PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

Algoritmia
Laboratorio 5
2014-1

Aplicación de los TDA

1.- (14 puntos) Emplee una pila para eliminar la recursión de las siguientes rutinas:

a) (7 puntos)

```
function comb (n, m : integer): integer;
  {calcula (n m) suponiendo que 0 ≤ m ≤ n y n ≥ 1}
  begin
     if (n = 1) or (m = 0) or (m = n) then
        return (1)
   else
      return (comb (n − 1, m) + comb(n − 1, m − 1))
end; {comb}
```

b) (7 puntos)

```
Algoritmo Torres de Hanoi
Rutina Principal
Inicio
     Anillos←Pide la cantidad de anillos a mover
     Mueve(Anillos, origen, intermedio, destino)
Fin

Rutina Mueve (a, o, i, d)
Inicio
     Si a = 1
          Entonces Muestra o "--->" d
          En otro caso Mueve(a-1, o, d, i)
                       Muestra o "--->" d
                       Mueve(a-1, i, o, d)
     Fin Si
Fin
```

2.- (6 puntos)

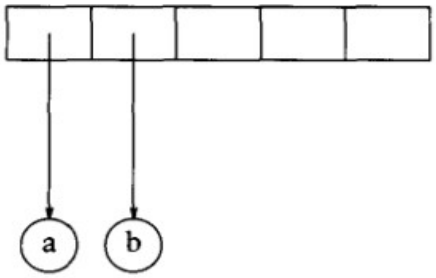
Constructing an Expression Tree

We now give an algorithm to convert a postfix expression into an expression tree. Since we already have an algorithm to convert infix to postfix, we can generate expression trees from the two common types of input. The method we describe strongly resembles the postfix evaluation algorithm of Section 3.2.3. We read our expression one symbol at a time. If the symbol is an operand, we create a one-node tree and push a pointer to it onto a stack. If the symbol is an operator, we pop pointers to two trees T_1 and T_2 from the stack (T_1 is popped first) and form a new tree whose root is the operator and whose left and right children point to T_2 and T_1 respectively. A pointer to this new tree is then pushed onto the stack.

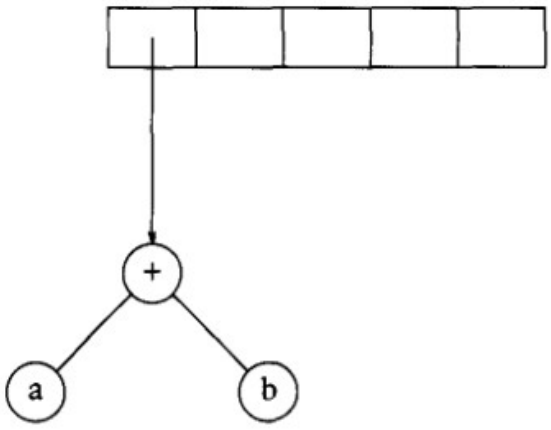
As an example, suppose the input is

a b + c d e + * *

The first two symbols are operands, so we create one-node trees and push pointers to them onto a stack.[†]

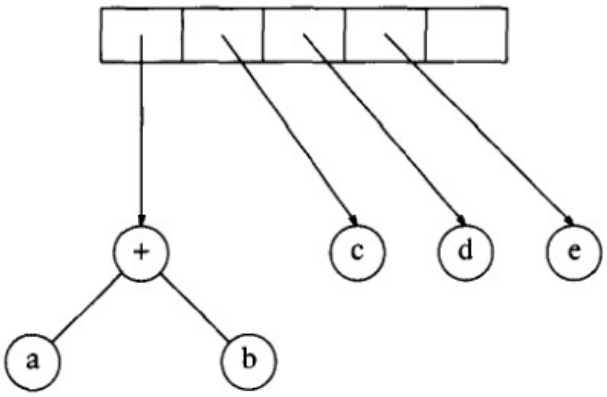


Next, a '+' is read, so two pointers to trees are popped, a new tree is formed, and a pointer to it is pushed onto the stack.

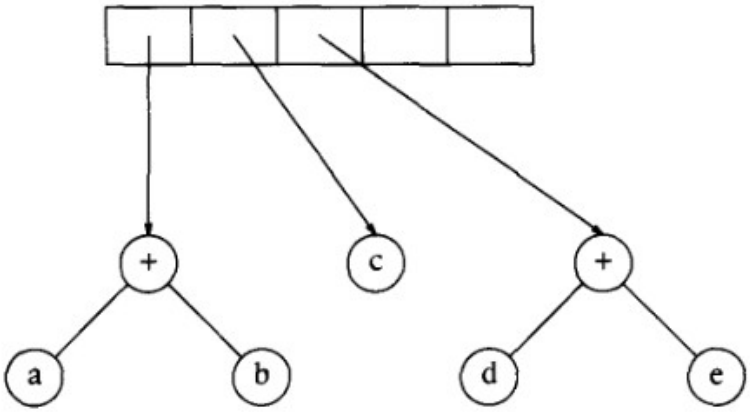


[†]For convenience, we will have the stack grow from left to right in the diagrams.

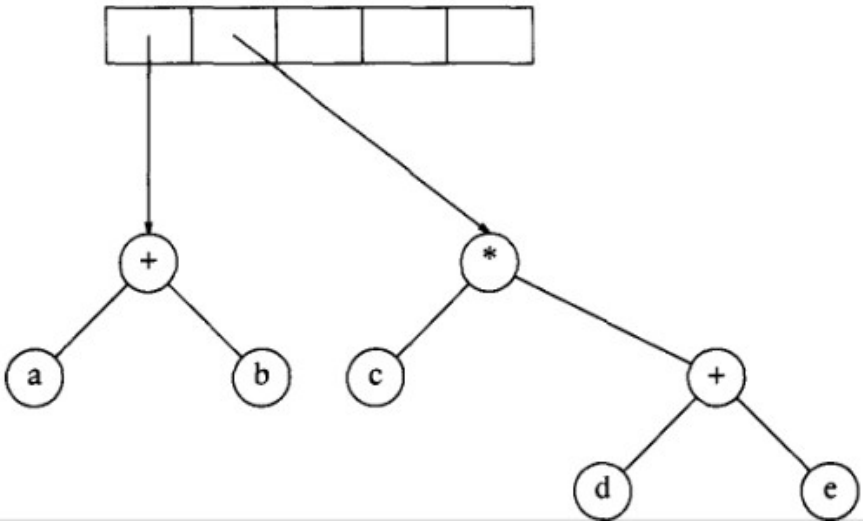
Next, c, d, and e are read, and for each a one-node tree is created and a pointer to the corresponding tree is pushed onto the stack.



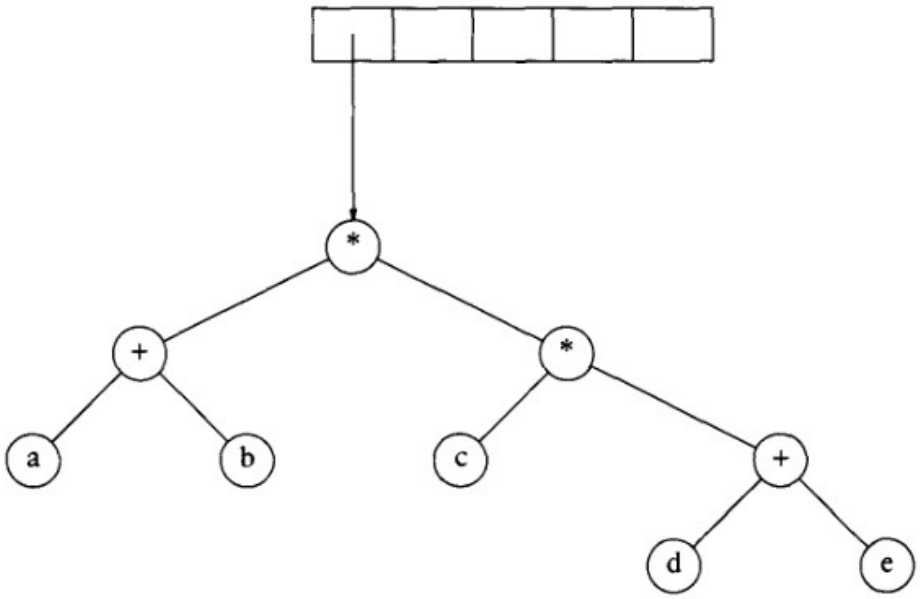
Now a '+' is read, so two trees are merged.



Continuing, a '*' is read, so we pop two tree pointers and form a new tree with a '*' as root.



Finally, the last symbol is read, two trees are merged, and a pointer to the final tree is left on the stack.



Implemente este algoritmo, siguiendo las indicaciones dadas por el autor.

Pando, 10 de junio del 2014

Prof. Alejandro T. Bello Ruiz