CORNSILK		#DC143C	CRIMSON
	#00FFFF	CYAN		#00008B	DARKBLUE
	#483D8B	DARKSLATEBLUE		#008B8B	DARKCYAN
	#B8860B	DARKGOLDENROD		#A9A9A9	DARKGRAY
	#FF1493	DEEPPINK		#00BFFF	DEEPSKYBLUE
	#696969	DIMGRAY		#1E90FF	DODGERBLUE
	#8B4513	FIREBRICK		#FFFAF0	FLORALWHITE
	#228B22	FORESTGREEN		#FF00FF	FUCHSIA
	#DCDCDC	GAINSBORO		#F8F8FF	GHOSTWHITE
	#FFD700	GOLD		#DAA520	GOLDENROD
	#808080	GRAY		#008000	GREEN
	#ADFF2F	GREENYELLOW		#F0FF00	HONEYDEW
	#FF69B4	HOTPINK		#CD5C5C	INDIANRED
	#4B0082	INDIGO		#FFFFFF	IVORY
	#F0E68C	KHAKY		#E6E6FA	LAVENDER
	#FFF0F5	LAVENDERBLUSH		#FFFACD	LEMONCHIFFON
	#ADD8E6	LIGHTBLUE		#F08080	LIGHTCORAL
	#E0FFFF	LIGHTCYAN		#FAFAD2	LIGHTGOLDENRODYELLOW
	#90EE90	LIGHTGREEN		#D3D3D3	LIGHTGRAY
	#FFB6C1	LIGHTPINK		#FFA07A	LIGHTSALMON
	#20B2AA	LIGHTSEAGREEN		#87CEFA	LIGHTSKYBLUE
	#778899	LIGHTSLATEGRAY		#B0C4DE	LIGHTSTEELBLUE
	#FFFFE0	LIGHTYELLOW		#00FF00	LIME
	#32CD32	LIMEGREEN		#FAF0E6	LINEN
	#FF00FF	MAGENTA		#800000	MAROON
	#66CDAA	MEDIUMAQUAMARINE		#0000CD	MEDIUMBLUE
	#BA55D3	MEDIUMMORCHID		#9370DB	MEDIUMPURPLE
	#3CB371	MEDIUMSEAGREEN		#7B68EE	MEDIUMSLATEBLUE
	#00FA9A	MEDIUMSPRINGGREEN		#48D1CC	MEDIUMTORQUOISE
	#C71585	MEDIUMVIOLETRED		#191970	MIDNIGHTBLUE
	#F5FFFA	MINTCREAM		#FFE4E1	MISTYROSE
	#FFDEAD	NAVAJOWHITE		#000080	NAVY
	#FDF5E6	OLDLACE		#808000	OLIVE
	#6B8E23	OLIVEDRAB		#FFA500	ORANGE
	#FF4500	ORANGERED		#DA70D6	ORCHID

	#EEE8AA	PALEGOLDENROD		#98FB98	PALEGREEN
	#AFEEEE	PALETURQUOISE		#DB7093	PALEVIOLETRED
	#FFefd5	PAPAYAWHIP		#FFDAB9	PEACHPUFF
	#CD853F	PERU		#FFC0CB	PINK
	#DDA0DD	PLUM		#B0E0E6	POWDERBLUE
	#800080	PURPLE		#FF0000	RED
	#BC8F8F	ROSYBROWN		#4169E1	ROYALBLUE
	#8B4513	SADDLEBROWN		#FA8072	SALMON
	#F4A460	SANDYBROWN		#2E8B57	SEAGREEN
	#FFF5EE	SEASHELL		#A0522D	SIENNA
	#C0C0C0	SILVER		#87CEEB	SKYBLUE
	#6A5ACD	SLATEBLUE		#708090	SLATEGRAY
	#FFFAFA	SNOW		#00FF7F	SPRINGGREEN
	#4682B4	STEELBLUE		#D2B48C	TAN
	#008080	TEAL		#D8BFD8	THISTLE
	#FF6347	TOMATO		#40E0D0	TURQUOISE
	#EE82EE	VIOLET		#F5DEB3	WHEAT
	#FFFFFF	WHITE		#F5F5F5	WHITESMOKE
	#FFFF00	YELLOW		#9ACD32	YELLOWGREEN

BOOK IN PROGRESS

APPUNTI SU Nvu

Competenze	Abilità	Conoscenze
Utilizzare, con autonomia e responsabilità, le reti e gli strumenti informatici nelle attività di studio, ricerca e approfondimento disciplinare.	Utilizzare la rete Internet per ricercare dati e fonti multidisciplinari e per attività di comunicazione interpersonale	Principali servizi e strumenti per la comunicazione su Internet
	Analizzare e risolvere semplici problemi con i principi della programmazione strutturata	

A cura della prof.ssa Giorgia Martina
con la collaborazione di Ribezzi Antonio

Creare una pagina WEB per iniziare ...

Per poter creare una pagina Web normalmente è necessario conoscere un linguaggio di programmazione specifico tipo HTML. L'interesse per la creazione e la gestione di siti più o meno complessi da pubblicare sul WEB, ha attivato una serie di aziende per la produzione di software a pagamento e open source che ,semplificando le procedure di programmazione con codice specifico, riescono a far creare pagine di questo tipo anche ai cosiddetti “utenti finali” (utilizzatori di PC).

Tra i tanti applicativi utilizzati per la creazione di pagine WEB, in questo capitolo introdurremo NVU.

Nvu  (pronunciato N-view, inteso come "New View", ovvero “nuova visualizzazione”) è un editor di testo in HTML, cioè serve per scrivere e salvare una pagina Web. E' un applicativo Freeware e si può scaricare dal sito “www.html.it” o da “<http://sourceforge.net/projects/mozilla-italia/files/NVU/1.0/nvu-1.0.it-IT.win32.zip/download>”.

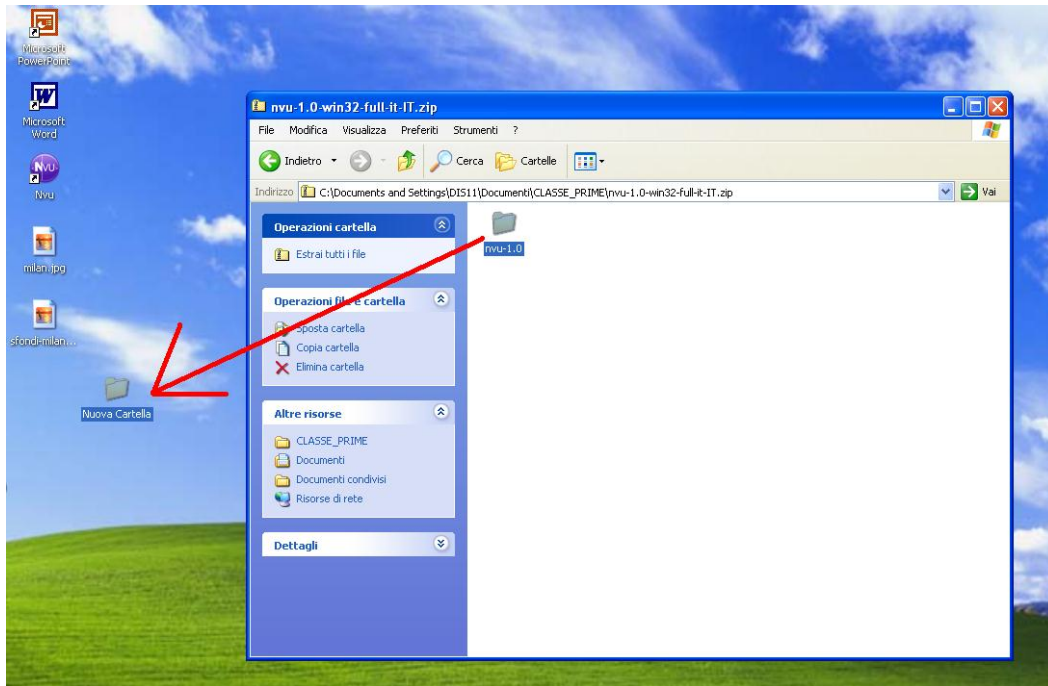
Questo applicativo è molto utilizzato e quindi l'azienda produttrice per meglio monitorare il numero degli utilizzatori sta divulgando su internet un link dal quale è possibile scaricare il logo “Made with Nvu” come marchio ufficiale da visualizzare su tutte le pagine Web create.

