

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

──────── \* ───────

BÁO CÁO CUỐI KỲ

MÔN: THỰC HÀNH KIẾN TRÚC MÁY TÍNH

Giảng viên hướng dẫn: Thầy Phạm Ngọc Hưng

Sinh viên thực hiện: Lê Thị Nhung - 20210662

Lớp: Thực hành kiến trúc máy tính - IT3280 - 139365

Hà Nội, tháng 7 năm 2023

## Bài 5: Biểu thức trung tố hậu tố

Viết chương trình tính giá trị biểu thức bất kỳ bằng phương pháp duyệt biểu thức hậu tố.

Các yêu cầu cụ thể:

1. Nhập vào biểu thức trung tố, ví dụ:

9 + 2 + 8 \* 6

1. In ra biểu thức ở dạng hậu tố, ví dụ: 9 2 + 8 6 \* + 3.Tính ra giá trị của biểu thức vừa nhập

Các hằng số là số nguyên, trong phạm vi từ 0 →99.

Toán tử bao gồm các phép toán cộng, trừ, nhân, chia lấy thương(/), chia lấy dư (%), đóng mở ngoặc.

**Phân tích cách làm và thuật toán:**

\*\* Để kiểm tra tính hợp lệ của biểu thức trung tố nhập vào:

+ Kiểm xem có đủ (): duyệt biểu thức nhập vào bắt đầu từ trái -> phải. Nếu gặp ‘(‘ ta push vào stack. Nếu gặp ‘)’, kiểm tra nếu stack rỗng => không đủ cặp (), nếu chưa rỗng ta pop ‘)’.

+Kiểm tra xem bthuc có bắt đầu bằng 1 toán tử hay (, cặp () rỗng không có giá trị bên trong, nhập phép chia /0, toán hạng lớn hơn 99, hai toán hạng cạnh nhau,.. và một số trường hợp không hợp lệ khác.

\*\* Để chuyển từ biểu thức trung tố sang biểu thức hậu tố, ta đọc lần lượt từng ký tự trong biểu thức trung tố, giả sử ký tự đọc được là c:

+ Nếu c là số -> thêm vào biểu thức hậu tố

+ Nếu c là toán tử -> thực hiện thêm toán tử vào stack. Nếu stack rỗng, thêm c vào stack, nếu stack không rỗng, so sánh thứ tự ưu tiên của c với toán tử ở đỉnh stack, nếu độ ưu tiên của c <= độ ưu tiên của toán tử ở đỉnh stack, lấy toán tử ở đỉnh ra và thêm vào biểu thức hậu tố, lặp lại việc so sánh trên cho đến khi độ ưu tiên của c > độ ưu tiên của toán tử ở đỉnh stack, sau đó thêm c vào stack.

+ Nếu c là toán tử ( -> thêm luôn c vào stack.

+ Nếu c là toán tử ) -> lấy lần lượt các toán tử trong stack ra và thêm vào biểu thức hậu tố cho đến khi gặp toán tử ( . Lưu ý, không thêm toán tử ) vào trong biểu thức hậu tố.

+ Khi duyệt hết biểu thức trung tố, nếu còn toán tự trong stack, thực hiện lấy lần lượt các toán tử ra và thêm vào biểu thức trung tố cho đến khi stack rỗng.

Vì các toán hạng có giá trị từ 0 -> 99 nên khi thêm vào biểu thức hậu tố, cần thêm các dấu cách để ngăn cách giữa các phần tử.

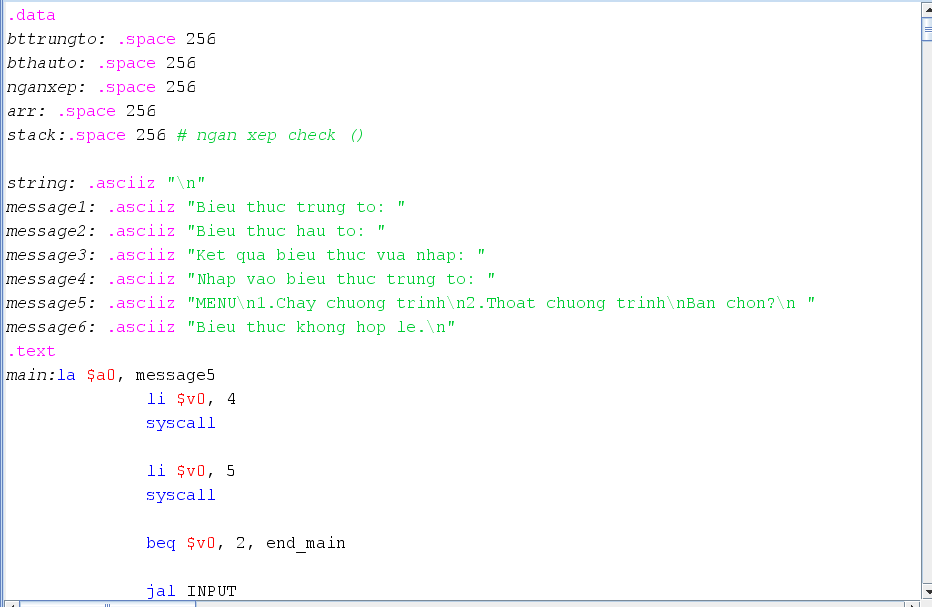
\*\*Để tính giá trị biểu thức hậu tố:

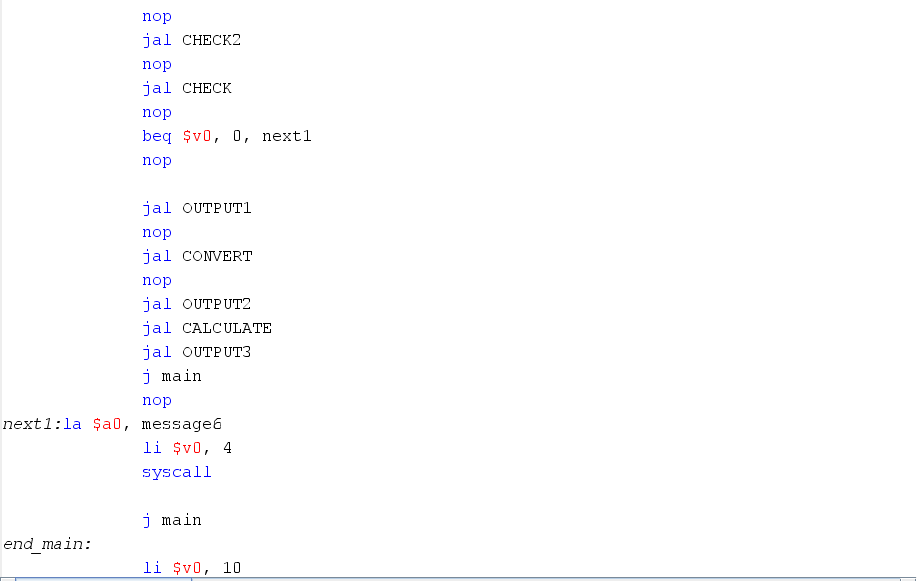
+ Thực hiện duyệt biểu thức hậu tố, nếu gặp toán hạng thì thêm toán hạng vào stack, nếu gặp toán tử, lấy hai toán hạng trong stack ra và tính toán phép tính theo toán tử tương ứng, kết quả của phép tính được lưu trở lại vào trong stack. Lặp lại cho đến khi duyệt hết biểu thức hậu tố, kết quả cuối cùng được lưu trong stack.

