



UNIVERSITÀ DI PARMA

# Le Espressioni

*We must either institute conventional forms  
of expression or else pretend that we have  
nothing to express.*

*George Santayana, Soliloquies in England*

- Operatori & Operandi
  - Tipologie
- Espressioni
- Valutazione
- Precedenza & Associatività
- Effetti collaterali

SUMMARY

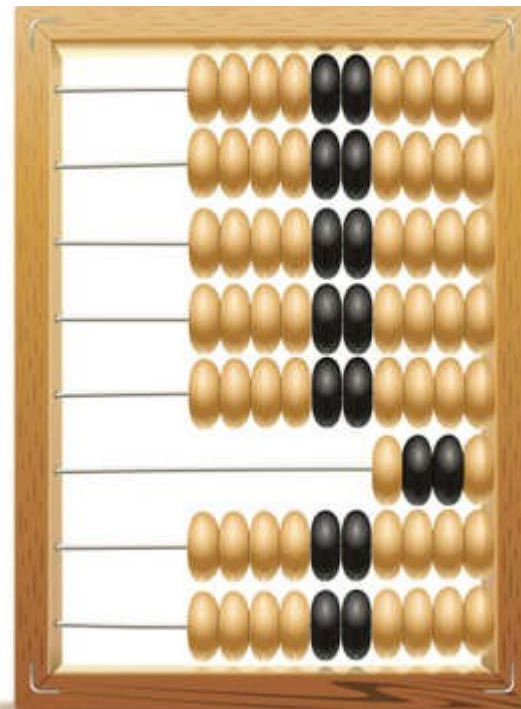


- Piccolo preambolo
  - Statement  $\rightarrow$  “Fai qualcosa” ovvero azione
  - Expression  $\rightarrow$  “Valuta qualcosa” ovvero “calcolo”
    - Alcune espressioni sono anche statement  $\rightarrow =$
- Le espressioni sono combinazioni di
  - Operatori
  - Operandi

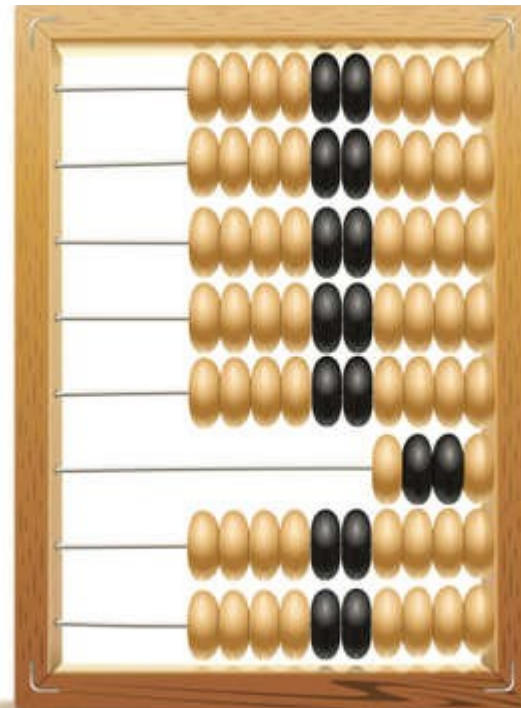
$$A = B * C + D * 17 / K$$

- Combinano uno o più operandi e permettono di valutare “operazioni”
  - Aritmetiche
  - Relazionali
  - Logiche
  - Bit a bit
  - Assegnamento
  - Condizionali e speciali
- A seconda del numero di operandi coinvolti:
  - Unari
  - Binari
  - Ternari

- Binari:
  - $+$   $\rightarrow$  somma
  - $-$   $\rightarrow$  sottrazione
  - $*$   $\rightarrow$  moltiplicazione
  - $/$   $\rightarrow$  divisione
  - $\%$   $\rightarrow$  resto della divisione intera (modulo)
- Unari:
  - $+ \text{ o } -$   $\rightarrow$  segno
  - $++$   $\rightarrow$  incremento di 1
  - $--$   $\rightarrow$  decremento di 1



- Permettono calcoli tra numeri interi o a virgola mobile
  - Eccetto %
- Attenzione a /
  - Se operandi interi
    - Divisione intera
      - $5/6 \rightarrow 0$
    - Se operando negativo problemi arrotondamento
      - $-9/7 \rightarrow -1$  oppure  $-2$



- Operatori di incremento o decremento
  - Incremento variabile “a” di 1  $\rightarrow ++a$ ;
  - Decremento variabile “a” di 1  $\rightarrow --a$ ;
- Operazioni comuni
- **Caveat:** comportamento differente se prefisso o postfixo
  - $++a \neq a++$
  - Incrementano entrambi “a” ma se in espressioni:
    - Prefisso  $\rightarrow$  eseguo incremento prima di altre operazioni
    - Postfisso  $\rightarrow$  eseguo incremento dopo tutte le altre operazioni

- Permettono assegnamento valore a variabile
- Semplice:
  - $=$   $\rightarrow$  assegnamento
- Composti:
  - $+=$   $\rightarrow$  assegnamento con somma
  - $-=$   $\rightarrow$  ...
  - $*=$
  - $\%=$
  - ...



