ALGEBRA DI BOOLE

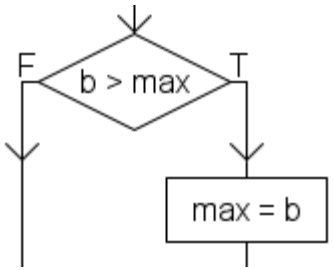
Fondamenti di Programmazione 2021/2022

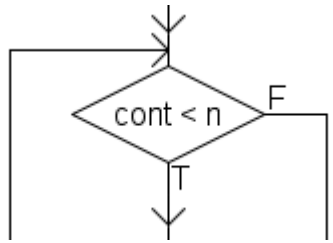
Francesco Tortorella

(si ringrazia la prof. Sabrina Senatore per parte del materiale)

Usiamo la logica!

- All'interno dei costrutti di selezione e ciclici abbiamo usato delle espressioni il cui risultato è un **valore logico**.
  - `b > max`
  - `cont < n`
- Un'espressione di questo tipo può assumere solo uno tra due valori possibili:
  - **`vero  (true)`**
  - **`falso (false)`**





Usiamo la logica!

- Anche nella vita reale abbiamo spesso a che fare con espressioni del genere:
  - Oggi piove
  - Paolo è più alto di Antonio
  - Il mio numero di anni è maggiore di 18
- … e ci capita di usarli in contesti condizionali e ciclici
  - Se hai più di 18 anni, voti
  - Sali finché arrivi al 5° piano

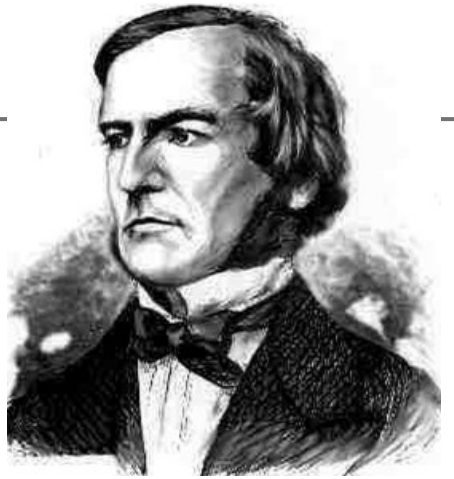
Espressioni logiche

- Nell'ambito delle espressioni possibili che possiamo ritrovare all'interno di un algoritmo, le più semplici sono quelle basate su un confronto:
  - Uguale: `a == b`
  - Diverso: `a != b`
  - Maggiore: `a > b`
  - Maggiore o uguale: `a >= b`
  - Minore: `a < b`
  - Minore o uguale: `a <= b`

Espressioni logiche

- Queste però da sole non bastano per esprimere condizioni più complesse.
  - "Se x è compreso tra 18 e 30 …"
- In questo caso la condizione è vera se `x >= 18` **e contemporaneamente** `x <= 30`
- È quindi necessario avere la possibilità di combinare espressioni logiche semplici per ottenere espressioni logiche più complesse.
- In matematica, questo è possibile grazie all'algebra elementare che consente di scrivere espressioni di tipo numerico.

Algebra di Boole



- Lo strumento analogo in logica è l'**algebra di Boole**, da George Boole, matematico del 19° sec. (1815-1864)
- Boole ha sviluppato un sistema matematico (algebra) con l'obiettivo di meccanizzare i processi logici.
- *An Investigation of the Laws of Thought, on Which are founded the Mathematical Theories of Logic and Probabilities* (1854)
  - propone una nuova impostazione della logica: dopo aver rilevate le analogie fra oggetti dell'algebra e oggetti della logica, riconduce le composizioni degli enunciati a semplici operazioni algebriche (Wikipedia)

Algebra di Boole

- È un'algebra in cui le variabili (e le funzioni) possono assumere solamente i valori vero (true) e falso (false)
- Componenti dell'algebra di Boole:
  - Gli **operatori booleani**: legano insieme le espressioni logiche (o proposizioni)
  - Gli **operandi booleani** (proposizioni): sono valori logici che possono assumere uno tra due i due valori possibili **`vero`** (**`true`**) e **`falso`** (**`false`**)

Algebra di Boole

- Operatori dell'algebra di Boole:
  - **and** congiunzione
  - **or** disgiunzione
  - **not** negazione
- Gli operatori and e or sono **binari**, cioè lavorano su due operandi.
- L'operatore not è **unario**, cioè lavora su un solo operando.

Algebra di Boole

- Come accade per gli operatori aritmetici
  - gli operatori binari si scrivono tra i due operandi
  - l'operatore unario si antepone all'unico operando
- Assumiamo che A e B siano due **variabili booleane**, cioè due variabili che possono assumere uno tra i due valori **`vero`** (**`true`**) e **`falso`** (**`false`**). Si scrive:
  - A and B
  - A or B
  - not A

Operazioni booleane

- Per capire quale sia il risultato delle operazioni booleane che possiamo effettuare con gli operatori visti, consideriamo quali siano le diverse combinazioni che possono assumere i loro operandi

A
F(alse)
T(rue)

A	B
F	F
F	T
T	F
T	T

Operazioni booleane

- L'operazione **A and B** restituisce **`true`** se e solo se sia A che B sono **`true`**, altrimenti restituisce **`false`**
- L'operazione **A or B** restituisce **`false`** se e solo se sia A che B sono **`false`**, altrimenti restituisce **`true`**
- L'operazione **not A** restituisce **`true`** se A è **`false`**, restituisce **`false`** se A è **`true`**

