**BÁO CÁO THỰC HÀNH KIẾN TRÚC MÁY TÍNH (IT3280) TUẦN 6**

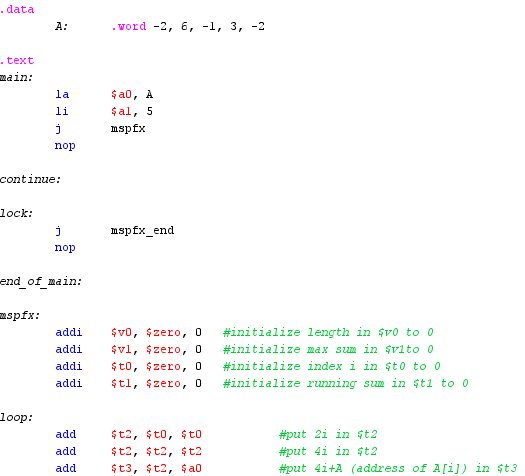
***Họ và tên: Phạm Đức Long***

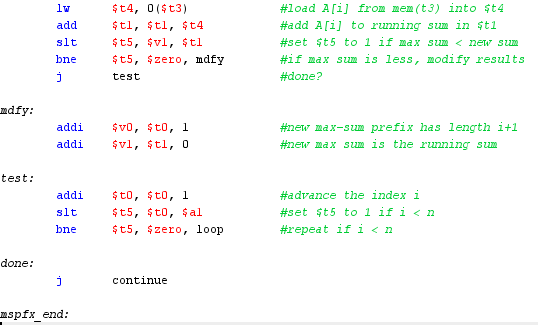
***MSSV: 20225737***

Assignment 1:

1. Thực hiện bài Home Assignment 1:

* Code:





* Kết quả:



* Giải thích: Chương trình này dùng để tìm ra dãy con bắt đầu từ vị trí đầu tiên đến vị trí N sao cho tổng của N số là lớn nhất và kết quả trả về là độ dài của dãy và tổng dãy.
* Khởi tạo chuỗi [-2, 6, -1, 3, -2]
* Khởi tạo $a0 = địa chỉ đầu tiên chuỗi (A[0]), $a1 = số phần tử chuỗi ( = 5)
* Khởi tạo 2 thanh ghi $v0 để lưu độ dài chuỗi, $v1 lưu tổng lớn nhất
* $t0 = i, Step = $t2 = 4\*i, $t3 = địa chỉ của A[step] = A[0 + i]
* Load word $t4 = A[i]
* $t1 = sum = tổng các phần tử sau mỗi bước lặp = $t1 + A[i]
* So sánh $t1 là tổng hiện thời với $v1 (tổng lớn nhất, ban đầu được gán bằng 0), nếu $t1 > $v1 thì thực hiện cập nhật lại giá trị của $v1 ( = $t1) và $v0 ( = i + 1), sau đó sẽ tăng i lên, nếu i < n tiếp tục thực hiện vòng lặp, ngược lại sẽ kết thúc và kết luận độ dài dãy có tổng lớn nhất và tổng lớn nhất. Như ở trên ta thu được $v0 = 4 và $v1 = 6 🡪 Đúng với tính toán theo lý thuyết.

1. Thực hiện nhập mảng có N phần tử và in ra độ dài và tổng của dãy có tổng tính từ phần tử đầu tiên đến N lớn nhất.

* Code:

