Informatica di base e Laboratorio
a.a. 2022/2023

Sommario della Lezione

- **La codifica binaria dell'informazione**
  - Codifica dei numeri
  - Codifica dei caratteri
  - Codifica delle immagini

Introduzione

- In un calcolatore i dati e le istruzioni sono codificati in forma binaria, ovvero come sequenze finite di cifre 0 e 1
  - il **bit** è il più piccolo dato in un calcolatore
    - un bit può avere valore 0 oppure 1
    - la parola "**bit**" è una forma contratta per **bi**nary digi**t** (cifra binaria)
- Un bit è quindi un dato che può essere usato soltanto per rappresentare una informazione binaria
- Per rappresentare altre tipologie di informazioni sono necessarie sequenze di bit

Introduzione

- Le ragioni di questa scelta sono prevalentemente di tipo tecnologico:
  - Due possibili stati di polarizzazione di una sostanza magnetizzabile;
  - Passaggio/non passaggio di corrente attraverso un conduttore;
  - Passaggio/non passaggio della luce attraverso una fibra ottica.

Informazione binaria

- Dato un interruttore come posso rappresentare l'informazione sul suo stato?

- Possiamo rappresentare queste informazioni con un bit
  - Interruttore acceso = 1
  - Interruttore spento = 0

Informazioni più complesse

- E se abbiamo a che fare con una scelta tra più di due alternative?

  Useremo più di 1 bit!

- Per rappresentare informazioni più 'complesse' si concatenano più simboli

01 = donna	10 = uomo
00 = bambino	11 = bambina

Informazioni più complesse

- Supponiamo di voler rappresentare lo 'stato' di un semaforo con i bit.

- Con 3 bit potremmo rappresentare tutti gli stati:
  - 100 ⇔ ROSSO ACCESO, giallo spento, verde spento
  - 010 ⇔ Rosso spento, GIALLO ACCESO, verde spento
  - 001 ⇔ Rosso spento, Giallo spento, VERDE ACCESO

Informazioni più complesse

- In effetti per rappresentare tutti gli stati del semaforo sono sufficienti 2 bit!

- Metodo più "compatto"
  - 00 ⇔ ROSSO ACCESO
  - 01 ⇔ GIALLO ACCESO
  - 10 ⇔ VERDE ACCESO
  - 11 ⇔ Semaforo guasto

Informazioni più complesse

- In genere una sequenza di n bit potrà assumere un numero diverso di combinazioni pari a 2^n

- Un **byte** è una sequenza di 8 bit, quindi può assumere 256 valori diversi (2^8 combinazioni), ad esempio valori numerici da 0..255

Numero di bit	Informazioni rappresentabili
1	2
2	4
4	16
8	256
16	65.536
32	4.294.967.296

Notazione posizionale in base 10

- Un numero (es. 5) può essere rappresentato in molti modi :
  - cinque, *five*, *5*, *V*,
- Rappresentazioni diverse hanno proprietà diverse
  - moltiplicare due numeri in notazione romana è molto più difficile che moltiplicare due numeri in notazione decimale ….
- Noi siamo abituati a lavorare con numeri rappresentati in notazione posizionale in base 10

Base ed alfabeto

- I sistemi di numerazione posizionale sono caratterizzati da una base b ed un alfabeto a:
  - **Alfabeto (a)**: è l'insieme delle cifre disponibili per esprimere i numeri. Ad ogni cifra corrisponde un valore compreso tra 0 e (b-1)
    - Ad esempio, nel sistema decimale a={0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
  - **Base (b)**: è il numero degli elementi che costituiscono l'alfabeto (dimensione dell'alfabeto)
    - Ad esempio, nel sistema decimale b=10

Esempi:

- Base 8 (ottale) a={0, 1, 2, 3, 4, 5, 6, 7}
- Base 16 (esadecimale) a={0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}

Notazione posizionale in base 10

- La **rappresentazione di un numero intero in base 10 è una** sequenza di cifre scelte fra 0 1 2 3 4 5 6 7 8 9:
  - es: 23, 118, 4
- Il **valore di una rappresentazione $c_N...c_0$ *è dato da***

  $c_N * 10^N + C_{N-1} * 10^{N-1} ....+ c_1 * 10^1 + c_0 * 10^0$

  esempi :

