Informatica Applicata ai Beni Culturati
A.A. 2012/2013

La Codifica e la Rappresentazione dei dati

Dott.ssa Annamaria Bria
www.mat.unical.it/bria

Problema

Molte azioni che si compiono sono finalizzate alla *soluzione* di *problemi*

Ogni ***problema*** è caratterizzato da un insieme di ***dati partenza*** e da un ***risultato***

Ogni sua soluzione è una procedura che genera un risultato sulla base dei dati indicati

Problema

La ***soluzione*** deve essere descritta in una forma che il soggetto esecutore sa interpretare in modo corretto.

I calcolatori sono esecutori di soluzioni che esseri umani hanno identificato e descritto.

Esempio

- Quanti quadrati di lato 2 cm posso inserire in un rettangolo di base 6 e altezza 4?

Dati di partenza:

Esempio

- Quanti quadrati di lato 2 cm posso inserire in un rettangolo di base 6 e altezza 4?

Risultato:

Numero di

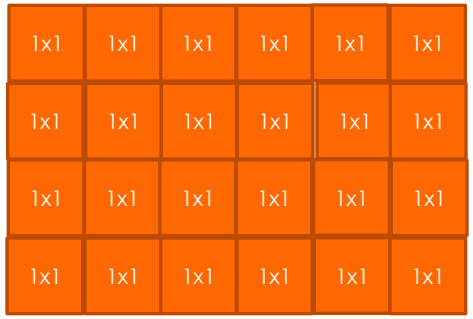
Quad. 2x2

Esempio

Calcolare l'area del rettangolo

Area = Base x altezza = 6 x 4 = 24 cm^2

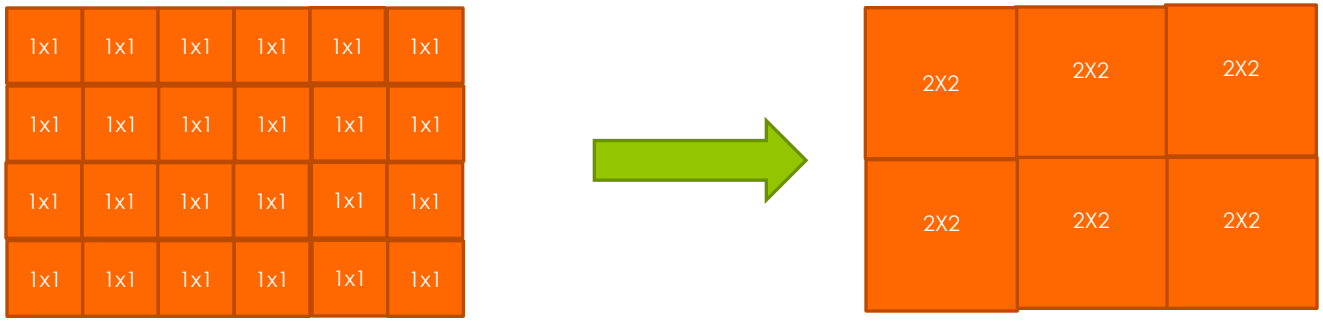
Rettangolo
6x4



Esempio

Dividere a Metà L'area

N° quadrati 2X2 = Area/2 = 24 : 2 = 12



Programma

La descrizione della soluzione di una problema per Il calcolatore è chiamata ***Programma***

Insieme di azioni finite e non abigue che deve compiere un calcolatore per risolvere un problema

Il programma è descritto in un linguaggio formale:
linguaggio di Programmazione

Sviluppo di un programma

1. *Analisi* del problema e identificazione di una soluzione
2. *Formalizzazione* della soluzione: *Algoritmo*
3. *Programmazione* descrizione della soluzione in un linguaggio di programmazione
4. *Traduzione** del programma nel linguaggio del calcolatore:

Linguaggio Macchina

* Questa fase avviene in modo automatico attraverso opportuni strumenti software: compilatori e interpreti

Esempio Algoritmo

Base = 6;
Altezza =4;
Lato quadrato = 2;
Area = Base x Altezza
Numero Quadrati = Area/2
Stampa Numero Quadrati

