**Thuật toán**

Giả sử M là một biểu thức được cho ở dạng trung tố. Khởi tạo 2 Stack: Sh và St.

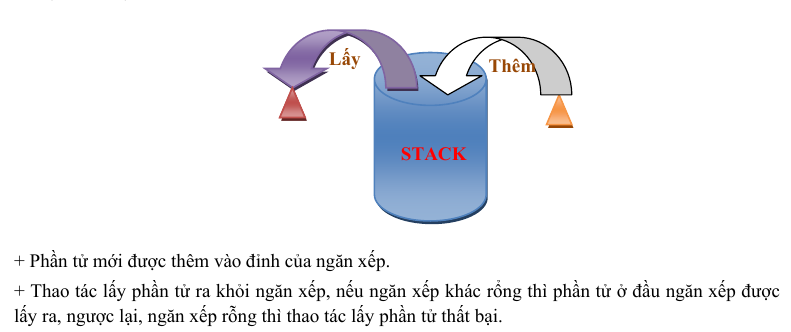
Stack Sh dùng để lưu trữ toán hạng, stack St dùng để lưu trữ toán tử.

Duyệt M từ trái qua phải:

* Nếu M[i] là toán hạng thì đưa vô Sh.
* Nếu M[i] là “(“ thì đưa vô St.
* Nếu M[i] là toán tử có độ ưu tiên cao hơn toán tử hiện có trên đỉnh St thì đưa nó vào St.
* Nếu M[i] là toán tử có độ ưu tiên thấp hơn hoặc bằng toán tử hiện có trên đỉnh St thì lấy 2 toán tử trên đỉnh Sh thực hiện phép tính với toán tử là phần tử trên đỉnh của St, kết quả cất vào Sh. Sau đó cất M[i] vào St.
* Nếu M[i] là dấu “)” , nếu phần tử trên đỉnh St khác “(“ thì thực hiện phép tính với toán tử là phần tử trên đỉnh của St, kết quả cất vào Sh. Loại dấu ngoặc “(“ gặp phải đầu tiên ra khỏi St.
* Thực hiện đến khi nào St rỗng và Sh còn 1 phần tử duy nhất đó là kết quả.

## Stack và ứng dụng của Stack

Stack là một danh sách có thứ tự mà phép chèn và xóa được thực hiện tại đầu cuối của danh sách và người ta gọi đầu cuối này là đỉnh (top) của stack.



**Ứng dụng của Stack**  
Stack có rất nhiều ứng dụng trong tin học như :  
– Chuyển đổi các cơ số (nhị phân, thập phân, bát phân,…)  
– Chuyển biểu thức trung tố sang hậu tố, tính toán các biểu thức hậu tố,…

## Danh Sách Liên Kết Đơn

## Danh sách liên kết là 1 cấu trúc dữ liệu cơ bản, được sử dụng để khắc phục hạn chế của mảng (cố định về kích thước). C++ nói chung và cụ thể là thư viện STL đã cung cấp sẵn một kiểu dữ liệu List. Tuy nhiên tôi vẫn muốn chia sẻ bài viết này để nêu rõ về bản chất của danh sách liên kết và một số thao tác cơ bản trên nó.

Cũng giống như mảng, danh sách liên kết cũng bao gồm các phần tử, có mối liên hệ với nhau. Tôi gọi mỗi phần tử đó là một Node. Node được xem là trái tim của danh sách liên kết, mỗi Node sẽ lưu trữ 2 thông tin:

* Thông tin dữ liệu: Lưu trữ các thông tìn về chính Node đó.
* Thông tin liên kết: Lưu trữ địa chỉ của phần tử kế tiếp trong danh sách, hoặc lưu trữ giá trị NULL nếu phần tử đó nằm cuối danh sách.