  $23 = 2*10^1 + 3 * 10^0 = 20 + 3$

  $118 = 1*10^2 + 1*10^1 + 8 * 10^0 = 100 + 10 + 8$
- Simboli uguali assumono valori diversi a seconda della loro posizione nel numero
- Somma delle potenze del 10 pesate per il valore del simbolo corrispondente

Notazione posizionale in base 2

- La rappresentazione di un numero intero in base 2 è una sequenza di cifre scelte fra 0 1

  es: 10, 110, 1

- Il valore di una rappresentazione $c_N...c_0$ è dato da

$$c_N * 2^N + C_{N-1} * 2^{N-1} \ldots + c_1 * 2^1 + c_0 * 2^0$$

esempi :

$$10 = 1*2^1 + 0*2^0 = 2$$

$$110 = 1*2^2 + 1*2^1 + 0 * 2^0 = 4 + 2 + 0 = 6$$

$$1 = 1*2^0 = 1$$

Conversione da base 10 a base 2

Dato un numero X si cerca $c_N \ldots c_0$ sua rappresentazione in base 2

- Conversione per divisione :
  - si divide ripetutamente X per 2
  - il resto ottenuto nella divisione i-esima è la i esima cifra (c_i) della rappresentazione binaria

Conversione da base 10 a base 2

Come si converte X nella sua rappresentazione in base 2 $c_N \ldots c_0$ usando il metodo della divisione

- Es : convertiamo il numero 13
  - 13 / 2 da quoziente 6 e resto 1 (c_0)
  - 6 / 2 da quoziente 3 e resto 0 (c_1)
  - 3 / 2 da quoziente 1 e resto 1 (c_2)
  - 1 / 2 da quoziente 0 e resto 1 (c_3)
- La rappresentazione di 13 è 1101

Conversione da base 2 a base 10

binario → decimale (esempio)

$$1001_2 = (1x2^3 + 0x2^2 + 0x2^1 + 1x2^0)_{10} =$$
$$(1x8 + 0x4 + 0x2 + 1x1)_{10} =$$
$$9_{10}$$

La rappresentazione dei numeri all'interno di un computer

- Usa la notazione binaria
- Ogni numero viene rappresentato con un numero finito di cifre binarie (*bit)*
- Numeri di 'tipo' diverso hanno rappresentazioni diverse
  - es. interi positivi, interi (pos. e neg.), razionali, reali, complessi

Unità di misura

- BIT
- Byte = 8 bit
- Word = 16 bit
- Duble-word = 32 bit
- Quad-word = 64 bit
- 1 KB (kilobyte) = 1024 byte = 2^{10} byte
- 1 MB (Megabyte) = 1024 Kb = 2^{20} byte
- 1 GB (Gigabyte) = 1024 Mb = 2^{30} byte
- 1 TB (Terabyte) = 1024 Gb = 2^{40} byte
- 1 PT (Petabyte) = 1024 Tb = 2^{50} byte

Rappresentazione di un insieme finito di oggetti

- Vogliamo rappresentare i giorni della settimana :

  {Lu, Ma, Me, Gio, Ve, Sa, Do}

  usando sequenze 0 e 1

- Questo significa costruire un 'codice', cioè una tabella di corrispondenza che ad ogni giorno associa una opportuna sequenza
- In principio possiamo scegliere in modo del tutto arbitrario.

Rappresentazione di un insieme finito di oggetti

- Una possibile codifica binaria per i giorni della settimana

Lunedì	0100010001
Martedì	001
Mercoledì	1100000
Giovedì	1
Venerdì	101010
Sabato	111111
Domenica	000001

Rappresentazione di un insieme finito di oggetti

- Problema: la tabellina di corrispondenza fra codifiche tutte di lunghezza diversa
  - spreco di memoria
  - devo capire come interpretare una sequenza di codifiche

    11000001**1** = Me Gio Gio

    1**1**0000011 = Gio Gio Do Gio

- Di solito si usa un numero di bit uguale per tutti: il minimo indispensabile

Rappresentazione di un insieme finito di oggetti

- Per rappresentare 7 oggetti diversi servono almeno 3 bit (minima potenza di due che supera 7 è 8= 2^3) quindi:

