Lab Report 03

Assignment 1

Code:

```
# Laboratory Exercise 3, Home Assignment 1

.text

init:   addi    $s1, $zero, 1       # i = 1
        addi    $s2, $zero, 2       # j = 2
if:     slt     $t0, $s2, $s1       # j < i
        bne     $t0, $zero, else    # branch to else if j < i
        addi    $t1, $t1, 1         # then part: x = x + 1
        addi    $t3, $zero, 1       # z = 1
        j       endif               # skip “else” part
else:   addi    $t2, $t2, -1        # begin else part: y = y - 1
        add     $t3, $t3, $t3       # z = 2 * z
endif:
```

Comments:

- Trong init:
  - Khởi tạo i = 1 tại $s1
  - Khởi tạo j = 2 tại $s2
- Trong if:
  - Gán $t0 = 0 vì j > i
  - Lệch bne không nhảy sang else vì $t0 = 0
  - Update x từ 0 lên 1
  - Update z từ 0 lên 1
  - Nhảy tới endif để kết thúc

Assignment 2

Code:

```
# Laboratory 3, Home Assigment 2

.data

A:      .word       1, 2, 3, 4, 5, 6, 7
```

```
.text
init:   addi    $s1, $zero, 0       # i = 0
        la      $s2, A
        addi    $s3, $zero, 7       # n = 7
        addi    $s4, $zero, 1       # step = 1
        addi    $s5, $zero, 0       # sum = 0
loop:   slt     $t2, $s1, $s3       # $t2 = i < n ? 1 : 0
        beq     $t2, $zero, endloop
        add     $t1, $s1, $s1       # $t1 = 2 * $s1
        add     $t1, $t1, $t1       # $t1 = 4 * $s1
        add     $t1, $t1, $s2       # $t1 store the address of A[i]
        lw      $t0, 0($t1)         # load value of A[i] in $t0
        add     $s5, $s5, $t0       # sum = sum + A[i]
        add     $s1, $s1, $s4       # i = i + step
        j       loop                # goto loop
endloop:
```

Comments:

- Trong init:
  - Khởi tạo i = 0 tại $s1
  - Lưu địa chỉ của A vào $s2
  - Khởi tạo n = 7 tại $s3
  - Khởi tạo step = 1 tại $s4
  - Khởi tạo sum = 0 tại $s5
- Trong if:
  - Kiểm tra điều kiện i < n
  - Nếu sai (0) thì nhảy tới endloop
  - Nếu đúng (1) thì cộng 4i vào địa chỉ bắt đầu mảng A
  - Lấy dữ liệu A[i] tại địa chỉ vừa tính được
  - Update i = i + 1
  - Lặp lại vòng lặp

Assignment 3

Code:

```
# Laboratory Exercise 3, Home Assignment 3
.data
test:       .word 1
.text
init:       addi    $s2, $zero, 1       # a = 1
            addi    $s3, $zero, 2       # b = 2
switch:     la      $s0, test           # load the address of test variable
            lw      $s1, 0($s0)         # load the value of test to register $t1
            li      $t0, 0              # load value for test case
            li      $t1, 1
            li      $t2, 2
            beq     $s1, $t0, case_0
            beq     $s1, $t1, case_1
            beq     $s1, $t2, case_2
            j       default
case_0:     addi    $s2, $s2, 1         # a = a + 1
            j       continue
case_1:     sub     $s2, $s2, $t1       # a = a - 1
            j       continue
case_2:     add     $s3, $s3, $s3       # b = 2 * b
            j       continue
default:
continue:
```

Comments:

- Trong init:
  - Khởi tạo a = 1 tại $s2
  - Khởi tạo b = 2 tại $s3
- Trong switch:
  - Vì test = 1, nên chương trình sẽ nhảy tới case_1
- Trong case_1:
  - Thực hiện phép trừ a = a - 1 được kết quả a = 0

Assignment 4

a. i < j

```
.text
init:   addi    $s1, $zero, 1       # i = 1
        addi    $s2, $zero, 2       # j = 2
if:     slt     $t0, $s1, $s2       # i < j
        beq     $t0, $zero, else    # branch to else if i >= j
        addi    $t1, $t1, 1         # then part: x = x + 1
        addi    $t3, $zero, 1       # z = 1
        j       endif               # skip “else” part
else:   addi    $t2, $t2, -1        # begin else part: y = y - 1
        add     $t3, $t3, $t3       # z = 2 * z
endif:
```

b. i >= j

```
.text
init:   addi    $s1, $zero, 1       # i = 1
        addi    $s2, $zero, 2       # j = 2
if:     slt     $t0, $s1, $s2       # i < j
        bne     $t0, $zero, else    # branch to else if i < j
        addi    $t1, $t1, 1         # then part: x = x + 1
        addi    $t3, $zero, 1       # z = 1
        j       endif               # skip “else” part
else:   addi    $t2, $t2, -1        # begin else part: y = y - 1
        add     $t3, $t3, $t3       # z = 2 * z
endif:
```

c. i + j <= 0

```
.text
init:   addi    $s1, $zero, 1       # i = 1
```

```
        addi    $s2, $zero, 2       # j = 2
        add     $s3, $s1, $s2       # $s3 = i + j
if:     slt     $t0, $s1, $s3       # 0 < i + j
        bne     $t0, $zero, else    # branch to else if 0 < i + j
        addi    $t1, $t1, 1         # then part: x = x + 1
        addi    $t3, $zero, 1       # z = 1
        j       endif               # skip “else” part
else:   addi    $t2, $t2, -1        # begin else part: y = y - 1
        add     $t3, $t3, $t3       # z = 2 * z
endif:
```