Per poterlo inserire nella propria pagina bisogna digitare le seguenti istruzioni:

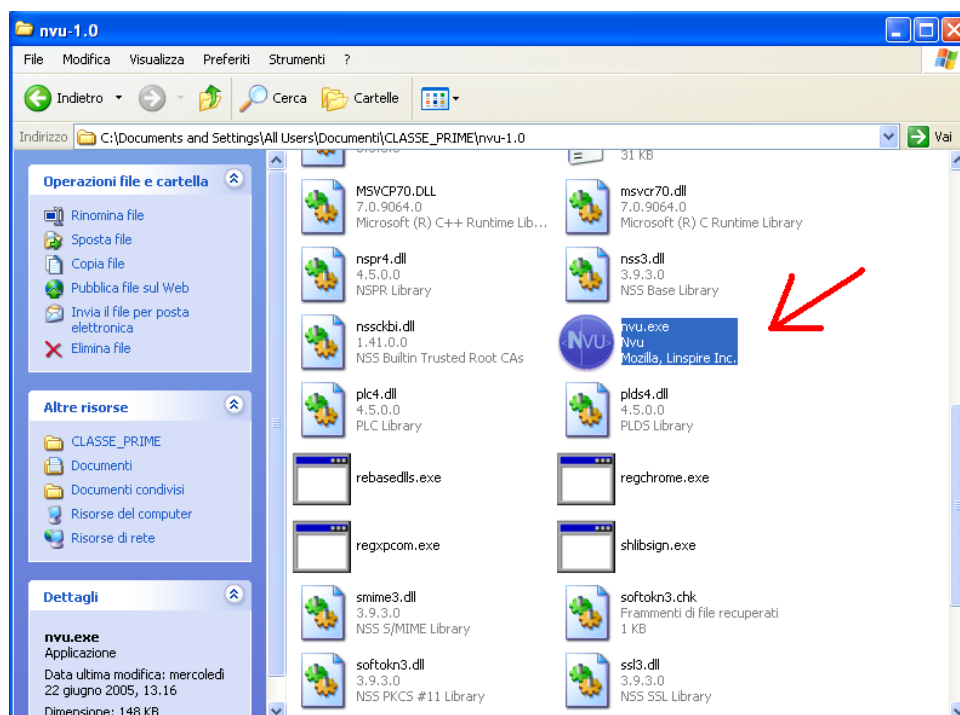
```
<a href="http://www.nvu.com/"></a>
```

Installazione di Nvu

Una volta scaricato il file “nvu-1.0-win32-full-it-IT.zip”, per installarlo basta fare doppio clic nella cartella creata di nome “nvu-1.0”, verrà aperto il programma “WinZip”, il quale permetterà l'estrazione dei vari file contenuti nella cartella stessa tra cui anche l'esecutivo “nvu.exe”.

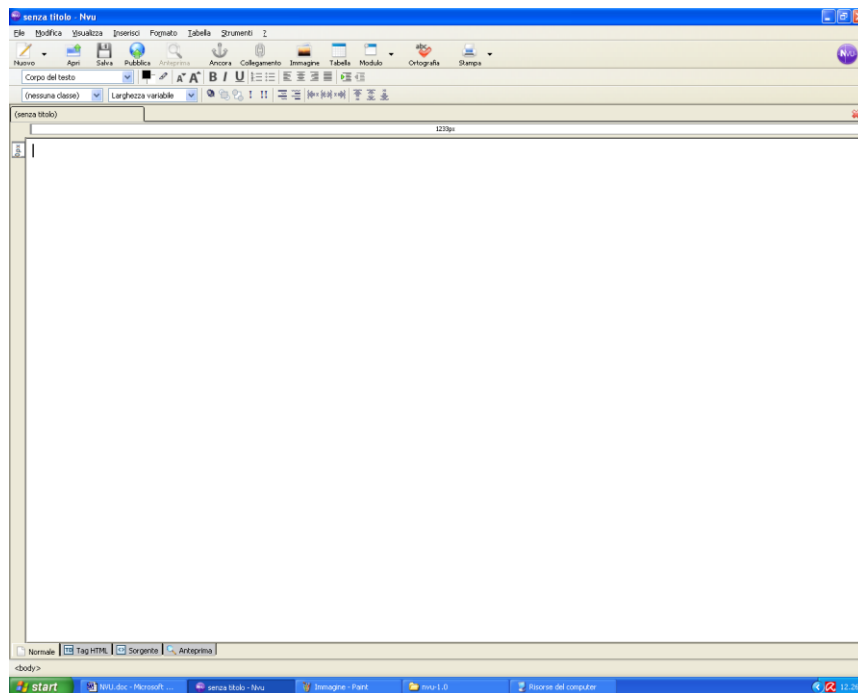


Per eseguire il programma bisogna fare doppio click su nvu.exe (vedi figura in basso)

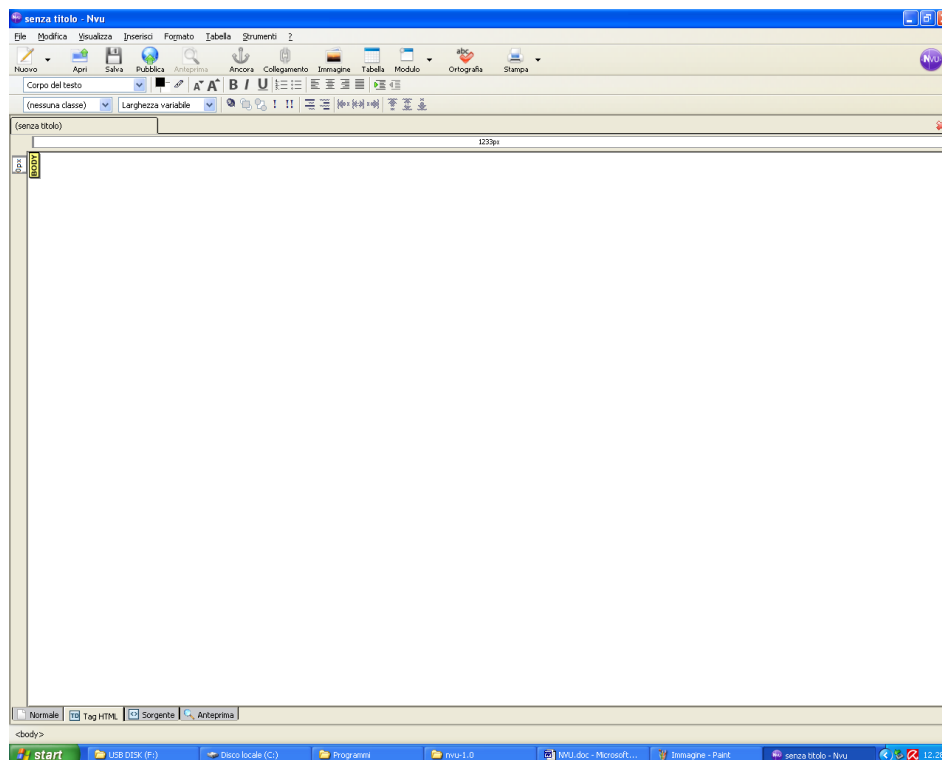


Interfaccia grafica dell'editor NVU

Composizione normale:



Tag HTML:



All'inizio la struttura del codice presentata da Nvu è sempre di questo tipo:

```
<html>

<head>

<title>Untitled</title>

</head>

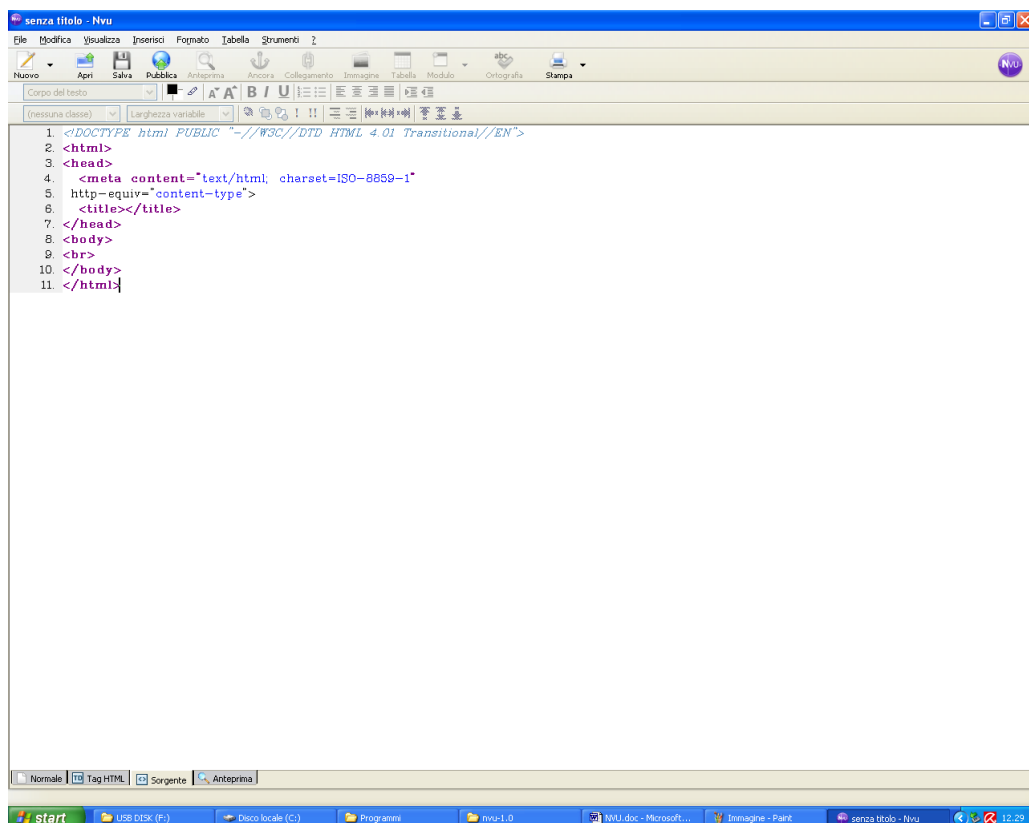
<body>

</body>

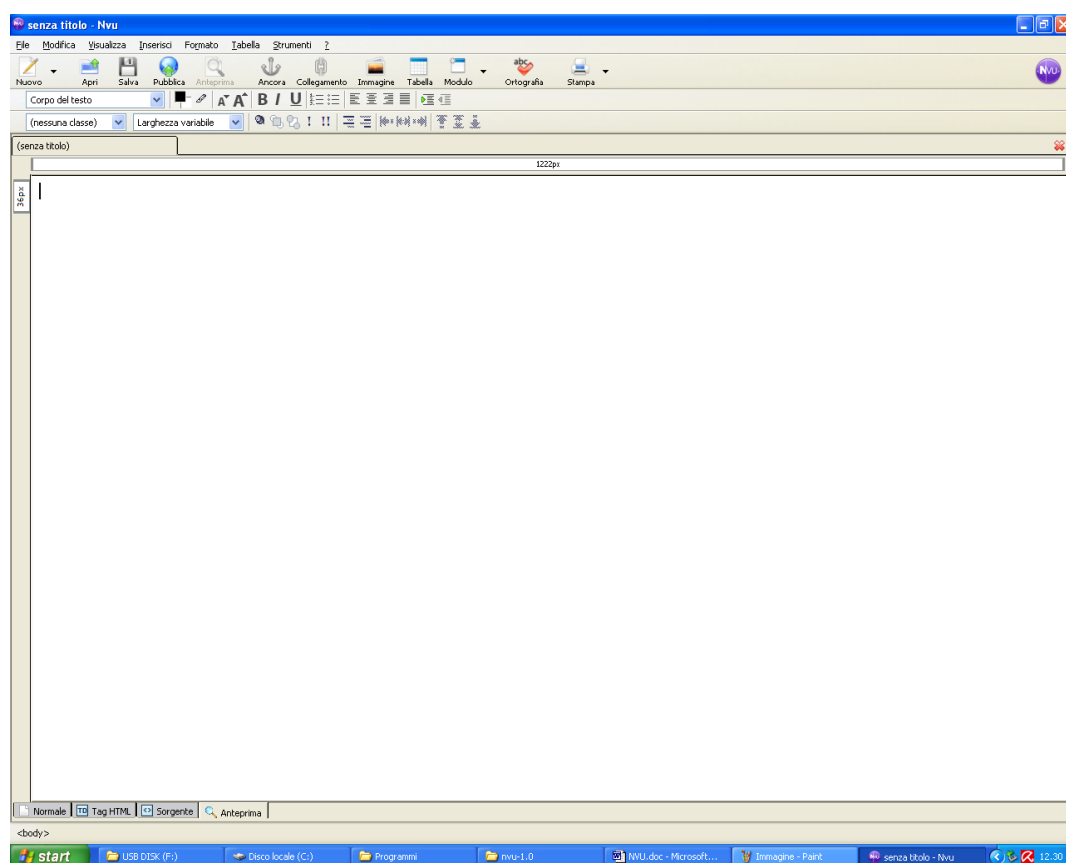
</html>
```