000 Lunedì 110 Domenica

001 Martedì 111 *non ammesso*

010 Mercoledì

011 Giovedì

100 Venerdì

101 Sabato

Rappresentazione di caratteri e stringhe

- I caratteri sono un insieme finito di oggetti e seguono la strategia vista per i giorni della settimana
- Perché due diversi calcolatori si possano parlare correttamente è necessario che usino lo stesso codice
- Un **testo** non è altro che una ***successione di caratteri***, e i caratteri di base – quelli compresi nell'alfabeto della lingua usata - sono in un numero che varia col variare delle lingue, ma che è comunque finito.
- E' sufficiente allora stabilire una tabella di corrispondenza fra caratteri da un lato e numeri binari dall'altro

Rappresentazione di caratteri e stringhe

- Dovremo ricordarci di includere fra i caratteri da codificare tutti quelli che vogliamo effettivamente differenziare in un testo scritto: se vogliamo poter distinguere fra lettere maiuscole e minuscole dovremo dunque inserirvi l'intero alfabeto sia maiuscolo che minuscolo, se vogliamo poter inserire nei nostri testi anche dei numeri decimali dovremo inserire le dieci cifre (0, 1, 2 ,3 ,4 ,5 ,6 ,7, 8, 9), se vogliamo poter utilizzare segni di interpunzione (punto, virgola, punto e virgola...), dovremo inserire i caratteri corrispondenti, e cosi via... senza dimenticare naturalmente di includere lo spazio per separare una parola dall'altra!

Rappresentazione di caratteri e stringhe

- Codifiche di uso comune:
  - il codice ASCII (American Standard code For Information Interchange) su 7 o 8 bit
  - Il codice UNICODE su 16 bit (più recente permette di rappresentare anche alfabeti diversi e simboli per la scrittura di lingua orientali)

- Le stringhe sono generalmente sequenze di caratteri terminate in modo particolare

Rappresentazione di caratteri e stringhe

Byte	Cod.	Char
00000000	0	Null
00000001	1	Start of heading
00000010	2	Start of text
00000011	3	End of text
00000100	4	End of transmit
00000101	5	Enquiry
00000110	6	Acknowledge
00000111	7	Audible bell
00001000	8	Backspace
00001001	9	Horizontal tab
00001010	10	Line feed
00001011	11	Vertical tab
00001100	12	Form Feed
00001101	13	Carriage return
00001110	14	Shift out
00001111	15	Shift in
00010000	16	Data link escape
00010001	17	Device control 1
00010010	18	Device control 2
00010011	19	Device control 3
00010100	20	Device control 4
00010101	21	Neg. acknowledge
00010110	22	Synchronous idle
00010111	23	End trans. block
00011000	24	Cancel
00011001	25	End of medium
00011010	26	Substitution
00011011	27	Escape
00011100	28	File separator
00011101	29	Group separator
00011110	30	Record Separator
00011111	31	Unit separator

Byte	Cod.	Char
00100000	32	Spc
00100001	33	!
00100010	34	"
00100011	35	#
00100100	36	$
00100101	37	%
00100110	38	&
00100111	39	'
00101000	40	(
00101001	41	)
00101010	42	*
00101011	43	+
00101100	44	,
00101101	45	-
00101110	46	.
00101111	47	/
00110000	48	0
00110001	49	1
00110010	50	2
00110011	51	3
00110100	52	4
00110101	53	5
00110110	54	6
00110111	55	7
00111000	56	8
00111001	57	9
00111010	58	:
00111011	59	;
00111100	60	<
00111101	61	=
00111110	62	>
00111111	63	?

Byte	Cod.	Char
01000000	64	@
01000001	65	A
01000010	66	B
01000011	67	C
01000100	68	D
01000101	69	E
01000110	70	F
01000111	71	G
01001000	72	H
01001001	73	I
01001010	74	J
01001011	75	K
01001100	76	L
01001101	77	M
01001110	78	N
01001111	79	O
01010000	80	P
01010001	81	Q
01010010	82	R
01010011	83	S
01010100	84	T
01010101	85	U
01010110	86	V
01010111	87	W
01011000	88	X
01011001	89	Y
01011010	90	Z
01011011	91	[
01011100	92	\
01011101	93	]
01011110	94	^
01011111	95	_

Byte	Cod.	Char
01100000	96	`
01100001	97	a
01100010	98	b
01100011	99	c
01100100	100	d
01100101	101	e
01100110	102	f
01100111	103	g
01101000	104	h
01101001	105	i
01101010	106	j
01101011	107	k
01101100	108	l
01101101	109	m
01101110	110	n
01101111	111	o
01110000	112	p
01110001	113	q
01110010	114	r
01110011	115	s
01110100	116	t
01110101	117	u
01110110	118	v
01110111	119	w
01111000	120	x
01111001	121	y
01111010	122	z
01111011	123	{
01111100	124	|
01111101	125	}
01111110	126	~
01111111	127	Del