Esempio Programma

```
int main(){
int base = 6;
int altezza = 4;
int lato = 2;
int area, risultato;
area = base x altezza;
risultato = area / 2;
cout<<‘Numero quadrati = ‘<< risultato;
Return 0;
}
```

Esempio Linguaggio Macchina

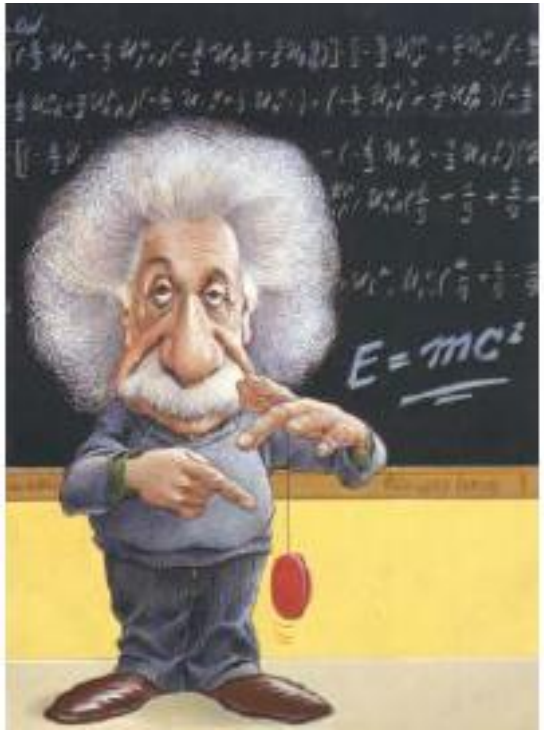
```
00100010 001001001
0010 00010 0001001001
00010101 0101010 101010000
10001001 00101001 010010
```

La codifica dell'Informazione

I codici

- Impiegati per codificare l'informazione in ambiti dove non è consentita ambiguità (logica, matematica, informatica…)

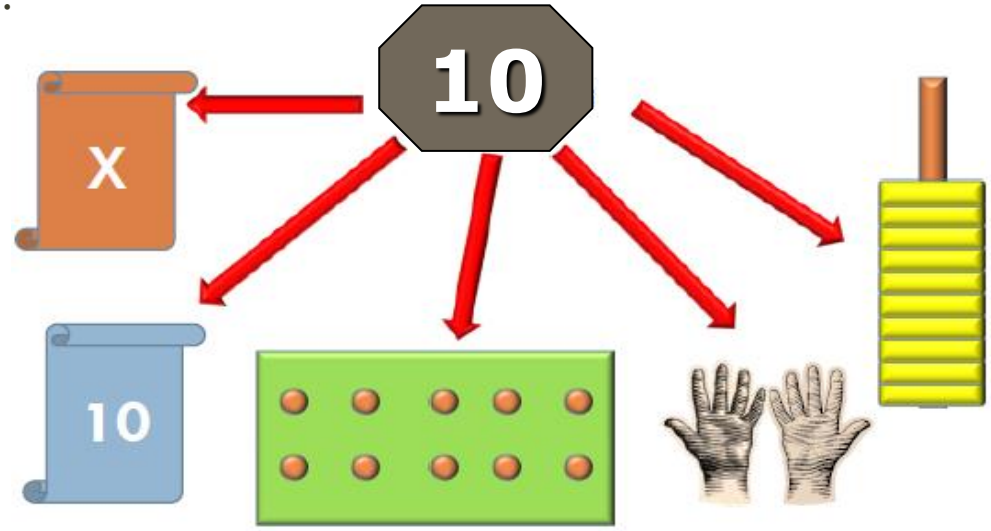
- Definiti da:
  - **alfabeto**: elenco (finito) di simboli
  - **grammatica**: regole sintattiche che specificano le combinazioni valide di simboli
  - **semantica**: attribuisce significato alle combinazioni valide di simboli



La codifica dell'Informazione

Codifica

- **Una codifica è una regola per associare in modo univoco i valori di un dato da codificare con sequenze di simboli.**
- La corrispondenza definita dalla codifica è arbitraria, è una convenzione.
- Essa deve pertanto essere nota e rispettata da chi genera e da chi utilizza i dati.



