0. ALGEBRA DI BOOLE E SISTEMI DI NUMERAZIONE

ALGEBRA DI BOOLE

Nel lavoro di programmazione capita spesso di dovere ricorrere ai principi della logica degli enunciati ed occorre conoscere almeno alcuni concetti base dell'**algebra delle preposizioni** detta anche **algebra booleana** o di **Boole** dal matematico inglese George Boole (1815-1864).

Gli oggetti dell'algebra di Boole sono gli **enunciati.**

DEF: Si definisce **enunciato** una **proposizione** che può essere soltanto vera oppure falsa e non può mai essere né contemporaneamente entrambe le cose né indeterminata.
DEF: Con il termine **valore di verità di enunciato** si intende la sua verità oppure la sua falsità.

Gli enunciati possono essere **semplici** oppure **composti.**
DEF: Un. **enunciato composto** è formato da due o più sottoenunciati collegati tra loro attraverso appositi **connettivi logici.**
La **proprietà fondamentale** di un **enunciato composto** è che il suo valore di verità viene completamente definito dai valori di verità dei suoi sottoenunciati e dal connettivo logico che li unisce.

I CONNETTIVI LOGICI FONDAMENTALI

A) CONGIUNZIONE (AND oppure **e** oppure $\wedge$ oppure **et** oppure **&)**
Il connettivo logico **AND** è un operatore **binario** (ossia agisce su due enunciati per crearne un altro) completamente definito dalla seguente **tavola di verità**

p	q	p AND q
V	V	V
V	F	F
F	V	F
F	F	F

Quindi l'**enunciato composto p AND q** risulta **VERO** solo nel caso in cui entrambi gli **enunciati semplici p** e **q** sono **VERI** mentre risulta **FALSO** in tutti gli altri casi.

B) DISGIUNZIONE (OR oppure **o** oppure $\vee$ oppure **vel** oppure **|)**
Il connettivo logico **OR** è un operatore **binario** (ossia agisce su due enunciati per crearne un altro) completamente definito dalla seguente **tavola di verità**

p	q	p OR q
V	V	V
V	F	V
F	V	V
F	F	F

Quindi l'**enunciato composto p OR q** risulta **FALSO** solo nel caso in cui entrambi gli **enunciati semplici p** e **q** sono **FALSI** mentre risulta **VERO** in tutti gli altri casi (ossia quando almeno uno degli enunciati semplici è VERO).

C) NEGAZIONE (NOT oppure **non)**
Il connettivo logico **OR** è un operatore **unario** (ossia agisce su un solo enunciato per crearne un altro) completamente definito dalla seguente **tavola di verità**

p	NOT p
V	F
F	V

Quindi l'**enunciato composto NOT p** (detto anche **"negazione di p")** che risulta **VERO** se l'**enunciato semplice p** è **FALSO** mentre risulta **FALSO** se l'**enunciato semplice p** è **VERO.**

I tre connettivi logici sopra definiti - **AND**, **OR** e **NOT** - costituiscono un insieme **"funzionalmente completo"** di operatori nell'algebra di Boole.
Tutti gli altri connettivi logici possibili sono chiamati **"connettivi derivati"** in quanto possono essere espressi mediante un'opportuna combinazione di uno o più dei connettivi funzionalmente completi (ossia **AND**, **OR** e **NOT**)

I CONNETTIVI LOGICI DERIVATI

Esempio di "connettivi derivati" sono **XOR**, **NAND**, **NOR** e **XNOR**

D) DISGIUNZIONE ESCLUSIVA (XOR oppure **o esclusivo** oppure **aut)**
Il connettivo logico X**OR** è un operatore **binario** (ossia agisce su due enunciati per crearne un altro) completamente definito dalla seguente **tavola di verità**

p	q	p XOR q
V	V	F
V	F	V
F	V	V
F	F	F

Quindi l'**enunciato composto p OR q** risulta **VERO** solo nel caso in cui **al più uno solo** dei due **enunciati semplici p** e **q** risulti **VERO** mentre risulta **FALSO** in tutti gli altri casi

oppure in altri termini

L'**enunciato composto p XOR q** risulta **VERO** solo nel caso in cui i due **enunciati semplici p** e **q** hanno **valori di verità diversi** mentre risulta **FALSO** se i due **enunciati semplici p** e **q** hanno **valori di verità uguali.**

N.B. Lo XOR è un connettivo derivato in quanto si potrà dimostrare utilizzando il concetto di equivalenza logica illustrato più avanti che

$$p \text{ XOR } q \equiv (p \text{ AND } (\text{NOT} q)) \text{ OR } ((\text{NOT} p) \text{ AND } q)$$

E) OPERATORE LOGICO NAND

Il connettivo logico **NAND** è un operatore **binario** (ossia agisce su due enunciati per crearne un altro) completamente definito dalla seguente **tavola di verità**

p	q	p NAND q
V	V	F
V	F	V
F	V	V
F	F	V

Quindi l'**enunciato composto p NAND q** risulta **FALSO** quando entrambi gli enunciati semplici **p** e **q** sono **VERI** mentre risulta **VERO** in tutti gli altri casi

N.B. Il NAND è un connettivo derivato in quanto si potrà dimostrare utilizzando il concetto di equivalenza logica illustrato più avanti che

p NAND q ≡ NOT (p AND q)

F) OPERATORE LOGICO NOR

Il connettivo logico **NOR** è un operatore **binario** (ossia agisce su due enunciati per crearne un altro) completamente definito dalla seguente **tavola di verità**

p	q	p NOR q
V	V	F
V	F	F
F	V	F
F	F	V

Quindi l'**enunciato composto p NOR q** risulta **VERO** quando entrambi gli enunciati semplici **p** e **q** sono **FALSI** mentre risulta **FALSO** in tutti gli altri casi

N.B. Il NOR è un connettivo derivato in quanto si potrà dimostrare utilizzando il concetto di equivalenza logica illustrato più avanti che

p NOR q ≡ NOT (p OR q)

G) OPERATORE LOGICO XNOR

Il connettivo logico **XNOR** è un operatore **binario** (ossia agisce su due enunciati per crearne un altro) completamente definito dalla seguente **tavola di verità**

p	q	p XNOR q
V	V	V
V	F	F
F	V	F
F	F	V

Quindi l'**enunciato composto p XNOR q** risulta **VERO** solo nel caso in cui gli **enunciati semplici p** e **q** hanno lo stesso valore di verità mentre risulta **FALSO** in tutti gli altri casi

N.B. Lo XNOR è un connettivo derivato in quanto si potrà dimostrare utilizzando il concetto di equivalenza logica illustrato più avanti che

p XNOR q ≡ NOT (p XOR q)

LE TAVOLE DI VERITÀ

Combinando in vario modo gli enunciati semplici del tipo p, q, r ed i connettivi logici **AND**, **OR**, **NOT** e **XOR** si possono ottenere enunciati molto più complessi.
La **forma enunciativa** è un enunciato composto **P(p, q, r, ..)** da enunciati semplici variabili attraverso i connettivi logici

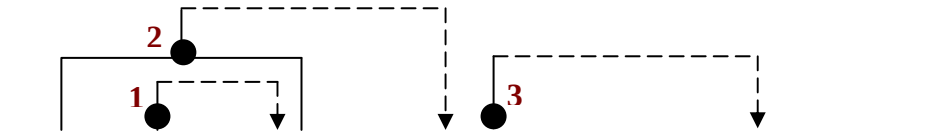
Il **valore di verità di una forma enunciativa** è noto quando si conoscono i valori di verità delle sue variabili. Un modo semplice per conoscerlo è quello che prevede la *costruzione delle tavole di verità.*

Esempio:
Supponiamo di voler conoscere i valori di verità della seguente forma enunciativa

NOT(p AND (NOT q))

Costruiamo la tavola di verità seguendo i seguenti passi:
*- prima occorre individuare gli enunciati semplici contenuti nella forma enunciativa. Nel nostro caso **p** e **q**;*
*- poi occorre individuare gli enunciati composti contenuti nella forma enunciativa partendo dall'enunciato composto più interno fino all'enunciato composto totale. Nel nostro caso **NOT q**, **(p AND (NOT q))** e **NOT(p AND (NOT q))**;*
- disegnare una tabella con tante colonne quanti sono gli elementi individuati nei primi due punti;
- applicare nell'ordine le tavole di verità dei connettivi logici fondamentali AND, OR, XOR e NOT.

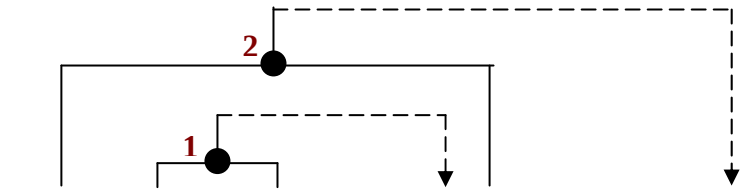
Legenda:
● *applicazione di una tavola di verità fondamentale (AND, OR, XOR o NOT)*
– – – *risultato di una tavola di verità fondamentale (AND, OR, XOR o NOT)*



p	*q*	*NOT q*	*p AND (NOT q)*	*NOT(p AND (NOT q))*
V	**V**	**F**	**F**	**V**
V	**F**	**V**	**V**	**F**
F	**V**	**F**	**F**	**V**
F	**F**	**V**	**F**	**V**

Esempio:
Supponiamo di voler conoscere i valori di verità della seguente forma enunciativa

p OR (q AND r))



p	*q*	*r*	*q AND r*	*p OR (q AND r))*
V	**V**	**V**	**V**	**V**
V	**V**	**F**	**F**	**V**
V	**F**	**V**	**F**	**V**
V	**F**	**F**	**F**	**V**
F	**V**	**V**	**V**	**V**
F	**V**	**F**	**F**	**F**
F	**F**	**V**	**F**	**F**
F	**F**	**F**	**F**	**F**

L'EQUIVALENZA LOGICA

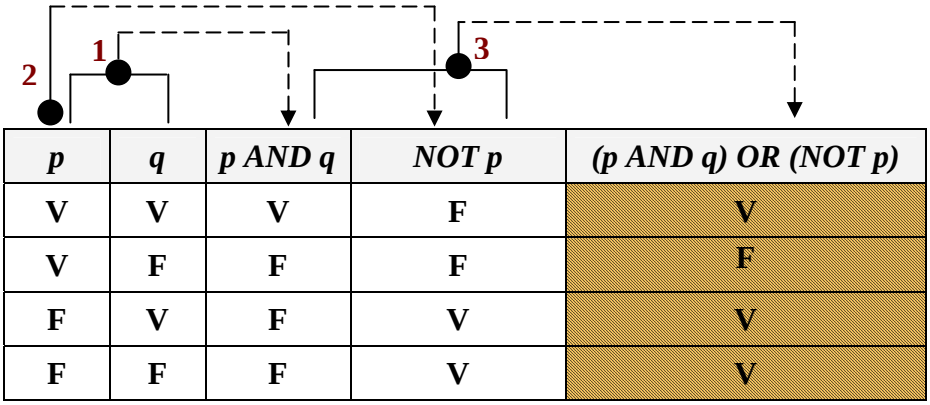
DEF: Due forme enunciative sono **equivalenti** se hanno la medesima tavola di verità.
L'equivalenza si in dica con il simbolo ≡

Esempio:
Proviamo che le due forme enunciative

(p AND q) OR (NOT p) ***e*** ***(NOT p) OR q***

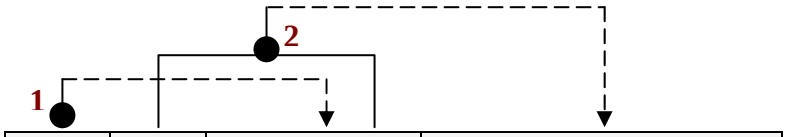
sono equivalenti.
Dobbiamo innanzitutto costruire le due tavole di verità con seguendo i passi prima specificati.

Tavola di verità di: ***(p AND q) OR (NOT p)***



p	*q*	*p AND q*	*NOT p*	*(p AND q) OR (NOT p)*
V	V	V	F	V
V	F	F	F	F
F	V	F	V	V
F	F	F	V	V

Tavola di verità di: ***(NOT p) OR q***



p	*q*	*NOT p*	*(NOT p) OR q*
V	V	F	V
V	F	F	F
F	V	V	V
F	F	V	V

Poichè le due forme enunciative hanno la stessa tavola di verità esse risultano essere equivalenti.

Quindi possiamo scrivere che ***(p AND q) OR (NOT p) ≡ (NOT p) OR q***

LEGGI O PROPRIETÀ DELL'ALGEBRA BOOLEANA

1) IDEMPOTENZA **p OR p ≡ p** **p AND p ≡ p**

2) ASSOCIATIVITA' **(p OR q) OR r ≡ p OR (q OR r)**
(p AND q) AND r ≡ p AND (q AND r)

3) COMMUTATIVITA' **p OR q ≡ q OR p** **p AND q ≡ q AND p**

4) DISTRIBUTIVITA' **p OR (q AND r) ≡ (p OR q) AND (p OR r)**
p AND (q OR r) ≡ (p AND q) OR (p AND r)

5) DOPPIA NEGAZIONE **NOT (NOT p) ≡ p**

6) LEGGI DI DE MORGAN **(1) NOT (p OR q) ≡ (NOT p) AND (NOT q)**
(2) NOT (p AND q) ≡ (NOT p) OR (NOT q)

7) PRINCIPIO DI NON CONTRADDIZIONE **p AND (NOT p) è sempre FALSO**

Nota Bene
Tutte queste proprietà si dimostrano costruendo le tavole di verità di entrambi i membri e verificando che esse coincidono

SISTEMI DI NUMERAZIONE

Sistema di numerazione posizionali o pesati

Il nostro sistema di numerazione viene chiamato **in base 10** o **decimale.**
Tutti i numeri che noi scriviamo sono sequenze finite di cifre scelte all'interno di un **alfabeto** A ammesso costituto esattamente da **10 simboli** ossia dalle cifre **0, 1, 2, 3, .. fino a 9**

$$\mathbf{A = \{\ 0, 1, 2, 3, 4, 5, 6, 7, 8\ 9\}}$$

Il nostro sistema di numerazione (in base 10) è un sistema di numerazione **posizionale** o **pesato** nel senso che una stessa cifra assume valori diversi ossia *pesi diversi* a seconda della *posizione* che essa occupa all'interno del numero.

Esempio:
*Nel numero **1415,67** che in base 10 leggiamo **"millequattrocentoquindici"** la prima cifra **"1"** che incontriamo a partire da destra verso sinistra ha un significato diverso dalla seconda cifra **"1"** incontrata. Infatti il primo "1" sta per **decine** mentre il secondo "1" indica le **migliaia.***
*Essi hanno quindi **pesi** diversi ed il peso dipende dalla posizione relativa della cifra. Tali pesi risultano sempre più significativi man mano che procediamo da destra verso sinistra.*

Il numero **1415,67** può come ben sappiamo essere anche espresso in base 10 con la seguente espressione:

$$\underbrace{1415{,}67}_{\text{Notazione compatta}} = \underbrace{1 \times 10^3 + 4 \times 10^2 + 1 \times 10^1 + 5 \times 10^0 + 6 \times 10^{-1} + 7 \times 10^{-2}}_{\text{Notazione espansa}}$$

Le due notazioni prendono rispettivamente il nome di **notazione compatta** e **notazione espansa** del numero dato.

Generalizzando qualunque numero espresso da una sequenza di **2n + 1** simboli

$$\mathbf{C_n\ C_{n-1}\ \ldots\ldots..C_1\ C_0\ ,\ C_{-1}\ \ldots\ldots\ldots C_{-(n-1)}\ C_{-n}}$$

ammessi nell'alfabeto di riferimento di una qualsiasi base di numerazione **b** può essere espressa nel seguente modo

$$\mathbf{C_n\ b^n + C_{n-1}\ b^{n-1}\ldots\ldots..C_1\ b^1 + C_0\ b^0 + C_{-1}\ b^{-1} + \ldots\ldots\ldots C_{-(n-1)}\ b^{-(n-1)} + C_{-n}\ b^{-n}}$$

con le cifre $\mathbf{C_i \in \{\ 0, 1, 2,\ldots..\ (b\text{-}1)\}}$

Sistema di numerazione binario o in base 2

Il sistema di numerazione binario o in base 2 è ovviamente posizionale ed utilizza come alfabeto di simboli l'insieme $\mathbf{A = \{\ 0, 1\}}$.

(*) Conversione da binario a decimale ossia da base 2 a base 10
Si parte dalla notazione compatta del numero in base 2 moltiplicando ciascuna cifra per il peso relativo e sommandone i parziali relativi, se ne calcola il totale.

Esempio
$\mathbf{(101)_2} = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = (4 + 0 + 1)_{10} = \mathbf{(5)_{10}}$
$\mathbf{(101{,}11)_2} = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} = (4 + 0 + 1 + 0{,}5 + 0{,}25)_{10} = \mathbf{(5{,}75)_{10}}$

(*) Conversione da decimale a binario ossia da base 10 a base 2
PARTE INTERA DEL NUMERO
Si parte dalla notazione compatta del numero in base 10 **dividendo successivamente per 2** (metodo delle divisioni successive) finchè il quoziente non è 0. A questo punto **la sequenza dei resti ottenuti letta dal basso verso l'alto** darà il numero decimale convertito in base 2.

PARTE DECIMALE DEL NUMERO
Si parte dalla notazione compatta del numero in base 10 **moltiplicando successivamente per 2** solo ed esclusivamente la parte decimale dei prodotti intermedi ottenuti. Il procedimento si arresta nel caso tale parte decimale si annulli oppure dopo un certo numero di moltiplicazioni ottenendo un valore approssimato. La parte decimale convertita sarà data dalla sequenza delle parti intere delle moltiplicazioni effettuate **lette dall'alto verso il basso.**

Esempio
Convertire in base 2 il numero decimale $(108,75)_{10}$

PARTE INTERA
Dobbiamo prima di tutto convertire in base 2 il numero decimale ***$(108)_{10}$***

108 : 2
Metodo delle divisioni successive
0 54 **:** 2
0 27 **:** 2
1 13 **:** 2
1 6 **:** 2
0 3 **:** 2
1 1 **:** 2
1 **0** *(il procedimento si arresta perché il quoziente è uguale a 0)*

Quindi come si può vedere:
$(108)_{10} = (1101100)_2$

PARTE DECIMALE
Occorre poi convertire in base 2 il numero decimale ***$(0,75)_{10}$***

Metodo delle moltiplicazioni successive
0,75 x 2 = **1**, 50
0,50 x 2 = **1**, **00** *(il procedimento si deve arrestare perché la parte decimale è uguale a 0)*

Quindi come si può vedere:
$(0,75)_{10} = (0,11)_2$

Riassumendo abbiamo ottenuto che
$(108,75)_{10} = (1101100,11)_2$

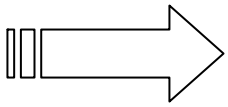
N.B. Come possiamo notare, abbiamo ottenuto in questo caso che la conversione in base 2 di un numero decimale finito, abbia dato origine ad un numero binario decimale finito.
Questa non è la regola generale: numeri decimali finiti in base 10 possono anche dare origine dopo la conversione a numeri binari illimitati aperiodici e viceversa.

Esempio
Convertire in base 2 il numero decimale $(2,63)_{10}$

PARTE INTERA
Dobbiamo prima di tutto convertire in base 2 il numero decimale ***$(2)_{10}$***

2 : 2
0 1 **:** 2
1 0

Metodo delle divisioni successive

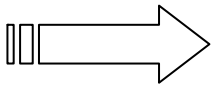


Quindi come si può vedere abbiamo ottenuto che:
$(2)_{10} = (10)_2$

PARTE DECIMALE
Occorre poi convertire in base 2 il numero decimale ***$(0,63)_{10}$***

0,63 x 2 = **1**, 26
0,26 x 2 = **0**, 52
0,52 x 2 = **1**, 04
0,04 x 2 = **0**, 08
0,08 x 2 = **0**, 16
0,16 x 2 = **0**, 32
……..

Metodo delle moltiplicazioni successive



Quindi come si può vedere abbiamo ottenuto che:
$(0,63)_{10} \approx (0,101000……)_2$

Se arrestiamo il procedimento approssimando il risultato ottenuto alla 3° cifra dopo la virgola otteniamo che

$(2,63)_{10} \approx (10,101)_2$ ***approssimazione per difetto perché il quarto simbolo è = 0***

<u>Sistema di numerazione ottale o in base 8</u>
Il sistema di numerazione ottale o in base 8 è ovviamente posizionale ed utilizza come alfabeto di simboli l'insieme **A = { 0, 1, 2, 3, 4, 5, 6, 7 }**.

(*) Conversione da ottale a decimale ossia da base 8 a base 10
Si parte dalla notazione compatta del numero in base 8 moltiplicando ciascuna cifra per il peso relativo e sommandone i parziali relativi, se ne calcola il totale.

Esempio
$(253,45)_8 = 2 \times 8^2 + 5 \times 8^1 + 3 \times 8^0 + 4 \times 8^{-1} + 5 \times 8^{-2} = (128 + 40 + 3 + 4/8 + 5/64)_{10} =$
$(171 + 37/64) = (171 + 0,578125) =$ ***$(171,578125)_{10}$***

Se arrestiamo il procedimento approssimando il risultato ottenuto alla 3° cifra dopo la virgola otteniamo che

$(253,45)_8 \approx (171,578)_{10}$ ***approssimazione per difetto perché il quarto simbolo è = 1***

(*) Conversione da decimale a ottale ossia da base 10 a base 8

PARTE INTERA DEL NUMERO

Si parte dalla notazione compatta del numero in base 10 **dividendo successivamente per 8** (metodo delle divisioni successive) finchè il quoziente non è 0. A questo punto **la sequenza dei resti ottenuti letta dal basso verso l'alto** darà il numero decimale convertito in base 8.

PARTE DECIMALE DEL NUMERO

Si parte dalla notazione compatta del numero in base 10 **moltiplicando successivamente per 8** solo ed esclusivamente la parte decimale dei prodotti intermedi ottenuti. Il procedimento si arresta nel caso tale parte decimale si annulli oppure dopo un certo numero di moltiplicazioni ottenendo un valore approssimato. La parte decimale convertita sarà data dalla sequenza delle parti intere delle moltiplicazioni effettuate **lette dall'alto verso il basso.**

Esempio

Convertire in base 8 il numero decimale $(144,26)_{10}$

PARTE INTERA

Dobbiamo prima di tutto convertire in base 8 il numero decimale ***$(144)_{10}$***

Metodo delle divisioni successive

144 : 8
0 18 **:** 8
2 2 **:** 8
2 0 *(il procedimento si arresta perché il q*

Quindi come si può vedere abbiamo ottenuto che:
$(144)_{10} = (220)_8$

PARTE DECIMALE

Occorre poi convertire in base 8 il numero decimale ***$(0,26)_{10}$***

0,26 x 8 = **2**, 08
0,08 x 8 = **0**, 64
0,64 x 8 = **5**, 12
0,12 x 8 = **0**, 96
0,96 x 8 = **7**, 68
……..

Metodo delle moltiplicazioni successive

Quindi come si può vedere abbiamo ottenuto che:
$(0,26)_{10} \approx (0,20507….)_8$

Se arrestiamo il procedimento approssimando il risultato ottenuto alla 3° cifra dopo la virgola otteniamo che

Riassumendo abbiamo ottenuto che

$(144,26)_{10} \approx (220,205)_8$ ***approssimazione per difetto perché il quarto simbolo è = 0***

N.B. Come possiamo notare, abbiamo ottenuto in questo caso che la conversione in base 8 di un numero decimale finito, abbia dato origine ad un numero ottale illimitato aperiodico.

Sistema di numerazione esadecimale o in base 16

Il sistema di numerazione esadecimale o in base 16 è ovviamente posizionale ed utilizza come alfabeto di simboli l'insieme **A = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F }** con i simboli che valgono rispettivamente**. A = 10, B=11, C=12, D=13, E=14, F=15**

(*) Conversione da esadecimale a decimale ossia da base 16 a base 10

Si parte dalla notazione compatta del numero in base 16 moltiplicando ciascuna cifra per il peso relativo e sommandone i parziali relativi, se ne calcola il totale.

Esempio

$(B95D)_{16} = B \times 16^3 + 9 \times 16^2 + 5 \times 16^1 + D \times 16^0 = (47453)_{10}$

(*) Conversione da decimale a esadecimale ossia da base 10 a base 16

PARTE INTERA DEL NUMERO

Si parte dalla notazione compatta del numero in base 10 **dividendo successivamente per 16** (metodo delle divisioni successive) finchè il quoziente non è 0. A questo punto **la sequenza dei resti ottenuti letta dal basso verso l'alto** darà il numero decimale convertito in base 16.

PARTE DECIMALE DEL NUMERO

Si parte dalla notazione compatta del numero in base 10 **moltiplicando successivamente per 16** solo ed esclusivamente la parte decimale dei prodotti intermedi ottenuti. Il procedimento si arresta nel caso tale parte decimale si annulli oppure dopo un certo numero di moltiplicazioni ottenendo un valore approssimato. La parte decimale convertita sarà data dalla sequenza delle parti intere delle moltiplicazioni effettuate **lette dall'alto verso il basso.**

N.B. In entrambi i casi ai valori numerici eventuali ammissibili per i resti o per le parti intere 10, 11, 12, 13, 14, 15 occorre sostituire la rappresentazione relativa consentita nell'alfabeto del sistema di numerazione con base 16 ossia rispettivamente le lettere **A**, **B**, **C**, **D**, **E**, **F**.

Esempio

Convertire in base 16 il numero decimale $(156,26)_{10}$

PARTE INTERA

Dobbiamo prima di tutto convertire in base 16 il numero decimale $(156)_{10}$

Metodo delle divisioni successive

156 : 16
12 9 **:** 16
9 **0** *(il procedimento si arresta perché il quoziente è uguale a 0)*

Quindi come si può vedere (ricordando che C=12) è:
$(156)_{10} = (9C)_{16}$

PARTE DECIMALE

Occorre poi convertire in base 16 il numero decimale $(0,26)_{10}$

0,26 x 16 = **4**, 16
0,16 x 16 = **2**, 56
0,56 x 16 = **8**, 96
0,96 x 16 = **15**, 36
0,36 x 16 = **5**, 76
……..

Metodo delle moltiplicazioni successive

Quindi come si può vedere (ricordando che F=15) è:
$(0,26)_{10} \approx (0,428F5\ldots\ldots)_{16}$

Se arrestiamo il procedimento approssimando il risultato ottenuto alla 3° cifra dopo la virgola otteniamo che

Riassumendo abbiamo ottenuto che

$(156,26)_{10} \approx (9C,429)_{16}$ ***approssimazione per eccesso perché il quarto simbolo è = F***

N.B. Come possiamo notare, abbiamo ottenuto anche in questo caso che la conversione in base 16 di un numero decimale finito, abbia dato origine ad un numero esadecimale illimitato aperiodico.

Conversioni esadecimali-ottali-binarie dirette

PREMESSA
Se volessimo conoscere il numero **n** di elementi distinti è possibile rappresentare su di un certo numero **x** di **bit** (**bi**nary digi**t**), dovremmo utilizzare la relazione:

$$\mathbf{n = 2^x}$$

Esempio
*Con **x = 2** bit è possibile rappresentare soltanto **4** elementi, ossia 2^2 , mentre con **x = 3** bit è possibile rappresentare soltanto **8** elementi, ossia 2^3 , **x = 4** bit è possibile rappresentare soltanto **16** elementi, ossia 2^4 etc.*

Se invece volessimo conoscere il minimo numero di **bit** (**bi**nary digi**t**), indicato con **x** ,con i quali possiamo rappresentare in binario l'insieme dei simboli o alfabeto di una determinata base di numerazione **b** dovremmo utilizzare la relazione:

$$\mathbf{x = \log_2 b}$$

Esempio
Qual è il numero minimo di bit necessario per rappresentare 4 simboli distinti?
Esattamente 2 bit ossia $\mathbf{\log_2 4}$
Qual è il numero minimo di bit necessario per rappresentare 8 simboli distinti?
Esattamente 3 bit ossia $\mathbf{\log_2 8}$
Qual è il numero minimo di bit necessario per rappresentare 16 simboli distinti?
Esattamente 4 bit ossia $\mathbf{\log_2 16}$
Qual è il numero minimo di bit necessario per rappresentare 10 simboli distinti?
Poiché $\mathbf{\log_2 10}$ *= 3,162… da ciò deriva che 3 bit non bastano (infatti servono per rappresentare come già visto 8 simboli) : bisogna quindi prevedere l'intero successivo ossia 4 bit che come sappiamo permette la rappresentazione di 16 simboli (6 in più di quanto richiesto).*
*In questo caso siamo obbligati a scegliere 4 bit ma la rappresentazione che ne deriva è **ridondante** ossia prevede una sovrabbondanza di spazio rispetto a quanto richiesto.*

Per convertire un numero ad esempio da una base qualunque $\mathbf{b_1}$ **(diversa dalla decimale)** ad una seconda base $\mathbf{b_2}$ **(diversa dalla decimale)** potremmo chiaramente procedere nel seguente modo.
Dapprima convertire il numero nella base $\mathbf{b_1}$ nel suo equivalente in base 10 utilizzando la notazione espansa ed i pesi relativi e poi convertire il numero così trasformato in base 10 nella nuova base $\mathbf{b_2}$ utilizzando il metodo delle divisioni successive per la sua eventuale parte intera e delle moltiplicazioni successive per la sua eventuale parte decimale.

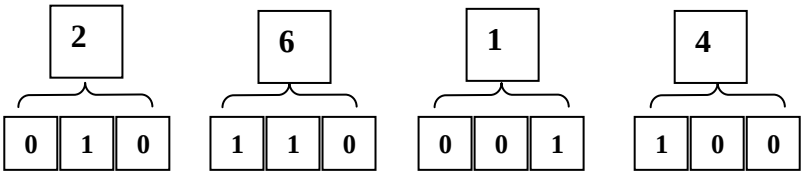
Tale metodo si semplifica di molto nel caso che una delle basi sia quella binaria e l'altra sia una base caratterizzata dall'essere esprimibile come potenza di 2 (ad esempio base8, base 16, base 32, etc.)

Vediamo ora nel dettaglio come si procede.

Conversione ottale-binario-ottale

(*) da base $\mathbf{b_1 = 8}$ a base $\mathbf{b_2 = 2}$
Poiché come abbiamo già visto $\mathbf{\log_2 8 = 3}$ per convertire un numero espresso in ottale nell'equivalente espresso in binario occorre **convertire ciascuna cifra ottale in binario utilizzando tre bit per la rappresentazione.**

Esempio
Supponiamo di volere convertire il numero $\boldsymbol{(2614)_8}$ *in binario utilizzando il metodo sopra spiegato.*

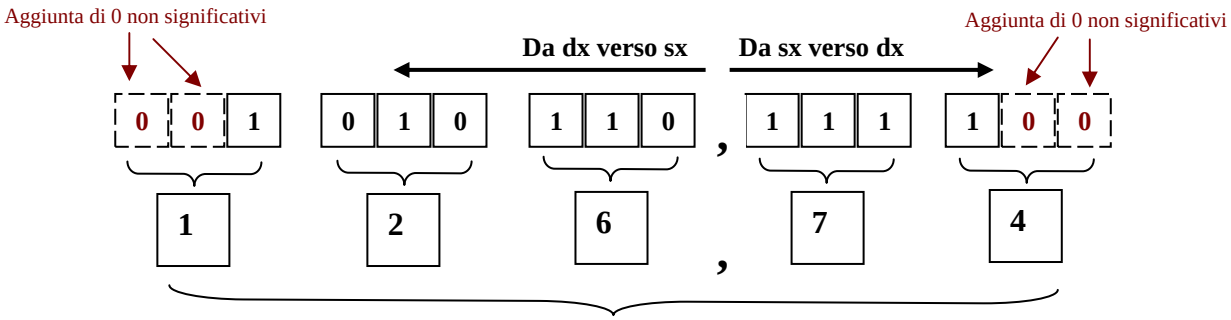


Da ciò deriva che $\boldsymbol{(2614)_8 = (010110001010)_2}$

(*) da base $\mathbf{b_1 = 2}$ a base $\mathbf{b_2 = 8}$
Poiché come abbiamo già visto $\mathbf{\log_2 8 = 3}$ per convertire un numero espresso in binario nell'equivalente espresso in ottale occorre dapprima **raggruppare a 3 a 3,** partendo dalla virgola, le cifre del numero binario per la parte intera da destra verso sinistra e per la parte decimale da sinistra verso destra aggiungendo degli zeri se necessario. Poi occorre **convertire ciascun gruppo di 3 cifre binarie** precedentemente individuato **nella sua cifra equivalente in ottale.**

Esempio
Supponiamo di volere convertire il numero $\boldsymbol{(1010110,1111)_2}$ *in ottale utilizzando il metodo sopra spiegato.*

Si parte da $\boldsymbol{(1010110,1111)_2}$ *raggruppando le cifre a gruppi di* ***3***

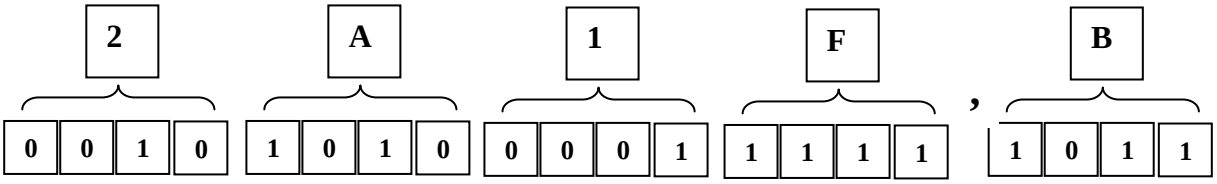


$\mathbf{(126,74)_8}$

Conversione esadecimale-binario-esadecimale

(*) da base $\mathbf{b_1 = 16}$ a base $\mathbf{b_2 = 2}$
Poiché come abbiamo già visto $\mathbf{\log_2 16 = 4}$ per convertire un numero espresso in esadecimale nell'equivalente espresso in binario occorre **convertire ciascuna cifra esadecimale in binario utilizzando quattro bit per la rappresentazione.**

Esempio
Supponiamo di volere convertire il numero $\boldsymbol{(2A1F,B)_{16}}$ *in binario utilizzando il metodo sopra spiegato.*



Da ciò deriva che $\boldsymbol{(2A1F,B)_{16} = (0010101000011111,1011)_2}$

(*) da base $\mathbf{b_1 = 2}$ a base $\mathbf{b_2 = 16}$
Poiché come abbiamo già visto $\mathbf{\log_2 16 = 4}$ per convertire un numero espresso in binario nell'equivalente espresso in esadecimale occorre dapprima **raggruppare a 4 a 4,** partendo dalla virgola, le cifre del numero binario per la parte intera da destra verso sinistra e per la parte decimale da sinistra verso destra aggiungendo degli zeri se necessario. Poi occorre **convertire ciascun gruppo di 4 cifre binarie** precedentemente individuato **nella sua cifra equivalente in esadecimale.**

Esempio
Supponiamo di volere convertire il numero $\boldsymbol{(1010110,101)_2}$ *in ottale utilizzando il metodo sopra spiegato.*

Si parte da $\boldsymbol{(1010110,101)_2}$ *raggruppando le cifre a gruppi di* ***4***