Rappresentazione di caratteri e stringhe

- Quindi una stringa di caratteri sarà rappresentata dal computer come una successione di gruppi di 8 bit (un byte)

O	G	G	I		P	I	O	V	E
01001111	01000111	01000111	01001001	00100000	01010000	01001001	01001111	01010110	01000101

Rappresentazione di caratteri e stringhe

- Consideriamo il problema inverso: data una sequenza di bit, il testo che essa codifica può essere così ottenuto:
  - Si divide la sequenza in gruppi di 8 bit
  - Si determina il carattere corrispondente ad ogni byte

- Esempio

01101001	01101100	00100000	01110000	01101111	00101110
i	l		p	o	.

Rappresentazione di immagini

- Le immagini sono un ‘continuo’ e non sono formate da sequenze di oggetti ben definiti come i numeri e le stringhe

- Bisogna quindi prima ‘discretizzarle’ ovvero trasformarle in un insieme di parti distinte che possono essere codificate separatamente con sequenze di bit

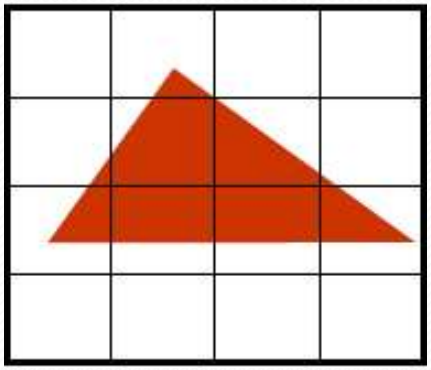
Pixels (picture elements)

L'idea di base:

- L'immagine viene suddivisa in una griglia di cellette
- Ogni celletta corrisponde a un 'puntino' (**pixel**) dell'immagine
- Tanto più è fitta la griglia (più numerose sono le cellette), tanto migliore è la risoluzione dell'immagine.

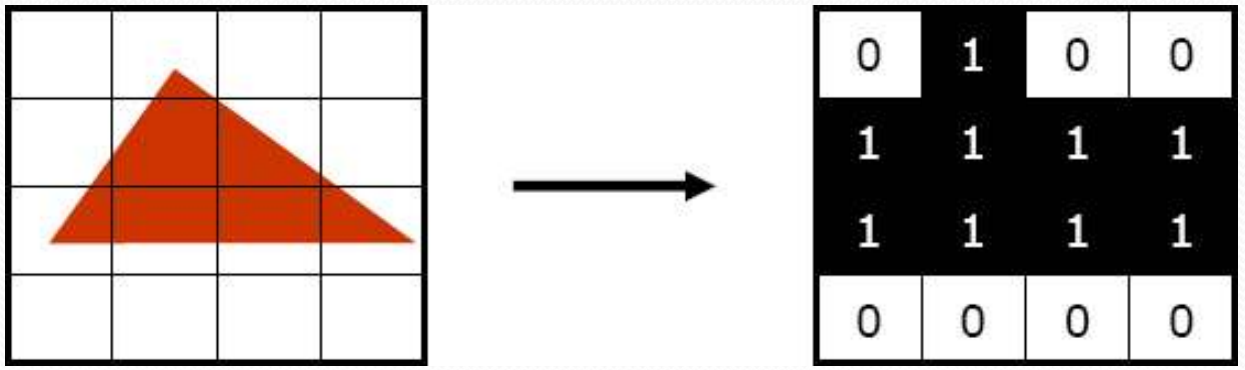
Rappresentazione di immagini

- La scomposizione di un'immagine secondo una griglia di punti è una modalità standard non solo di visualizzare un'immagine ma anche di gestirla e salvarla tramite un computer.