Sistemi per codificare l'informazione numerica

Lampione:

- Due stati
- 2 informazioni

Semaforo:

- Tre stati
- 3 informazioni

- Condizione necessaria perché un sistema sia in grado di portare informazione è che esso possa assumere configurazioni differenti (a ognuna delle quali può essere associata una differente entità d'informazione)

Dispositivi bistabili

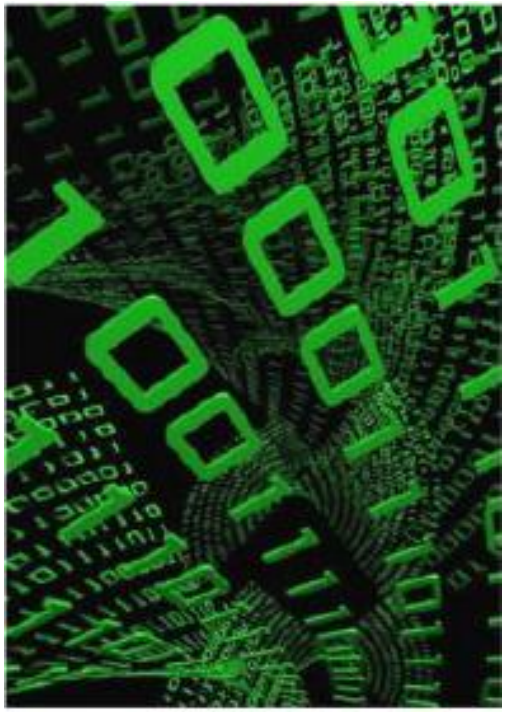
Dispositivi fisici in grado di assumere in modo stabile una configurazione tra due diverse possibili

ALFABETO = 0,1

Codifica dell'informazione

- Il calcolatore memorizza ed elabora vari tipi di informazioni
  - Numeri, testi, immagini, suoni
- Occorre rappresentare tale informazione in formato facilmente manipolabile dall'elaboratore
- A causa della **natura fisica a due stati degli elementi di base che costituiscono la memoria di un calcolatore**, l'unico alfabeto che è compreso dalla macchina è il binario.

Codifica Binaria dell'Informazione



- Nei moderni calcolatori tutta l'informazione (numerica, testuale, grafica, audio, video…) è codificata mediante sequenze di bit
- Usiamo cioè una rappresentazione binaria (a due valori) dell'informazione
- L'unità minimale di rappresentazione è il **BIT** (**BInary digiT** – cifra digitale): "**0**" e "**1**"

Informazioni complesse

Con 1 bit rappresentiamo 2 diverse informazioni

Esempio

Stato di una porta: Codifica

0 → Porta Chiusa
1 → Porta Aperta

Stato di una porta: De-Codifica

0 → Chiusa
1 → Aperta

Esempio

Le stagioni

0 → Primavera

1 → Estate

0 → Autunno

1 → Inverno

Le Stagioni: De-Codifica

0 → Primavera o Autunno???

1 → Estate o Inverno???

Informazioni complesse

Con 1 bit rappresentiamo **solo** 2 diverse informazioni:

si/no – vero/falso - on/off - 0/1

Mettendo insieme più bit possiamo rappresentare più informazioni:

00 / 01 / 10 / 11

Informazioni complesse si memorizzano come sequenze di bit

Esempio

Le stagioni

00 → Primavera

10 → Estate

01 → Autunno

11 → Inverno

Le Stagioni: De-Codifica

00 → Primavera

11 → Inverno

Sequenze di bit

Con **2 bit** riesco a rappresentare 4 ($=2^2$) sequenze possibili:

00

01

10

11

e quindi **4 informazioni diverse**

Quanti bit mi servono per rappresentare i giorni della settimana?

Informazioni complesse

Per rappresentare **M** informazioni dobbiamo usare **N** bit, in modo che $2^N >= M$

$2^3 > 7$

viceversa

In generale, con **N** bit, ognuno dei quali può assumere **2** valori, possiamo rappresentare 2^N informazioni diverse (**tutte le possibili combinazioni di 0 e 1 su N posizioni**)

Sequenze di bit

- 2 bit: 4 (=2^2) sequenze possibili:
  00 01 10 11
- 3 bit: 8 (=2^3) sequenze possibili:
  000 001 010 011 100 101 110 111
- …
- 8 bit: 256 (=2^8) sequenze possibili:
  00000000 00000001 00000010 … 11111100 11111101 11111110 11111111
- n bit: 2^n sequenze possibili

Esempio

Per rappresentare **57** informazioni diverse dobbiamo usare gruppi di almeno **6** bit. Infatti:

$$2^6 = 64 > 57$$

Cioè un gruppo di 6 bit può assumere 64 configurazioni diverse:
000000 / 000001 / 000010 …/ 111110 / 111111

Il Byte

Una sequenza di **8 bit** viene chiamata ***Byte***

- 0 0 0 0 0 0 0 0
- 0 0 0 0 0 0 0 1
- .....

byte = 8 bit = 2^8 = 256 informazioni diverse

Usato come unità di misura per indicare

- le dimensioni della memoria
- la velocità di trasmissione
- la "potenza" di un elaboratore

Usando sequenze di byte (e quindi di bit) si possono rappresentare caratteri, numeri immagini, suoni.

Altre unità di misura

- KiloByte (**KB**), MegaByte (**MB**), GigaByte (**GB**)
- Per ragioni storiche in informatica Kilo, Mega, e Giga indicano le ***potenze di 2*** che più si avvicinano alle corrispondenti potenze di 10
- Più precisamente
  - 1 KB = 1024 x 1 byte = $2^{10} \sim 10^3$ byte
  - 1 MB = 1024 x 1 KB = $2^{20} \sim 10^6$ byte
  - 1 GB = 1024 x 1 MB = $2^{30} \sim 10^9$ byte

Codifica dei numeri

Informatica Applicata ai Beni Culturati
A.A. 2012/2013 Dott.ssa Annamaria Bria

Il sistema decimale

- 10 cifre di base: 0, 1, 2, …, 9

- **Notazione posizionale:** la posizione di una cifra in un numero indica il suo **peso** in potenze di **10**. I pesi sono:
  - unità = $10^0 = 1$ (posiz. 0-esima)
  - decine = $10^1 = 10$ (posiz. 1-esima)
  - centinaia = $10^2 = 100$ (posiz. 2-esima)
  - migliaia = $10^3 = 1000$ (posiz. 3-esima)
  - … .. .. …

Esempio di numero rappresentato in notazione decimale

Il **numerale** 2304 in notazione decimale (o in base 10) rappresenta la quantità:

$2304 = 2*10^3 + 3*10^2 + 0*10^1 + 4*10^0 =$

$2000 + 300 + 0 + 4 = 2304$ (**numero**)

Nota: numero e numerale qui coincidono, perché il sistema decimale è quello adottato come sistema di riferimento.

Il sistema binario

- 2 Cifre di base: 0 e 1.

- **Notazione posizionale:** la posizione di una cifra in un numero binario indica il suo **peso** in potenze di **2**. I pesi sono:
  - $2^0 = 1$ (posiz. 0-esima)
  - $2^1 = 2$ (posiz. 1-esima)
  - $2^2 = 4$ (posiz. 2-esima)
  - $2^3=8$; $2^4=16$; $2^5=32$; $2^6=64$; $2^7=128$; $2^8=256$; $2^9=512$; $2^{10} = 1024$; $2^{11}=2048$, $2^{12}=4096$;...

Esempio di numero rappresentato in notazione binaria

Il **numerale** 10100101 in notazione binaria (o in base 2) rappresenta la quantità:

10100101

$1*2^7+0*2^6+1*2^5+0*2^4+0*2^3+1*2^2+0*2^1+1*2^0$

128 + 0 + 32 + 0 + 0 + 4 + 0 + 1 =

165 (**numero**)

Il numero più grande rappresentato con N cifre

- Sist. Decimale = 99…99 = $10^N - 1$
- Sist. Binario= 11..11 = $2^N - 1$
- **Esempio**: 11111111 (8 bit binari) = $2^8 - 1 = 255$.