**.data**

**A: .word**

**Message1: .asciiz "Nhap so luong phan tu: "**

**Message2: .asciiz "Nhap so: "**

**Message3: .asciiz "Do dai cua mang co tong lon nhat la: "**

**Message4: .asciiz "Tong lon nhat la: "**

**Newline: .asciiz "\n"**

**.text**

**input\_number:**

**# Nhap so luong phan tu:**

**li $v0, 4**

**la $a0, Message1**

**syscall**

**li $v0, 5**

**syscall**

**move $a1, $v0 # Gan N vao $a1**

**sub $a1, $a1, 1 # Tru $a1 di 1**

**la $a0, A**

**addi $s0, $a0, 0 # Gan dia chi mang A vao $s0**

**input\_array:**

**# Nhap cac phan tu cua mang:**

**bgt $t0, $a1, main**

**# Neu $t0 > $a1 thi ket thuc nhap phan tu**

**li $v0, 4**

**la $a0, Message2**

**syscall**

**li $v0, 5**

**syscall**

**move $t1, $v0**

**sw $t1, 0($s0) # Gan cac gia tri vua nhap vao mang A**

**addi $s0, $s0, 4**

**# Tro den dia chi cua phan tiep theo trong mang**

**addi $t0, $t0, 1**

**j input\_array**

**main: la $a0, A**

**j mspfx**

**nop**

**continue:**

**# In ra output**

**addi $s0, $v0, 0**

**addi $s1, $v1, 0**

**li $v0, 4**

**la $a0, Message3**

**syscall**

**li $v0, 1**

**la $a0, 0($s0)**

**syscall**

**li $v0, 4**

**la $a0, Newline**

**syscall**

**li $v0, 4**

**la $a0, Message4**

**syscall**

**li $v0, 1**

**la $a0, 0($s1)**

**syscall**

**lock: li $v0, 10**

**syscall**

**end\_of\_main:**

**mspfx:**

**addi $v0, $zero, 0 # max\_length = 0**

**addi $v1, $zero, 0 # max\_sum = 0**

**addi $t0, $zero, 0 # i = 0**

**addi $t1, $zero, 0 # sum = 0**

**loop: add $t2, $t0, $t0 # 2\*i**

**add $t2, $t2, $t2 # 4\*i**

**add $t3, $t2, $a0 # $t3 = (address of A) + 4\*i**

**lw $t4, 0($t3) # $t4 = A[i]**

**add $t1, $t1, $t4 # sum += A[i]**

**slt $t5, $v1, $t1 # if max\_sum < sum**

**bne $t5, $zero, mdfy # j mdfy**

**j test**

**mdfy: addi $v0, $t0, 1 # max\_length = i + 1**

**addi $v1, $t1, 0 # max\_sum = sum**

**test: addi $t0, $t0, 1 # i++**

**sle $t5, $t0, $a1 # if i <= n thi loop**

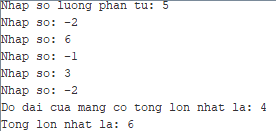
**bne $t5, $zero, loop**

**done:**

**j continue**

**mspfx\_end:**

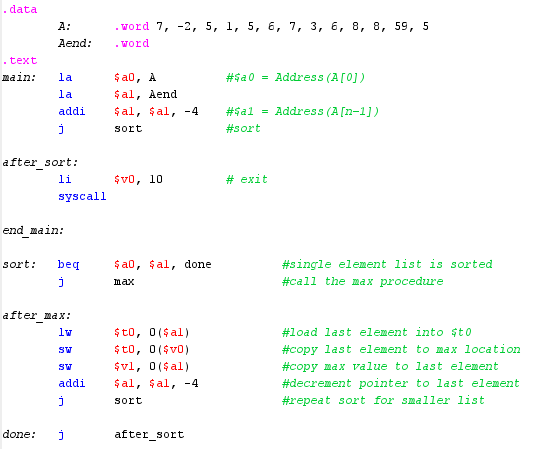
* Kết quả thu được:

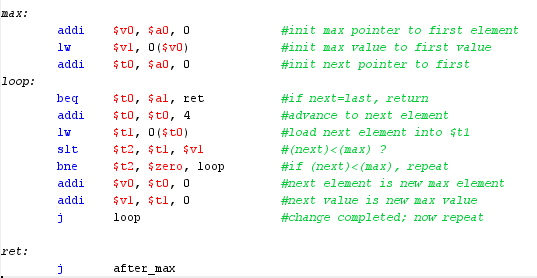


* Kết quả thu được hoàn toàn đúng theo lý thuyết.

Assignment 2:

1. Thực hiện bài Home Assignment 2:





* Trước khi sắp xếp:



* Sau khi sắp xếp:



* Giải thích:
* Chương trình sắp xếp chuỗi bằng cách sử dụng một biến (thanh ghi) để lưu **giá trị** và một con trỏ (thanh ghi) để lưu **vị trí** của phần tử **giá trị** lớn nhất chuỗi, sau khi tìm được giá trị lớn nhất ta hoán đổi giá trị của số lớn nhất và số cuối cùng của chuỗi
* Khởi tạo chuỗi [7, -2, 5, 1, 5, 6, 7, 3, 6, 8, 8, 59, 5]
* Khởi tạo $a0 = địa chỉ đầu tiên chuỗi (A[0]), $a1 = địa chỉ cuối chuỗi (A[n-1])
* Chạy hàm sort nếu địa chỉ của A[i] = A[n-1] thì chuỗi đã sắp xếp xong và nhảy sang hàm done và kết thúc chương trình thu được chuỗi tăng dần ngược lại ta thực hiện hàm after\_sort để in ra các chuỗi sắp xếp
* Hàm max để tìm ra con trỏ và giá trị lớn nhất có $v0 = địa chỉ của A[i] và $v1 = A[i], khởi tạo i = 0
* Hàm loop so sánh $t0 và $a1 <=> so sánh địa chỉ của A[i] và A[n-1]. Nếu bằng thì ta thực hiện hàm ret nhảy đến hàm after\_max để hoán đổi giá trị của số lớn nhất và số ở vị trí cuối cùng bằng cách lưu giá trị của địa chỉ A[n-1] vào $t0, sau đó truyền giá trị $t0 và địa chỉ ptr, cuối cùng truyền giá trị max và địa chỉ của A[n-1] và giảm n đi 1 để chuyển vị trí cuối cùng sang n-2
* Ngược lại thì lưu giá trị của A[i + 1] vào $t1 và so sánh $t1 và $v1. Nếu $t1 nhỏ hơn $v1 (giá trị tiếp theo nhỏ hơn giá trị ban đầu) thì thực hiện lại vòng loop còn ngược lại sẽ cập lại lại $v0 = địa chỉ của A[i + 1] và $v1 = A[i + 1] sau đó cứ loop lại cho đến khi i = n