*Quelle scritte sono la base del documento web scritto in linguaggio HTML.
Le parole "html", "body", "head" sono i cosiddetti TAG e vengono sempre racchiusi
tra i segni minore "<" e maggiore ">".
Il documento scritto e pubblicato in questo modo sarebbe una pagina bianca.
Per poter realizzare una pagina con dei contenuti bisognerà riempire questi spazi con
scritte oppure istruzioni.*

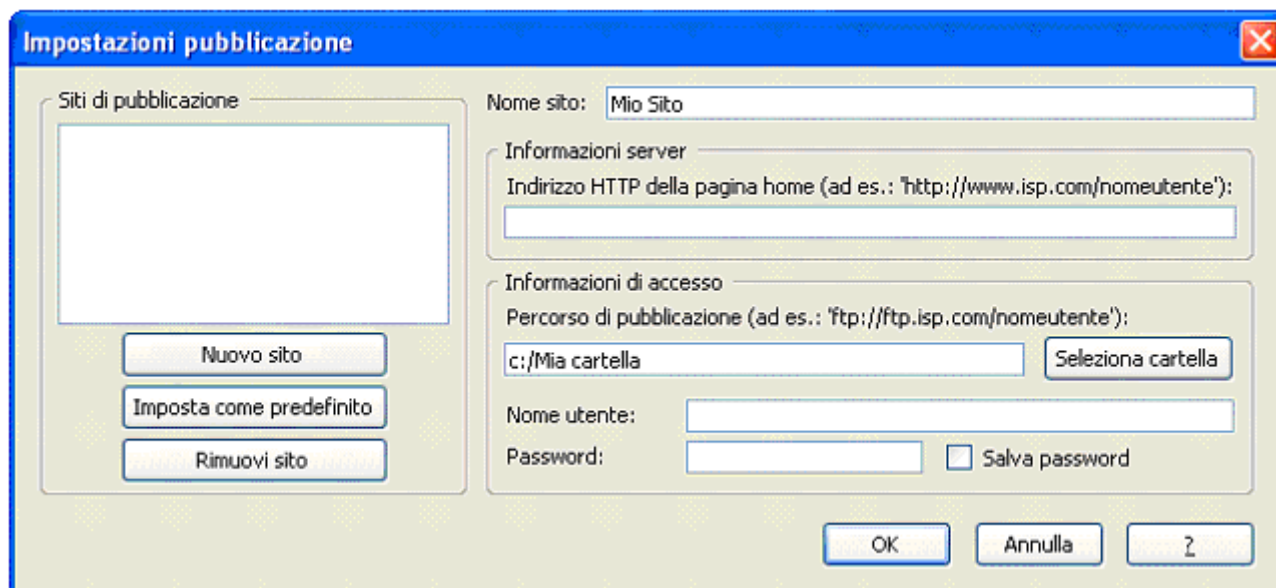
Esempio di Sorgente:



Anteprima:

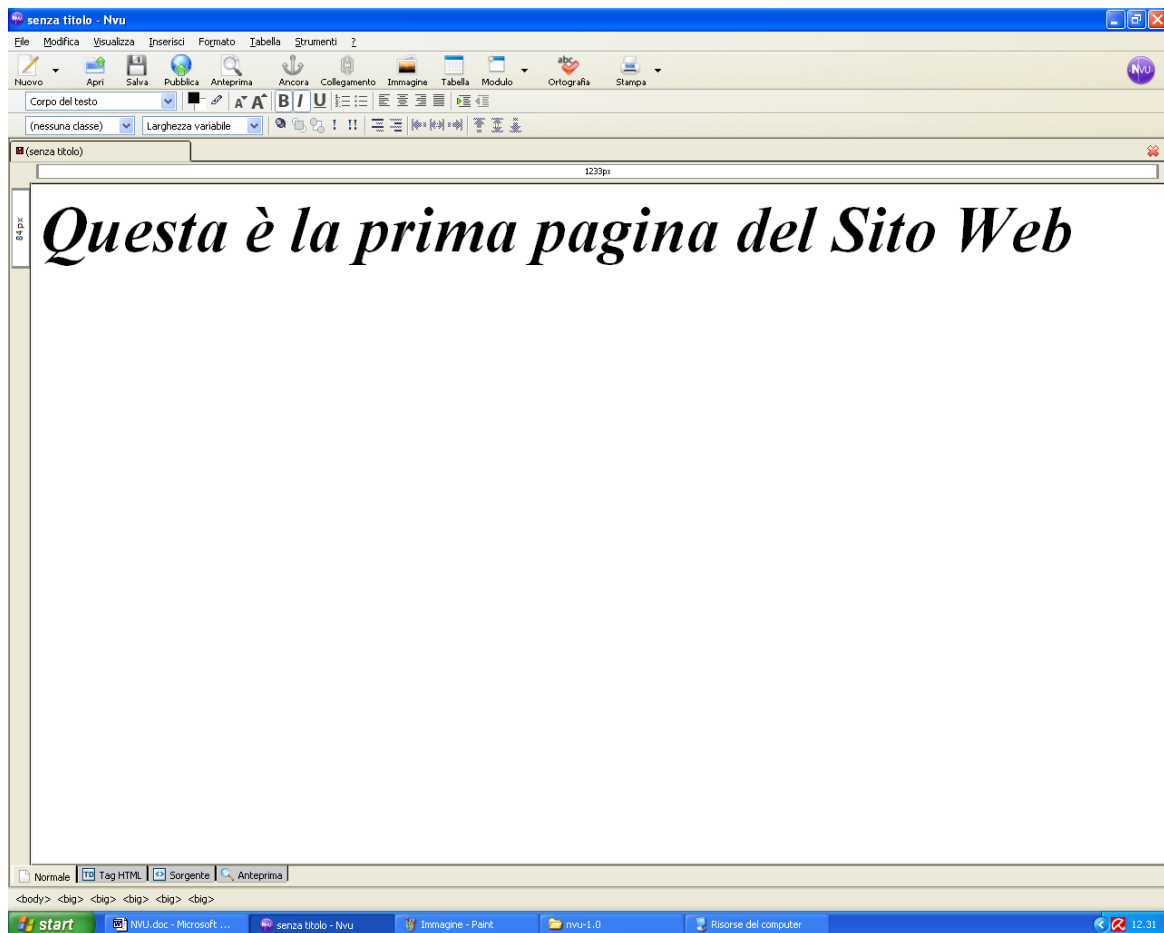


Si procede, innanzitutto, con il creare un proprio sito personale andando su Visualizza -> Mostra/Nascondi -> Gestione Siti e clickando su Modifica Siti. Quindi si dà il nome al sito e si seleziona la cartella in cui verranno salvati i files e si fa click su OK.



Si può scrivere in formato normale, in quanto il testo verrà convertito direttamente in formato HTML, sia in HTML modificando la sorgente.

Per iniziare, proviamo, per esempio a scrivere semplicemente del testo nella nostra pagina:



In alto, esempio di pagina scritta in composizione normale.

