



Un ripasso di aritmetica: Rappresentazione binaria - operazioni

- A queste rappresentazioni si possono applicare le operazioni aritmetiche:

riporti

1

1 0 +

1 0 =

1 0 0



Un ripasso di aritmetica: Rappresentazione binaria - operazioni

- A queste rappresentazioni si possono applicare le operazioni aritmetiche:

riporti

1

1 1 +

1 0 =

1 0 1



Un ripasso di aritmetica: Rappresentazione binaria - operazioni

- A queste rappresentazioni si possono applicare le operazioni aritmetiche:

riporti

$$\begin{array}{r} 1 \quad 1 \\ \quad 1 \quad 1 \quad + \\ \quad 1 \quad 1 \quad = \\ 1 \quad 1 \quad 0 \end{array}$$



Un ripasso di aritmetica: Rappresentazione binaria - operazioni

- A queste rappresentazioni si possono applicare le operazioni aritmetiche:

riporti

$$\begin{array}{r} 1 \ 1 \ 1 \\ 1 \ 1 \ 1 \ + \\ 1 \ 1 \ = \\ 1 \ 0 \ 1 \ 0 \end{array}$$



Un ripasso di aritmetica: Conversione dalla base 10 alla base 2

- Dato un numero N rappresentato in base dieci, la sua rappresentazione in base due sarà del tipo:

$c_m c_{m-1} \dots c_1 c_0$ (le " c_i " sono cifre binarie)

- Come possiamo determinare queste cifre?
 - Si deve calcolare la divisione intera di N per 2: $N/2=N'$ con resto R'
 - R' è la cifra più a destra nella rappresentazione binaria di N , cioè $c_0 = R'$
 - Si divide $N'/2$ ottenendo $N'/2 = N''$ con resto R'' e si ha che $c_1 = R''$
 - Si ripete il procedimento fino a quando il risultato della divisione è uguale a 0



Un ripasso di aritmetica: Conversione dalla base 10 alla base 2

- Consideriamo ad esempio il numero 13_{10} e calcoliamo la sua rappresentazione in base due:

$$\begin{array}{rcll} 13/2 & = & 6 & \text{resto } 1 \\ 6/2 & = & 3 & \text{resto } 0 \\ 3/2 & = & 1 & \text{resto } 1 \\ 1/2 & = & 0 & \text{resto } 1 \end{array}$$

- Leggendo i resti dal basso verso l'alto, si ha che la rappresentazione binaria del numero 13_{10} è 1101_2



Un ripasso di aritmetica: Conversione dalla base 10 alla base 2

- Consideriamo ad esempio il numero 42_{10} e calcoliamo la sua rappresentazione in base due:

| | | | |
|--------|---|----|---------|
| $42/2$ | = | 21 | resto 0 |
| $21/2$ | = | 10 | resto 1 |
| $10/2$ | = | 5 | resto 0 |
| $5/2$ | = | 2 | resto 1 |
| $2/2$ | = | 1 | resto 0 |
| $1/2$ | = | 0 | resto 1 |

- Leggendo i resti dal basso verso l'alto, si ha che la rappresentazione binaria del numero 42_{10} è 101010_2



Un ripasso di aritmetica: Conversione dalla base 10 alla base 2

- Consideriamo ad esempio il numero 345_{10} e calcoliamo la sua rappresentazione in base due:

| | | | | |
|---------|-----|-----|-------|---|
| $345/2$ | $=$ | 172 | resto | 1 |
| $172/2$ | $=$ | 86 | resto | 0 |
| $86/2$ | $=$ | 43 | resto | 0 |
| $43/2$ | $=$ | 21 | resto | 1 |
| $21/2$ | $=$ | 10 | resto | 1 |
| $10/2$ | $=$ | 5 | resto | 0 |
| $5/2$ | $=$ | 2 | resto | 1 |
| $2/2$ | $=$ | 1 | resto | 0 |
| $1/2$ | $=$ | 0 | resto | 1 |

- Leggendo i resti dal basso verso l'alto, si ha che la rappresentazione binaria del numero 345_{10} è 101011001_2