1. Thực hiện nhập mảng có N phần tử và sắp xếp mảng đó theo thứ tự tăng dần bằng sắp xếp lựa chọn:

* Code:

**.data**

**A: .word**

**Message: .asciiz "Nhap so luong phan tu cua mang: "**

**Message0: .asciiz "Nhap so: "**

**Message1: .asciiz "Mang sau khi thuc hien selection sort la:"**

**Message2: .asciiz ", "**

**Newline: .asciiz "\n"**

**.text**

**input\_number:**

**# Nhap N la so luong phan tu:**

**li $v0, 4**

**la $a0, Message**

**syscall**

**li $v0, 5**

**syscall**

**move $a1, $v0**

**sub $a1, $a1, 1**

**addi $s0, $s0, 0**

**la $a0, A**

**addi $s0, $a0, 0**

**input\_array:**

**# Nhap cac gia tri cua mang:**

**bgt $t0, $a1, end\_input**

**li $v0, 4**

**la $a0, Message0**

**syscall**

**li $v0, 5**

**syscall**

**move $t1, $v0**

**sw $t1, 0($s0)**

**addi $s0, $s0, 4**

**addi $t0, $t0, 1**

**j input\_array**

**end\_input:**

**main: la $a0, A # address of A[0]**

**addi $s3, $a1, 1**

**mul $s4, $s3, 4**

**add $t6, $a0, $s4**

**la $a1, 0($t6)**

**sub $a1, $a1, 4**

**j sort**

**after\_sort:**

**# Print message1**

**li $v0, 4**

**la $a0, Message1**

**syscall**

**# Print newLine**

**la $a0, Newline**

**syscall**

**la $s0, A**

**la $s1, 0($t6)**

**lw $s2, 0($s0)**

**li $v0, 1**

**la $a0, 0($s2) # print number1 of array**

**syscall**

**addi $t3, $zero, 0 # i = 0**

**print\_array:**

**addi $t3, $t3, 4 # i += 1**

**add $t4, $s0, $t3 # $t1 = address of A[0] + 4\*i**

**lw $t5, 0($t4) # x = A[i]**

**beq $t4, $s1, end\_main # if i > (n-1) end**

**li $v0, 4**

**la $a0, Message2**

**syscall # Print Message2**

**li $v0, 1**

**la $a0, 0($t5)**

**syscall # Print A[i]**

**j print\_array**

**end\_main:**

**li $v0, 10**

**syscall**

**sort: bgt $a0, $a1, after\_sort # if i = n done**

**j max**

**after\_max:**

**lw $t0, 0($a1) # $t0 = value of adress A[n-1]**

**sw $t0, 0($v0) # value of address ptr = $t0**

**sw $v1, 0($a1) # value of A[n-1] = max**

**addi $a1, $a1, -4 # n -=**

**j sort**

**max:**

**la $a0, A**

**addi $v0, $a0, 0 # ptr = address of A[0]**

**lw $v1, 0($v0) # max = A[0]**

**addi $t0, $a0, 0 # i = 0**

**loop:**

**beq $t0, $a1, ret # if i = n ret**

**addi $t0, $t0, 4 # i += 1**

**lw $t1, 0($t0) # temp = A[i]**

**slt $t2, $t1, $v1 # temp < max**

**bne $t2, $zero, loop # if temp < max loop**

**addi $v0, $t0, 0 # ptr = address of A[i]**

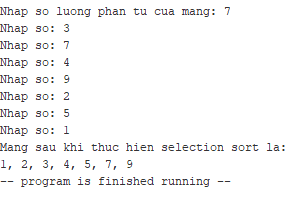
**addi $v1, $t1, 0 # max = A[i]**

**j loop**

**ret:**

**j after\_max**

* Kết quả:



* Chương trình hoàn toàn hoạt động chính xác.