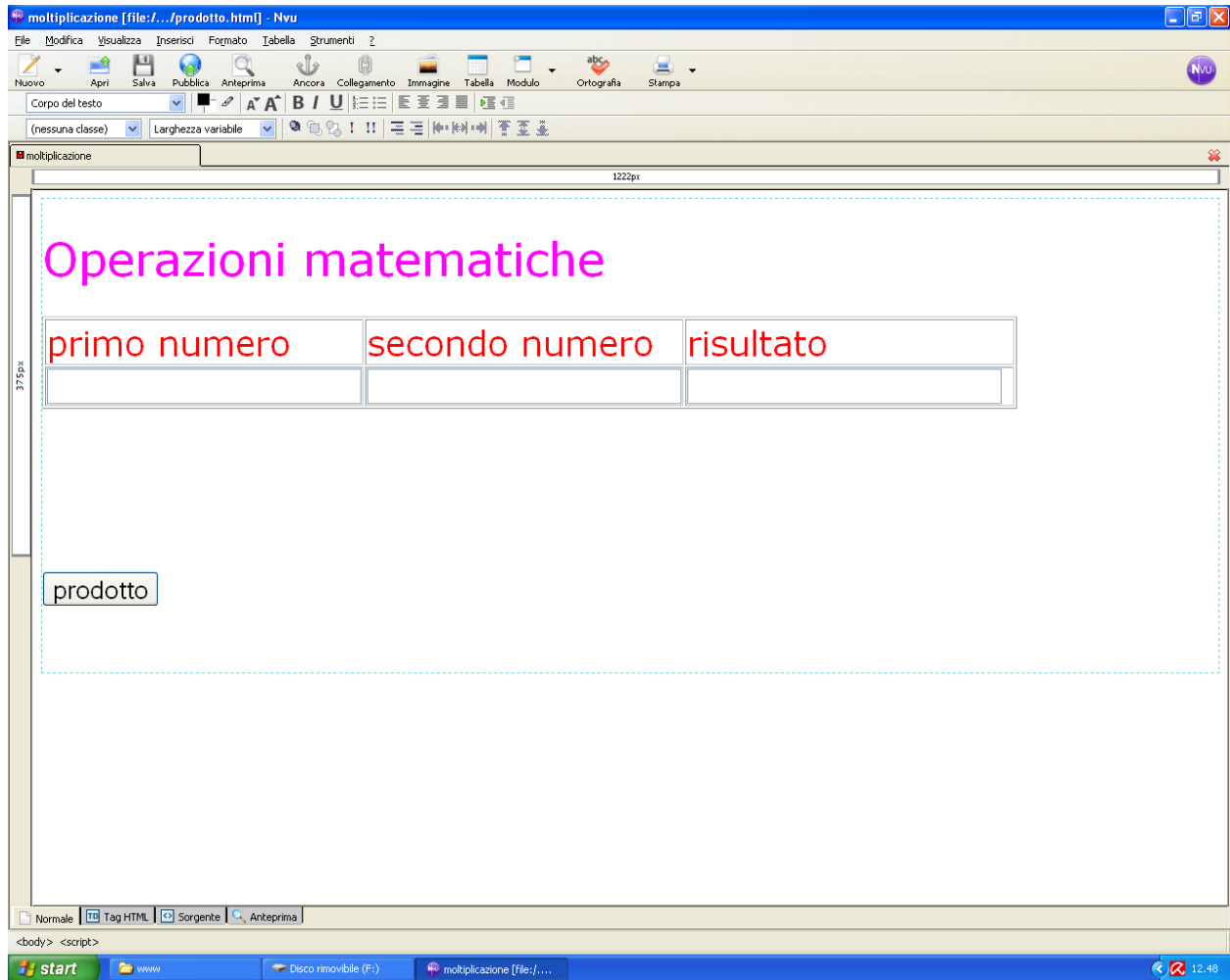
Visualizzando il file sorgente generato in HTML, otterremo le seguenti istruzioni:

```
<html>
<head>
  <meta content="text/html; charset=ISO-8859-1"
  http-equiv="content-type">
  <title></title>
</head>
<body>
  <big style="font-weight: bold; font-style: italic;"><big><big><big><big>Questa
  &egrave; la prima pagina del Sito Web</big></big></big></big>
</body>
</html>
```

Una volta completato il progetto, occorre salvare il file nella directory indicata in precedenza.

Su NVU si possono inserire degli script (dei particolari programmi) realizzati sia in Javascript sia in PHP. Proviamo ora a fare qualche calcolo, per esempio vogliamo calcolare il prodotto di due numeri.

Esempio di pagina HTML in cui viene usata la funzione “**Prodotto**” in Javascript:



Codice della Pagina:

<html> *Il tag che indica l'inizio del codice HTML*

<head>

<title>moltiplicazione</title> *Il tag che racchiude il titolo della pagina*

</head>

<body> *Viene indicato l'inizio del corpo della pagina*

<script> *Tag che indica l'inizio del Codice in Javascript*

function prodotto () { *Nome della funzione*

a=1*document.modulo.casella4.value; *il valore a sarà uguale al prodotto di 1 per il contenuto della casella 4*

b=1*document.modulo.casella5.value; *il valore b sarà uguale al prodotto di 1 per il contenuto della casella 5*

document.modulo.casella6.value=a*b; *il valore della casella 6 sarà il prodotto del valore a per il valore b*

```

} fine della funzione "prodotto"
</script> fine del codice in Javascript
<form method="post" name="modulo"> inizio del form
  <p><font color="#ff00ff" face="Verdana"
size="5">Operazioni matematiche </font></p>
  <table 100="" border="1" width=""> indica la creazione di una tabella
    <tbody> indica l'inizio del corpo della tabella
      <tr> indica l'inizio della prima riga della tabella
        <td width="33%"><font color="#ff0000"
face="verdana" size="4"> primo numero </font></td> indica il contenuto del primo riquadro
della tabella
        <td width="33%"><font color="#ff0000"
face="verdana" size="4"> secondo numero </font></td> indica il contenuto del secondo
riquadro della tabella
        <td width="34%"><font color="#ff0000"
face="verdana" size="4"> risultato </font></td> indica il contenuto del terzo riquadro della
tabella
      </tr> indica la fine della prima riga della tabella
      <tr> indica l'inizio della seconda riga della tabella
        <td width="33%"><input name="casella4" size="23" style="" type="text"></td>
indica il contenuto del quarto riquadro della tabella, in questo caso una casella di testo in cui
verrà richiamata la funzione in Javascript scritta in precedenza
        <td width="33%"><input name="casella5" size="23" style="" type="text"></td>
indica il contenuto del quinto riquadro della tabella
        <td width="34%"><input name="casella6" size="23" style="" type="text"></td>
indica il contenuto del quarto riquadro della tabella
      </tr> indica la fine della seconda riga della tabella
    </tbody> indica la fine del corpo della tabella
  </table> indica la fine della tabella
  <p>&nbsp;</p>
  <p>&nbsp;</p>
  <p><input value="prodotto" onclick="prodotto()" name="B1" type="button"></p>
crea un bottone che eseguirà la funzione Javascript "prodotto"
  <br>
</form> indica la fine del Form
</body> indica la fine del corpo del codice
</html> indica la fine del codice in HTML

```

Una volta salvato il file HTML, si può visualizzare con un comune browser come Internet Explorer, Mozilla Firefox ecc...

Buon lavoro e buon divertimento !!!

BOOK IN PROGRESS

ALGORITMI E DIAGRAMMI DI FLUSSO

Competenze	Abilità	Conoscenze
Utilizzare, con autonomia metodologica ed esecutiva, procedure e tecniche per trovare soluzioni efficaci ed efficienti, in relazione a semplici problemi nei campi di propria competenza.	Rappresentare la soluzione di un problema con diagrammi di flusso.	Concetto di algoritmo.
		Fasi risolutive di un problema e rappresentazione con diagrammi di flusso o pseudocodice
	Analizzare e risolvere semplici problemi con i principi della programmazione strutturata	Fondamenti di programmazione strutturata.

A cura del prof. Giuseppe Cavallo

INDICE

CONCETTI FONDAMENTALI SUGLI ALGORITMI; 3

DEFINIZIONE FORMALE DI ALGORITMO; 5

ESEMPIO DI ALGORITMO; 6

RAPPRESENTAZIONE DELL'ALGORITMO MEDIANTE LA TABELLA DI VERITÀ; 8

I DIAGRAMMI DI FLUSSO; 9

TIPI DI ISTRUZIONI; 15

LE STRUTTURE DI CONTROLLO; 16

LA SELEZIONE; 17

LA SEQUENZA; 17

L'ITERAZIONE; 18

ACCUMULATORI E CONTATORI; 19

CONCETTI FONDAMENTALI SUGLI ALGORITMI

Per molto tempo si pensò che il termine *algoritmo* derivasse da una storpiatura del termine *logaritmo*. L'opinione attualmente diffusa è invece che il termine derivi da *al-Khuwarizmi*, nome di un matematico arabo, autore di un libro di aritmetica e di uno di algebra: nel libro di aritmetica si parla della cosiddetta numerazione araba (quella attualmente usata) e si descrivono i procedimenti per l'esecuzione delle operazioni dell'aritmetica elementare. Questi procedimenti vennero in seguito chiamati algoritmi e il termine passò ad indicare genericamente qualunque *procedimento di calcolo*.

L'algoritmo esprime le *azioni* da svolgere su determinati *oggetti* al fine di produrre gli *effetti* attesi. Una azione che produce un determinato effetto è chiamata **istruzione** e gli oggetti su cui agiscono le istruzioni possono essere **costanti** (valori che restano sempre uguali nelle diverse esecuzioni dell'algoritmo) e **variabili** (contenitori di valori che variano ad ogni esecuzione dell'algoritmo). Si potrà dire brevemente che un algoritmo è una elaborazione di dati: i dati, cioè l'insieme delle informazioni che devono essere elaborate, sono manipolati, secondo le modalità descritte dalle istruzioni, per produrre altri dati. Ciò porta l'algoritmo ad essere una funzione di trasformazione dei dati di un insieme A (dati di input) in dati di un insieme B (dati di output).