Un ripasso di aritmetica: Conversione dalla base 10 alla base 16

- Dato un numero N rappresentato in base dieci, la sua rappresentazione in base sedici sarà del tipo:

$c_m c_{m-1} \dots c_1 c_0$ (le " c_i " sono cifre esadecimale)

- Come possiamo determinare queste cifre?
 - Si deve calcolare la divisione intera di N per 16: $N/16=N'$ con resto R'
 - R' è la cifra più a destra nella rappresentazione esadecimale di N , cioè $c_0 = R'$
 - Si divide $N'/16$ ottenendo $N'/16 = N''$ con resto R'' e si ha che $c_1 = R''$
 - Si ripete il procedimento fino a quando il risultato della divisione è uguale a 0



Un ripasso di aritmetica: Conversione dalla base 10 alla base 16

- Consideriamo ad esempio il numero 345_{10} e calcoliamo la sua rappresentazione in base sedici:

$$\begin{array}{rcl} 345/16 = & 21 & \text{resto } 9 \\ 21/16 = & 1 & \text{resto } 5 \\ 1/16 = & 0 & \text{resto } 1 \end{array}$$

- Leggendo i resti dal basso verso l'alto, si ha che la rappresentazione esadecimale del numero 345_{10} è 159_{16}



Un ripasso di aritmetica: Conversione dalla base 2 alla base 10

- Sia $c_m c_{m-1} \dots c_1 c_0$ un numero rappresentato in base 2, per trovare la rappresentazione decimale di questo numero dobbiamo considerare le potenze successive della base 2

$$c_0 \times 2^0 + c_1 \times 2^1 + \dots + c_{m-1} \times 2^{m-1} + c_m \times 2^m = N$$

- Esempio: 101011001_2

$$1 \times 2^0 + 0 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 + 1 \times 2^4 + 0 \times 2^5 + 1 \times 2^6 + 0 \times 2^7 + 1 \times 2^8 = 1 + 8 + 16 + 64 + 256 = 345$$



Un ripasso di aritmetica: Conversione dalla base 16 alla base 10

- Sia $c_m c_{m-1} \dots c_1 c_0$ un numero rappresentato in base 16, per trovare la rappresentazione decimale di questo numero dobbiamo considerare le potenze successive della base 16

$$c_0 \times 16^0 + c_1 \times 16^1 + \dots + c_{m-1} \times 16^{m-1} + c_m \times 16^m = N$$

- Esempio: 159_{16}

$$9 \times 16^0 + 5 \times 16^1 + 1 \times 16^2 = 9 + 80 + 256 = 345$$



Rappresentazione digitale dei numeri

- **FINE DEL RIPASSO DI ARITMETICA!!!!**
- Come si ottiene la rappresentazione usando bit?
 - si calcola la rappresentazione in base 2 del numero
 - si associa ad ogni cifra binaria della rappresentazione in base 2 un bit
 - se la cifra binaria vale 0 si associa un bit che vale 0
 - se la cifra binaria vale 1 si associa un bit che vale 1



Rappresentazione digitale dei numeri

- Lo stesso numero può essere codificato in modi diversi:

ASCII: 37 00110011 00110111 (2 byte)

3 7

BINARIA: 37 00100101 (1 byte)

- Il primo modo è usato per le comunicazioni con l'esterno (input/output: ingresso/uscita)
- Il secondo modo è usato all'interno del calcolatore per fare i calcoli; a questo fine non è possibile usare direttamente le codifiche ASCII:

| Esempio: | Numero | ASCII |
|----------|--------|----------|
| | 3 + | 00110011 |
| | 2 = | 00110010 |
| | ---- | ----- |
| | e | 01100101 |

- Esistono dei programmi di conversione che trasformano i numeri da una codifica all'altra



Rappresentazione di numeri negativi e dei numeri reali

- In realtà, una semplice codifica binaria come quella discussa fino ad ora non è sufficiente, per due motivi:
 - numeri negativi
 - numeri con la virgola
- Per questi numeri vengono utilizzate delle rappresentazioni differenti