Rappresentazione di immagini

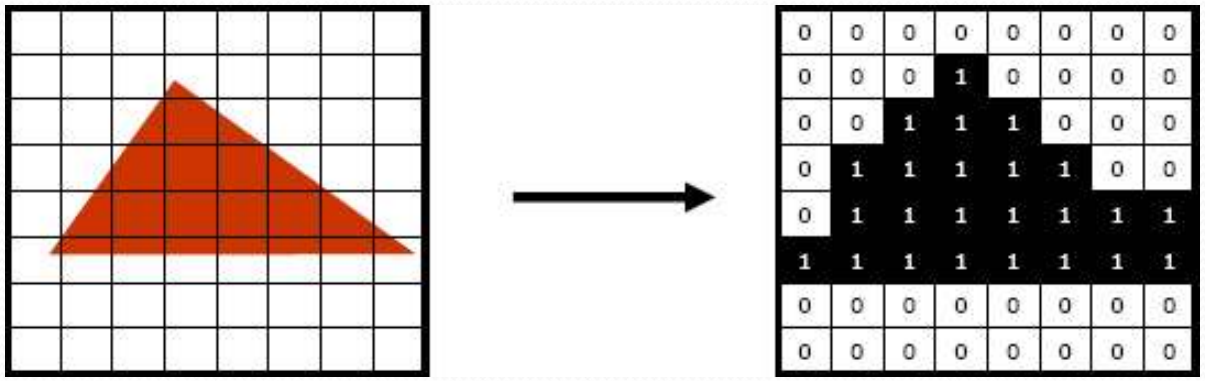
- Supponiamo ora che ogni cella (pixel) della nostra griglia possa assumere solo un valore **0** o **1** (immagine in bianco e nero) e coloriamo in nero i quadrati dove c'è l'immagine



Il nostro triangolo digitalizzato ad una risoluzione 4x4 pixel (occupa 2 byte)

Rappresentazione di immagini

- Se aumentiamo la risoluzione con cui campioniamo la nostra immagine otterremo una rappresentazione sempre più fedele, ad esempio con una griglia 8x8 abbiamo:



Il nostro triangolo digitalizzato ad una risoluzione 8x8 pixel (occupa 8 byte)

Rappresentazione di immagini

Rappresentazioni dei pixel:

- La rappresentazione in 'toni di grigio': un byte per pixel, con 256 gradazioni di grigio per ogni punto
- Rappresentazione a colori RGB (red, green, blu): comunemente 3 byte per pixel che definiscono l'intensità di ciascun colore base. In questo modo ho circa 16 milioni di colori diversi definibili.

Rappresentazione di immagini

- Problema:
  - La rappresentazione accurata di una immagine dipende
    - dal numero di pixel (definizione)
    - dalla codifica del pixel
  - … e richiede generalmente molta memoria, ad esempio

tipo	*definizione*	*num. Colori*	*num. byte*
Imm. Televisiva	720x625	256	440 KB
SVGA	1024x768	65536	1.5 MB

Riepilogando

- Per informazione intendiamo tutto quello che viene manipolato da un calcolatore:
  - numeri (naturali, interi, reali, . . .)
  - caratteri
  - immagini
  - suoni
  - programmi

- La più piccola unità di informazione memorizzabile o elaborabile da un calcolatore, il bit, corrisponde allo stato di un dispositivo fisico che viene interpretato come 1 o 0.
- In un calcolatore tutte le informazioni sono rappresentate in forma binaria, come sequenze di 0 e 1.
- Per motivi tecnologici: distinguere tra due valori di una grandezza fisica è più semplice che non ad esempio tra dieci valori.

…motivi tecnologici

- All'interno del computer le informazioni vengono rappresentate da 2 possibili valori di tensione elettrica
- Per convenzione, i simboli di un alfabeto binario sono 0 e 1
- In generale, a seconda del tipo di dispositivo considerato, i valori zero e uno sono rappresentati:
  - da una tensione elettrica (alta, bassa)
  - da un differente stato di polarizzazione magnetica (positiva, negativa)
  - da luce e buio
  - …

Riepilogando

- Comunicazione è scambio di messaggi (Mondo reale) Messaggio: successione di simboli appartenenti a più di un alfabeto (lettere, cifre, note musicali, …)
  - Testo = una successione di caratteri
  - Numero = una successione di cifre e simboli
  - Suono = successione di note

- (Computer) Ogni messaggio è una successione di simboli appartenenti ad un solo alfabeto: l'alfabeto binario

Riepilogando

- Per rappresentare l'informazione del mondo reale nel computer è necessario 'tradurre' (codificare) i "messaggi reali" in messaggi binari

- Codificare vuol dire determinare delle regole di corrispondenza dette codifiche

- La codifica mette in corrispondenza biunivoca ogni simbolo appartenente ad un alfabeto più grande con una stringa di simboli appartenente ad un alfabeto meno grande