Báo cáo thực hành KTMT Tuần 6

Phạm Thành Lập – 20215076

Assignment 1

```
.data
A:      .word -2, 6, -1, 3, -2
.text
main:   la $a0,A
        li $a1,5                # n = 5
        j mspfx
        nop
continue:
lock:   j lock
        nop
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) + 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 = max
test:   addi $t0,$t0,1          # i++
        slt $t5,$t0,$a1         # if i<n
        bne $t5,$zero,loop
done:   j continue
mspfx_end:
```

$v0	2	4
$v1	3	6

- Chương trình 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
- 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]
- Lưu giá trị của $t3 vào $t4 ($t4 = A[step])
- Gắn $t1 là thanh ghi để lưu tổng tạm thời, $t1 += A[step]

- So sánh $t1 (tổng tạm thời) với $v1 (tổng lớn nhất), nếu $t1 > $v1 thì thực hiện cập nhật lại giá trị của $v1 và $v0 ngược lại 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
- Sự thay đổi giá trị thanh ghi

Trạng thái	$v0	$v1	$t0	$t1	$t3
Ban đầu	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
Sau loop 1 (Trước khi i++)	-	-	-	0xfffffffe	0x10010000
Sau loop 2 (Trước khi i++)	0x00000002	0x00000002	0x00000001	0x00000004	0x10010004
Sau loop 3 (Trước khi i++)	-	-	0x00000002	0x0000003	0x10010008
Sau loop 4 (Trước khi i++)	0x00000006	0x0000004	0x00000003	0x00000006	0x1001000c
Sau loop 5 (Trước khi i++)	-	-	0x00000004 (End loop)	0x00000004	0x10010010

⇨ Như vậy kết quả thu được là $v0 = 6 và $v1 = 4 (Thỏa mãn)

- **TextCode**

```
.data
A:        .word
Message1:     .asciiz  "Enter number of array: "
Message2:     .asciiz  "Enter number: "
Message3:     .asciiz  "Length of max array is: "
Message4:     .asciiz  "Sum max is: "
Newline:      .asciiz  "\n"
.text
input_number:
        li $v0,4
        la $a0, Message1
        syscall
        li $v0,5
        syscall
```

```
        move $a1, $v0
        subi $a1, $a1, 1
        addi $s0, $s0, 0
        la $a0, A
        addi $s0, $a0, 0
input_array:
        bgt $t0, $a1, end_input
        li $v0,4
        la $a0, Message2
        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
        j mspfx
        nop
continue:
        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) + 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 = max
test:   addi $t0,$t0,1       # i++
        slt $t5,$t0,$a1      # if i<n
        bne $t5,$zero,loop
done:   j continue
mspfx_end:
```

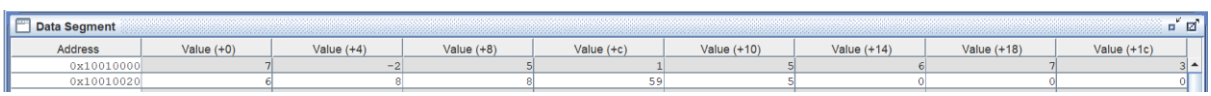
- **Result**

```
Enter number of array: 5
Enter number: -2
Enter number: 6
Enter number: -1
Enter number: 3
Enter number: -2
Length of max array is: 4
Sum max is: 6
-- program is finished running --
```

Assignment 2

```
.data
        A: .word 7, -2, 5, 1, 5,6,7,3,6,8,8,59,5
        Aend: .word
.text
main:   la $a0,A                #$a0 = Address(A[0])
        la $a1,Aend
        addi $a1,$a1,-4         #$a1 = Address(A[n-1])
        j sort                  #sort
after_sort: li $v0, 10          #exit
        syscall
end_main:
sort:   beq $a0,$a1,done        #single element list is sorted
        j max                   #call the max procedure
after_max: lw $t0,0($a1)        #load last element into $t0
        sw $t0,0($v0)           #copy last element to max location
        sw $v1,0($a1)           #copy max value to last element
        addi $a1,$a1,-4         #decrement pointer to last element
        j sort                  #repeat sort for smaller list
done:   j after_sort
max:
        addi $v0,$a0,0          #init max pointer to first element
        lw $v1,0($v0)           #init max value to first value
        addi $t0,$a0,0          #init next pointer to first
loop:
        beq $t0,$a1,ret         #if next=last, return
        addi $t0,$t0,4          #advance to next element
        lw $t1,0($t0)           #load next element into $t1
        slt $t2,$t1,$v1         #(next)<(max) ?
        bne $t2,$zero,loop      #if (next)<(max), repeat
        addi $v0,$t0,0          #next element is new max element
        addi $v1,$t1,0          #next value is new max value
        j loop                  #change completed; now repeat
ret:
        j after_max
```

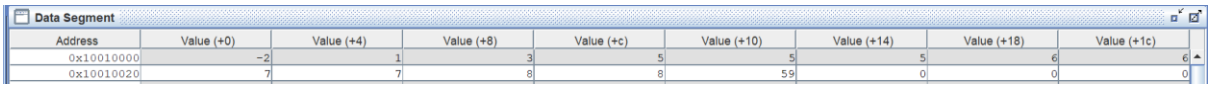
- **Trước khi sort**



Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	7	-2	5	1	5	6	7	3
0x10010020	6	8	8	59	5	0	0	0

- **Sau khi sort**



- Chương trình sắp xếp chuỗi bằng cách sử dụng 1 biến để lưu giá trị và 1 con trỏ lưu vị trí của giá trị của giá trị lớn nhất của 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 nếu chưa i < n thì truyền $t1 địa chỉ của A[i] và lưu giá trị ở địa chỉ A[i] vào $t1 so sánh $t2 và $t1 nếu nhỏ hơn thì thực hiện lại vòng loop còn ngược lại sẽ cập lại lại $v0 = địa chỉ của A[i] và $v1 = A[i] sau đó loop lại cho đến khi i = n

- **TextCode**

```
.data
        A:      .word
        Message:        .asciiz  "Enter number of array: "
        Message0:       .asciiz  "Enter number: "
        Message1:       .asciiz "Array after each round is:"
        Message2:       .asciiz " , "
        Newline:        .asciiz "\n"
.text
input_number:
        li $v0,4
        la $a0, Message
        syscall
        li $v0,5
        syscall
        move $a1, $v0
        subi $a1, $a1, 1
        addi $s0, $s0, 0
        la $a0, A
        addi $s0, $a0, 0
input_array:
        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)
        subi $a1, $a1, 4
        j sort
after_sort:
        li $v0, 4
        la $a0, Message1
        syscall                 #print message1
        la $a0, Newline
        syscall                 #print newLine
        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
        j print_array
print_array:
        addi $t3, $t3, 4                # i++
        add $t4, $s0, $t3               # $t1 = address 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_main:
        li $v0, 10
        syscall
end:
        li $v0, 4
        la $a0, Newline
        syscall
        j max
sort:   bgt $a0,$a1,done        # if i = n done
```

```
        j after_sort
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
done:   j end_main
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 = 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
```

- **Result**

```
-- program is finished running --

Enter number of array: 5
Enter number: 1
Enter number: -9
Enter number: -5
Enter number: 9
Enter number: 5
Array after each round is:
1 , -9 , -5 , 9 , 5
Array after each round is:
1 , -9 , -5 , 5 , 9
Array after each round is:
1 , -9 , -5 , 5 , 9
Array after each round is:
-5 , -9 , 1 , 5 , 9
Array after each round is:
-9 , -5 , 1 , 5 , 9

-- program is finished running --
```

Assignment 3

- **TextCode**

```
.data
        A:      .word
        Message:        .asciiz  "Enter number of array: "
        Message0:       .asciiz  "Enter number: "
        Message1:       .asciiz "Array after each round bubble sort is:"
        Message2:       .asciiz " , "
        Newline:        .asciiz "\n"
.text
input_number:
        li $v0,4
        la $a0, Message
        syscall
        li $v0,5
        syscall
        move $a1, $v0
        subi $a1, $a1, 1
        addi $s0, $s0, 0
        la $a0, A
        addi $s0, $a0, 0
input_array:
        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]1
        addi    $a2, $a0, 4     # i = 2
        addi $s3, $a1, 1
        mul  $s4, $s3, 4
        add $t6, $a0, $s4
        la $a1, 0($t6)
        subi $a1, $a1, 4
        addi $t0, $zero, 0
        j bubble_sort
swat:
        lw      $t2, 0($a2)     # $t2 = value of address A[i]
        sw      $t2, 0($t1)     # value of address A[j] = $t2
        sw      $v0, 0($a2)     # value of A[i] = A[j]
        j continue
reset:
        la      $a0, A
```

```
        addi    $t0, $zero, 0   # j = 0
        addi    $a2, $a2, 4     # i++
        j bubble_sort
print_sort:
        li      $v0, 4
        la      $a0, Message1
        syscall                 #print message1
        la      $a0, Newline
        syscall                 #print newLine
        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
        j print_array
bubble_sort:
        bgt     $a2, $a1, end_main   # if i>(n-1) end
        add     $t1, $a0, $t0        # $t1 = address of A[0] + 4*j (A[j])
        beq     $t1, $a2, print_sort # if j = i print
        lw      $v0, 0($t1)          # $v0 = A[j]
        lw      $v1, 0($a2)          # $v1 = A[i]
        blt     $v1, $v0, swat       # if A[i] > A[j] swat
continue:
        addi    $t0, $t0, 4          # j++
        j bubble_sort
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
        j       reset
end_main:
        li      $v0, 10
        syscall                      # exit
```

- **Result**

```
Enter number of array: 5
Enter number: -6
Enter number: -9
Enter number: 5
Enter number: -3
Enter number: 1
Array after each round bubble sort is:
-9 , -6 , 5 , -3 , 1
Array after each round bubble sort is:
-9 , -6 , 5 , -3 , 1
Array after each round bubble sort is:
-9 , -6 , -3 , 5 , 1
Array after each round bubble sort is:
-9 , -6 , -3 , 1 , 5

-- program is finished running --
```

Assignment 4

- **TextCode**

```
.data
    A:      .word
    Message:    .asciiz  "Enter number of array: "
    Message0:   .asciiz  "Enter number: "
    Message1:   .asciiz "Array after each round insertion sort is:"
    Message2:   .asciiz " , "
    Newline:    .asciiz "\n"
.text
input_number:
    li $v0,4
    la $a0, Message
    syscall
    li $v0,5
    syscall
    move $a1, $v0
    subi $a1, $a1, 1
    addi $s0, $s0, 0
    la $a0, A
    addi $s0, $a0, 0
```

```
input_array:
        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)
        subi $a1, $a1, 4
        addi $t0, $zero, 0
        j insertion_sort
print_sort:
        li      $v0, 4
        la      $a0, Message1
        syscall                 #print message1
        la      $a0, Newline
        syscall                 #print newLine
        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
        j print_array
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
        j       insertion_sort
insertion_sort:
        la      $a0, A
        addi    $t0, $t0, 4     # i++
        add     $v0, $a0, $t0   # i = 1
        bgt     $v0, $a1, end_main
        lw      $s0, 0($v0)     # key = A[i]
        subi    $v1, $v0, 4     # j = i-1
loop:
        blt     $v1, $a0, end_loop    # if j < 0 end_loop
        lw      $s1, 0($v1)           # $s1 = A[j]
        blt     $s1, $s0, end_loop    # if A[j] < key end_loop
```

```
        addi    $t2, $v1, 4             # $t2 = j+1
        sw      $s1, 0($t2)             # A[j+1] = A[j]
        subi    $v1, $v1, 4             # j--
        j       loop
end_loop:
        addi    $t2, $v1, 4     # $t2 = j+1
        sw      $s0, 0($t2)     # A[j+1] = key
        j print_sort
end_main:
        li      $v0, 10
        syscall
```

- **Result**

```
Enter number of array: 5
Enter number: -6
Enter number: -9
Enter number: 5
Enter number: 4
Enter number: 0
Array after each round insertion sort is:
-9 , -6 , 5 , 4 , 0
Array after each round insertion sort is:
-9 , -6 , 5 , 4 , 0
Array after each round insertion sort is:
-9 , -6 , 4 , 5 , 0
Array after each round insertion sort is:
-9 , -6 , 0 , 4 , 5

-- program is finished running --
```