In questi appunti, dato che ci si pone il fine di una introduzione alla programmazione, più che una definizione rigorosa di algoritmo se ne fornirà una definizione intuitiva. In questo senso si può definire l'algoritmo come "*.. un insieme di istruzioni che definiscono una sequenza di operazioni mediante le quali si risolvono tutti i problemi di una determinata classe*".

Per chiarire meglio il concetto di algoritmo è bene fare riferimento ad alcune proprietà che un insieme di istruzioni deve possedere affinché possa chiamarsi algoritmo:

- La **finitezza**. Il numero di istruzioni che fanno parte di un algoritmo è finito. Le operazioni definite in esso vengono eseguite un numero finito di volte.

- Il **determinismo**. Le istruzioni presenti in un algoritmo devono essere definite senza ambiguità. Un algoritmo eseguito più volte e da diversi esecutori, a parità di premesse, deve giungere a medesimi risultati. L'effetto prodotto dalle azioni descritte nell'algoritmo non deve dipendere dall'esecutore o dal tempo.
- La **realizzabilità pratica**. Tutte le azioni descritte devono essere eseguibili con i mezzi di cui si dispone.
- La **generalità**. Proprietà già messa in evidenza nella definizione che si è data: un algoritmo si occupa della risoluzione di famiglie di problemi.

DEFINIZIONE FORMALE DI ALGORITMO

- Un algoritmo è un insieme ben ordinato di operazioni non ambigue ed effettivamente calcolabili che eseguito , produce un risultato e termina in una quantità finita di tempo.

Insieme ben ordinato

- Ordinamento delle operazioni da eseguire; deve essere chiaro e non ambiguo
- Il controllo deve procedere senza ambiguità da una operazione alla successiva

Di solito nei linguaggi di programmazione

- Le operazioni sono elencate come passi numerati
- In assenza di altra indicazione (operazioni condizionali o iterative) il controllo passa sempre al passo successivo

Operazioni non ambigue calcolabili

- Tutti i passi devono essere chiari per chi l'esegue le operazioni .
- Una operazione non ambigua è detta primitiva
- Analizzando la stesura della prima versione di un algoritmo occorre verificare che tutte le operazioni coinvolte siano primitive

Produce un risultato

- Gli algoritmi risolvono problemi
- Per comprendere se una soluzione algoritmica è corretta , il suo risultato deve poter essere confrontato con il risultato atteso
- Se il risultato non è determinabile , l'algoritmo

deve produrre un messaggio di errore , o fornire un approssimazione del risultato corretto

Termina in una quantità finita di tempo

- L'algoritmo deve produrre un risultato dopo l'esecuzione di un numero finito di operazioni
- E' sempre buona abitudine chiudere l'algoritmo con una frase di chiusura
- È facile incorrere in situazioni dove gruppi di istruzioni vengono ripetute all'infinito perché non vengono mai a verificarsi le condizioni di chiusura di ciclo

ESEMPIO DI ALGORITMO

Vogliamo costruire un algoritmo che stampi 4 volte "prova"
Una possibile soluzione potrebbe essere:

1. poni $k = 0$
2. stampa "prova "
3. ripeti il passo 2 sino a quando $k < 4$
4. Fine

Questo algoritmo dato in input ad un agente esecutore non termina mai perchè :

- ESEGUE IL PASSO 1: inserisce il valore zero in K
- ESEGUE IL PASSO 2: scrive "prova "
- ESEGUE IL PASSO 3 CHE LO RIPORTA AL PASSO 2 PERCHE' K CHE E' ZERO QUINDI E' SICURAMENTE MINORE DI 4

In questo modo il passo 2 ed il passo 3 sono ripetuti all'infinito .

Come possiamo modificare l'algoritmo affinché possa terminare ottenendo il risultato voluto cioè la stampa di quattro volte "prova" ?

Basta aggiungere una ulteriore istruzione nell'algoritmo, così :

1. poni $k = 0$
2. stampa "prova "
3. **aggiungi 1 a K**
4. ripeti il passo 2 ed il 3 sino a quando $k < 4$
5. Fine

Questo algoritmo termina perchè :

- ESEGUE IL PASSO 1: $0 \rightarrow K$
- ESEGUE IL PASSO 2: ("prova " in output)
- ESEGUE IL PASSO 3: $1 + K \rightarrow K$ ($K = 1$)
- ESEGUE IL PASSO 4 CHE RIMANDA IL CONTROLLO AL PASSO 2 E POI IN SEQUENZA AL PASSO 3 PERCHE' K, CHE VALE **1**, E' SICURAMENTE MINORE DI 4
- ESEGUE IL PASSO 2: ("prova " in output)
- ESEGUE IL PASSO 3: $1 + K \rightarrow K$ ($K = 2$) PERCHE' PRIMA ERA $k = 1$
- ESEGUE IL PASSO 4 CHE RIMANDA IL CONTROLLO AL PASSO 2 E POI IN SEQUENZA AL PASSO 3 PERCHE' K, CHE VALE **2**, E' SICURAMENTE MINORE DI 4
- ESEGUE IL PASSO 2 : ("prova " in output)
- ESEGUE IL PASSO 3: $1 + K \rightarrow K$ ($K = 3$) PERCHE' PRIMA ERA $k = 2$
- ESEGUE IL PASSO 4 CHE RIMANDA IL CONTROLLO AL PASSO 2 E POI IN SEQUENZA AL PASSO 3 PERCHE' K, CHE VALE **3**, E' SICURAMENTE MINORE DI 4
- ESEGUE IL PASSO 2: ("prova " in output)
- ESEGUE IL PASSO 3: $1 + K \rightarrow K$ ($K = 4$) PERCHE' PRIMA ERA $k = 3$
- ESEGUE IL PASSO 4; POICHE' CHE LA CONDIZIONE $K < 4$ NON E' PIU' VERA. IN QUANTO K E' UGUALE A 4, PASSA IL CONTROLLO AL PASSO SUCCESSIVO, IL 5
- PASSO 5 (**FINE**) TERMINE DELL'ALGORITMO.

RAPPRESENTAZIONE DELL'ALGORITMO MEDIANTE LA TABELLA DI VERITÀ

Un modo diverso di rappresentare un algoritmo si basa sulla tabella di verità.

La prima colonna indica la posizione ordinale del passo che viene eseguito

Le altre colonne indicano i valori degli oggetti così come evolvono con il susseguirsi dei passi eseguiti .

PASSO	K		
1	0		
2		OUTPUT (PROVA)	
3	1		
4			K<4 ? SI TORNA A 2
2		OUTPUT (PROVA)	
3	2		
4			K<4 ? SI TORNA A 2
2		OUTPUT (PROVA)	
3	3		
4			K<4 ? SI TORNA A 2
2		OUTPUT (PROVA)	
3	4		
4			K<4 ? NO PROSEGUI
5	Termina algoritmo		

I Diagrammi di Flusso

I **diagrammi di flusso** sono dei disegni che rappresentano graficamente un algoritmo, permettendo così una comprensione immediata del funzionamento del nostro percorso logico e un controllo accurato sulla funzionalità e la casistica del ragionamento.

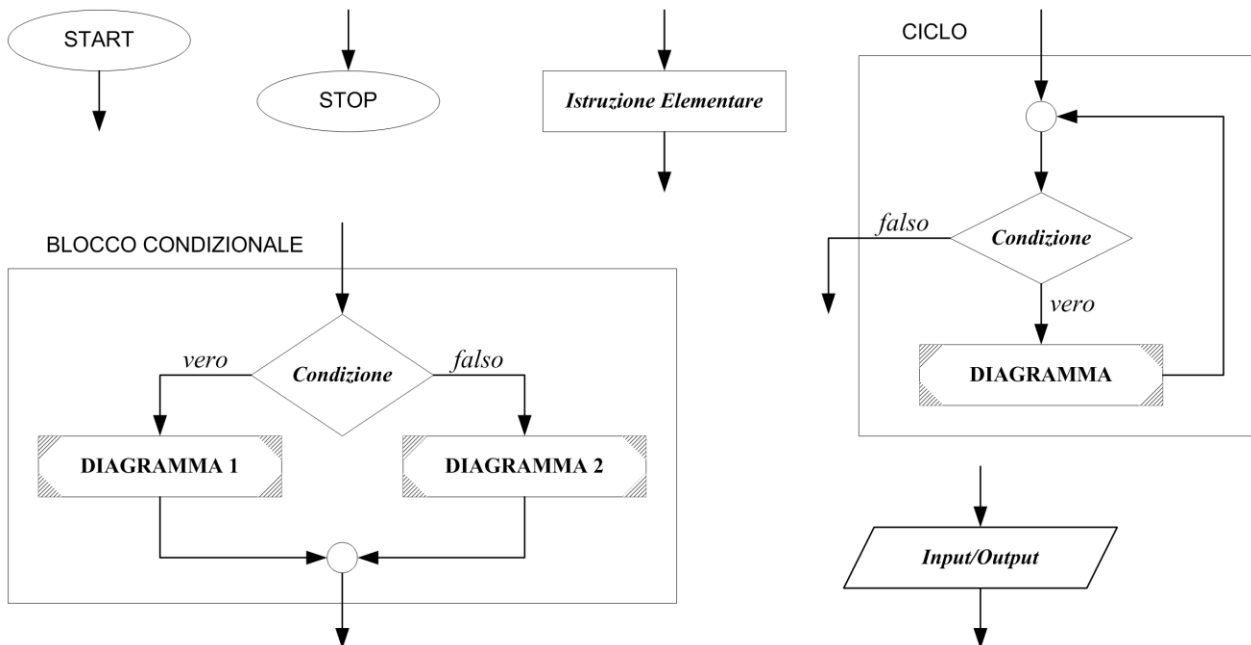
Un diagramma a flusso (o flow chart) è un linguaggio formale di tipo grafico utilizzato per rappresentare algoritmi (in senso lato), attraverso la descrizione delle differenti operazioni sotto forma di uno schema in cui le diverse fasi del processo e le differenti condizioni che devono essere rispettate vengono rappresentati da simboli grafici detti blocchi. I blocchi sono collegati tra loro tramite frecce che indicano la cronologia.

