

Espressioni

Un'espressione rappresenta un valore, solitamente vincolato ad un tipo. Per poter associare un valore ad un'espressione, questa deve essere valutata. I valori rappresentabili dalle espressioni sono detti **esprimibili**:

$$Eval = \text{bool} \cup \text{int}$$
 (ovvero i terminali della grammatica)

Le caratteristiche principali delle espressioni sono:

- Arietà, che indica il numero di operatori ai quali si applica l'espressione (es: arietà 2 == operatore binario);
- Tipo di notazione, che può essere infissa, suffissa o postfissa;
- Regole di associatività e precedenza (es: ordine di precedenza tra operatori dello stesso livello);
- Ordine di valutazione degli operatori (es: da sinistra verso destra);
- Generazione di side-effects;
- Possibilità di overloading degli operatori;
- Possibilità di accettare tipi misti.

Notazione

A seconda di come si rappresenta l'espressione varia il modo in cui si determina la semantica e di conseguenza la sua valutazione. La notazione infuisce anche sulle regole di associatività e precedenza.

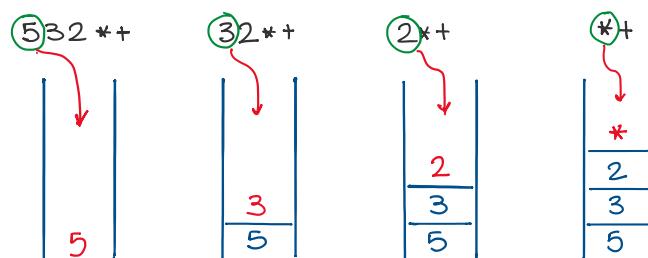
Notazione postfissa La notazione postfissa viene valutata usando una pila LIFO, secondo il seguente algoritmo:

1. Leggi il prossimo simbolo dell'espressione;
2. Se simbolo == operatore:
 - Applica l'operatore agli operandi in cima alla pila e poi cancellali;
 - Memorizza il risultato in cima alla pila.

Se il simbolo letto è un operando inseriscilo in cima alla pila;

3. Torna a (1).

Esempio:



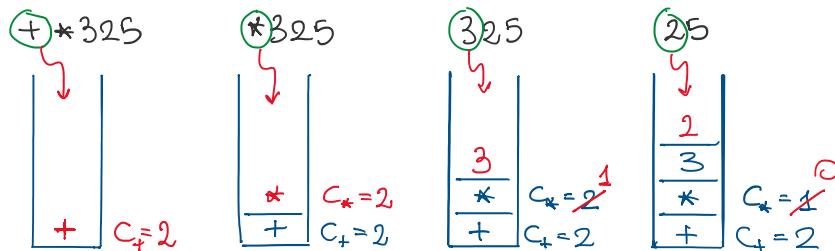
OPERANDO IN CIMA ARIETÀ = 2



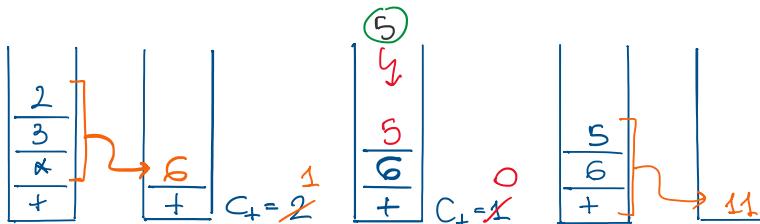
Notazione prefissa La notazione prefissa non necessita di regole di associatività e parentesi. Viene valutata come segue:

1. Leggi il prossimo simbolo dell'espressione e mettilo in cima alla lista;
2. Se simbolo == operatore, inizializzo una variabile C con l'arietà dell'operatore.
Se il simbolo letto è un operando inseriscilo in cima alla pila e decrementa l'operatore ($C--$);
3. Se $C \neq 0$, torna a (1);
Se $C == 0$:
 - Applica l'operatore agli operandi in cima alla pila e poi cancellali;
 - Salva il risultato in cima alla pila;
 - $C = n - m$, con n arietà del nuovo operatore in cima alla pila e m numero operandi sopra l'operatore.
4. Se la pila non è vuota, torna a (1).

Esempio:



$C = 0$ DELL'OPERATORE IN CIMA \rightarrow ESEGNO



Ordine di valutazione

I principali problemi legati all'ordine di valutazione si hanno a causa di: operandi non definiti, effetti collaterali e aritmetica finita.

Operandi non definiti Se devo valutare la seguente espressione $a == 0?b : b/a$, posso avere dei problemi calcolo il risultato di b/a se $a = 0$. Il problema si pone solo se ho una valutazione totale degli operandi.

La valutazione può essere:

- Lazy, ovvero valuta solo gli operandi necessari. Nel caso sopra, se $a == 0$ valuta solo b ;
- Eager, ovvero valuta tutti gli operandi a prescindere dal risultato.

Generazione di effetti collaterali Se devo valutare il seguente codice:

```
a = 10;
b = a + fun(a);
```

```
fun(a) = a++;
```

Se valuto prima $fun(a)$ ottengo come risultato finale $b = 11 + 11$.

Se valuto prima a ottengo $b = 10 + 11$.

Aritmetica finita il problema si pone quando eseguo operazioni sul massimo numero rappresentabile.