La codifica delle immagini

Lettere e numeri non costituiscono le uniche informazioni utilizzate dagli elaboratori ma si stanno diffondendo sempre di più applicazioni che utilizzano ed elaborano anche altri tipi di informazione: **diagrammi, immagini, suoni, filmati**. Spesso in questi casi si parla di applicazioni di tipo

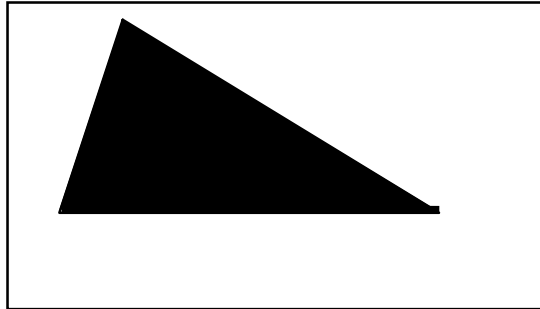
MULTIMEDIALE



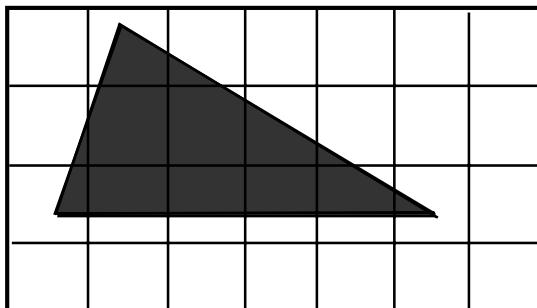
La codifica delle immagini

- Vediamo dapprima il caso delle immagini
- Esistono numerose tecniche che vengono utilizzate per la memorizzazione digitale e l'elaborazione di un'immagine
- Consideriamo un'immagine in bianco e nero, senza ombreggiature o livelli di chiaroscuro

La codifica delle immagini



Suddividiamo l'immagine mediante una griglia formata da righe orizzontali e verticali a distanza costante



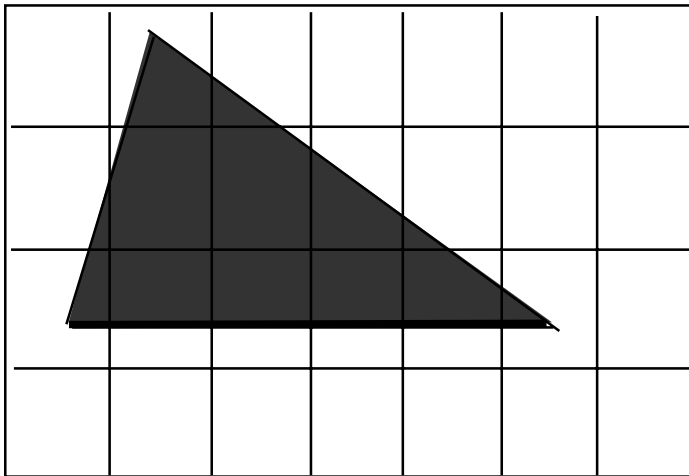


La codifica delle immagini

- Ogni quadratino derivante da tale suddivisione prende il nome di **pixel** (**picture element**) e può essere codificato in binario secondo la seguente convenzione:
 - Il simbolo "0" viene utilizzato per la codifica di un pixel corrispondente ad un quadratino bianco (in cui il bianco è predominante)
 - Il simbolo "1" viene utilizzato per la codifica di un pixel corrispondente ad un quadratino nero (in cui il nero è predominante)

La codifica delle immagini

- Poiché una sequenza di bit è lineare, è necessario definire delle convenzioni per ordinare la griglia dei pixel in una sequenza. Assumiamo che i pixel siano ordinati dal basso verso l'alto e da sinistra verso destra