I diagrammi trovano la loro applicazione in diversi ambiti: a partire dall'impiego nella programmazione informatica, si sono diffusi in campo industriale, come strumenti per schematizzare i processi produttivi e/o organizzativi. Da qui l'impiego anche in campo valutativo, in particolare nelle metodologie della **qualità**, per la rappresentazione dei processi da valutare e da tenere sotto controllo (sia di tipo organizzativo, sia di tipo educativo).

Come si costruiscono

Il linguaggio della programmazione individua cinque tipi di blocchi elementari, collegati tra loro attraverso frecce:

- un blocco iniziale, rappresentato da un ovale contenente la parola "INIZIO"
- un blocco finale, rappresentato da un ovale contenente la parola "FINE"
- i blocchi di azione, rappresentati da un rettangolo indicante una data operazione
- i blocchi di lettura/scrittura, rappresentati da un parallelogramma
- i blocchi di controllo, rappresentati da un rombo con un ingresso e due uscite



Osserviamo la presenza del blocco condizionale; esse serve per eseguire un blocco di istruzioni quando è verificata una condizione oppure, in alternativa, un altro blocco di istruzioni se la condizione è falsa.

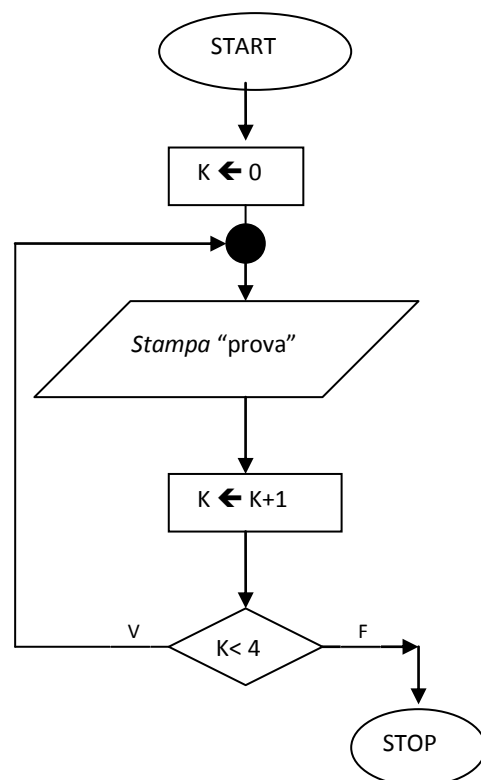
In realtà l'uso dei diagrammi di flusso in campo organizzativo risulta più libero nelle forme di rappresentazione, mantenendo l'idea di fondo di rappresentare un processo attraverso un algoritmo composto da una sequenza di passaggi, tra loro collegati logicamente e cronologicamente.

La costruzione di un diagramma di flusso richiede di individuare i passaggi chiave che caratterizzano un dato processo e li rappresenta, evidenziando con frecce le loro reciproche relazioni. Può essere utile accompagnare un algoritmo con una breve descrizione verbale che aiuti a riconoscere la logica connessa alla sua costruzione.

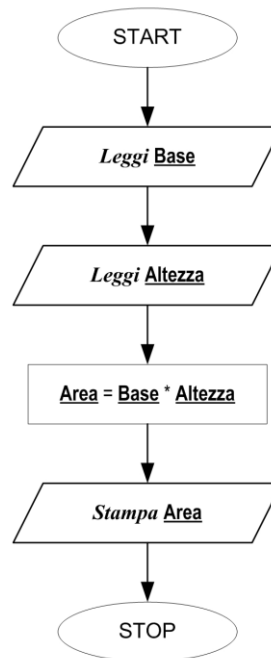
L'algoritmo precedentemente analizzato :

1. poni $k = 0$
2. stampa "prova "
3. **aggiungi 1 a K**
4. **ripeti il passo 2 ed il 3**
sino a quando $k < 4$
5. Fine

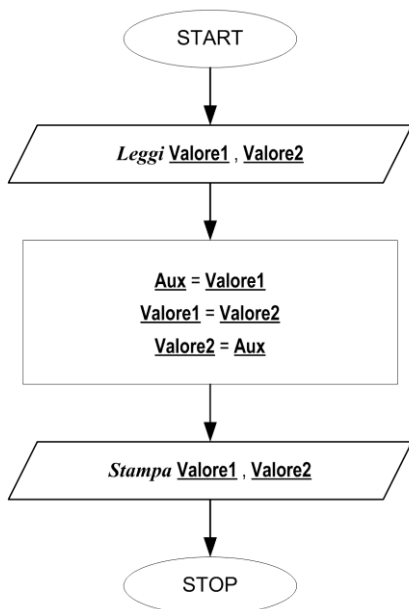
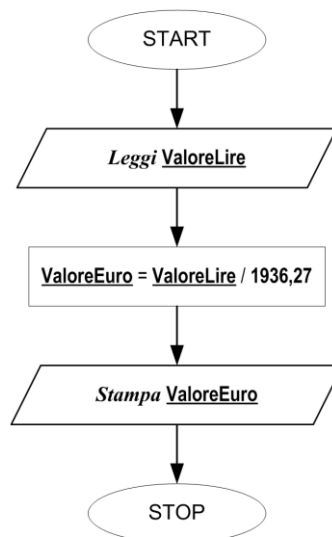
possiamo rappresentarlo graficamente con il diagramma di flusso a lato:



Qui a lato è rappresentato
Il Diagramma di Flusso
per il calcolo dell'area
di un rettangolo

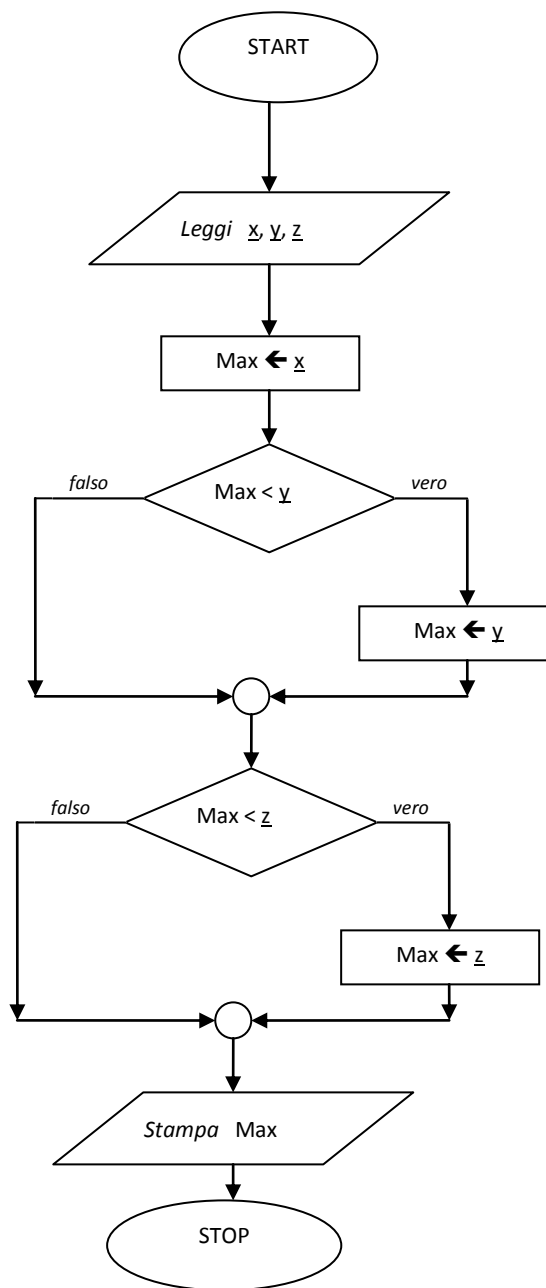


Qui a lato è rappresentato
Il Diagramma di Flusso
per conversione Lira/Euro



Qui a lato è rappresentato il Diagramma di Flusso
per lo scambio dei valori contenuti in due variabili.
La variabile Aux serve per “appoggiare”
temporaneamente i valori delle due altre variabili.