  Per rappresentare il n. 256 ci vuole un bit in più:

  100000000 = $1*2^8 = 256$.

Quindi…

Fissate quante cifre (bit) sono usate per rappresentare i numeri, si fissa anche il numero più grande che si può rappresentare:

- con 16 bit: $2^{16} - 1 = 65.535$
- con 32 bit: $2^{32} - 1 = 4.294.967.295$
- con 64 bit: $2^{64} - 1 =$ circa $1,84 * 10^{19}$

Conversione da base 2 a base 10

Basta moltiplicare ogni bit per il suo peso e sommare il tutto:

Esempio:

$$10100$$

$$1*2^4 + 0*2^3 + 1*2^2 + 0*2^1 + 0*2^0 =$$

$$16 + 4 = 20$$

la conversione e' una **somma di potenze**

(N.B. se il numero binario termina per 1 e' dispari altrimenti e' pari).

Conversione da base 10 a base 2

Idea di fondo: usare le potenze di 2 che, sommate, danno il numero **N** da convertire:

- Prendere le potenze di 2 <= di N nell'ordine dalla più grande alla più piccola (cioè 2^0)
- Associare il bit 1 alle potenze che vengono usate nella somma per ricostruire **N**
- Associare il bit 0 alle potenze non usate.

Conversione da base 10 a base 2

- Dividere il numero per 2 ripetutamente finchè il risultato non è 0

- Scrivere i resti in ordine inverso.

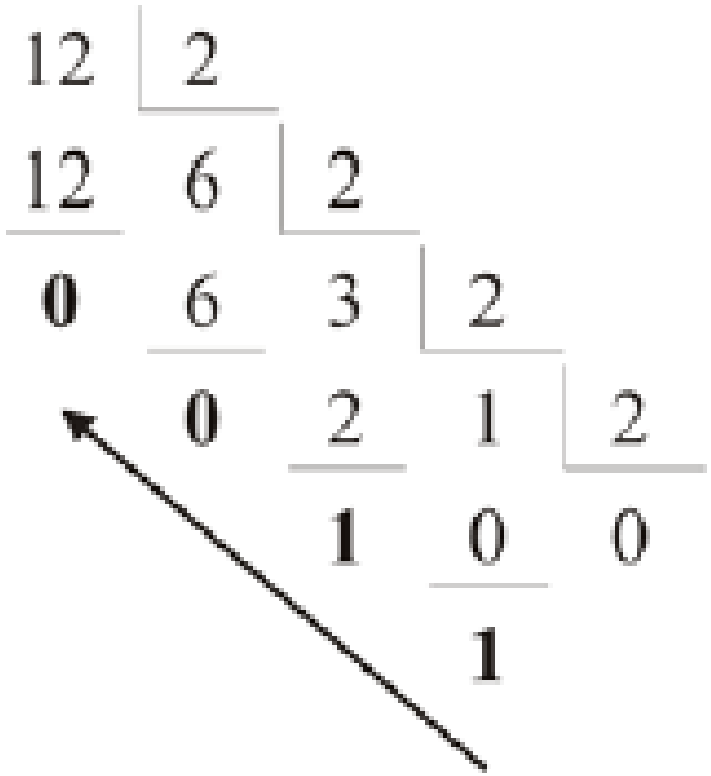
Esempio: conversione del numero 12

Divisioni: 12/2 = 6/2 = 3/2 = 1/2 = 0

Resti: 0 0 1 1

12 = 1100

Conversione da base 10 a base 2



La Codifica dei Caratteri

Informatica Applicata ai Beni Culturati
A.A. 2012/2013 Dott.ssa Annamaria Bria

Codici per i simboli dell'alfabeto

- Per rappresentare i simboli dell'alfabeto anglosassone
  (0 1 2 ... A B ... a b ...) bastano 7 bit
  - Nota: *B* e *b* sono simboli diversi
  - 26 maiuscole + 26 minuscole + 10 cifre + 30 segni di interpunzione+… -> circa 120 oggetti

Codifica ASCII

- La codifica **ASCII** (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange) utilizza codici su 7 bit
  (2^7 = 128 caratteri diversi)
- Ad esempio
  - 1 0 0 0 0 0 1 rappresenta A
  - 1 0 0 0 0 1 0 rappresenta B
  - 1 0 0 0 0 1 1 rappresenta C

- Le parole si codificano utilizzando sequenze di byte
  - 1000010 1000001 1000010 1000001
    B A B A

Altri codici di codifica

- **ASCII ESTESO**
  - Ogni carattere è rappresentato da un byte
  - Comprende anche simboli quali &, %, $, ...
  - 256 caratteri diversi
  - non è standard (cambia con la lingua usata)
- **UNICODE**
  - standard proposto a 16 bit (65.536 caratteri) per manipolare un numero maggiore di simboli

ASCII esteso

00000000	Null	00100000	Spc	01000000	@	01100000	`
00000001	Start of heading	00100001	!	01000001	A	01100001	a
00000010	Start of text	00100010	"	01000010	B	01100010	b
00000011	End of text	00100011	#	01000011	C	01100011	c
00000100	End of transmit	00100100	$	01000100	D	01100100	d
00000101	Enquiry	00100101	%	01000101	E	01100101	e
00000110	Acknowledge	00100110	&	01000110	F	01100110	f
00000111	Audible bell	00100111	'	01000111	G	01100111	g
00001000	Backspace	00101000	(	01001000	H	01101000	h
00001001	Horizontal tab	00101001	)	01001001	I	01101001	i
00001010	Line feed	00101010	*	01001010	J	01101010	j
00001011	Vertical tab	00101011	+	01001011	K	01101011	k
00001100	Form Feed	00101100	,	01001100	L	01101100	l
00001101	Carriage return	00101101	-	01001101	M	01101101	m
00001110	Shift out	00101110	.	01001110	N	01101110	n
00001111	Shift in	00101111	/	01001111	O	01101111	o
00010000	Data link escape	00110000	0	01010000	P	01110000	p
00010001	Device control 1	00110001	1	01010001	Q	01110001	q
00010010	Device control 2	00110010	2	01010010	R	01110010	r
00010011	Device control 3	00110011	3	01010011	S	01110011	s
00010100	Device control 4	00110100	4	01010100	T	01110100	t
00010101	Neg. acknowledge	00110101	5	01010101	U	01110101	u
00010110	Synchronous idle	00110110	6	01010110	V	01110110	v
00010111	End trans. block	00110111	7	01010111	W	01110111	w
00011000	Cancel	00111000	8	01011000	X	01111000	x
00011001	End of medium	00111001	9	01011001	Y	01111001	y
00011010	Substitution	00111010	:	01011010	Z	01111010	z
00011011	Escape	00111011	;	01011011	[	01111011	{
00011100	File separator	00111100	<	01011100	\	01111100	|
00011101	Group separator	00111101	=	01011101	]	01111101	}
00011110	Record Separator	00111110	>	01011110	^	01111110	~
00011111	Unit separator	00111111	?	01011111	_	01111111	Del

ASCII esteso: esempio

- "Il cane"

- 01001001 ("I")
- 01101100 ("l")
- 00100000 (SP)
- 01100011 ("c")
- 01100001 ("a")
- 01101110 ("n")
- 01100101 ("e")

Numeri in ASCII

Le cifre 0..9 rappresentate in Ascii sono simboli o caratteri **NON** quantità numeriche

Non possiamo usarle per indicare quantità e per le operazioni aritmetiche. (Anche nella vita di tutti giorni usiamo i numeri come simboli e non come quantità: i n. telefonici)

Formato dei File con Informazioni Testuali

- **plaintext**: solo caratteri (**.txt**)
- **formati generati da word processor**: caratteri + meta-informazioni sulla impaginazione (es: **.doc** generato da Word)
- **linguaggi di marcatura** (**markup language**): caratteri + **annotazioni**
  - **HTML**(**HypertextMarkup Language**): per ipertesti nel World Wide Web, annotazioni tra parentesi angolari ("<" e ">"), visualizzato da **web browser**
  - **XML**(**eXtensibleMarkup Language**): linguaggio per definire linguaggi di marcatura
- **PDF** (**PortableDocumentFormat**): visualizzatore gratuito, molto comune in Internet