Progetto POO # 1 - A.A. 2017/2018: Valutazione di una espressione aritmetica intera

Si ammettono gli operatori +,-,*,%,^ e valgono le usuali priorità della matematica, cioè (π priorità):

$\pi(\hat{}) > \pi(*,/,\%) > \pi(+,-)$

A parità di priorità, si assume l'associatività a sinistra. Eventualmente, si possono usare le parentesi per alterare le priorità intrinseche: un'espressione in parentesi () viene sempre valutata prima.

Algoritmo di valutazione

Si usano due stack: uno stack di operandi, ed uno stack di caratteri operatori. Quando arriva un operando, lo si inserisce in cima allo stack di operandi. Quando arriva un operatore, sia esso opc (*operatore corrente*), si procede come segue:

- A) Se opc è più prioritario dell'operatore affiorante dallo stack di operatori o tale stack è vuoto, si inserisce opc in cima allo stack degli operatori.
- B) Se opc non è più prioritario rispetto alla cima dello stack operatori, si preleva l'operatore **op** al top dello stack operatori, quindi si prelevano due operandi o2 (top) e o1 (top-1) dallo stack operandi (in caso di eccezioni, l'espressione è malformata). Si esegue o1 **op** o2. Il risultato è inserito in cima allo stack operandi. Si continua ad eseguire il passo B) se opc risulta ancora non più prioritario dell'operatore affiorante la cima dello stack operatori. Dopo questo, o perché opc è più prioritario dell'operatore in cima allo stack operatori o perché lo stack è vuoto, si applica il caso A).

Quando termina una (sotto)espressione, se lo stack operatori non è vuoto, si estraggono uno alla volta gli operatori presenti e si applicano ai rispettivi due operandi prelevati dallo stack operandi, come spiegato al punto B), inserendo ogni volta il risultato in cima allo stack operandi.
Quando lo stack operatori è vuoto, allora lo stack operandi dovrebbe contenere un solo elemento che è il risultato dell'espressione. Ogni altra situazione (stack operandi vuoto o con più di un elemento) denota una situazione di espressione malformata.
Si suggerisce di dividere il compito della valutazione in due metodi: int valutaOperando(…) e int valutaEspressione(…). Questi due metodi possono ricevere come parametro l'oggetto string tokenizer "acceso" sulla stringa espressione, in modo da ripartire sempre dall'ultima posizione raggiunta.
Per il confronto di operatori, introdurre una classe Precedenza che implementa Comparator<Character> il cui metodo compare(…) riceve due caratteri operatori e ritorna il loro confronto secondo le priorità della matematica.

Cosa succede se ci sono parentesi ?

Quando in valutaOperando(…) si incontra una parentesi aperta '(' anziché un normale operando intero, si invoca ricorsivamente la procedura valutaEspressione(…). Quando in valutaEspressione(…) si incontra una parentesi chiusa ')' come operatore, occorre ritornare l'operando in cima allo stack operandi (un solo elemento o l'espressione è malformata). Attenzione: al termine di un'espressione (o sotto espressione) se esistono operatori sullo stack di operatori, questi vanno ordinatamente processati, uno alla volta. Alla fine il risultato dell'espressione (o sotto espressione) si dovrebbe trovare in cima allo stack operandi (a meno di malformazioni).
Il metodo valutaEspressione(…) è opportuno che introduca i due stack come proprie variabili locali, in quanto assistono la valutazione della (sotto) espressione.

Il progetto dovrebbe realizzare un programma valutatore interattivo di espressioni aritmetiche, fornite a fronte di un prompt ">> ". Ogni espressione aritmetica deve essere seguita, sulla linea successiva, dal suo risultato o dalla segnalazione "Espressione malformata!". Si esce quando l'utente digita una pseudo espressione ".".

Sviluppi futuri

Nel seguito del corso si provvederà ad estendere il progetto prevedendo una minima GUI di interazione con l'utente e la verifica di casi evidenti di malformazioni in una espressione aritmetica mediante l'uso di espressioni regolari.