Assignment 3:

* Thuật toán Bubble Sort:

void BubbleSort(int a[], int n){

int temp;

for (int i = 0; i < n; i++) {

for (int j = i + 1; j < n; j++) {

if (a[j] > a[j + 1]){

temp = a[j];

a[j] = a[j + 1];

a[j + 1] = temp;

}

}

}

}

* Code:

**.data**

**A: .word**

**Message: .asciiz "Nhap so luong phan tu N: "**

**Message0: .asciiz "Nhap so: "**

**Message1: .asciiz "Mang sau khi thuc hien bubble sort la: "**

**Message2: .asciiz ", "**

**Newline: .asciiz "\n"**

**.text**

**input\_number:**

**# Nhap so luong phan tu N:**

**li $v0, 4**

**la $a0, Message**

**syscall**

**li $v0, 5**

**syscall**

**move $a1, $v0**

**sub $a1, $a1, 1 # n -= 1**

**addi $s0, $s0, 0**

**la $a0, A**

**addi $s0, $a0, 0**

**input\_array:**

**# Nhap cac phan tu cua A:**

**bgt $t0, $a1, end\_input**

**li $v0, 4**

**la $a0, Message0**

**syscall**

**li $v0, 5**

**syscall**

**move $t1, $v0**

**sw $t1, 0($s0)**

**addi $s0, $s0, 4**

**addi $t0, $t0, 1**

**j input\_array**

**end\_input:**

**main:**

**# Khoi tao cac gia tri**

**la $a0, A # address of A[0]**

**addi $a2, $a0, 4 # address of A[1]**

**mul $s3, $a1, 4 # $s3 = (n - 1) \* 4**

**add $a1, $s3, $a0 # $a1 = (n - 1) \* 4 + A[0] = address of A[n - 1]**

**addi $t6, $a1, 4 # $t6 = address of A[n]**

**addi $t0, $zero, 0 # $t0 to count the loop**

**j bubble\_sort**

**swap:**

**lw $t2, 0($a2) # $t2 = A[i]**

**sw $t2, 0($t1) # A[j] = $t2 = A[i]**

**sw $v0, 0($a2) # A[i] = \*($v0) = A[j]**

**j continue**

**reset:**

**la $a0, A**

**addi $t0, $zero, 0 # j = 0**

**addi $a2, $a2, 4 # i += 1**

**j bubble\_sort**

**bubble\_sort:**

**bgt $a2, $a1, print\_sort # if i > (n-1) end**

**add $t1, $a0, $t0 # $t1 = address of A[0] + 4\*j = A[j]**

**beq $t1, $a2, reset # if j = i print**

**lw $v0, 0($t1) # $v0 = A[j]**

**lw $v1, 0($a2) # $v1 = A[i]**

**blt $v1, $v0, swap # if A[i] > A[j] swap**

**continue:**

**addi $t0, $t0, 4 # j++**

**j bubble\_sort**

**print\_sort:**

**# Print message1**

**li $v0, 4**

**la $a0, Message1**

**syscall**

**# Print newLine**

**la $a0, Newline**

**syscall**

**la $s0, A**

**la $s1, 0($t6)**

**lw $s2, 0($s0)**

**li $v0, 1**

**la $a0, 0($s2) # Print number1 of array**

**syscall**

**addi $t3, $zero, 0 # i = 0**

**print\_array:**

**addi $t3, $t3, 4 # i++**

**add $t4, $s0, $t3 # $t1 = adrress of A[0] + 4\*i = A[i]**

**lw $t5, 0($t4) # x = A[i]**

**beq $t4, $s1, end # if i > (n-1) end**

**li $v0, 4**

**la $a0, Message2**

**syscall # Print Message2**

**li $v0, 1**

**la $a0, 0($t5)**

**syscall # Print A[i]**

**j print\_array**

**end:**

**li $v0, 4**

**la $a0, Newline**

**syscall**

**end\_main:**

**# Exit**

**li $v0, 10**

**syscall**

* Kết quả:

A white text box with black text

Description automatically generated

* Chương trình hoàn toàn hoạt động chính xác.

Assignment 4:

* Thuật toán Insertion Sort:

**void** insertionSort(**int** arr[], **int** n)

{

**int** i, key, j;

**for** (i = 1; i < n; i++) {

        key = arr[i];

        j = i - 1;

**while** (j >= 0 && arr[j] > key) {

            arr[j + 1] = arr[j];

            j = j - 1;

        }

        arr[j + 1] = key;

    }

}

* Code: