Scuola di Ingegneria Industriale

Laurea in Ingegneria Energetica

Laurea in Ingegneria Meccanica

POLITECNICO DI MILANO

Dipartimento di Elettronica, Informazione e Bioingegneria





Informatica B

Sezione D

Franchi Alessio Mauro, PhD *alessiomauro.franchi@polimi.it*

Elementi di logica

Data la formula:

not(A or B) and (A or not(C))

1) Si costruisca la sua tabella di verità.

A	B	C	A or B	Not(A or B)	not(C)	A or not(C)	=
0	0	0	0	1	1	1	1
1	0	0	1	0	1	1	0
0	1	0	1	0	1	1	0
1	1	0	1	0	1	1	0
0	0	1	0	1	0	0	0
1	0	1	1	0	0	1	0
0	1	1	1	0	0	0	0
1	1	1	1	0	0	1	0

Elementi di logica

2) Si consideri ora la condizione scritta in linguaggio C, in cui x e y siano due variabili di tipo int:

!(x > 4 || x < 9) && (x > 4 || !(x == y))

ottenuta dalla prima formula sostituendo la variabile A con x > 4, B con x < 9 e C con x==y.

Si risponda alle seguenti domande:

(i) La condizione è vera o falsa quando x vale 1 e y vale 2?

(ii) La condizione è vera per qualsiasi valore di x e y?

(iii) La condizione è falsa per qualsiasi valore di x e y?

i: in questo caso A = 0; B = 1; C = 0. Dalla tabella di verità leggo quindi che la condizione è falsa!

ii: No, abbiamo appena visto un esempio in cui è falsa;

iii: L'unico caso in cui è vera, dalla tabella di verità, accade quando A=B=C=0; icioè quando x<=4, x>=9, e x!=y; questo è impossibile, quindi è sempre falsa.

Elementi di logica

not(A or B) and (A or not(C))

(c) Semplificare l'espressione nelle variabili A, B, C usando le proprietà dell'algebra di Boole.

Applico De Morgan:
not A and not B and (A or not(C))

Applico la proprietà distributiva dell'AND
(not(A) and not(B) and A) or (not(A) and not(B) and not(C))

Principio di contraddizione
FALSE or (not(A) and not(B) and not(C))

(not(A) and not(B) and not(C))

Elementi di logica

Assumendo che c1 e c2 siano due variabili di tipo char, che memorizzano rispettivamente i valori ‘e’ ed ‘m’, indicare nella seguente tabella, per ognuna delle espressioni logiche

- se l’espressione è vera o falsa (per i valori delle variabili sopra indicati),
- se è sempre vera per qualsiasi valore che le due variabili possono assumere,
- se è sempre falsa per qualsiasi valore che le due variabili possono assumere.

Si fornisca una giustificazione per ogni risposta inserita nella tabella

((c1 != ‘e’) && (c2 == ‘m’)) || ((c1 != ‘h’) && (c2 == ‘m’))
(c1 < ‘g’) || !((c1 <= ‘g’) && (c1 != ‘g’))
(c1 <= ‘m’) || ((c2 > ’m’) || !(c2 > c1))

Elementi di logica

	V/F?	Sempre vera?	Sempre falsa?
((c1 != ‘e’) && (c2 == ‘m’)) || ((c1 != ‘h’) && (c2 == ‘m’))	V	NO	NO
(c1 < ‘g’) || !((c1 <= ‘g’) && (c1 != ‘g’))	V	SI	NO
(c1 <= ‘m’) || ((c2 > ’m’) || !(c2 > c1))	V	Si	NO

c1 = 'e'
c2 = 'm'

Elementi di logica

	V/F?	Sempre vera?	Sempre falsa?
((c1 != ‘e’) && (c2 == ‘m’)) || ((c1 != ‘h’) && (c2 == ‘m’))	V	NO	NO
(c1 < ‘g’) || !((c1 <= ‘g’) && (c1 != ‘g’))	V	SI	NO
(c1 <= ‘m’) || ((c2 > ’m’) || !(c2 > c1))	V	Si	NO

c1 = 'e'
c2 = 'm'

Richiamo di codifica binaria

-1313,3125

La parte intera vale 1313 (non si considera il segno, si usa un bit extra apposito)

1313 → 10100100001

La parte frazionaria vale 0,3125, che in binario diventa 0101.

Quindi si ha che 1313,3125 in binario vale 10100100001,0101

Lo normalizzo: $1,01001000010101 \times 2^{10}$ (lo scrivo nella forma $1,M \times B^E$); E è uguale a 10 perchè ho spostato la virgola a sinistra di 10 posizioni.

Converto l'esponente con la codifica “eccesso 127”; converto quindi in binario
10+127 = 137 → 10001001

Quindi -1313.3125 in binario è 1 10001001 01001000010101000000000

Richiamo di codifica binaria

0.1015625

La parte intera vale 0

La parte frazionaria vale 0,1015625, che in binario diventa 0001101

Quindi si ha che 0.1015625 in binario vale 0,0001101

Lo normalizzo: $0.00011012 = 1.1012 \times 2^{-4}$. (lo scrivo nella forma $1,MxB^E$); E è uguale a -4 perchè ho spostato la virgola a destra di 4 posizioni.

Converto l'esponente con la codifica “eccesso 127”; converto quindi in binario
-4+127 = 123 ⟶ 01111011

Quindi 0.1015625 in binario è 00111101101000000000000000000

Programmazione C

Esercizio 1

1. Il programa legge da tastiera una matrice quadrata DIM x DIM e calcola la somma degli elementi di una matrice colonna per colonna.
2. 6 6 6

Esercizio 2

1. Il programma legge da tastiera una sequenza di numeri, sino a quando l'utente inserisce uno 0 oppure si raggiunge il limite massimo di elementi nell'array; legge poi un ulteriore intero (compreso tra 0 e la dimensione massima dell'array). A questo punto “suddivide” l'array in due parti in base al numero letto e calcola la differenza tra la somma dei numeri nella prima porzione e quella dei numeri nella prima.

   Se tale differenza è pari a zero, il programma stampa il valore che è stato utilizzato per dividere la sequenza in due parti. Se la differenza è minore di zero il proramma stampa tutti i valori nella seconda porzione della sequenza.
2. Viene stampato il valore 4 (la differenza è nulla!)

Programmazione C

Esercizio 3

1. Il programa calcola e stampa il triangolo di tartaglia;
2. Si potrebbe “incolonnare” bene ogni elemento, a forma di triangolo, inserendo un numero adeguato di spazi che dipende dalla riga attuale!