Diagramma di flusso
Per il calcolo del massimo
tra tre numeri



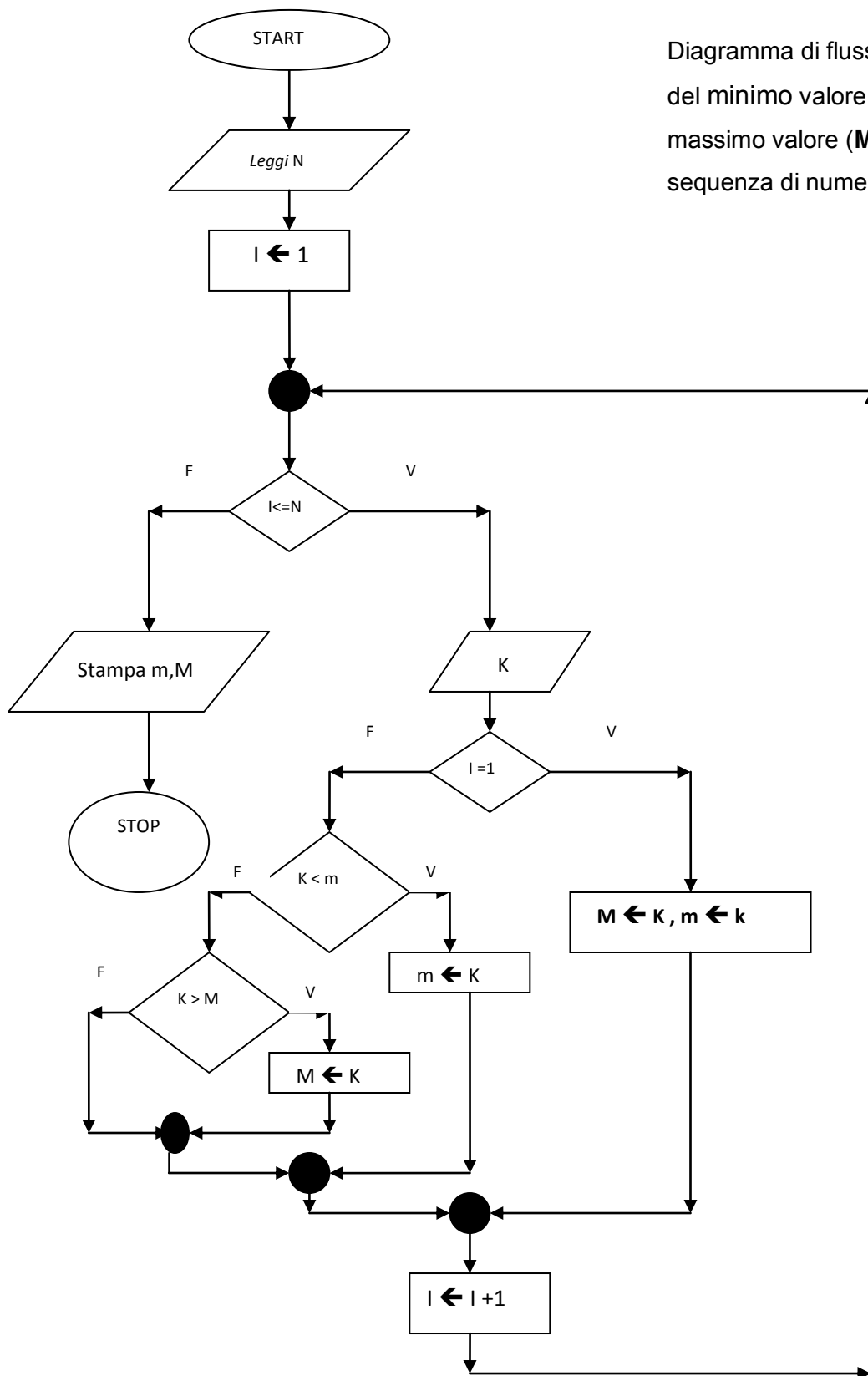
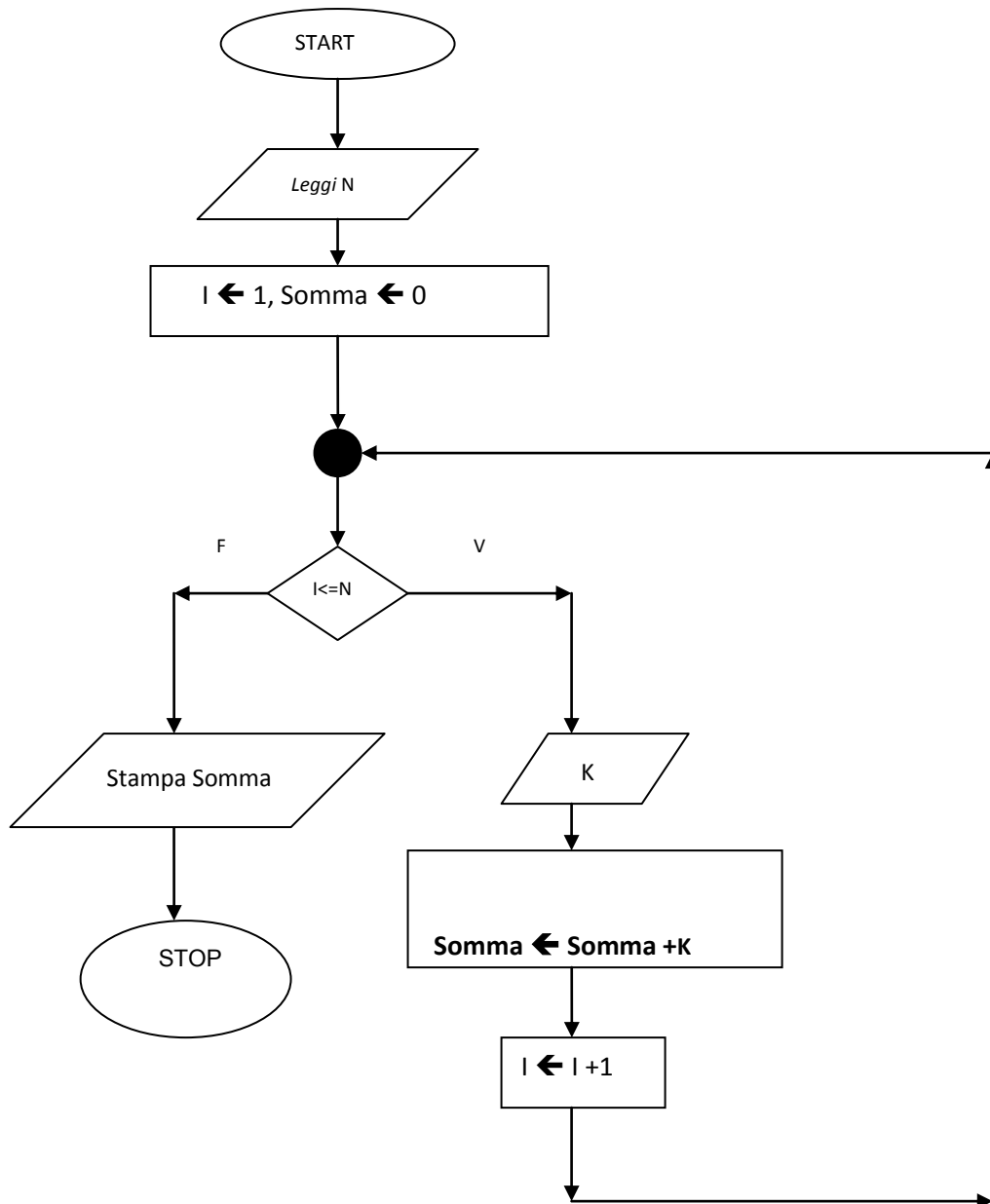


Diagramma di flusso per il calcolo del minimo valore (**m**) e del massimo valore (**M**) data una sequenza di numeri generica

Diagramma di flusso per il calcolo della somma di una sequenza di numeri generica



TIPI DI ISTRUZIONI

Gli algoritmi operano principalmente su variabili che conterranno i dati sui quali si vuole svolgere una determinata elaborazione. I valori da elaborare devono essere *assegnati* alle variabili prima di effettuare l'elaborazione. Si pensi infatti ad una variabile come ad un contenitore. Le istruzioni operano sui valori contenuti: se questi non ci sono non ci si può attendere alcuna elaborazione. Ogni variabile è identificata da un nome che permette di distinguerla dalle altre.

- **L'istruzione di assegnamento** fa in modo che un determinato valore sia conservato in una variabile. In questo modo si prepara la variabile per l'elaborazione o si conserva nella variabile un valore intermedio prodotto da una elaborazione precedente. Si può assegnare ad una variabile un valore costante come anche il valore risultante da una espressione aritmetica.
- **L'istruzione di input** fa in modo che l'algoritmo riceva dall'esterno un valore da assegnare ad una variabile. Nel caso di algoritmi eseguiti da un elaboratore, questi attende che da una unità di input (per esempio la tastiera) arrivi una sequenza di caratteri tipicamente terminanti con la pressione del tasto <Invio> . Il dato verrà assegnato alla variabile appositamente predisposta. Praticamente si tratta di una istruzione di assegnamento solo che stavolta il valore da assegnare proviene dall'esterno.
- **L'istruzione di output** fa in modo che l'algoritmo comunichi all'esterno i risultati della propria elaborazione. Nel caso di un elaboratore viene inviato su una unità di output (per esempio il video) il valore contenuto in una determinata variabile.

Esaminiamo adesso due versioni di un algoritmo per il calcolo dell'area di un rettangolo (i nomi delle variabili sono scritti in maiuscolo):

Algoritmo A

Assegna a BASE il valore 3
 Assegna a ALTEZZA il valore 7
 Assegna a AREA valore
 $BASE * ALTEZZA$
 Comunica AREA

Algoritmo B

Ricevi BASE
 Ricevi ALTEZZA
 Assegna a AREA valore $BASE * ALTEZZA$
 Comunica AREA

Nell'algoritmo A si assegnano alle variabili BASE ed ALTEZZA dei valori costanti, l'algoritmo calcola l'area di un rettangolo particolare e se si vuole applicare l'algoritmo ad un diverso rettangolo è necessario modificare le due istruzioni di assegnamento a BASE e ALTEZZA: l'algoritmo ha perso la sua caratteristica di generalità. Questo esempio non deve portare a concludere che non ha senso parlare di costanti in un algoritmo perché, per esempio, se si fosse trattato di un triangolo il calcolo dell'area avrebbe assunto l'aspetto: Assegna a AREA valore $BASE * ALTEZZA / 2$. La costante 2 prescinde dai valori diversi che possono essere assegnati a BASE e ALTEZZA, essendo parte della formula del calcolo dell'area di un triangolo *qualsiasi*.