- A differenza di altri operatori, gli assegnamenti hanno limiti su cosa posso avere come operandi
- A destra dell'operatore di assegnamento
  - Numeri, variabili, espressioni ecc.
  - Right Value o R-Value
- A sinistra dell'operatore di assegnamento
  - Solo variabili o comunque indirizzo memoria
  - Left Value o L-Value



- A volte necessità di valutare situazioni
  - Il numero  $x$  è pari o dispari?
  - $a$  è più piccolo o più grande di  $b$ ?
- Vero o Falso, in C:
  - Vero  $\rightarrow$  qualunque numero diverso da 0
  - Falso  $\rightarrow 0$

- $== \rightarrow$  controllo uguaglianza (no virgola mobile!)
- $!= \rightarrow$  controllo differenza (non esiste  $\neq$ )
- $> \rightarrow$  piú grande di
- $< \rightarrow$  piú piccolo di
- $>= \rightarrow$  piú grande o eguale di
- $<= \rightarrow$  piú piccolo o eguale di
- Hanno come risultato:
  - Vero  $\rightarrow 1$  (diverso da  $\emptyset$ )
  - Falso  $\rightarrow \emptyset$

Name	Operator	Expression	Evaluates
Equals	<b>==</b>	5 == 5 K == 10	true depends
Not Equals	<b>!=</b>	50 != 25 100 != 100	true false
Less than	<b>&lt;</b>	100 < 200 P < Q	true depends
Greater than	<b>&gt;</b>	100 > 200 P > Q	false depends
Less than or equals	<b>&lt;=</b>	25 <= 26 25 <= 25	true true
Greater than or equals	<b>&gt;=</b>	1000 >= 1000 K >= (P + Q)	true depends

- Operatori relazionali  $\rightarrow$  test “semplici”
  - $a$  è compreso nell’intervallo 10-18?
  - $a > 10$  ma anche  $a < 18$ !
- Gli operatori logici permettono di costruire test più complessi
- Combinando:
  - Espressioni che coinvolgono operatori relazionali
  - Espressioni che assumono valori vero/falso

- Operatori il cui risultato è vero(1)/falso(0)
- `&&` → and logico
- `||` → or logico
- `!` → not logico

&&	!alive	alive
!dead		
dead		

# Operatori logici: tabelle di verità

<b>a</b>	<b>b</b>	<b>a &amp;&amp; b</b>	<b>a    b</b>	<b>!a</b>
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

- Combinazioni di operatori e operandi
- Esempi:
  - $17$  → espressione senza operandi
  - $17 * 2 + 3$  → espressione costante
  - $a = b * 17 + c / 4 + e \% 5 * 2$  → calcolata in esecuzione



- Operatori differenti possono avere precedenze differenti
  - $3*4+2$
  - Nella valutazione delle espressioni si parte dagli operatori con maggiore precedenza
- A parità di precedenza gli operatori vengono valutati in base alla associatività
  - $a\%b\%c$              $(\rightarrow)$
  - $a = b = c;$          $(\leftarrow)$
- **Nel dubbio ()**

Operator	Description	Associativity
() [] . -> ++ --	Parentheses or function call Brackets or array subscript Dot or Member selection operator Arrow operator Postfix increment/decrement	left to right
++ -- + - ! ~ (type) * & sizeof	Prefix increment/decrement Unary plus and minus not operator and bitwise complement type cast Indirection or dereference operator Address of operator Determine size in bytes	right to left
* / %	Multiplication, division and modulus	left to right
+ -	Addition and subtraction	left to right
<< >>	Bitwise left shift and right shift	left to right
< <= > >=	relational less than/less than equal to relational greater than/greater than or equal to	left to right
== !=	Relational equal to or not equal to	left to right
&&	Bitwise AND	left to right
^	Bitwise exclusive OR	left to right
	Bitwise inclusive OR	left to right
&&	Logical AND	left to right
	Logical OR	left to right
? :	Ternary operator	right to left
= += -= *= /= %= &= ^=  = <<= >>=	Assignment operator Addition/subtraction assignment Multiplication/division assignment Modulus and bitwise assignment Bitwise exclusive/inclusive OR assignment	right to left
,	comma operator	left to right

- Condizionali
  - ?
  - Unico operatore ternario del C
- Speciali
  - Li dettaglieremo man mano che li incontreremo
    - sizeof() dimensione di un dato
    - & indirizzo di una variabile
    - \* puntatori o accesso memoria
    - , concatenazione di espressioni
    - [] indice array
    - ...

- Il C riesce ad operare anche a basso livello sui dati
  - Singolo bit
  - Dati interi
- Perché si usano?
  - Efficienza
  - Rappresentazione compatta di valori vero/falso ovvero 1/0

- Lavorano a livello dei singoli bit di ciascun dato
  - $\&$   $\rightarrow$  and bit a bit
  - $|$   $\rightarrow$  or bit a bit
  - $\sim$   $\rightarrow$  not bit a bit
  - $\wedge$   $\rightarrow$  xor bit a bit
  - $\ll$   $\rightarrow$  left shift
  - $\gg$   $\rightarrow$  right shift



UNIVERSITÀ DI PARMA

# Le Espressioni



KEEP  
CALM  
IT'S  
QUESTION  
TIME

*We must either institute conventional forms  
of expression or else pretend that we have  
nothing to express.*

*George Santayana, Soliloquies in England*