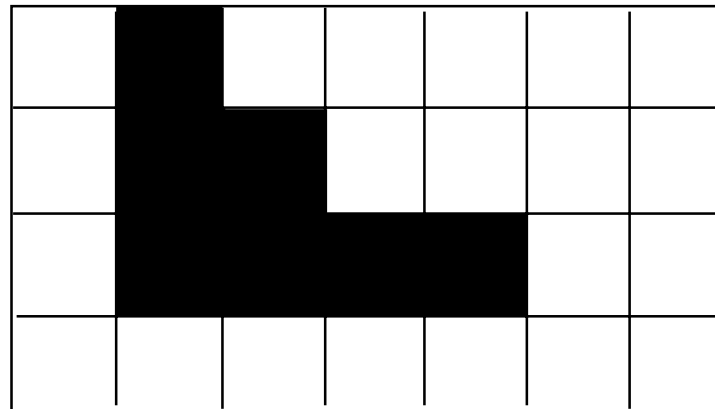
| | | | | | | |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| 0 ₂₂ | 1 ₂₃ | 0 ₂₄ | 0 ₂₅ | 0 ₂₆ | 0 ₂₇ | 0 ₂₈ |
| 0 ₁₅ | 1 ₁₆ | 1 ₁₇ | 0 ₁₈ | 0 ₁₉ | 0 ₂₀ | 0 ₂₁ |
| 0 ₈ | 1 ₉ | 1 ₁₀ | 1 ₁₁ | 1 ₁₂ | 0 ₁₃ | 0 ₁₄ |
| 0 ₁ | 0 ₂ | 0 ₃ | 0 ₄ | 0 ₅ | 0 ₆ | 0 ₇ |

Con questa convenzione la rappresentazione della figura sarà data dalla sequenza di bit

00000000111000100000100000

La codifica delle immagini

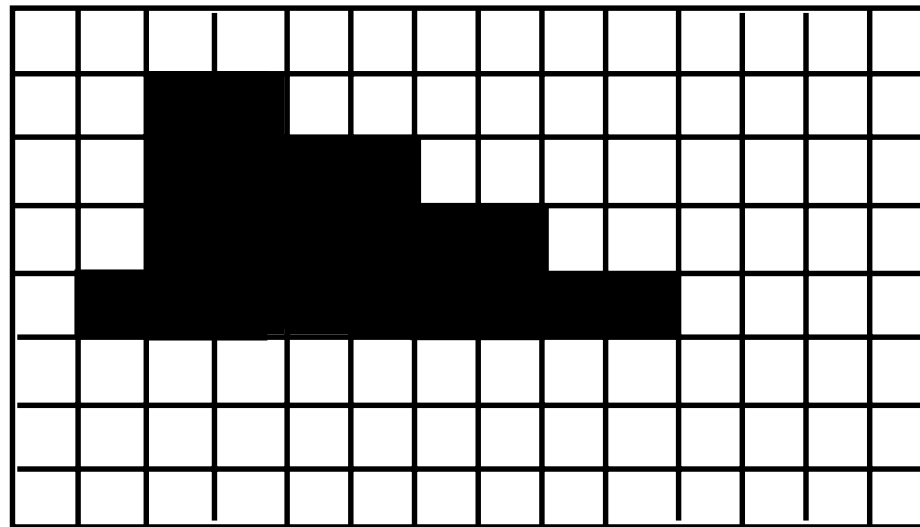
- Non sempre il contorno della figura coincide con le linee della griglia. Quella che si ottiene nella codifica è un'approssimazione della figura originaria
- Se riconvertiamo la stringa **0000000011110001100000100000** in immagine otteniamo



LA DIGITALIZZAZIONE COMPORTA PERDITA
DI QUALITÀ

La codifica delle immagini

- La rappresentazione sarà più fedele all'aumentare del numero di pixel, ossia al diminuire delle dimensioni dei quadratini della griglia in cui è suddivisa l'immagine



AUMENTIAMO LA QUALITÀ AUMENTANDO LA
GRANULARITÀ DELL'IMMAGINE



La codifica delle immagini

- Assegnando un bit ad ogni pixel è possibile codificare solo immagini senza livelli di chiaroscuro
- Le immagini in bianco e nero hanno delle sfumature (diversi livelli di intensità di grigio)
- Per codificare le immagini con diversi livelli di grigio si usa la stessa tecnica: per ogni pixel si stabilisce il livello medio di grigio cui viene assegnata convenzionalmente una rappresentazione binaria
- Per memorizzare un pixel non è più sufficiente un solo bit. Ad esempio, se utilizziamo quattro bit possiamo rappresentare $2^4=16$ livelli di grigio, mentre con otto bit ne possiamo distinguere $2^8=256$, ecc.



