Expresiones en notaciones infija, prefija y sufija

Cuando usted escribe una expresión aritmética como B * C, la forma de la expresión le proporciona información para que pueda interpretarla correctamente. En este caso sabemos que la variable B está siendo multiplicada por la variable C, ya que el operador de multiplicación * aparece entre ellos en la expresión. Este tipo de notación se conoce como **infija** ya que el operador está *entre* los dos operandos sobre los que está actuando.

Considere otro ejemplo de notación infija, A + B * C. Los operadores + y * siguen apareciendo entre los operandos, pero hay un problema. ¿Sobre qué operandos actúan? ¿Opera el + sobre A y B o el * opera sobre B y C? La expresión parece ambigua.

De hecho, usted ha estado leyendo y escribiendo estos tipos de expresiones durante mucho tiempo y no le causan ningún problema. La razón de esto es que usted sabe algo sobre los operadores + y *. Cada operador tiene un nivel de **precedencia**. Los operadores de mayor precedencia se utilizan antes que los operadores de menor precedencia. Lo único que puede cambiar ese orden es la presencia de paréntesis. El orden de precedencia para los operadores aritméticos ubica la multiplicación y la división por encima de la suma y la resta. Si aparecen dos operadores de igual precedencia, se utiliza un ordenamiento o asociatividad de izquierda a derecha.

Interpretemos la expresión problemática A + B * C usando la precedencia de los operadores. B y C se multiplican primero y A se añade a ese resultado. (A + B) * C forzaría la suma de A y B antes de la multiplicación. En la expresión A + B + C, por precedencia (vía asociatividad), el + que está más a la izquierda operaría primero.

Aunque todo esto puede ser obvio para usted, recuerde que las computadoras necesitan saber exactamente qué operadores deben ejecutarse y en qué orden. Una forma de escribir una expresión que garantice que no habrá confusión con respecto al orden de las operaciones es crear lo que se denomina expresión **completamente agrupada**. Este tipo de expresión utiliza una pareja de paréntesis para cada operador. Los paréntesis dictan el orden de las operaciones; no hay ambigüedad. Tampoco es necesario recordar las reglas de precedencia.

La expresión A + B * C + D se puede reescribir como ((A + (B * C)) + D) para mostrar que la multiplicación ocurre primero, seguida por la adición que está más a la izquierda. A + B + C + D se puede escribir como ((A + B) + C) + D) ya que las operaciones de adición se asocian de izquierda a derecha.

Hay otros dos formatos de expresión muy importantes que, al principio, pueden no parecer obvios. Considere la expresión infija A + B. ¿Qué pasaría si moviéramos el operador antes de los dos operandos? La expresión resultante sería + A B. Del mismo modo, podríamos mover el operador al final. Obtendríamos A B +. Estas expresiones se ven un poco extrañas.

Estos cambios en la posición del operador con respecto a los operandos crean dos nuevos formatos de expresión, la **notación prefija** y la **notación sufija (o postfija)**. La notación prefija requiere que todos los operadores precedan a los dos operandos sobre los que actúan. La notación sufija, por otro lado, requiere que sus operadores aparezcan después de los operandos correspondientes. Algunos ejemplos más deberían ayudar a hacer esto un poco más claro (ver la tabla siguiente).

A + B * C se escribiría como + A * B C en la notación prefija. El operador de multiplicación aparece inmediatamente antes de los operandos B y C, denotando que el * tiene precedencia sobre el +. El operador de adición aparece entonces antes de la A y del resultado de la multiplicación.

Expresión infija	Expresión prefija	Expresión sufija
A + B	+ A B	A B +
A + B * C	+ A * B C	A B C * +

En notación sufija, la expresión sería A B C * +. Una vez más, el orden de las operaciones se conserva ya que el * aparece inmediatamente después de la B y la C, denotando que el * tiene precedencia, con el + apareciendo después. Aunque los operadores se movieron y ahora aparecen antes o después de sus respectivos operandos, el orden de cada operando se mantuvo exactamente igual en relación con los demás.

Ahora considere la expresión infija (A + B) * C. Recuerde que en este caso, la notación infija requiere los paréntesis para forzar que se lleve a cabo la adición antes de la multiplicación. Sin embargo, cuando A + B fue escrito en notación prefija, el operador de adición fue movido simplemente antes de los operandos, + A B. El resultado de esta operación se convierte en el primer operando para la multiplicación. El operador de multiplicación se mueve delante de toda la expresión, dándonos * + A B C. Igualmente, en notación sufija A B + obliga a que la adición ocurra primero. La multiplicación se puede hacer sobre ese resultado y el operando restante C. La expresión sufija correcta es entonces A B + C *.

Considere nuevamente estas tres expresiones (ver la tabla siguiente). Ha ocurrido algo muy importante. ¿A dónde se fueron los paréntesis? ¿Por qué no los necesitamos en las notaciones prefija y sufija? La respuesta es que los operadores ya no son ambiguos con respecto a los operandos sobre los que actúan. Solamente la notación infija requiere los símbolos adicionales. El orden de las operaciones dentro de las expresiones prefijas y sufijas está completamente determinado por la posición del operador y nada más. De muchas maneras, esto hace que la notación infija sea la notación menos deseable de usar.

Expresión infija	**Expresión prefija**	**Expresión sufija**
(A + B) * C	* + A B C	A B + C *

La tabla siguiente muestra algunos ejemplos adicionales de expresiones infijas y las expresiones equivalentes en notaciones prefija y sufija. Asegúrese de entender cómo son equivalentes en términos del orden de las operaciones que se están realizando.

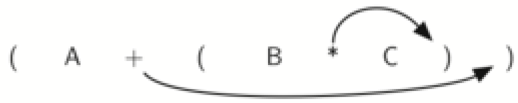
Expresión infija	**Expresión prefija**	**Expresión sufija**
A + B * C + D	+ + A * B C D	A B C * + D +
(A + B) * (C + D)	* + A B + C D	A B + C D + *
A * B + C * D	+ * A B * C D	A B * C D * +
A + B + C + D	+ + + A B C D	A B + C + D +

Conversión de expresiones infijas a notaciones prefija y sufija

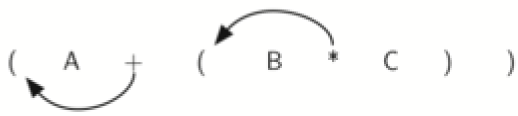
Hasta el momento, hemos utilizado métodos ad hoc para convertir entre expresiones infijas y las expresiones equivalentes en notaciones prefija y sufija. Como es de esperar, hay formas algorítmicas para realizar la conversión que permiten transformar correctamente cualquier expresión de cualquier complejidad.

La primera técnica que vamos a considerar utiliza la noción de una expresión completamente agrupada que se discutió anteriormente. Recordemos que A + B * C se puede escribir como (A + (B * C)) para mostrar explícitamente que la multiplicación tiene precedencia sobre la adición. Sin embargo, al observar más de cerca, puede verse que cada pareja de paréntesis también indica el comienzo y el final de un par de operandos con el operador correspondiente en la mitad.

Observe el paréntesis derecho en la subexpresión (B * C) anterior. Si tuviéramos que mover el símbolo de multiplicación a esa posición y quitar el paréntesis izquierdo correspondiente, nos daría B C *, de hecho, habríamos convertido la subexpresión a notación sufija. Si el operador de adición también es movido a la posición de su paréntesis derecho correspondiente y se elimina el paréntesis izquierdo que le corresponde, se produciría la expresión sufija completa (ver la figura siguiente).

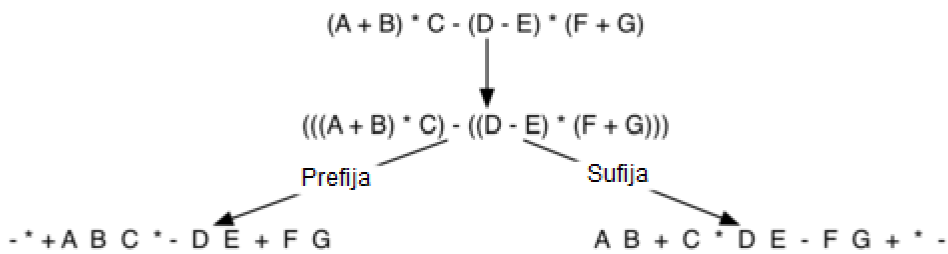


Si hacemos lo mismo pero en lugar de mover el símbolo a la posición del paréntesis derecho, lo movemos a la izquierda, obtenemos la notación prefija (ver la siguiente figura). La posición de la pareja de paréntesis es en realidad una pista sobre la posición final del operador encerrado entre ellos.



Así que, para convertir una expresión, independientemente de su complejidad, ya sea a notación prefija o a notación sufija, agrúpela completamente utilizando el orden de las operaciones. A continuación, mueva cada operador a la posición correspondiente de los paréntesis izquierdo o derecho dependiendo de si desea obtener la expresión en notación prefija o en notación sufija.

La siguiente es una expresión más compleja: (A + B) * C - (D - E) * (F + G).
La figura siguiente muestra la conversión a las notaciones sufija y prefija.



Expresiones matemáticas en Scheme.

En Scheme se pueden representar varios tipos de números: enteros, racionales y reales.
Por ejemplo: 5, −5, 2/3, 7/3, #i1, 4142135623731.

Los números no tienen distinción de algún tipo, todos son simplemente números.

Scheme permite también sumar, restar multiplicar, dividir y en general realizar muchas operaciones con números. Las operaciones tienen notación prefija, esto es: el operador se escribe primero.

Ejemplo:
5 + 3 se escribe (+ 5 3) y 3*5+15 ¿Cómo será? ---- NOTE QUE TODA EXPRESIÓN SE ESCRIBE ENTRE PARÉNTESIS ----

Al igual que en la aritmética, para evitar diferentes interpretaciones usamos paréntesis:
(3*5)+15 se escribe (+ (* 3 5) 15)

¿Qué es una Expresión? Es un conjunto de palabras que permiten expresar algo. Una expresión en Scheme es como una frase en español. Existen reglas para construir expresiones en Scheme:

Todas las expresiones en Scheme empiezan y terminan con paréntesis. Los operadores van primero (notación prefija).

Expresiones que contienen operaciones numéricas:
(+ 2 2) equivale a ???
(* 3 1) equivale a ???
(/ 10 0) equivale a ???
(* 7 (+ 4 5)) equivale a??

Al igual que en álgebra se pueden escribir muchas operaciones dentro de otras usando paréntesis:
(* (+ 2 2) (/ (* (+ 3 5) (/ 30 10)) 2)) equivale a 48. ¿Está de acuerdo?
Los paréntesis internos deben resolverse primero!

Escriba las expresiones en Scheme equivalentes a:
((5+6) * 4) / (17 + 9)
(5 + (4/8 - 2/8)) * ((1 - 18) / (23 / 7))

¿Y cómo se escriben operaciones como potencias, raíces, logaritmos y otras?
Estas operaciones se escriben siguiendo las reglas anteriormente vistas, pero no se pueden escribir signos como 3^4 o $\log_{10} 2$ por tanto se usan palabras reservadas (funciones predefinidas del lenguaje).
Ejemplo:
3^4 se escribe en Scheme (expt 3 4)
$\log_{10} 2$ se escribe en Scheme (/ (log 2) (log 10))
$\log_3 2$ se escribe en Scheme (/ (log 2) (log 3))
ln 2, se escribe en Scheme (log 2)

Algunas operaciones matemáticas en Scheme:

(sqrt z)	Raíz cuadrada principal de z (si z es real, la raíz cuadrada positiva).
(abs x)	Valor absoluto de x.
(sin z)	Seno de z.
(cos z)	Coseno de z.
(tan z)	Tangente de z.
(asin z)	Arcoseno de z.
(acos z)	Arcocoseno de z.
(atan z)	Arcotangente de z.
(max x1 [x2 . . . xk])	Máximo entre los argumentos.
(min x1 [x2 . . . xk])	Mínimo entre los argumentos.
(quotient n1 n2)	(n2 distinto de cero), cociente de n1 entre n2.
(remainder n1 n2)	(n2 distinto de cero), resto de n1 entre n2.
(expt z1 z2)	La potencia de z1 elevado a la z2 (con 0 elevado a la 0 = 1).
(exp z)	La potencia de e elevado a la z.
(log z)	Logaritmo natural de z.
(/ (log N) (log B))	Logaritmo en base B de N.
(gcd [n1 . . . nk])	Máximo común divisor entre los argumentos; sin argumentos, 0.
(lcm [n1 . . . nk])	Mínimo común múltiplo entre los argumentos; sin argumentos, 1.
(floor x)	Mayor entero menor o igual que x.
(ceiling x)	Menor entero mayor o igual que x.
(truncate x)	Parte entera de x.
(round x)	Entero más cercano a x, en caso de equidistancia número entero par más cercano.
Pi	3.141592653589793

EJERCICIOS:
-Construya ejemplos de uso de las funciones listadas anteriormente, analice la respuesta y concluya.
-Resuelva en Racket los ejercicios planteados a continuación.

$$\frac{35{,}7 \cdot 64 - 7^3}{45 + 5^2}$$

$$\frac{5}{4} \cdot 7 \cdot 6^2 + \frac{3^7}{(9^3 - 652)}$$

$$(2+7)^3 + \frac{273^{2/3}}{2} + \frac{55^2}{3}$$

$$2^3 + 7^3 + \frac{273^3}{2} + 55^{3/2}$$

$$\frac{3^7 \log(76)}{7^3 + 546} + \sqrt[3]{910}$$

$$43 \cdot \frac{\left(\sqrt[4]{250} + 23\right)^2}{e^{(45 - 3^3)}}$$

$$\cos^2\left(\frac{5\pi}{6}\right) \operatorname{sen}\left(\frac{7\pi}{8}\right)^2 + \frac{\tan\left(\frac{\pi}{6}\ln 8\right)}{\sqrt{7}}$$

$$\cos\left(\frac{5\pi}{6}\right)^2 \operatorname{sen}^2\left(\frac{7\pi}{8}\right) + \frac{\tan\left(\frac{\pi}{6}\ln 8\right)}{7 \cdot \frac{5}{2}}$$