**Bài thực hành số 7: Mảng và con trỏ**

Họ tên: Nguyễn Văn Hiếu-20225717

**Assignment 1:**

.data

*A: .space 400 # Cấp phát không gian cho 100 từ (mỗi từ 4 byte) để lưu trữ mảng*

*.text*

*main:*

*# Đọc số lượng phần tử*

*li $v0, 5 # Load mã syscall để đọc một số nguyên*

*syscall # Đọc số nguyên*

*move $a1, $v0 # Lưu số lượng phần tử vào $a1*

*# Nhập các phần tử của mảng*

*li $t1, 0 # Khởi tạo chỉ số cho vòng lặp*

*la $a0, A # Load địa chỉ cơ sở của mảng*

*input\_loop:*

*bge $t1, $a1, mspfx # Nếu chỉ số >= số lượng phần tử, thoát khỏi vòng lặp*

*li $v0, 5 # Load mã syscall để đọc một số nguyên*

*syscall # Đọc số nguyên*

*sw $v0, ($a0) # Lưu giá trị vào mảng tại vị trí hiện tại*

*addi $t1, $t1, 1 # Tăng chỉ số*

*addi $a0, $a0, 4 # Tăng địa chỉ lên 4 byte để chuyển đến phần tử tiếp theo của mảng*

*j input\_loop # Quay lại vòng lặp nhập*

*nop*

*mspfx:*

*# Đoạn mã tính toán tổng dãy con lớn nhất sẽ được thêm vào sau đây*

*# Khởi tạo các biến*

*la $a0, A # Load địa chỉ cơ sở của mảng*

*addi $v0, $zero, 0 # Khởi tạo độ dài tổng dãy con lớn nhất trong $v0 là 0*

*addi $v1, $zero, 0 # Khởi tạo tổng dãy con lớn nhất trong $v1 là 0*

*addi $t0, $zero, 0 # Khởi tạo chỉ số i trong $t0 là 0*

*addi $t1, $zero, 0 # Khởi tạo tổng chạy trong $t1 là 0*

*loop:*

*add $t2, $t0, $t0 # Nhân i với 2*

*add $t2, $t2, $t2 # Nhân kết quả với 2*

*add $t3, $t2, $a0 # Thêm A[i] vào 4i để lấy địa chỉ của A[i]*

*lw $t4, 0($t3) # Load giá trị A[i] vào $t4*

*add $t1, $t1, $t4 # Cộng A[i] vào tổng chạy $t1*

*slt $t5, $v1, $t1 # So sánh tổng chạy với tổng dãy con lớn nhất*

*bne $t5, $zero, mdfy # Nếu tổng chạy lớn hơn tổng dãy con lớn nhất, điều chỉnh kết quả*

*j test # Chuyển đến bước kiểm tra tiếp theo*

*mdfy:*

*addi $v0, $t0, 1 # Cập nhật độ dài của dãy con lớn nhất*

*addi $v1, $t1, 0 # Cập nhật tổng dãy con lớn nhất*

*test:*

*addi $t0, $t0, 1 # Tăng chỉ số i lên 1*

*slt $t5, $t0, $a1 # Kiểm tra xem đã duyệt qua tất cả các phần tử chưa*

*bne $t5, $zero, loop # Nếu chưa, tiếp tục vòng lặp*

*# In ra tổng dãy con lớn nhất*

*li $v0, 1 # Load mã syscall để in một số nguyên*

*move $a0, $v1 # Di chuyển tổng dãy con lớn nhất vào $a0*

*syscall # In tổng dãy con lớn nhất*

*# Kết thúc chương trình*

*li $v0, 10 # Load mã syscall để kết thúc chương trình*

*syscall*

Nhập số lượng phần tử trong là 6 bao gồm: 1,-3,2,4,-5,4. Kết quả thu được là 4





Giá trị thanh ghi $v0 tăng từ 0 lên 5, sau đó lại bị giảm về 0.



* Đúng với lí thuyết.

**Assignment 2:**

*.data*

*A: .space 100 # Khai báo mảng A với không gian 100 từ*

*Message3: .asciiz "\n" # Ký tự xuống dòng mới*

*ms: .asciiz " " # Khoảng trắng*

*.text*

*main:*

*li $v0, 5 # Syscall để đọc một số nguyên (độ dài của mảng)*

*syscall*

*move $t0, $v0 # Lưu độ dài của mảng vào $t0*

*la $a3, A # Gán $a3 là địa chỉ của phần tử đầu tiên trong mảng*

*j insert*

*after\_insert:*

*la $a0, A # $a0 = Địa chỉ(A[0])*

*la $t8, ($t0)*

*mul $t7, $t0, 4*

*add $a1, $a0, $t7*

*add $a1, $a1, -4*

*j sort # Sắp xếp*

*after\_sort:*

*li $v0, 10 # Thoát*

*syscall*

*j end\_main*

*print:*

*beq $t9, $t8, after\_print*

*la $a0, A*

*mul $t6, $t9, 4*

*add $t7, $a0, $t6*

*lw $a0, ($t7)*

*li $v0, 1*

*syscall*

*li $v0, 4*

*la $a0, ms*

*syscall*

*addi $t9, $t9, 1*

*j print*

*insert:*

*li $t1, 0*

*loop\_insert:*

*beq $t1, $t0, after\_insert # Quay lại main*

*li $v0, 5 # Syscall để đọc một số nguyên*

*syscall*

*sw $v0, 0($a3)*

*addi $t1, $t1, 1*

*add $a3, $a3, 4*

*j loop\_insert*

*sort:*

*beq $a0,$a1,done # Mảng chỉ có một phần tử đã được sắp xếp*

*j max # Gọi thủ tục max*

*after\_max:*

*lw $t0,0($a1) # Load phần tử cuối cùng vào $t0*

*sw $t0,0($v0) # Sao chép phần tử cuối cùng vào vị trí lớn nhất*

*sw $v1,0($a1) # Sao chép giá trị lớn nhất vào phần tử cuối cùng*

*addi $a1,$a1,-4 # Giảm con trỏ xuống phần tử cuối cùng*

*li $v0, 4 # Syscall để in chuỗi*

*la $a0, Message3*

*syscall*

*li $t9, 0*

*j print*

*after\_print:*

*j sort*

*done:*

*j after\_sort*

*max:*

*la $a0, A # Đặt con trỏ A[0]*

*addi $v0,$a0,0 # Khởi tạo con trỏ lớn nhất tại phần tử đầu tiên*

*lw $v1,0($v0) # Khởi tạo giá trị lớn nhất là giá trị đầu tiên*

*addi $t0,$a0,0 # Khởi tạo con trỏ tiếp theo*

*loop:*

*beq $t0,$a1,ret # Nếu con trỏ tiếp theo bằng phần tử cuối cùng, trở về*

*addi $t0,$t0,4 # Tăng con trỏ đến phần tử tiếp theo*

*lw $t1,0($t0) # Load phần tử tiếp theo vào $t1*

*slt $t2,$t1,$v1 # (tiếp theo) < (lớn nhất) ?*

*bne $t2,$zero,loop # Nếu (tiếp theo) < (lớn nhất), lặp lại*

*addi $v0,$t0,0 # Phần tử tiếp theo là phần tử lớn nhất mới*

*addi $v1,$t1,0 # Giá trị tiếp theo là giá trị lớn nhất mới*

*j loop # Đã hoàn thành việc thay đổi; bây giờ lặp lại*

*ret:*

*j after\_max*

*end\_main:*

Ban đầu nhập mảng có 6 phần tử gồm các giá trị 1,3,2,9,4,5.

Khi đó kết quả thu được là:



* Đúng với lí thuyết.

**Assignment 3:**

*.data*

*A: .word 3,2,-5,6,7,10,-9,12,5 # Khai báo mảng và khởi tạo giá trị ban đầu*

*Aend: .word # Kết thúc mảng*

*.text*

*la $a0, A # $a0 = Địa chỉ của A[0]*

*la $a1, Aend # $a1 = Địa chỉ của Aend*

*la $a2, Aend # $a2 = Địa chỉ của Aend*

*addi $a1, $a1, -4 # $a1 = Địa chỉ của A[n-1] (phần tử cuối cùng của mảng)*

*move $a3, $a1 # Sao chép địa chỉ phần tử cuối cùng vào $a3 (sẽ được sử dụng sau này)*

*j sort # Gọi tiểu trình sắp xếp*

*after\_sort:*

*li $v0, 10 # Số nguyên 10 đại diện cho syscall exit*

*syscall # Thực hiện syscall để thoát khỏi chương trình*

*end\_main:*

*sort: li $s0, 0 # Khởi tạo chỉ số sắp xếp*

*move $t7, $a1 # Sao chép địa chỉ của phần tử cuối cùng vào $t7*

*loop:*

*beq $t7, $a0, end\_loop # Nếu $t7 đến phần tử đầu tiên của mảng thì thoát vòng lặp*

*add $t0, $s0, $s0 # $t0 = 2 \* $s0*

*add $t0, $t0, $t0 # $t0 = 4 \* $s0*

*add $t1, $a0, $t0 # $t1 = Địa chỉ của A[s0]*

*lw $t2, 0($t1) # $t2 = A[s0]*

*addi $t3, $t1, 4 # $t3 = Địa chỉ của A[s0 + 1]*

*lw $t4, 0($t3) # $t4 = A[s0 + 1]*

*slt $t5, $t2, $t4 # Nếu A[s0] < A[s0 + 1] thì $t5 = 1, ngược lại $t5 = 0*

*bne $t5, $zero, tang\_$s0 # Nếu A[s0] < A[s0 + 1] thì nhảy đến nhãn tang\_$s0*

*move $t6, $t4 # Sao chép A[s0 + 1] vào $t6*

*move $t4, $t2 # Sao chép A[s0] vào A[s0 + 1]*

*move $t2, $t6 # Sao chép $t6 (giá trị ban đầu của A[s0 + 1]) vào A[s0]*

*sw $t2, 0($t1) # Lưu giá trị mới của A[s0] vào bộ nhớ*

*sw $t4, 0($t3) # Lưu giá trị mới của A[s0 + 1] vào bộ nhớ*

*addi $s0, $s0, 1 # Tăng chỉ số sắp xếp*

*addi $t7, $t7, -4 # Giảm địa chỉ phần tử cuối cùng xuống*

*j loop # Quay lại vòng lặp*

*tang\_$s0:*

*addi $s0, $s0, 1 # Tăng chỉ số sắp xếp*

*addi $t7, $t7, -4 # Giảm địa chỉ phần tử cuối cùng xuống*

*j loop # Quay lại vòng lặp*

*end\_loop:*

*addi $a3, $a3, -4 # Giảm địa chỉ phần tử cuối cùng xuống*

*#print*

*print:*

*li $s1, 0 # Khởi tạo chỉ số in*

*la $s4, A # $s4 = Địa chỉ của A*

*print\_char:*

*add $s2, $s1, $s1 # $s2 = 2 \* $s1*

*add $s2, $s2, $s2 # $s2 = 4 \* $s1*

*add $s3, $s4, $s2 # $s3 = Địa chỉ của A[s1]*

*lw $s5, 0($s3) # $s5 = A[s1]*

*beq $s3, $a2, in\_xuong\_dong*

*li $v0, 1 # Số nguyên 1 đại diện cho syscall để in một số nguyên*

*move $a0, $s5 # Đặt $a0 bằng giá trị cần in*

*syscall # Thực hiện syscall để in giá trị số nguyên*

*addi $s6, $a2, -4 # $s6 = Địa chỉ của Aend - 4*

*beq $s3, $s6, qua*

*li $v0, 11 # Số nguyên 11 đại diện cho syscall để in một ký tự*

*li $a0, ' ' # ' ' là ký tự cách*

*syscall # Thực hiện syscall để in ký tự cách*

*qua:*

*addi $s1, $s1, 1 # Tăng chỉ số in lên một đơn vị*

*j print\_char # Quay lại vòng lặp để in các phần tử còn lại của mảng*

*in\_xuong\_dong:*

*li $v0, 11 # Số nguyên 11 đại diện cho syscall để in một ký tự*

*li $a0, '\n' # '\n' là ký tự xuống dòng*

*syscall # Thực hiện syscall để in ký tự xuống dòng*

*la $a0, A # Đặt $a0 bằng địa chỉ bắt đầu của mảng A để chuẩn bị in lại từ đầu*

*beq $a3, $a0, after\_sort #kiem tra xem sap xep het mang chua*

*j sort # Nếu chưa in xong, quay lại tiểu trình sắp xếp*

Đoạn chương trình trên mô tả thuật toán buble sort sắp xếp các các phần tử của mảng theo giá trị tăng dần. Ban đầu khởi tạo các phần tử của mảng là 3, 2, -5, 6, 7, 10, -9, 12, 5. Kết quả thu được:

Ảnh có chứa văn bản, biên lai, ảnh chụp màn hình, đại số

Mô tả được tạo tự động

* Đúng với lí thuyết.

**Assignment 4:**

.data

A: .word 2,0,2,2,5,7,1,7 # Mảng cần sắp xếp

Aend: .word # Địa chỉ kết thúc mảng (khởi tạo sau)

.text

la $a0, A # $a0 lưu địa chỉ của A[0]

la $a1, Aend

add $a2, $a0, 4 # $a2 lưu địa chỉ của A[1]

la $a3, Aend

add $a1, $a1, -4 # $a1 lưu địa chỉ của A[n-1]

j sort # Nhảy đến phần con sắp xếp

after\_sort:

li $v0, 10 # syscall để kết thúc chương trình

syscall

end\_main:

sort:

li $s0, 1 # i (vị trí phần tử cần sắp xếp)

loop:

li $s1, 0 # Biến kiểm tra xem có cần chèn không?

add $t0, $s0, $s0

add $t0, $t0, $t0

add $t0, $a0, $t0 # Lấy địa chỉ của phần tử A[i]

addi $t2, $t0, -4 # Lấy địa chỉ của phần tử A[j] (j = i - 1)

move $t5, $t0 # $t5 lưu vị trí sẽ chèn (khởi tạo tại vị trí A[i])

loop1:

lw $t1, 0($t0) # $t1 lưu giá trị của A[i]

lw $t3, 0($t2) # $t3 lưu giá trị của A[j]

slt $t4, $t1, $t3 # Nếu A[j] < A[i]

beq $t4, $zero, tru\_tiep # Thì nhảy đến j--

move $t5, $t2 # A[j] > A[j] thì vị trí chèn sẽ là $t2

li $s1, 1 # Biến kiểm tra = 1

tru\_tiep:

beq $t2, $a0, chen # Nếu j-- đã đến vị trí A[0] thì bắt đầu chèn

addi $t2, $t2, -4 # j--

j loop1

chen:

beq $s1, 0, print # Kiểm tra xem A[i] đã đúng vị trí chưa? Nếu $s1 = 0 thì đã đúng và in dãy luôn

loop\_chen:

addi $t0, $t0, -4 # A[j]

lw $s2, 0($t0) # Lưu A[j] vào vị trí A[i]

sw $s2, 4($t0)

bne $t0, $t5, loop\_chen # Nếu A[j] chưa đến vị trí cần chèn thì tiếp tục

sw $t1, 0($t5) # Nếu đã đến vị trí chèn thì thực hiện chèn A[i]

print:

li $s4, 0 # Biến chỉ mục cho việc in

print\_char:

la $a0, A

add $s5, $s4, $s4

add $s5, $s5, $s5

add $s5, $a0, $s5

lw $s6, 0($s5) # Lưu A[j] vào vị trí A[i]

beq $s5, $a3, in\_xuong\_dong

li $v0, 1 # In số nguyên

move $a0, $s6

syscall

addi $s7, $a3, -4

beq $s5, $s7, skip

li $v0, 11 # In ký tự dấu cách

li $a0, ' '

syscall

skip:

addi $s4, $s4, 1 # Tăng biến chỉ mục

j print\_char

in\_xuong\_dong:

li $v0, 11 # In ký tự xuống dòng

li $a0, '\n'

syscall

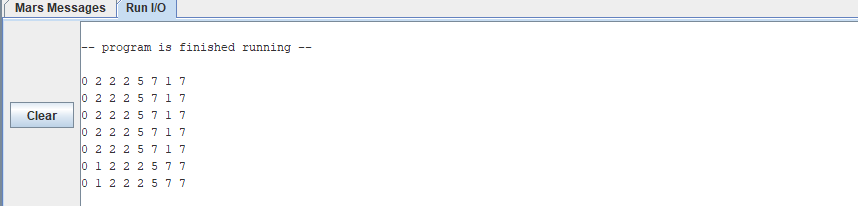
la $a0, A

beq $a1, $a2, after\_sort # Thực hiện vòng lặp đến A[n-1]

addi $s0, $s0, 1 # i++

addi $a2, $a2, 4 # Tăng $a2, thực hiện kiểm tra tiếp A[2]

j loop



=> Kết quả đúng với lý thuyết