



## BÀI TẬP THỰC HÀNH KHÓA HỌC CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT VỚI JAVA BÀI 3.2

**Bài 1.** Tạo stack generic sau đó dùng stack này để lưu trữ các đối tượng kiểu sinh viên. Trong đó thông tin sinh viên gồm: mã sinh viên, họ, đệm và tên, địa chỉ, email, tuổi, điểm trung bình. Thực hiện các chức năng sau:

- 1) Thêm mới sinh viên vào stack.
- 2) Hiển thị thông tin sinh viên đầu stack theo dạng bảng gồm các hàng, cột.
- 3) Kiểm tra xem stack hiện chứa bao nhiêu sinh viên.
- 4) Hiển thị các sinh viên có trong stack ra màn hình dưới dạng bảng gồm các hàng, các cột. Mỗi sinh viên hiển thị trên 1 dòng.
- 5) Thoát chương trình.

**Bài 2.** Viết chương trình đảo ngược các từ trong chuỗi kí tự sử dụng stack. Ví dụ nhập vào chuỗi `str = "Welcome to Branium Academy!"` thì kết quả là `"Academy! Branium to Welcome"`.

**Bài 3.** Viết chương trình chuyển biểu thức dạng trung tố sang dạng hậu tố. Giả định các phép toán cần thực hiện trong biểu thức là `+`, `-`, `*`, `/`, `^`. Biểu thức có thể chứa dấu ngoặc để gom nhóm ưu tiên. Thứ tự ưu tiên của các phép toán là lũy thừa = 3, nhân = chia = 2, cộng = trừ = 1.

Gọi thành phần cấu thành nên biểu thức là các phần tử. Nó có thể là các con số, dấu ngoặc `(, )` hoặc các phép toán. Bỏ qua dấu cách. Quy trình thực hiện chuyển đổi trung tố sang hậu tố:

- Lần lượt lấy từng phần tử `e` trong biểu thức:
  - o Nếu `e` là toán hạng, hiển thị ra màn hình.
  - o Nếu `e` là dấu ngoặc `(` thì thêm nó vào stack.
  - o Nếu `e` là dấu ngoặc `)`, thì pop các phần tử trong stack ra tới khi gặp ngoặc `(`. Sau đó pop bỏ ngoặc `(`.
  - o Nếu `e` là phép toán và stack không rỗng, trong khi thứ tự ưu tiên của `e`  $\leq$  thứ tự ưu tiên của phần tử đầu stack, pop phần tử đầu stack cho hiển thị ra màn hình. Sau đó push `e` vào stack.
- Trong khi stack còn chưa rỗng, pop các phần tử còn lại hiển thị ra màn hình.
- Input:
  - o Dòng đầu là số bộ test  $0 < t \leq 10$ .
  - o Các dòng sau mỗi dòng là một biểu thức trung tố cần chuyển đổi.
- Output: mỗi kết quả ghi trên một dòng biểu thức sau khi chuyển từ trung tố sang hậu tố.

Ví dụ

INPUT	OUTPUT
3	
10 * 25	10 25 *
100 * 20 + 6 - 2	100 20 * 6 + 2 -
20^5 / (5 * 90) + 7	20 5 ^ 5 90 * / 7 +



**Bài 4.** Viết chương trình tính giá trị biểu thức dạng hậu tố. Bỏ qua các khoảng trắng phân tách từng phần tử. Thuật toán được thực hiện như sau:

Lần lượt lấy từng giá trị  $e$  trong biểu thức hậu tố:

- Nếu  $e$  là một toán tử  $t$ :
  - o Pop phần tử đầu stack gán cho biến  $b$ .
  - o Pop phần tử đầu stack tiếp theo gán cho biến  $a$ .
  - o Tính kết quả  $a \ t \ b$  sau đó push kết quả vào stack.
- Nếu  $e$  là một toán hạng, push  $e$  vào stack.

Sau khi thực hiện xong, phần tử còn lại trong stack là kết quả của biểu thức.

Ví dụ tính giá trị của biểu thức  $100 \ 20 \ * \ 6 \ + \ 2 \ -$  ta xét lần lượt từng phần tử  $e$  của biểu thức:

- Khi  $e = '100'$ , đây là một toán hạng, push 100 vào stack. Stack hiện tại: [100].
- Khi  $e = '20'$ , đây là toán hạng, push 20 vào stack. Stack hiện tại: [100, 20] (20 là phần tử top).
- Khi  $e = '*'$ , đây là một toán tử,  $b = 20$ ,  $a = 100$ ,  $a \ e \ b = 100 \ * \ 20 = 2000$ . Push 2000 vào stack. Stack hiện tại: [2000].
- Khi  $e = '6'$ , đây là toán hạng, push vào stack. Stack hiện tại: [2000, 6].
- Khi  $e = '+'$ , đây là toán tử,  $b = 6$ ,  $a = 2000$ ,  $a \ + \ b = 2006$ . Push vào stack. Stack hiện tại: [2006].
- Khi  $e = '2'$ , đây là toán hạng, push vào stack. Stack hiện tại: [2006, 2].
- Khi  $e = '-'$ , đây là toán tử, pop các phần tử trong stack:  $a = 2006$ ,  $b = 2$ . Tính  $a - b = 2006 - 2 = 2004$ . Push 2004 vào stack. Stack hiện tại: 2004.
- Kết thúc biểu thức, phần tử còn lại trong stack là kết quả: 2004.

Input:

- Dòng đầu là số bộ test  $0 < t \leq 10$ .
- $t$  dòng sau mỗi dòng là một biểu thức hậu tố.

Output: ghi ra trên  $t$  dòng, mỗi dòng là kết quả của bộ test tương ứng.

Ví dụ:

INPUT	OUTPUT
3	
10 25 *	250
100 20 * 6 + 2 -	2004
5 3 + 6 2 * 3 5 * ++	35

Trang chủ: <https://braniumacademy.net>

Bài giải mẫu: [click vào đây](#).