d. l + j > m + n

```
.text
init:   addi    $s1, $zero, 1       # i = 1
        addi    $s2, $zero, 2       # j = 2
        addi    $s3, $zero, 3       # m = 3
        addi    $s4, $zero, 4       # n = 4
        add     $s5, $s1, $s2       # $s5 = i + j
        add     $s6, $s3, $s4       # $s6 = m + n
if:     slt     $t0, $s6, $s5       # m + n < i + j
        beq     $t0, $zero, else    # branch to else if m + n >= i + j
        addi    $t1, $t1, 1         # then part: x = x + 1
        addi    $t3, $zero, 1       # z = 1
        j       endif               # skip “else” part
else:   addi    $t2, $t2, -1        # begin else part: y = y - 1
        add     $t3, $t3, $t3       # z = 2 * z
endif:
```

Assignment 5

a. i >= j

```
.data
A:      .word       1, 2, 3, 4, 5, 6, 7, 8
.text
init:   addi        $s1, $zero, 0       # i = 0
        la          $s2, A
        addi        $s3, $zero, 7       # n = 7
        addi        $s4, $zero, 1       # step = 1
        addi        $s5, $zero, 0       # sum = 0
loop:   slt         $t2, $s3, $s1       # $t2 = n < i
        bne         $t2, $zero, endloop
        add         $t1, $s1, $s1       # $t1 = 2 * $s1
        add         $t1, $t1, $t1       # $t1 = 4 * $s1
        add         $t1, $t1, $s2       # $t1 store the address of A[i]
        lw          $t0, 0($t1)         # load value of A[i] in $t0
        add         $s5, $s5, $t0       # sum = sum + A[i]
        add         $s1, $s1, $s4       # i = i + step
        j           loop                # goto loop
endloop:
```

b. i + j <= 0

```
.data
A:      .word       1, 2, 3, 4, 5, 6, 7, 8, -37, 9
.text
init:   addi        $s1, $zero, 0       # i = 0
        la          $s2, A
        addi        $s3, $zero, 7       # n = 7
        addi        $s4, $zero, 1       # step = 1
        addi        $s5, $zero, 0       # sum = 0
loop:   slt         $t2, $s5, $zero     # $t2 = sum < 0
        bne         $t2, $zero, endloop
```

```
        add     $t1, $s1, $s1       # $t1 = 2 * $s1
        add     $t1, $t1, $t1       # $t1 = 4 * $s1
        add     $t1, $t1, $s2       # $t1 store the address of A[i]
        lw      $t0, 0($t1)         # load value of A[i] in $t0
        add     $s5, $s5, $t0       # sum = sum + A[i]
        add     $s1, $s1, $s4       # i = i + step
        j       loop                # goto loop
endloop:
```

c. l + j > m + n

```
.data
A:      .word   1, 2, 3, 4, 5, 6, 7, 8, 0, 9
.text
init:   addi    $s1, $zero, 0       # i = 0
        la      $s2, A
        addi    $s3, $zero, 7       # n = 7
        addi    $s4, $zero, 1       # step = 1
        addi    $s5, $zero, 0       # sum = 0
loop:   slt     $t2, $s1, $s3       # $t2 = i < n ? 1 : 0
        beq     $t2, $zero, endloop
        add     $t1, $s1, $s1       # $t1 = 2 * $s1
        add     $t1, $t1, $t1       # $t1 = 4 * $s1
        add     $t1, $t1, $s2       # $t1 store the address of A[i]
        lw      $t0, 0($t1)         # load value of A[i] in $t0
        beq     $t0, $zero, endloop # endloop if A[i] == 0
        add     $s5, $s5, $t0       # sum = sum + A[i]
        add     $s1, $s1, $s4       # i = i + step
        j       loop                # goto loop
endloop:
```

Assignment 6

Code:

```
.data
A:      .word       -2, 3, -5, -9, 4, -1, 0
.text
init:   addi        $s1, $zero, 0           # i = 0
        la          $s2, A
        addi        $s3, $zero, 7           # n = 7
        addi        $s4, $zero, 1           # step = 1
        addi        $s5, $zero, 0           # sum = 0
        lw          $s6, 0($s2)             # load value of A[0] in $s6
        abs         $s6, $s6                # max = abs(A[0])
        addi        $s7, $s7, 0             # max_id = 0
loop:   slt         $t2, $s1, $s3           # $t2 = i < n ? 1 : 0
        beq         $t2, $zero, endloop
        add         $t1, $s1, $s1           # $t1 = 2 * $s1
        add         $t1, $t1, $t1           # $t1 = 4 * $s1
        add         $t1, $t1, $s2           # $t1 store the address of A[i]
        lw          $t0, 0($t1)             # load value of A[i] in $t0
        abs         $t0, $t0                # abs(A[i])
        slt         $t6, $s6, $t0           # if max < abs(A[i])
        bne         $t6, $zero, update      # update if True
        add         $s1, $s1, $s4           # i = i + step
        j           loop                    # goto loop
update:
        add         $s6, $zero, $t0         # update max
        add         $s7, $zero, $s1         # update max_id
        add         $s1, $s1, $s4           # i = i + step
        j           loop
```

endloop: