

Esercizi sulla codifica binaria**Ricordiamo alcune definizioni**

codifica = regola per mettere in corrispondenza valori del dato da codificare
ottengo → stringa composta da elementi di altro alfabeto

K elementi dell'alfabeto di codifica

M le informazioni distinte da rappresentare (codificare)

N = $\lceil \log M \rceil$ il numero di elementi necessari per la stringa di codifica approssimati per eccesso all'intero successivo

1. Quanti bit sono necessari per codificare il numero di articoli da inserire in un carrello online che può contenere al massimo 20 prodotti ?

Nel nostro caso alfabeto (0,1)

devo rappresentare (codificare) $M = 21$ elementi diversi cioè 0.....20,

da "carrello vuoto" a "carrello pieno", e il numero di bit necessari è calcolato come

$$\log_2 20 + 1 = 4 + 1 = 5 \quad (\text{il log approssimato all'intero successivo})$$

Il calcolo vale anche per $\log_2 21$ perché ridondano le configurazioni

2. Quanti bit sono necessari per codificare 32 bandiere ?

Se vogliamo codificare 32 elementi diversi.

- Se considero la combinazione di 0 bandiere, ho 33 elementi da considerare 0...32,

il numero di bit necessari può essere calcolato come

$$\log_2 32 + 1 = 5 + 1 = 6 \quad 0..32 \text{ compreso}$$

- Se non considero la combinazione di 0 bandiere, ho 32 elementi da codificare 1..32,

ma per rappresentare 32 oggetti il numero di bit necessari sarà

$$\log_2 31 + 1 = 4 + 1 = 5 \quad \text{si riporta a } 0..31$$

Ricordiamo che

La codifica si basa sulla **notazione posizionale o «pesata»** che adottiamo per i numeri in base decimale. Nella notazione posizionale, in qualunque base, il valore numerico rappresentato da una cifra dipende

- dal valore della cifra
- dalla posizione della cifra nel numero,

Per un numero composto da N cifre le posizioni, A PARTIRE DA SINISTRA, sono indicate con $N-1 \dots 0$

Esempio in base 10 $231_{10} = 2 \cdot 10^2 + 3 \cdot 10^1 + 1 \cdot 10^0$

3. Converti 197_{10} in base 2. Verifica la correttezza della conversione, riconvertendo il risultato in base 10.

Possiamo innanzitutto calcolare il numero di cifre necessarie a codificare 197 in base 2

$$\rightarrow \log_2 197 + 1 = 7 + 1 = 8$$

L'algoritmo di conversione è il seguente :

197	2
98	1
49	0
24	1
12	0
6	0
3	0
1	1
0	1

$$\text{Quindi } 197_{10} = 11000101_2$$

Verifica del risultato:

$$11000101_2 = 1 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 128 + 64 + 4 + 1 = 197_{10}$$

4. Converti 584_{10} in base 11. Verifica la correttezza della conversione, riconvertendo il risultato in base 10.

584	11
53	1
4	9
0	4

$$\text{Quindi } 584_{10} = 491_{11}$$

$$\text{Verifica del risultato: } 491_{11} = 4 \cdot 11^2 + 9 \cdot 11^1 + 1 \cdot 11^0 = 484 + 99 + 1 = 584_{10}.$$

Anche in questo caso possiamo sapere a priori il numero di cifre necessarie per la codifica

$$\log_{11} 584 + 1 = 2 + 1 = 3$$

5. Definire le rappresentazioni in (a) modulo e segno, (b) complemento a uno, (c) complemento a due del numero decimale -512_{10} ; usando il numero minimo di bit necessari.

Per prima cosa, convertiamo il valore assoluto del numero in base 2

512	2
256	0
128	0
64	0
32	0
16	0
8	0
4	0
2	0
1	0
0	1

Abbiamo quindi che $512_{10} = 2^9 = 1000000000_2$

(a) Modulo e segno:

Per avere la rappresentazione MS aggiungiamo un bit avanti al valore assoluto del numero in base 2

il primo bit è il segno (0 per valori positivi, 1 per valori negativi)

$-512_{10} = 11000000000_{MS} \rightarrow$ numero minimo di bit per questa codifica

(b) Complemento a 1:

numeri positivi sono codificati come : 0 + codifica del valore assoluto in base 2

i numeri negativi sono ottenuti come : 1+ il complemento del valore assoluto in base 2

il primo bit è 0 per numeri positivi, 1 per numeri negativi.

$512_{10} = 01000000000_{C1} \rightarrow$ 0 + codifica in base2 di 512

$-512_{10} = 10111111111_{C1} \rightarrow$ 1 + complemento dei bit che compongono 512 in base2

(c) Complemento a 2

per rappresentare un numero in Complemento a 2 consideriamo che

- il primo bit è 0 per numeri positivi, 1 per numeri negativi, fa parte della rappresentazione del numero e non deve essere scartato

Vediamo come rappresentare il -512 a partire dal $512_{10} = 2^9 = 1000000000_2$

MODO 1:

- *Complemento tutti i bit del numero binario dato, cioè sostituisco 0 con 1 e viceversa*
- *Sommo 1 al risultato*

Prima calcoliamo il complemento di 512_{10} che è dato da 0111111111,

a questo sommiamo 1 di modo da ottenere la rappresentazione in complemento a due

$$-512_{10} = 0111111111 + 1 = 1000000000_{C2}.$$

MODO 2:

- *Scorro il numero da destra verso sinistra (dal bit meno significativo al più significativo)*
- *Copio tutti i bit che leggo nel risultato, sempre da destra verso sinistra, compreso il primo 1 che incontro*
- *Copio le cifre restanti complementate*

$$512_{10} = 1000000000_2$$

ATTENZIONE:

In questo caso abbiamo che è presente un solo 1 al bit più significativo così

abbiamo che la rappresentazione in complemento a due è data da $512_{10} = 1000000000_{C2}$

che è identica alla rappresentazione binaria di 512

- per numeri positivi il complemento a due è la rappresentazione binaria (coincidono le rapp.)
- per i numeri negativi è il complemento a 1 a cui viene sommato il valore 1.

Tuttavia ciò non causa ambiguità dato che utilizzando 10 bit in complemento a due non possiamo rappresentare il numero 512 e dunque 1000000000_{C2} può solamente corrispondere a -512.

Ricordiamo: con 10 bit in complemento a due possiamo rappresentare i numeri da -512 a 511

—

6. Spiegare cosa rappresenta il numero binario 1000101 se codificato in (a) modulo e segno, (b) complemento a 1 e (c) complemento a 2.

a) Modulo e segno:

il primo bit è il segno, 1 è per numeri negativi, il resto è la codifica binaria del valore assoluto del numero. $1000101_{MS} = -000101_2 = -5_{10}$

b) Complemento a uno:

il primo bit è il segno, il restante è il complemento del valore assoluto del numero. $1000101_{C1} = -111010_2 = -58_{10}$

c) Complemento a due:

è il complemento a uno del numero a cui viene sommato 1.

$$1000101_{C2} = 1000101 - 1 = 1000100_{C1} = -0111011_2 = -59_{10}$$

7. Spiegare cosa rappresenta il numero binario 100000

se codificato in: (a) modulo e segno, (b) complemento a 1 e (c) complemento a 2.

(a) Modulo e segno:

il primo bit è il segno, 1 è per numeri negativi, il resto è la codifica binaria del valore assoluto del numero. $100000_{MS} = -00000_2 = -0_{10}$

(b) Complemento a uno:

il primo bit è il segno, il restante è il complemento del valore assoluto del numero.

$$100000_{C1} = -11111_2 = -31_{10}$$

(c) Complemento a due:

è il complemento a uno del numero a cui viene sommato 1. il complemento a uno del numero a cui viene sommato 1. $100000_{C2} = -32_{10}$. In questo caso il valore binario è il limite inferiore dei valori rappresentabili, che per il complemento a 2 è per convenzione negativo, in questo caso ho 6 bit, quindi ho $-2^6 = -32$

8. Converti 10.25_{10} in base 2. Verifica la correttezza della conversione, riconvertendo il risultato in base 10.

Si considera separatamente la parte intera dalla parte decimale. Parte intera:

Parte intera : 10 la converto in binario

10	2
5	0
2	1
1	0
0	1

Parte decimale: .25 la converto in binario

$$0.25 \cdot 2 = 0.5 \rightarrow 0$$

$$0.5 \cdot 2 = 1.0 \rightarrow 1$$

La conversione quindi è data da $10.25_{10} \Rightarrow 1010.01_2$

Possiamo verificare il risultato:

$$1010.01_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} = 8 + 2 + \frac{1}{4} = 10.25_{10}$$

9. Converti 8.375_{10} in base 2. Verifica la correttezza della conversione, riconvertendo il risultato in base 10.

Si considera separatamente la parte intera dalla parte decimale.

Parte intera:

8	2
4	0
2	0
1	0
0	1

Parte decimale:

$$0.375 \cdot 2 = 0.75 \rightarrow 0$$

$$0.75 \cdot 2 = 1.5 \rightarrow 1$$

$$0.5 \cdot 2 = 1.0 \rightarrow 1$$

La conversione quindi data da $8.375_{10} = 1000.011_2$

Possiamo verificare il risultato:

$$1000.011_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} = 8 + 1/4 + 1/8 =$$

$$8.375_{10}$$

OPERAZIONI CON I NUMERI BINARI

10. Calcolare $8 - 5$ in base 2 in complemento a 2 usando 5 bit.

Per prima cosa convertiamo i due numeri in binario

Ora rappresentiamo i due numeri utilizzando 5 bit, aggiungendo all'occorrenza degli zeri a sinistra del bit piú significativo

$$8_{10} = 01000_2$$

$$5_{10} = 00101_2$$

convertiamo 5_{10} nel suo negativo con complemento a 2 usando 5 bit

$$-5_{10} = 11011_2$$

La somma tra i due numeri binari é adesso diretta e il suo risultato corrisponderá a quello della sottrazione $8_{10} - 5_{10}$

$$\begin{array}{r} 0 \ 1 \ 0 \ 0 \ 0 \\ 1 \ 1 \ 0 \ 1 \ 1 \\ \hline 1 \ 0 \ 0 \ 0 \ 1 \ 1 \end{array}$$

I segni degli addendi sono discordi: il risultato é corretto: $8_{10} - 5_{10} = 3_{10} = 00011_2$

Siccome **il risultato é positivo** il complemento a due é uguale alla rappresentazione binaria del numero corrispondente in base 10

11. Calcolare $-3_{10} - 2_{10}$ in base 2 in complemento a 2 usando 5 bit.

1) Per prima cosa convertiamo i due numeri in binario

2) Ora rappresentiamo i due numeri utilizzando 5 bit,

aggiungendo all'occorrenza degli zeri a sinistra del bit piú significativo

$$3_{10} = 00011_2$$

$$2_{10} = 00010_2$$

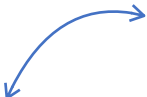
3) convertiamo 3_{10} e 2_{10} nel loro negativo con complemento a 2 usando 5 bit

$$-3_{10} = 11101_2$$

$$-2_{10} = 11110_2$$

3	2
1	1
0	1

2	2
1	0
0	1


$$\begin{array}{r} 1\ 1\ 1\ 0\ 1 \\ 1\ 1\ 1\ 1\ 0 \\ \hline 1\ 1\ 1\ 0\ 1\ 1 \end{array}$$

4) La somma tra i due numeri binari adesso é diretta e il risultato corrisponderá a quello della sottrazione $-3_{10} - 2_{10}$

I segni degli addendi sono concordi e uguali al segno del risultato: risultato corretto.

Siccome **il risultato é negativo** dobbiamo sottrarvi 1 e complementare i bit ottenuti per trovare il modulo del numero corrispondente:

$$11011 - 1 = 11010$$

complementiamo i bit \rightarrow 00101 che corrisponde a 5

quindi il risultato ottenuto é corretto, dato che $-3_{10} - 2_{10} = -5_{10} = 11011_2$

12. Calcolare $-5_{10} - 6_{10}$ in base 2 in complemento a 2 usando 4 bit

Per prima cosa convertiamo i due numeri in binario

5	2	6	2
2	1	3	0
1	0	1	1
0	1	0	1

Ora rappresentiamo i due numeri utilizzando 4 bit,

aggiungendo all'occorrenza degli zeri a sinistra del bit piú significativo

$$5_{10} = 0101_2$$


$$6_{10} = 0110_2$$

convertiamo 5_{10} e 6_{10} nel loro negativo con complemento a 2 usando 5 bit

$$-5_{10} = 1011_2$$

$$-6_{10} = 1010_2$$

La somma tra i due numeri binari é adesso diretta e il suo risultato corrisponderá a quello della sottrazione $-5_{10} - 6_{10}$


$$\begin{array}{r} 1011 \\ 1010 \\ \hline 1001 \end{array}$$

I segni degli addendi sono concordi ma diversi dal segno del risultato: *risultato errato*.

In particolare *il risultato esatto -11 ha bisogno di almeno di 5 bit per essere rappresentato in Complemento a 2*.

$$-5_{10} - 6_{10} = -11_{10} = 10101_2$$

Infatti se per esempio avessimo utilizzato 5 bit avremmo avuto che

$$5_{10} = 00101_2$$

$$6_{10} = 00110_2$$

convertiamo 5_{10} e 6_{10} nel loro negativo con complemento a 2 usando 5 bit

$$-5_{10} = 11011_2$$

$$-6_{10} = 11010_2$$

La somma tra i due numeri binari adesso é diretta e il suo risultato corrisponderá a quello della sottrazione $-5_{10} - 6_{10}$

$$\begin{array}{r} 1\ 1\ 0\ 1\ 1 \\ 1\ 1\ 0\ 1\ 0 \\ \hline 1\ 1\ 0\ 1\ 0\ 1 \end{array}$$

Ora i segni degli addendi sono concordi e uguali a quello del risultato: risultato corretto.

Siccome il risultato é negativo dobbiamo sottrarci 1 e complementare i bit ottenuti per trovare il modulo del numero corrispondente:

1 $10101 - 1 = 10100$

2 complementiamo i bit e otteniamo 01011 che corrisponde a 11,
ora il risultato ottenuto é corretto, dato che $-5_{10} - 6_{10} = -11_{10} = 10101_2$

13. Convertire il numero decimale 3.25 in base 2 in virgola mobile in singola precisione usando lo standard IEEE 754.

Ricordiamo : rappresentazione divisa in tre parti **S(1bit) E (8bit) M(23 bit)**

-caso 32 bit

-----codifico

Per prima cosa convertiamo il numero in binario.

Prima la parte intera

$$\begin{array}{r} 3\ 2 \\ 1\ 1 \\ \hline 0\ 1 \end{array}$$

poi la parte decimale

$$0.25 \cdot 2 = 0.5 \rightarrow 0$$

$$0.5 \cdot 2 = 1.0 \rightarrow 1$$

perció otteniamo $3.25_{10} = 11.01_2$

-----normalizzo

ora scriviamo la forma normalizzata del numero che abbiamo ottenuto $3.25_{10} = 1.101_2 \cdot 2^1$

----- calcolo esponente e segno

Segno $\rightarrow S = 0$

Esponente $\rightarrow E = e + k = 1 + 127 = 128 = 10000000_2$

Mantissa $\rightarrow M = 101000000000000000000000$

-----scrivo il numero

Il numero in virgola mobile in singola precisione usando lo standard IEEE 754 è quindi dato da

0 10000000 101000000000000000000000

S(1 bit) E(8 bit) M(23 bit)

14. Convertire il numero decimale -23.375 in base 2 in virgola mobile in singola precisione usando lo standard IEEE 754.

----codifica

Per prima cosa convertiamo il numero in binario. Prima la parte intera

23	2
11	1
5	1
2	1
1	0
0	1

poi la parte decimale $0.375 \cdot 2 = 0.75 \rightarrow 0$

$0.75 \cdot 2 = 1.5 \rightarrow 1$

$0.5 \cdot 2 = 1 \rightarrow 1$

perciò otteniamo $23.375_{10} = 10111.011_2$

-----normalizzo

ora scriviamo la forma normalizzata del numero che abbiamo ottenuto $23.375_{10} = 1.0111011_2 \cdot 2^4$

-----segno ed esponente

Segno $\rightarrow S = 1$

Esponente $\rightarrow E = e + k = 4 + 127 = 131 = 10000011_2$

Mantissa $\rightarrow M = 011101100000000000000000$

-----scrivo il numero

Il numero in virgola mobile in singola precisione usando lo standard IEEE 754 è quindi dato da

1 10000011 011101100000000000000000

S(1 bit) E(8 bit)

M(23 bit)

15. Convertire il numero decimale -127.25 in base 2 in virgola mobile in singola precisione usando lo standard IEEE 754.

Per prima cosa convertiamo il numero in binario. Prima la parte intera

127	2
63	1
31	1
15	1
7	1
3	1
1	1
0	1

poi la parte decimale $0.25 \cdot 2 = 0.5 \rightarrow 0$

$0.5 \cdot 2 = 1 \rightarrow 1$

perciò otteniamo $127.25_{10} = 1111111.01_2$

ora scriviamo la forma normalizzata del numero che abbiamo ottenuto $127.25_{10} = 1.11111101_2 \cdot 2^6$

Segno $\rightarrow S = 1$

Esponente $\rightarrow E = e + k = 6 + 127 = 133 = 10000101_2$ Mantissa $\rightarrow M = 111111010000000000000000$

Il numero in virgola mobile in singola precisione usando lo standard IEEE 754 è quindi dato da

1 10000101 111111010000000000000000
S(1 bit) E(8 bit) M(23 bit)

16. Tema d'esame novembre 2011 - esercizio 1 - terza domanda

Supponendo di avere un alfabeto di 43 simboli, indicare il numero minimo di bit necessari per definire una loro rappresentazione binaria.

$$\log \lceil 43 \rceil = 6$$

logaritmo di 43 è un valore tra 5 e 6, lo approssimo per eccesso.

17. Dal tema di esame 8 settembre 2020

Si ipotizzi un codice composto da 3 lettere maiuscole (supponiamo 20 possibili alternative), due numeri (10 alternative) e altre 2 lettere.

- ***Quanti codici diversi si possono ottenere?***

Il codice ha la forma LLL NN LL

Definiamo card L = 20 card N = 10

numero di bit per codificare LLL $\rightarrow 20^3 = 8000$

numero di bit per codificare NN $\rightarrow 10^2 = 100$

numero di bit per codificare LL $\rightarrow 20^2 = 400$

Quindi $8000 \times 100 \times 400 = 320\,000\,000$

- ***Quanti bit servirebbero per una rappresentazione minima del singolo codice?***

L ha bisogno di $\log_{20} + 1 = 5 \rightarrow \text{LLL} \rightarrow 5 \times 3 = 15 \text{ bit}$ LL $\rightarrow 5 \times 2 = 10 \text{ bit}$

N ha bisogno di $\log_{10} + 1 = 4 \rightarrow \text{NN} \rightarrow 4 \times 2 = 8 \text{ bit}$

$$15 + 8 + 10 = 33 \text{ bit}$$

- ***Sarebbe possibile usare la codifica ASCII per ogni singolo carattere e numero?***

Nella codifica ASCII ogni carattere ha bisogno di 8 bit per la sua rappresentazione,

LLL NN LL sono 7 caratteri \rightarrow se usiamo 8 bit per ogni carattere $7 \times 8 = 56 \text{ bit}$ necessari

Non si potrebbe codificare in ASCII disponendo dei 37 bit della configurazione minima