Operazioni booleane

A	not A
F	T
T	F

A	B	A or B
F	F	F
F	T	T
T	F	T
T	T	T

A	B	A and B
F	F	F
F	T	F
T	F	F
T	T	T

Operazioni booleane

- Esistono diverse notazioni alternative
  - and: $\wedge$, &&
  - or: $\vee$, ||
  - not: ¬, !
- Gli operatori &&, || e ! sono impiegati sia in Algobuild che nel linguaggio C e C++
- Esempio:
  - espressione vera se x è compreso tra 18 e 30: **(x >= 18) && (x <= 30)**

Precedenza degli operatori

- È possibile costruire un'espressione in cui siano presenti più operatori.
- In questo caso va tenuto conto di quale sia l'ordine di applicazione (precedenza) degli operatori in modo che non ci siano ambiguità sul risultato finale.
- L'ordine di precedenza degli operatori booleani è:
  1. not
  2. and
  3. or
- È possibile imporre un ordine di applicazione diverso tramite l'uso di parentesi.

Precedenza degli operatori

- Per quali valori di x e y sono vere le seguenti espressioni?
  - **(x >= 10) && (x <= 20) || (x < 5)**
  - **(x > 2) || (x < 0) && (x > -10)**
  - **(x == 1) || (x==10) || (x > 100)**
  - **((x == 0) || (x == 1)) && (y > 3)**
  - **(x == 3) && (x != 3)**
  - **(y > 2) || (y <= 2)**

Precedenza degli operatori

- Per quali valori di x e y sono vere le seguenti espressioni?
  - **(x >= 10) && (x <= 20) || (x < 5)**
  - **(x > 2) || (x < 0) && (x > -10)**
  - **(x == 1) || (x==10) || (x > 100)**
  - **((x == 0) || (x == 1)) && (y > 3)**
  - **(x == 3) && (x != 3)** — contraddizione **(A and not A)**
  - **(y > 2) || (y <= 2)** — tautologia **(A or not A)**

Proprietà degli operatori

Proprietà		
Elemento neutro	A && true = A	A || false = A
Minimo e massimo	A && false = false	A || true = true
Idempotenza	A && A = A	A || A = A
Complemento	A && (!A) = false	A || (!A) = true
Commutativa	A && B = B && A	A || B = B || A
Associativa	A && (B && C) = (A && B) && C	A || (B || C) = (A || B) || C
Distributiva	A && (B || C) = A && B || A && C	A || (B && C) = (A || B) && (A || C)
Assorbimento	A && (A || B) = A	A || (A && B) = A
Teoremi di De Morgan	!(A && B) = !A || !B	!(A || B) = !A && !B

Tabella della verità

- È possibile esprimere espressioni a più variabili booleane tramite una forma tabellare (**tabella della verità**)
- Nota l'espressione, è facile ottenere la tabella

a	b	c	a AND b	NOT(a) AND c	(a AND b) OR (NOT(a) AND c)
F	F	F	F	F	F
F	F	T	F	T	T
F	T	F	F	F	F
F	T	T	F	T	T
T	F	F	F	F	F
T	F	T	F	F	F
T	T	F	T	F	T
T	T	T	T	F	T

Tabella della verità

- In altri casi, potremmo conoscere la tabella, ma non sappiamo qual è l'espressione relativa

- Ci basta trovare una delle possibili espressioni equivalenti che hanno questa tabella della verità

a	b	c	Espressione?
F	F	F	F
F	F	T	F
F	T	F	F
F	T	T	T
T	F	F	F
T	F	T	T
T	T	F	T
T	T	T	T

Dalla tabella all'espressione

1. Identificare di tutte le righe che hanno valore di verità **T**
2. per ogni riga identificata si costruisce una sotto-espressione formata dall'**and** di tutte le lettere prese nella loro forma naturale o complementata (negata) seguendo le seguenti regole:
   - le lettere che nella riga in esame hanno valore **T** sono prese nella forma naturale;
   - le lettere che nella riga in esame hanno valore **F** sono prese nella forma complementata;
3. le sotto-espressioni così ottenute vengono connesse in **or** tra loro, per realizzare l'espressione complessiva desiderata.

Dalla tabella all'espressione

	a	b	c	espressione	
riga 0	F	F	F	F	
riga 1	F	F	T	F	
riga 2	F	T	F	F	
riga 3	**F**	**T**	**T**	**T**	⇦
riga 4	T	F	F	F	
riga 5	**T**	**F**	**T**	**T**	⇦
riga 6	**T**	**T**	**F**	**T**	⇦
riga 7	**T**	**T**	**T**	**T**	⇦

Dalla tabella all'espressione

	a	b	c	espressione		
riga 0	F	F	F	F		
riga 1	F	F	T	F		
riga 2	F	T	F	F		
riga 3	**F**	**T**	**T**	**T**	⇦	**(!a) && b && c**
riga 4	T	F	F	F		
riga 5	**T**	**F**	**T**	**T**	⇦	**a && (!b) && c**
riga 6	**T**	**T**	**F**	**T**	⇦	**a && b && (!c)**
riga 7	**T**	**T**	**T**	**T**	⇦	**a && b && c**

Dalla tabella all'espressione

	a	b	c	espressione		
riga 0	F	F	F	F		
riga 1	F	F	T	F		
riga 2	F	T	F	F		
riga 3	**F**	**T**	**T**	**T**	⇦	**(!a) && b && c**
riga 4	T	F	F	F		
riga 5	**T**	**F**	**T**	**T**	⇦	**a && (!b) && c**
riga 6	**T**	**T**	**F**	**T**	⇦	**a && b && (!c)**
riga 7	**T**	**T**	**T**	**T**	⇦	**a && b && c**

Espressione complessiva: (!a) && b && c || a && (!b) && c || a && b && (!c) || a && b && c