Nell'algoritmo B i valori da assegnare a BASE e ALTEZZA provengono da una unità di input. L'algoritmo ad ogni esecuzione si fermerà in attesa di tali valori e il calcolo verrà eseguito su tali valori: l'elaborazione è sempre la stessa (l'area di un rettangolo si calcola sempre allo stesso modo) ma i dati saranno di volta in volta diversi (i rettangoli hanno dimensioni diverse).

LE STRUTTURE DI CONTROLLO

Gli algoritmi, a causa della loro generalità, lavorano utilizzando variabili. Non si conoscono, al momento della stesura dell'algoritmo stesso, i valori che possono assumere le variabili. Ciò se permette di scrivere algoritmi generali può comportare problemi per alcune istruzioni: si pensi al problema apparentemente banale del calcolo del quoziente di due numeri:

Ricevi DIVIDENDO

Ricevi DIVISORE

Assegna a QUOZIENTE valore DIVIDENDO/DIVISORE

Comunica QUOZIENTE

Il quoziente può essere calcolato se DIVISORE contiene un valore diverso da 0, evento questo non noto in questo momento dipendendo tale valore dal numero proveniente da input. Inoltre è chiaro che, in questa eventualità, sarebbe priva di senso anche l'istruzione di output del valore di QUOZIENTE non essendoci nella variabile alcun valore.

È necessario introdurre, oltre alle istruzioni, degli strumenti che permettano di controllare l'esecuzione dell'algoritmo: le strutture di controllo. La **programmazione strutturata** (disciplina nata alla fine degli anni 60 per stabilire le regole per la scrittura di buoni algoritmi) impone l'uso di tre sole strutture per la composizione degli algoritmi:

LA SEQUENZA

È l'unica struttura di composizione che si è utilizzata finora. In poche parole questa struttura permette di specificare l'ordine con cui le istruzioni si susseguono: ogni istruzione produce un risultato perché inserita in un contesto che è quello determinato dalle istruzioni che la precedono. Nell'esempio di prima il calcolo di QUOZIENTE, per poter contenere il valore atteso, deve essere eseguito dopo gli input.

LA SELEZIONE

Questa struttura permette di scegliere tra due alternative la sequenza di esecuzione. È la struttura che ci permette, per esempio, di risolvere in modo completo il problema del calcolo del quoziente fra due numeri:

Ricevi DIVIDENDO

Ricevi DIVISORE

Se DIVISORE $\neq$ 0

Assegna a QUOZIENTE valore DIVIDENDO/DIVISORE

Comunica QUOZIENTE

Altrimenti

Comunica 'Operazione senza senso'

Fine-se

La condizione espressa nella struttura “Se” permette di scegliere, in relazione al valore di verità o falsità, quale elaborazione svolgere. La sequenza contenuta nella parte “Altrimenti” potrebbe mancare se si volesse soltanto un risultato laddove possibile: in tale caso se la condizione $\text{DIVISORE} \neq 0$ risultasse non verificata, non si effettuerebbe alcuna elaborazione.

L'ITERAZIONE

La struttura iterativa permette di ripetere più volte la stessa sequenza di istruzioni finché non si verifica una determinata condizione. Si voglia, a titolo di esempio, *scrivere un algoritmo che calcoli e visualizzi i quadrati di una serie di numeri positivi*. Si tratta in altri termini di effettuare la stessa elaborazione (calcolo e visualizzazione del quadrato di un numero) effettuata su numeri diversi (quelli che arriveranno dall'input):

```
Ricevi NUMERO
Mentre NUMERO > 0
  Assegna a QUADRATO valore NUMERO*NUMERO
  Comunica QUADRATO
Ricevi NUMERO
Fine-mentre
```

Dentro la struttura iterativa (la parte compresa fra le parole “Mentre” e “Fine-mentre”) sono specificate le istruzioni per il calcolo del quadrato di un numero: l'iterazione permette di ripetere tale calcolo per tutti i numeri che verranno acquisiti tramite l'istruzione di input inserita nell'iterazione stessa. La condizione $\text{NUMERO} > 0$ viene chiamata **condizione di controllo del ciclo** e specifica quando deve terminare l'elaborazione (il valore introdotto da input è non positivo): si ricorda che l'algoritmo deve essere finito e non si può iterare all'infinito. Il primo input fuori ciclo ha lo scopo di permettere l'impostazione della condizione di controllo sul ciclo stesso e stabilire, quindi, quando terminare le iterazioni. In generale si può dire che la struttura di una elaborazione ciclica, controllata dal verificarsi di una condizione, assume il seguente aspetto:

```
Considera primo elemento
Mentre elementi non finiti
  Elabora elemento
  Considera prossimo elemento
Fine mentre
```

Le strutture di controllo devono essere pensate come schemi di composizione: una sequenza può contenere una iterazione che, a sua volta, contiene una selezione ecc.. Rappresentano in pratica i mattoncini elementari di una scatola di montaggio le cui diverse combinazioni permettono la costruzione di architetture di varia complessità.

ACCUMULATORI E CONTATORI

L'elaborazione ciclica è spesso utilizzata per l'aggiornamento di totalizzatori o contatori. Per chiarire meglio il concetto di totalizzatore, si pensi alle azioni eseguite dal cassiere di un supermercato quando si presenta un cliente con il proprio carrello pieno di merce. Il cassiere effettua una elaborazione ciclica sulla merce acquistata: ogni oggetto viene esaminato per acquisirne il prezzo. Lo scopo della elaborazione è quello di cumulare i prezzi dei prodotti acquistati per stabilire il totale che il cliente dovrà corrispondere.

Dal punto di vista informatico si tratta di utilizzare una variabile (nell'esempio potrebbe essere rappresentata dal totalizzatore di cassa) che viene aggiornata per ogni prezzo acquisito: ogni nuovo prezzo acquisito non deve sostituire il precedente ma aggiungersi ai prezzi già acquisiti precedentemente. Tale variabile:

1. dovrà essere azzerata quando si passa ad un nuovo cliente (ogni cliente dovrà corrispondere solamente il prezzo dei prodotti che lui acquista)
2. si aggiornerà per ogni prodotto esaminato (ogni nuovo prezzo acquisito verrà cumulato ai precedenti)
3. finito l'esame dei prodotti acquistati la variabile conterrà il valore totale da corrispondere.

La variabile di cui si parla nell'esempio è quella che, nel linguaggio della programmazione, viene definita un **totalizzatore** o **accumulatore**: cioè una variabile nella quale ogni nuovo valore non sostituisce ma si accumula a quelli già presenti in precedenza. Se la variabile si aggiorna sempre di una quantità costante (per esempio viene sempre aggiunta l'unità) viene chiamata **contatore**.

In generale si può dire che l'uso di un totalizzatore prevede i seguenti passi:

Inizializzazione totalizzatore
ciclo aggiornamento totalizzatore

...

Aggiornamento totalizzatore

Fine ciclo

Uso del totalizzatore

L'inizializzazione serve sia a dare senso all'istruzione di aggiornamento (cosa significherebbe la frase: *aggiorna il valore esistente con il nuovo valore* se non ci fosse un valore esistente ?), sia a fare in modo che l'accumulatore stesso contenga un valore coerente con l'elaborazione da svolgere (nell'esempio di prima il nuovo cliente non può pagare prodotti acquistati dal cliente precedente: il totalizzatore deve essere azzerato, prima di cominciare l'elaborazione, affinché contenga un valore che rispecchi esattamente tutto ciò che è stato acquistato dal cliente esaminato).

L'aggiornamento viene effettuato all'interno di un ciclo. Se infatti si riflette sulla definizione stessa di totalizzatore, è facile prendere atto che avrebbe poco significato fuori da un ciclo: come si può cumulare valori se non si hanno una serie di valori ?

Quando i valori da esaminare terminano, il totalizzatore conterrà il valore cercato. Nell'esempio di prima tutto ciò si tradurrebbe: finito l'esame dei prodotti acquistati, si potrà presentare al cliente il totale da corrispondere.

Si voglia, a titolo di esempio di utilizzo di un accumulatore, risolvere il seguente problema: *data una sequenza di numeri positivi, se ne vuole calcolare la somma.*

Inizializza SOMMA con valore 0
 Ricevi NUMERO
 Mentre NUMERO > 0
Aggiorna SOMMA sommando NUMERO
 Ricevi NUMERO
 Fine-mentre
Comunica SOMMA

Il totalizzatore SOMMA è inizializzato, prima del ciclo, al valore nullo poiché deve rispecchiare la somma dei numeri introdotti da input e, quindi, non essendo ancora stata effettuata alcuna elaborazione su alcun numero tale situazione viene espressa assegnando il valore neutro della somma.

L'output di SOMMA alla fine del ciclo indica il fatto che si può utilizzare (in questo caso per la conoscenza del valore contenuto) il totalizzatore quando il suo contenuto è coerente con il motivo della sua esistenza: SOMMA deve accumulare tutti i valori e ciò avverrà quando tutti i numeri da considerare saranno stati elaborati, cioè in uscita dal ciclo.

NOTE

NOTE

NOTE

NOTE

NOTE