La codifica delle immagini

- Analogamente possiamo codificare le immagini a colori. In questo caso si tratta di individuare un certo numero di sfumature, gradazioni di colore differenti e di codificare ognuna mediante un'opportuna sequenza di bit
- Qualsiasi colore può essere rappresentato dalla composizione del rosso, del verde e del blu.
- Quindi, invece che rappresentare alcune sfumature di tanti colori diversi, possiamo rappresentare molte sfumature dei tre colori primari: dalla combinazione di essi otteniamo tanti altri colori.




La codifica delle immagini

- Codifica **RGB** (*Red, Green, Blu* - Rosso, Verde, Blu ovvero i tre colori primari).
- Ogni pixel viene rappresentato con una combinazione dei tre colori
- Per ogni colore primario si usa un certo numero di bit per rappresentarne la gradazione (la "quantità")
- Ad esempio, utilizzando 8 bit per colore primario, otteniamo 256 diverse gradazioni, ovvero $256 \times 256 \times 256 = 16777216$ colori diversi. In questo caso un pixel richiede tre byte di informazione



La codifica delle immagini

Le sequenze di bit relative ad ogni colore primario si possono interpretare come la rappresentazione di una quantità (la gradazione, la sfumatura) quindi si possono esprimere in base decimale:

Ad esempio, se il colore di un pixel fosse  sarebbe espresso con la seguente sequenza di bit: **100010111101001011011000** allora potremmo, più comodamente, scrivere **139 210 216**.

Avendo 8 bit a disposizione per rappresentare la gradazione di un colore fondamentale, tutti e tre i numeri sono compresi tra 0 e 255

Spesso, per comodità di scrittura, tale codifica è espressa in base esadecimale. In questo modo, lo stesso colore dell'esempio sarebbe espresso nel modo seguente: **8B D2 D8** ($8B_{16} = 139_{10}$, e così via...).

100010111101001011011000 equivale a **139 210 216** ed a **8B D2 D8**



La codifica delle immagini

- La rappresentazione di un'immagine mediante la codifica dei pixel, viene chiamata codifica **bitmap** o **raster**
- La **risoluzione** dell'immagine è il numero di pixel che la costituiscono, espressi in termini di *larghezza x altezza*. Ovviamente, aumentando il numero di pixel a disposizione, migliora la qualità dell'immagine.
- La **profondità** dell'immagine è invece il numero di bit che servono per rappresentare un singolo pixel dell'immagine.
- Il numero di bit richiesti per memorizzare un'immagine dipende dalla risoluzione e dalla profondità

numero di bit per immagine = risoluzione x profondità



La codifica delle immagini

- Per distinguere 16777216 colori sono necessari 24 bit per la codifica di ciascun pixel: la codifica di un'immagine formata da 640X480 pixel richiederà 7.372.800 bit (921.600 byte)
- Esistono delle tecniche di compressione delle informazioni che consentono di ridurre drasticamente lo spazio occupato dalle immagini
- **codifiche di compressione:** le più famose sono la *CompuServe Graphic Interface (GIF)* e la *Joint Photographic Experts Group (JPEG)*. I file che usano tali codifiche riportano rispettivamente le estensioni .gif e .jpg (o anche .jpeg)



La codifica delle immagini

- Tali formati (detti anche **codec**: *compression/decompression*), usano un sistema per comprimere l'informazione prima di memorizzarla e per decomprimerla prima di visualizzarla.
- entrambi i formati tendono ad eliminare i pixel ripetitivi,
- Entrambe le soluzioni sono *compressioni con perdita* di informazione. Tale perdita non può essere recuperata in alcun modo. La codifica JPEG consente di manipolare tale fattore di compressione.