

Infix - Postfix

Infix Expression :

Any expression in the standard form like " $2*3-4/5$ " is an Infix(Inorder) expression.

Postfix Expression :

The Postfix(Postorder) form of the above expression is " $23*45/-$ ".

Infix to Postfix Conversion :

In normal algebra we use the infix notation like $a+b*c$. The corresponding postfix notation is $abc*+$. The algorithm for the conversion is as follows :

- Scan the Infix string from left to right.
- Initialise an empty stack.
- If the scanned character is an operand, add it to the Postfix string. If the scanned character is an operator and if the stack is empty Push the character to stack.
 - If the scanned character is an Operand and the stack is not empty, compare the precedence of the character with the element on top of the stack (topStack). If topStack has higher precedence over the scanned character Pop the stack else Push the scanned character to stack. Repeat this step as long as stack is not empty and topStack has precedence over the character.
- Repeat this step till all the characters are scanned.
- (After all characters are scanned, we have to add any character that the stack may have to the Postfix string.) If stack is not empty add topStack to Postfix string and Pop the stack. Repeat this step as long as stack is not empty.
- Return the Postfix string.

Example :

Let us see how the above algorithm will be implemented using an example.

Infix String : $a+b*c-d$

Initially the Stack is empty and our Postfix string has no characters. Now, the first character scanned is 'a'. 'a' is added to the Postfix string. The next character scanned is '+'. It being an operator, it is pushed to the stack.

Postfix String

Stack

Next character scanned is 'b' which will be placed in the Postfix string. Next character is '*' which is an operator. Now, the top element of the stack is '+' which has lower precedence than '*', so '*' will be pushed to the stack.

Postfix String

Stack

The next character is 'c' which is placed in the Postfix string. Next character scanned is '-'. The topmost character in the stack is '*' which has a higher

precedence than '-'. Thus '*' will be popped out from the stack and added to the Postfix string. Even now the stack is not empty. Now the topmost element of the stack is '+' which has equal priority to '-'. So pop the '+' from the stack and add it to the Postfix string. The '-' will be pushed to the stack.

Postfix String

Stack

Next character is 'd' which is added to Postfix string. Now all characters have been scanned so we must pop the remaining elements from the stack and add it to the Postfix string. At this stage we have only a '-' in the stack. It is popped out and added to the Postfix string. So, after all characters are scanned, this is how the stack and Postfix string will be :

Postfix String

Stack

End result :

- Infix String : a+b*c-d
- Postfix String : abc*+d-

© 2002 [Premshree Pillai](#).