

## 1. Espressioni Condizionali

Si considera un semplice linguaggio per la formulazione di espressioni condizionali, ossia espressioni booleane i cui operandi sono generati da un confronto di due espressioni aritmetiche, con un operatore di relazione. Le espressioni aritmetiche si intendono costruite su numeri senza segno, manipolati dagli operatori addizione, sottrazione, moltiplicazione, divisione (quoziente), resto della divisione intera, elevamento a potenza. Nelle espressioni aritmetiche, l'elevamento a potenza è il più prioritario, seguono gli operatori moltiplicativi (moltiplicazione e divisione) ed infine quelli additivi (somma e sottrazione). Nelle espressioni booleane, NOT è l'operatore più prioritario, seguono le AND, quindi le OR. Gli operatori di relazione sono più prioritari degli operatori booleani.

Esempi di espressioni condizionali:

$a \leq b \ \&\& \ c > 1 \ || \ a == 4$  che è equivalente a:  $(a \leq b \ \&\& \ c > 1) \ || \ a == 4$   
 $a \% b == 0 \ \&\& \ (a > 2 \ || \ b - 12 > 0)$

L'obiettivo del progetto, basato sui pattern Composite, Builder (ricorsivo) e Visitor, è quello di acquisire da input un'espressione condizionale, farne l'analisi sintattica e costruire l'albero sintattico (composite) corrispondente. A partire dall'albero occorre valutare il risultato della condizione sulla base di un *contesto* o ambiente (fornito esternamente mediante un file properties), che stabilisce i valori delle variabili intere usate nelle espressioni aritmetiche (se una variabile non è presente nel contesto, si assume che il suo valore sia 0). Occorre altresì predisporre le operazioni di stampa in ordine simmetrico e postfixo del composite di una condizione.

Una particolarità del progetto è che l'operazione di valutazione deve applicare correttamente le condizioni di corto circuito sugli operatori AND e OR. Ad es. (corto circuito AND):  $b > 0 \ \&\& \ a/b > 2$ , se  $b$  è  $\leq 0$  tutta la AND deve valersi a false senza valutare il secondo "pezzo" che invece potrebbe dar luogo ad un'eccezione per divisione per 0. Similmente (corto circuito OR):  $b == 0 \ || \ a/b > 2$  se  $b$  è 0, tutta la OR deve valersi a true senza valutare il esplicitamente il secondo confronto della OR. Attenzione che come in Java, le condizioni di corto circuito si basano sull'ordine di elencazione dei confronti sinistro e destro dell'operatore AND o OR.

### 1.1 Grammatica

$\langle \text{cond} \rangle \rightarrow \langle \text{termb} \rangle [\text{OR } \langle \text{termb} \rangle]^*$   
 $\langle \text{termb} \rangle \rightarrow \langle \text{factb} \rangle [\text{AND } \langle \text{factb} \rangle]^*$   
 $\langle \text{factb} \rangle \rightarrow \langle \text{expr} \rangle \text{RELOP } \langle \text{expr} \rangle \mid \text{NOT } \langle \text{factb} \rangle \mid \text{OPAR } \langle \text{cond} \rangle \text{CPAR}$   
 $\langle \text{expr} \rangle \rightarrow [\text{PLUS} \mid \text{MINUS}] \langle \text{term} \rangle [(\text{PLUS } \langle \text{term} \rangle) \mid (\text{MINUS } \langle \text{term} \rangle)]^*$   
 $\langle \text{term} \rangle \rightarrow \langle \text{termp} \rangle [(\text{MULT } \langle \text{termp} \rangle) \mid (\text{DIV } \langle \text{termp} \rangle) \mid (\text{REM } \langle \text{termp} \rangle)]^*$   
 $\langle \text{termp} \rangle \rightarrow \langle \text{fact} \rangle [\text{POWER } \langle \text{fact} \rangle]^*$   
 $\langle \text{fact} \rangle \rightarrow \text{ID} \mid \text{NUM} \mid \text{OPAR1 } \langle \text{expr} \rangle \text{CPAR1}$

### 1.2 Simboli terminali

$\text{ID} \rightarrow ("A" \mid \dots \mid "Z" \mid "a" \mid \dots \mid "z") [("A" \mid \dots \mid "Z" \mid "a" \mid \dots \mid "z" \mid "0" \mid \dots \mid "9")]^*$   
 $\text{NUM} \rightarrow ("0" \mid \dots \mid "9") [("0" \mid \dots \mid "9")]^*$   
 $\text{OPAR} \rightarrow "("$   
 $\text{CPAR} \rightarrow ")"$   
 $\text{OPAR1} \rightarrow "["$   
 $\text{CPAR1} \rightarrow "]"$   
 $\text{RELOP} \rightarrow \text{EQ} \mid \text{NEQ} \mid \text{GT} \mid \text{GE} \mid \text{LT} \mid \text{LE}$   
 $\text{EQ} \rightarrow "="$

NEQ → "!="  
GT → ">"  
GE → ">="  
LT → "<"  
LE → "<="  
POWER → "^"  
DIV → "/"  
REM → "%"  
MULT → "\*"  
MINUS → "-"  
PLUS → "+"  
AND → "and" o "&&"  
OR → "or" o "||"  
NOT → "not" o "!"

Sotto espressioni aritmetiche in parentesi utilizzano le parentesi quadre. Sotto condizioni in parentesi usano le parentesi tonde.