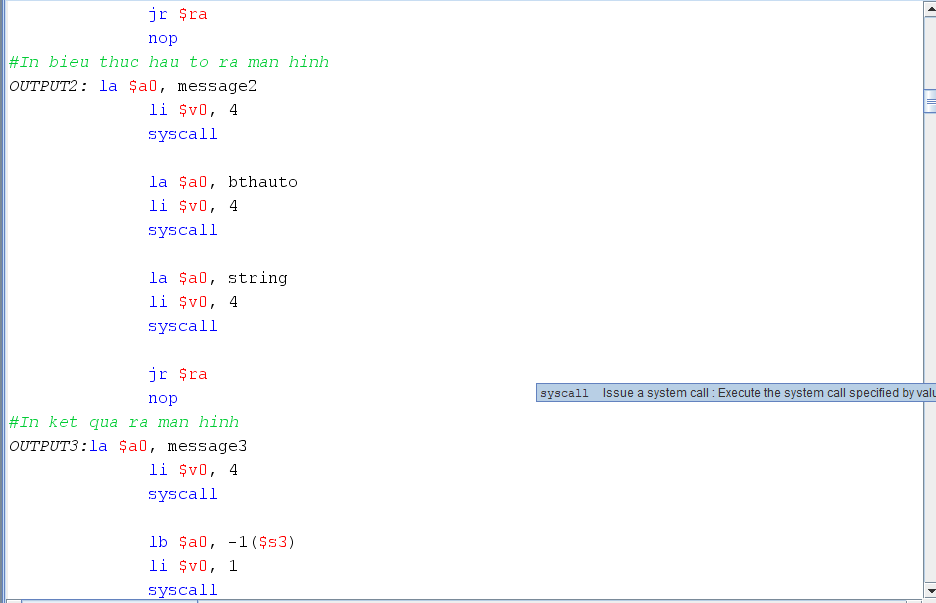
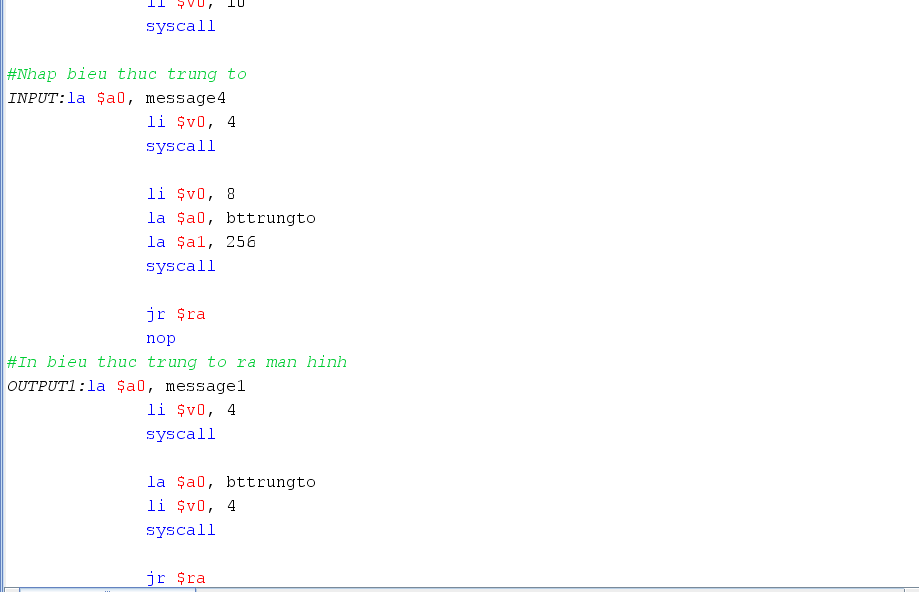
Độ ưu tiên của các toán tử:

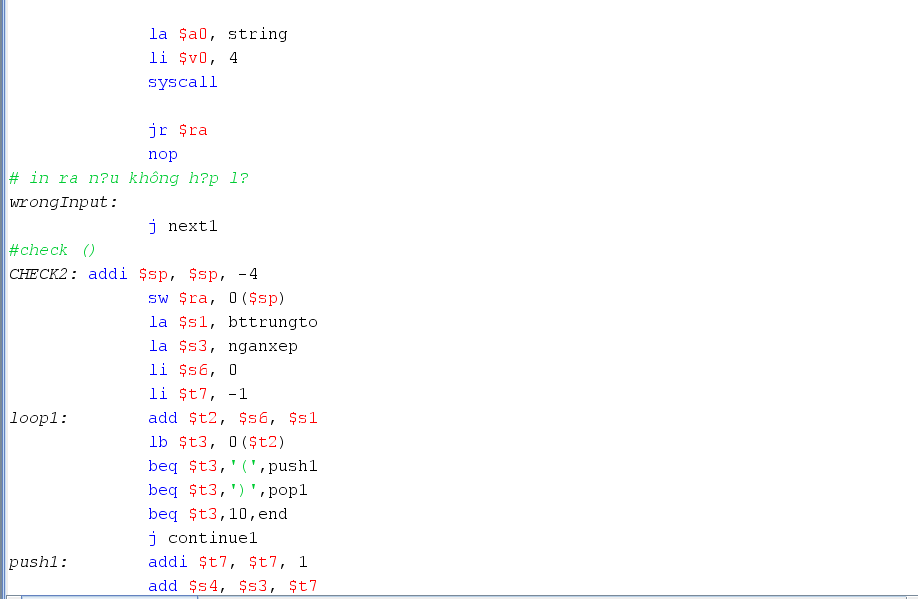
|  |  |
| --- | --- |
| Toán tử | Độ ưu tiên |
| ( | 0 |
| + | 1 |
| - | 1 |
| \* | 2 |
| / | 2 |
| % | 2 |
| ) | 3 |

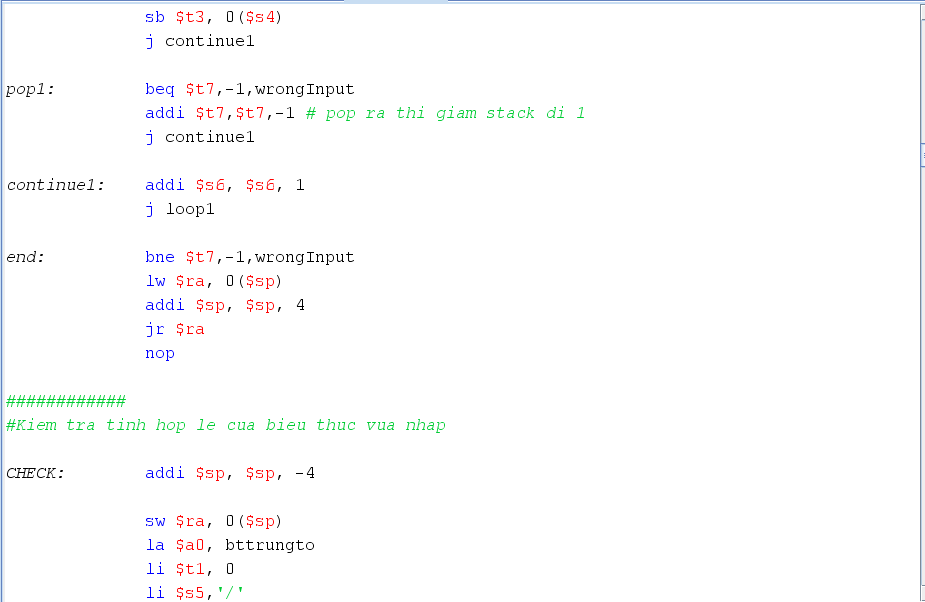
### Mã nguồn

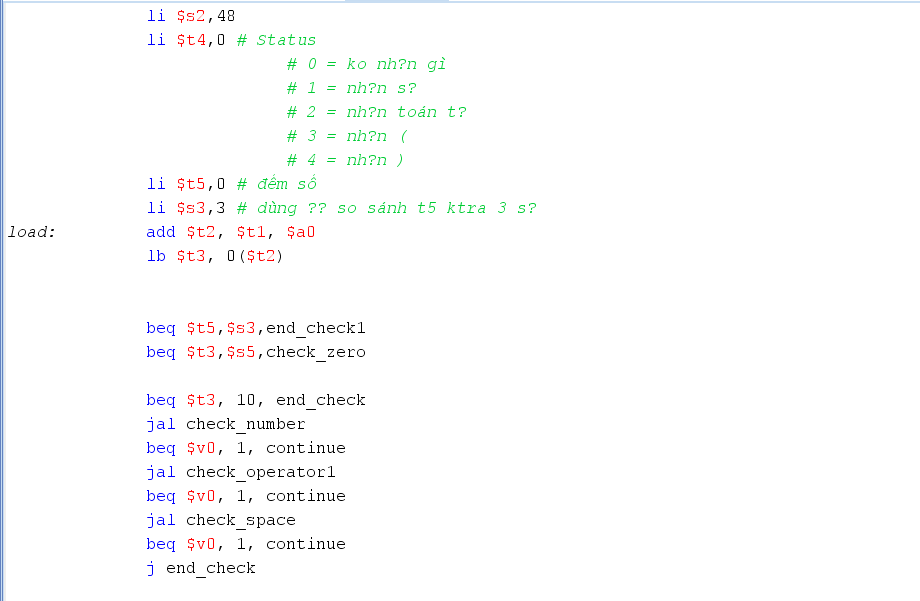


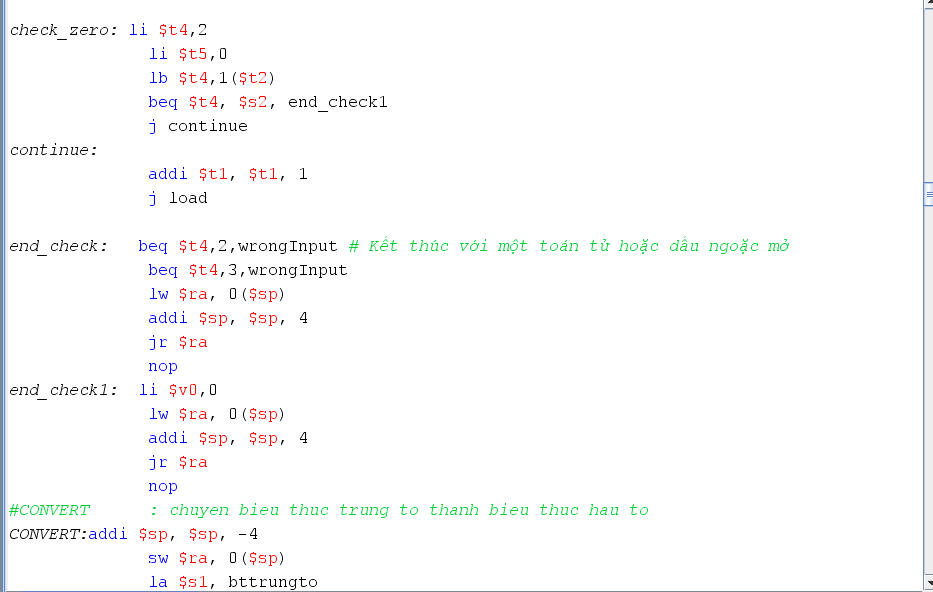


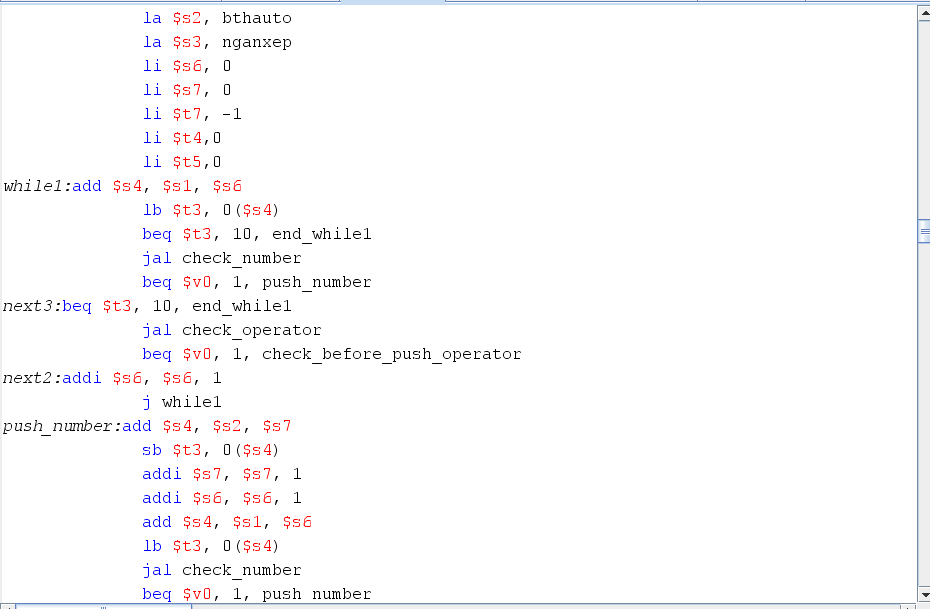


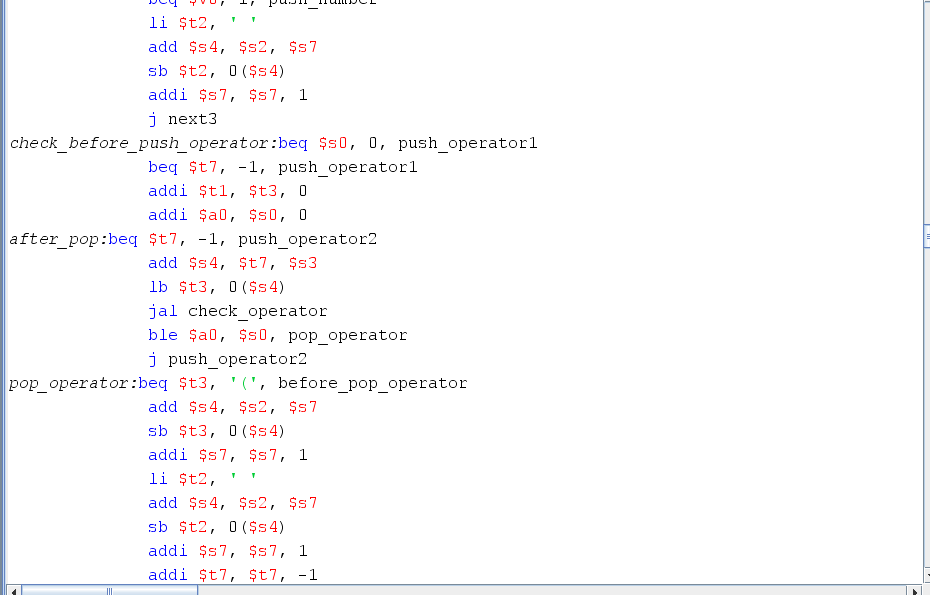


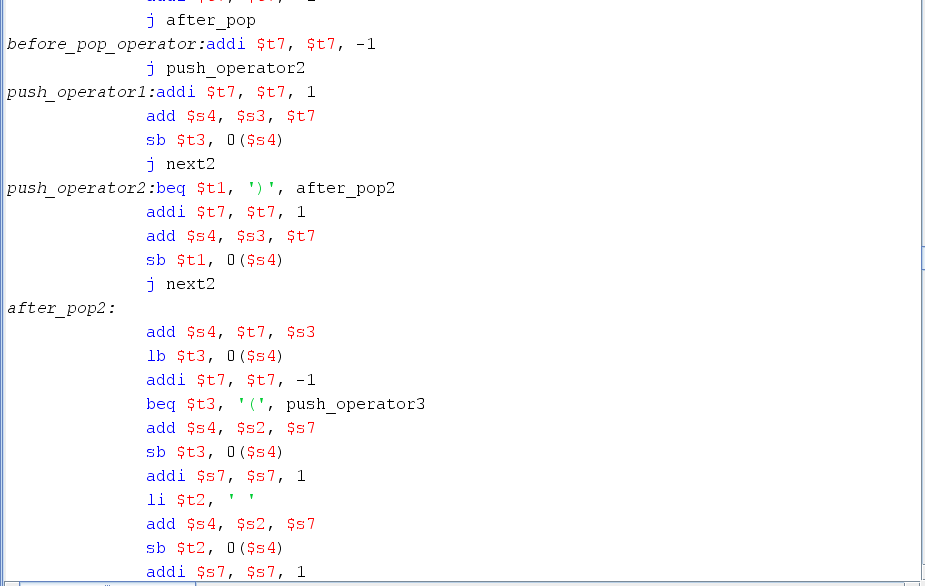


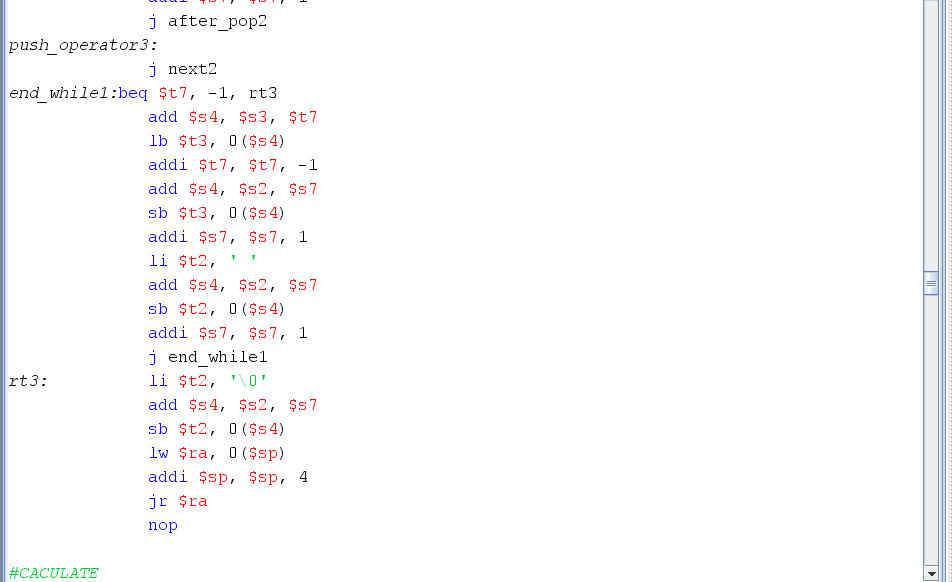


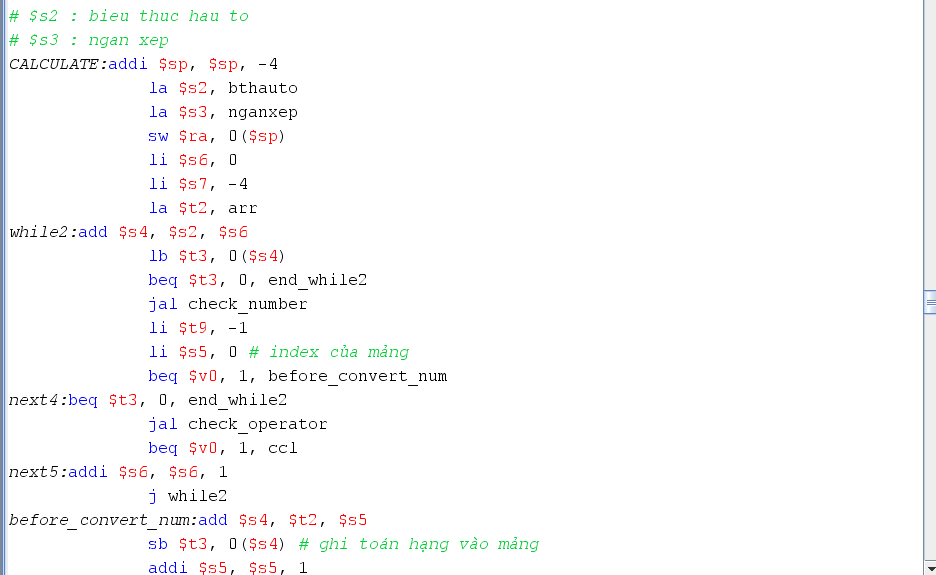


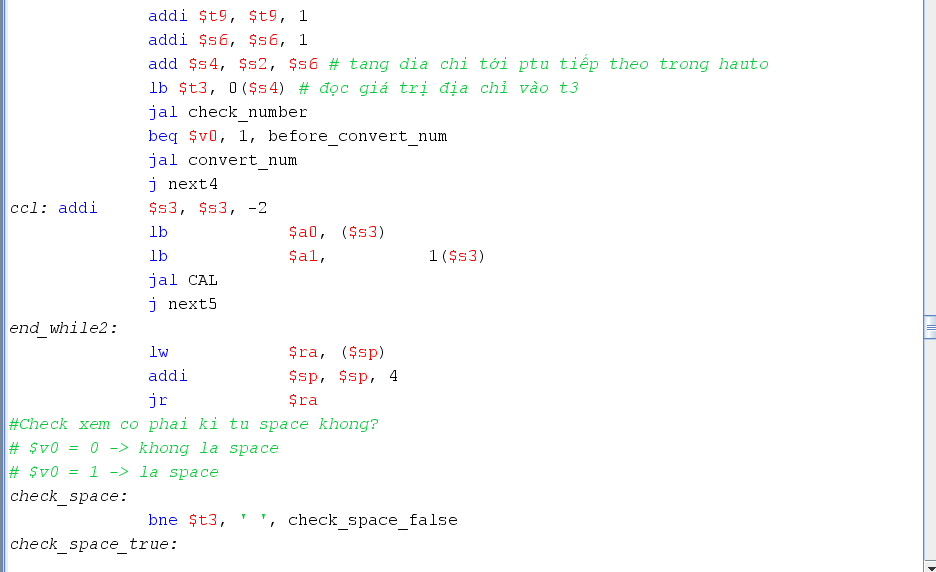


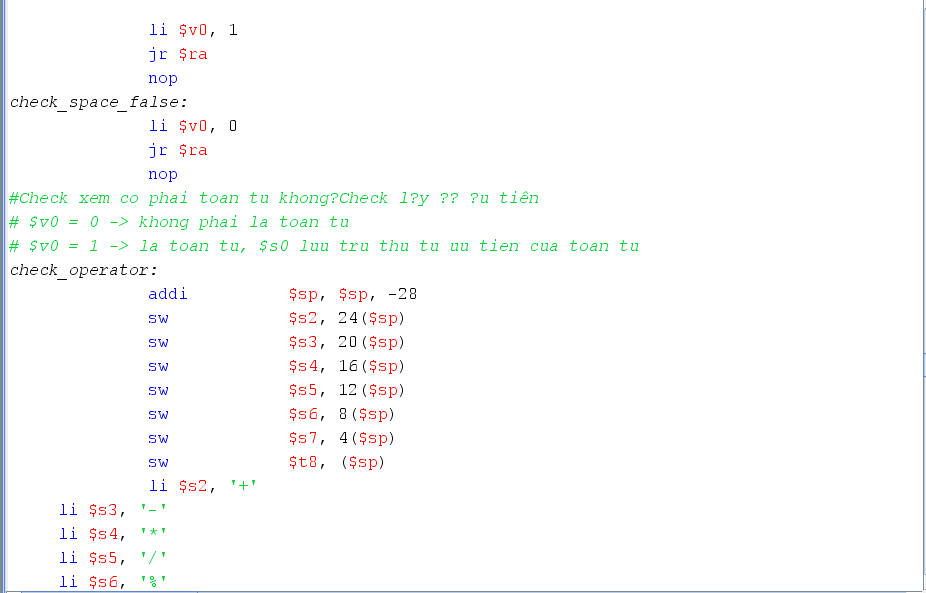


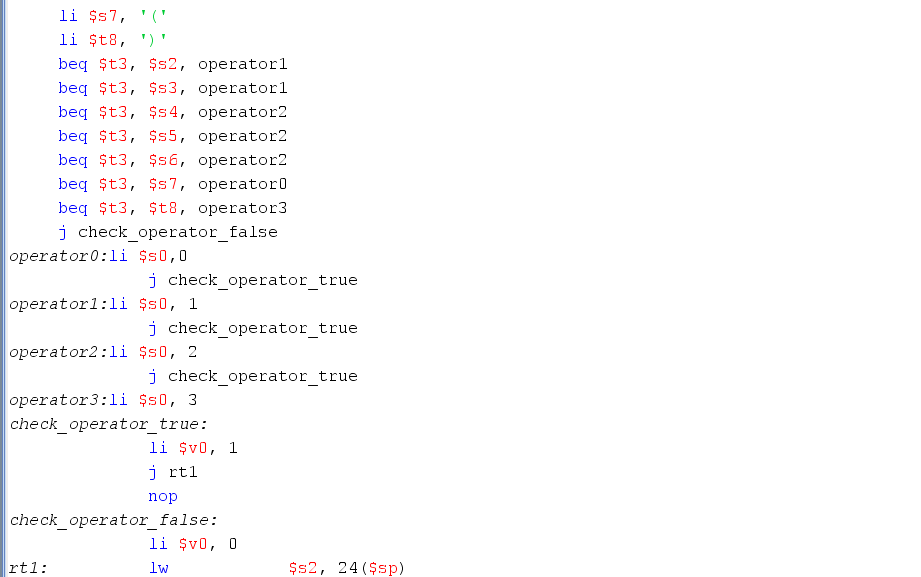


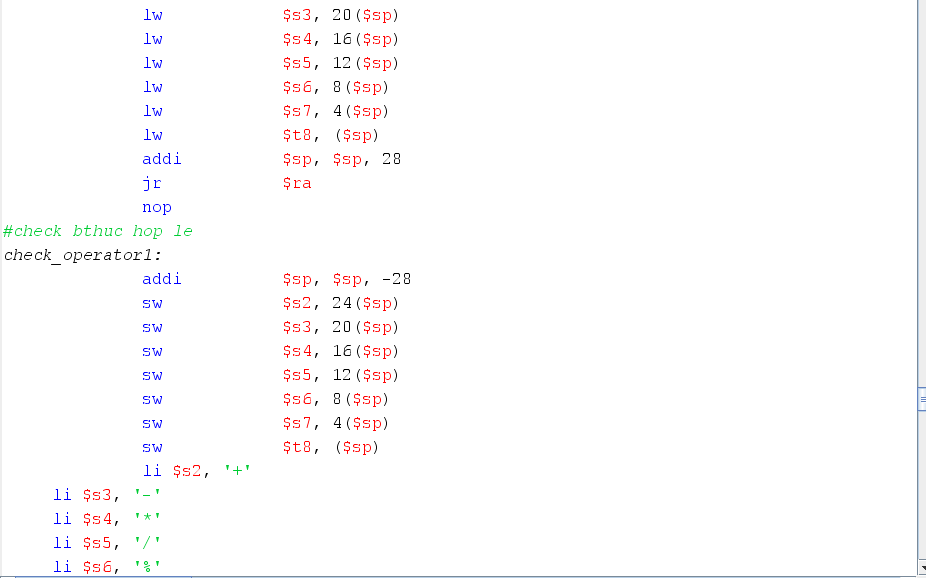


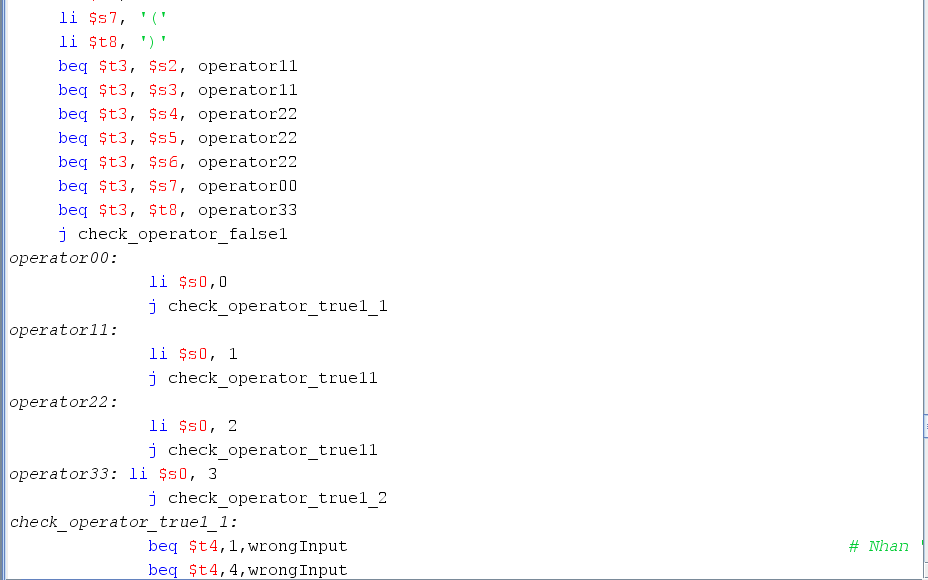


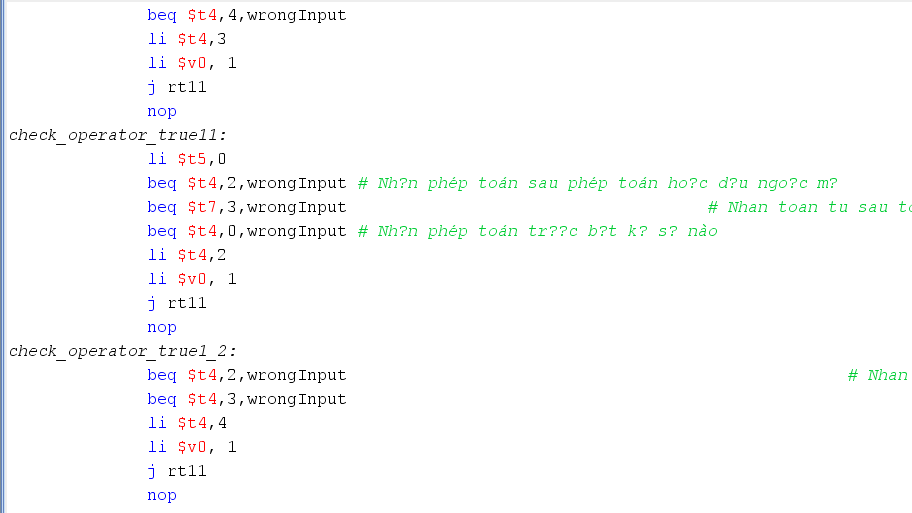


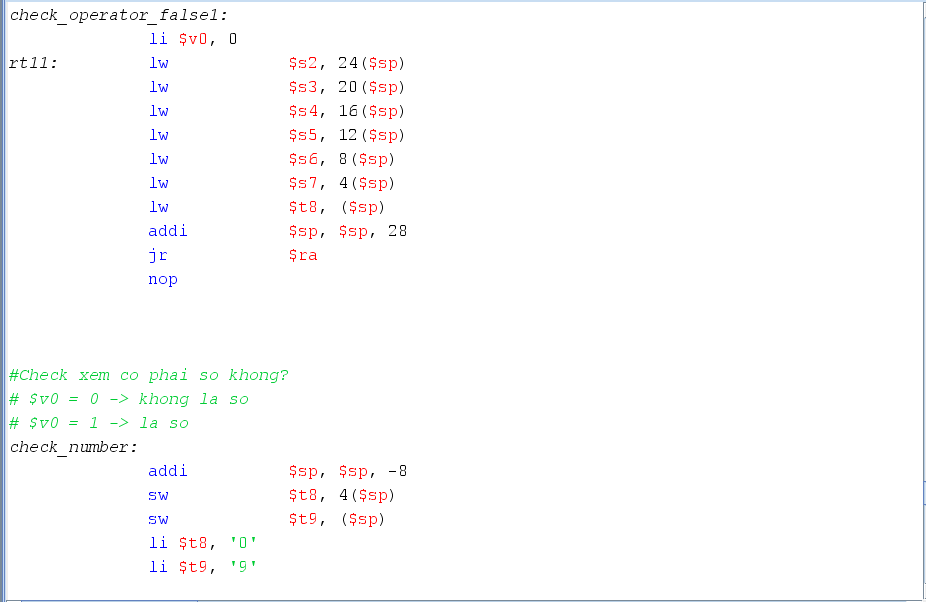


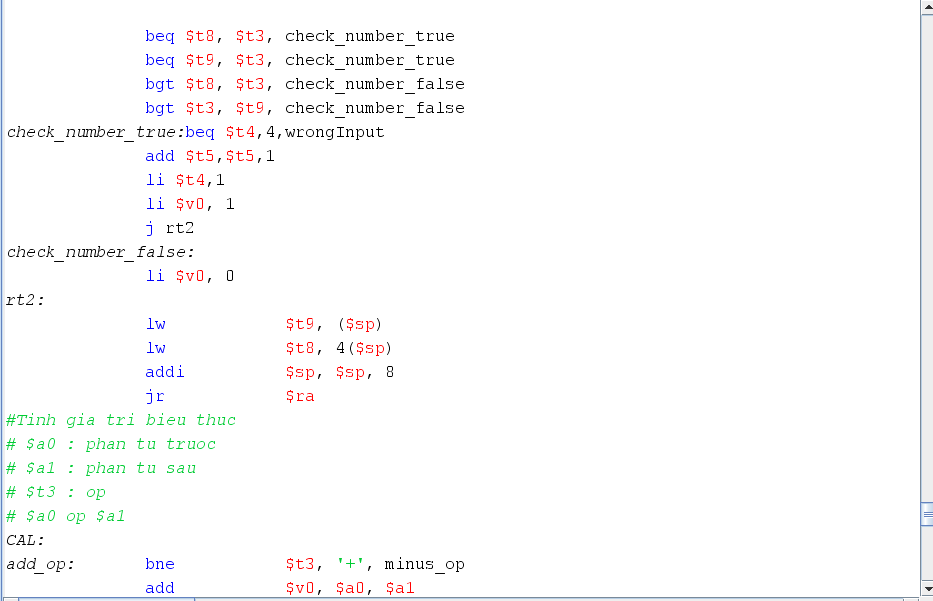


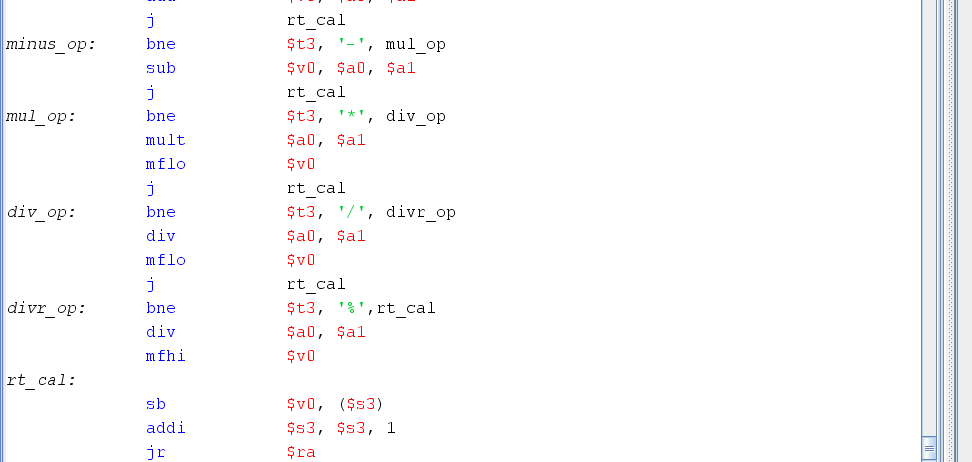


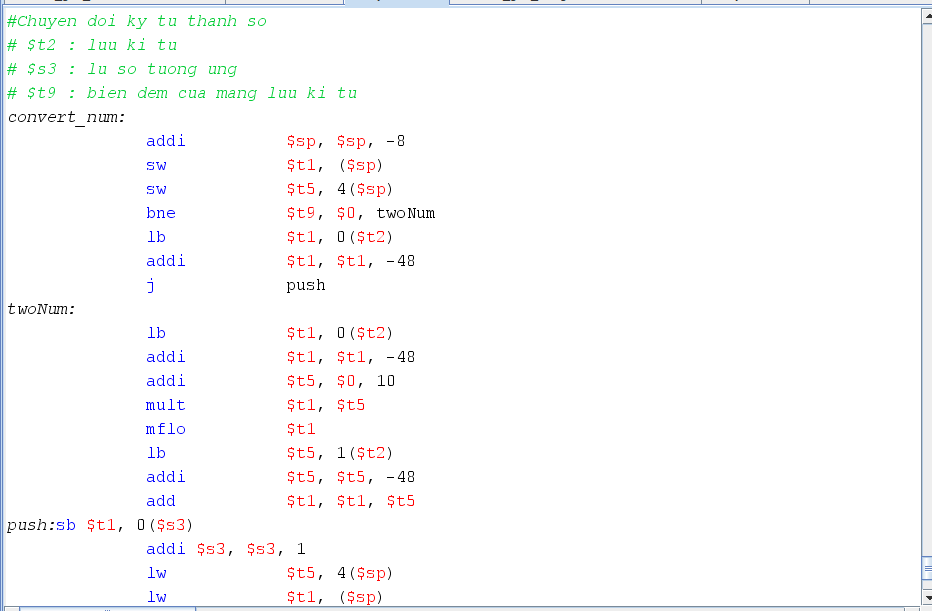


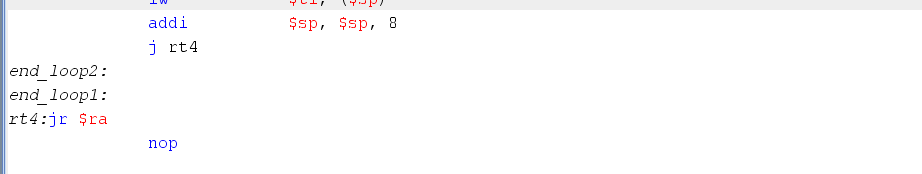






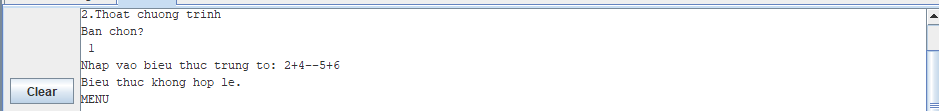


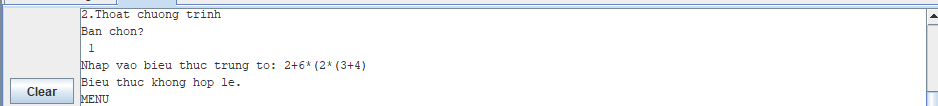




**Kết quả chạy mô phỏng:**

Một số trường hợp không hợp lệ:

****

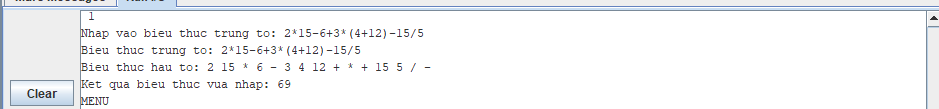
****

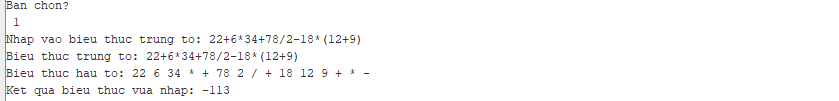
****

**…..**

Chuyển đổi và tính toán biểu thức

****

****

****

Thoát chương trình :

****

Giải thích về chương trình :

Check2 : dùng để kiểm tra xem có đủ cặp () trong biểu thức vừa nhập không với vòng lặp loop1 để duyệt từng phần tử từ trái sang phải và hai hàm push để đẩy ‘(‘ vào và pop để kiểm tra xem stack có rỗng ra không rồi pop ‘(‘ khi gặp ‘)’.

Check: lưu địa chỉ trả về ra vào stack tại địa chỉ con trỏ stack hiện tại.

gán địa chỉ chuỗi bttrungto cho thanh ghi a0. Và thanh ghi $t4 lưu trạng thái của kí tự được nhập trước đó.$t5 để đếm số chữ số

Load:

Lưu trữ giá trị đầu tiên của chuỗi bttrungto vào $t3.

Nếu t5= 3 – số có 3 chữ số hoặc 2 số cạnh nhau=> bthuc không hợp lệ.

Kiếm tra nếu là ‘/’ thì chuyển tới check\_zero xem ktu sau có là 0 hay không

Nếu $t3 là kí tự Null thì chuyển đến end\_check.

Nếu không chuyển tới check\_number: kiểm tra xem kí tự đó có là số không. Trả về v0= 1 nếu là số, 0 nếu không là số.

Nếu $t3 là số, chuyển tới kí tự tiếp theo để kiểm tra. Nếu không thì tới kiểm tra toán tử. Nếu kí tự là toán tử thì tiếp tục . Không thì kiểm tra tới dấu cách. Nếu là dấu cách thì bỏ qua chuyển tới continue để kiểm tra phần từ tiếp theo

CONVERT: Hàm đổi biểu thức trung tố thành biểu thức hậu tố.

Addi $sp, -4: dịch chuyển con trỏ ngăn xếp xuống 4 byte để cấp phát không gian lưu trữ cho biến cục bộ.

Sw: lưu giá trị trở về vào đỉnh ngắn xếp để tránh mất dữ liệu.

$s1: lấy giá trị địa chỉ của mảng bttrungto

$s2: lấy giá trị địa chỉ của mảng bthauto

$s3:lấy giá trị địa chỉ của mảng nganxep

$s6: biến đếm cho vòng lặp

$s7: biến đếm cho mảng

$t7: biến đếm cho ngăn xếp

WHILE : dùng để duyệt qua các ký tự trong bttrungto

Tính toán địa chỉ của ptu thứ $s6 trong BTTT luu vào $s4.

Lưu trữ giá trị địa chỉ s4 vào t3. Nếu t3 là ktu xuống dòng=> nhảy tới end\_ưhile1

Nhảy tới check\_number để kiểm tra t3 có là số không.

Nếu t3 là số ( v0=1) thì nhảy tới push\_number

Nếu không thì kiểm tra xem nó có là toán tử, nếu có chuyển tới CHECK\_BEEFORE\_PUSH\_OPERATPR.

PUSH\_NUMBER: đẩy số vào mảng bthauto

Tính toán địa chỉ thứ $s7 trong mang hto và lưu vào s4. Ghi giá trị của thanh ghi t3 vào địa chỉ s4 trong bthauto . Tăng s7 lên để trỏ tới phần tử tiếp theo trong bthauto. Tăng s6 lên để trỏ tới phần tử tiếp theo trong bttrungto.

Tính toán địa chỉ của phần tử tiếp theo trong bthuc trung tố và lưu vào s4.

Đọc giá trị s4 và lưu vào thanh ghi t3. Nhảy tới check\_number xem có là số xong. Nếu có thì nhảy tới push\_number. Gán ‘ ‘ vào t2. Tính toán địa chỉ tiếp theo trong mảng bthauto và lưu vào s4. Ghi giá trị khoảng trắng vào bthauto. Tăng s7 để trỏ tới ptu tiep theo hauto. Nhảy tới next 3 để kiểm tra phần tử tiếp theo trong trung tố.

NEXT3: kiểm tra nếu t3 là xuống dòng thì end thì tới endwhie1. Nhảy tới hàm CHECK\_operator.

Nếu t3 là toán tử thì nhảy tớ nhãn check\_before\_push\_operator.

(1) NEXt2: năng t6 để duyệt tới phần tử tiếp theo trong nttrungto. Nhảy tới while 1 để tiếp tục vòng lặp.

CHECK\_BEFORE\_PUSH\_OPERATOR: nhãn kiểm tra trước khi đẩy toán tử vào bthuc hauto.

Nếu s0 =0 thì chưa có toán tử trước đó, hoặc là dấu ( => nhảy tới nhãn push1.

Push\_operator 1: tăng t7 tới ptu tiếp theo trong ngăn xếp. tính toán địa chỉ trong ngắn xếp và lưu trữ giá trị tại vị trí thứ t7 vào t3. Nhảy tới next2 để kiểm tra phần tử tiếp theo trong trung tố.

Nếu t7=-1 tức là ngắn xếp rỗng. nhảy tới push\_operator1

Sao chép giá trị t3 vào t1 và sao chép s0 vào a0.

After\_pop: nhãn thực hiện thao tác khi loại bỏ toán tử từ ngắn xếp.

Nếu t7 bằng -1 : ngắn xếp rỗng, nhảy tới push\_operator2.

Push\_operator2: nếu $t1 là ( thì nhảy tới nhãn after\_pop2

After\_pop2: nhãn thực hiện thao tác sau khi loại bỏ toán tử từ ngăn xếp.

Tính toán địa chỉ thứ t7 và đọc giá trị tại địa chỉ đó lưu vào t3

Giảm t7 để trỏ tới phần tử trước trong ngắn xếp. nếu t3= ( thì nhảy tới Push\_operator3: tăng đến ptu tiếp trong bthuc trung to .

Tính toán phần tử tiếp theo trong bthauto và lưu vào s4. Ghi giá trị t3 vào dchi s4 trong bthauto. Thêm khoảng ‘ ‘ . tăng s7 để trỏ tới pthut tiếp trong bthuchauto.

Tiếp afterpop : Tính toán địa chỉ của ptu thứ t7 rồi lưu vào s4. Đọc giá trị tại s4 vào t3.

Nhảy tới hàm check\_operator xem t3 có là toán tử không.

So sánh nếu a0 <=s0 (so sánh độ ưu tiên toán tử hiện tại với toán tử đầu nganxep) thì nhảy tới pop\_operator. Nhảy tới nhãn push2 để thêm toán tử vào mảng hauto.

Pop\_operator: Lưu vào hậu tố, thêm ‘ ‘, rồi trỏ tới phần tử tiếp theo

BEFORE\_POP\_operator: giảm t7 để tro tới ptu trước trong ngăn xếp.

Nhảy tới push\_operator2 xử lí ( .

CACULATE: Thực hiện tính toán giá trị biểu thức của biểu thức hậu tố.

(S2 : bthuc hau to

S3: ngan xep)

Addi $sp, -4: dịch chuyển con trỏ ngăn xếp xuống 4 byte để cấp phát không gian lưu trữ cho biến cụ bộ.

Sw: lưu giá trị trở về vào đỉnh ngắn xếp để tránh mất dữ liệu.

Gán địa chỉ mảng arr vào thanh ghi t2.

While2: tính toán địa chỉ tiếp theo trong bthuc hau to lưu vào s4. Đọc giá trị tại s4 xem nó có là null 0. Nếu có thì end\_while2 . Nhảy tới kiểm tra có là số khoog. Nếu là số thì chuyển đổi ký tự thành số và lưu vào arr.

Nếu không phải số thì nhảy tới kiểm tra toán tử. nếu là toán tử thực hiện phép tính tuwowg ứng vào lưu vào arr.

Đọc giá trị từ arr lưu vào v0.

CAL:

Thực hiện phép tính giữa 2 ptu sdung toán tử t3.

COVERT\_NUM: Hàm chuyển đổi kí tự thành số rồi lưu vào mảng