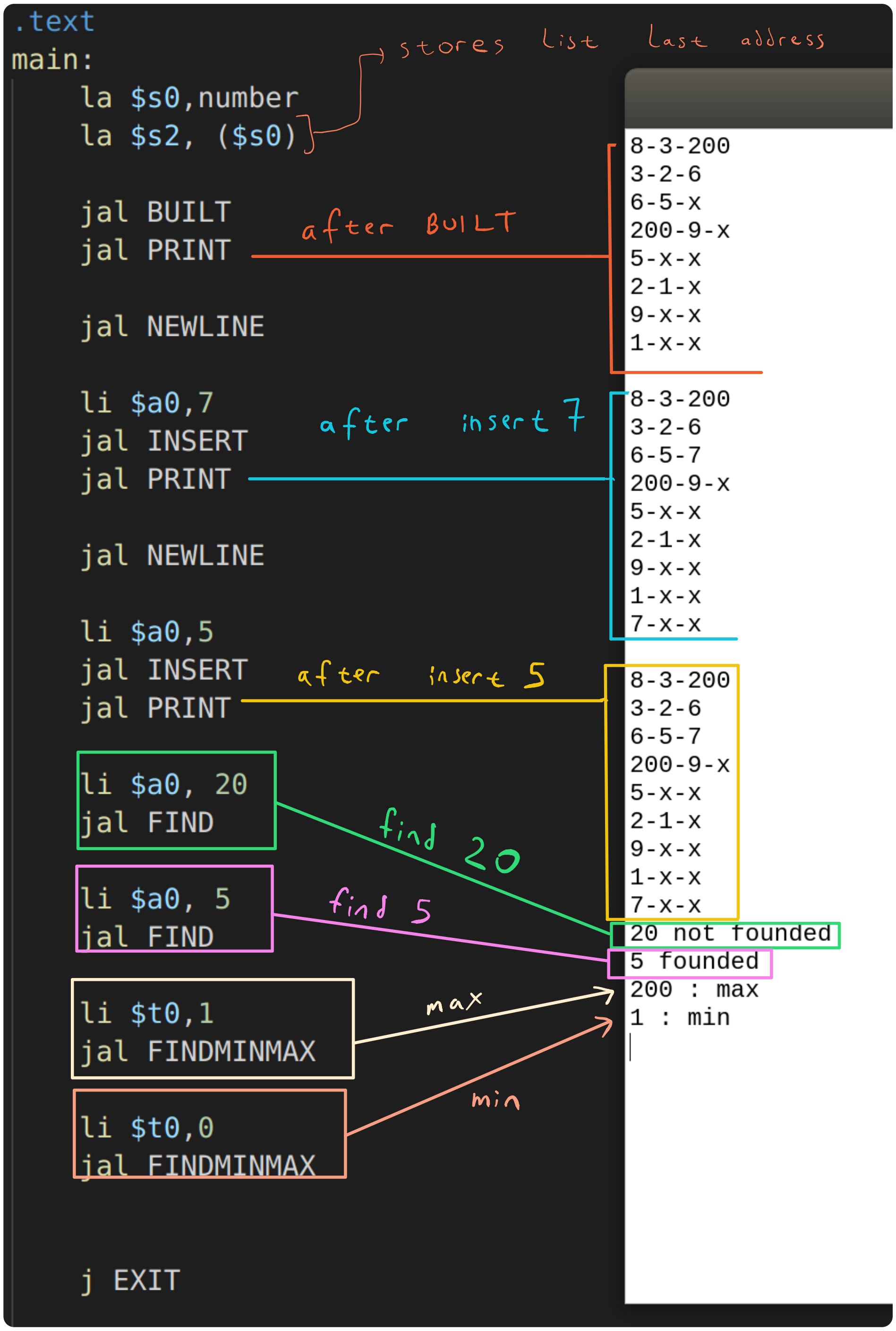
CENG311

REPORT OF
PROGRAMMING ASSIGNMENT 1

ELMAN HAMDİ
240201036

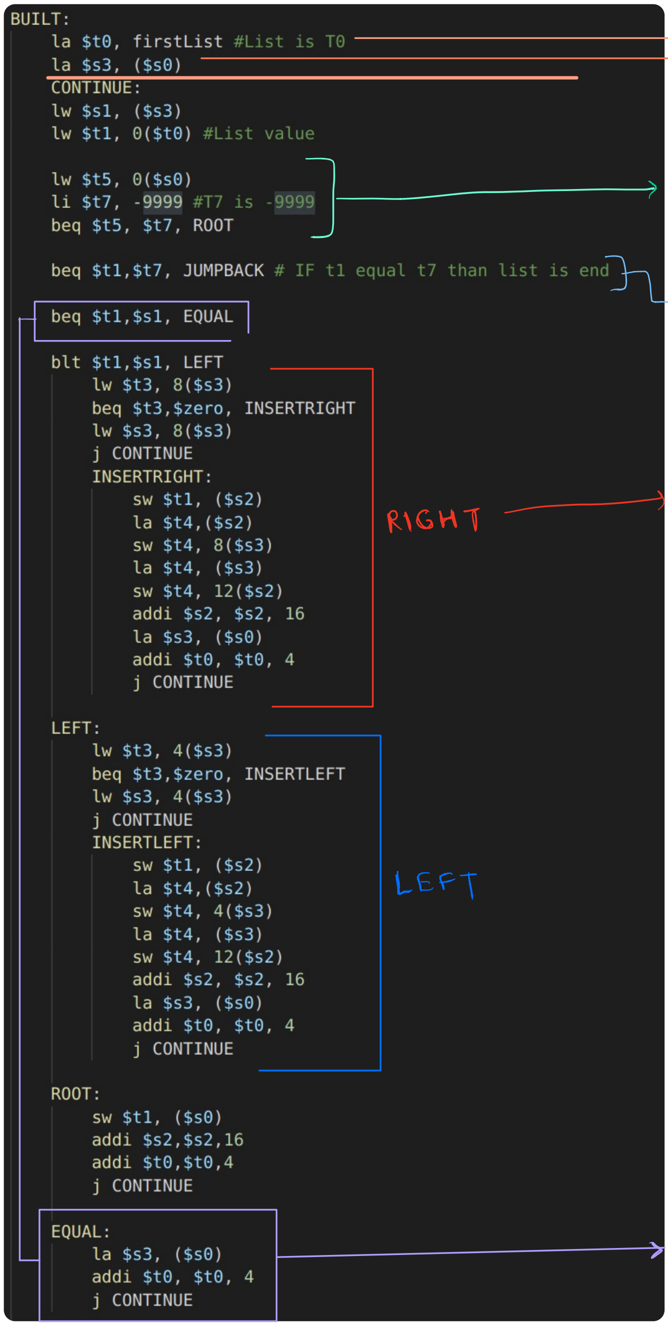
TEST PROGRAM WITH OUTPUTS



```
.text
main:
    la $s0,number
    la $s2, ($s0)

    jal BUILT
    jal PRINT

    jal NEWLINE

    li $a0,7
    jal INSERT
    jal PRINT

    jal NEWLINE

    li $a0,5
    jal INSERT
    jal PRINT

    li $a0, 20
    jal FIND

    li $a0, 5
    jal FIND

    li $t0,1
    jal FINDMINMAX

    li $t0,0
    jal FINDMINMAX


    j EXIT
```

```
8-3-200
3-2-6
6-5-x
200-9-x
5-x-x
2-1-x
9-x-x
1-x-x

8-3-200
3-2-6
6-5-7
200-9-x
5-x-x
2-1-x
9-x-x
1-x-x
7-x-x

8-3-200
3-2-6
6-5-7
200-9-x
5-x-x
2-1-x
9-x-x
1-x-x
7-x-x
20 not founded
5 founded
200 : max
1 : min
```

BUILT



```
BUILT:
    la $t0, firstList #List is T0
    la $s3, ($s0)
    CONTINUE:
    lw $s1, ($s3)
    lw $t1, 0($t0) #List value

    lw $t5, 0($s0)
    li $t7, -9999 #T7 is -9999
    beq $t5, $t7, ROOT

    beq $t1,$t7, JUMPBACK # IF t1 equal t7 than list is end

    beq $t1,$s1, EQUAL

    blt $t1,$s1, LEFT
        lw $t3, 8($s3)
        beq $t3,$zero, INSERTRIGHT
        lw $s3, 8($s3)
        j CONTINUE
        INSERTRIGHT:
            sw $t1, ($s2)
            la $t4,($s2)
            sw $t4, 8($s3)
            la $t4, ($s3)
            sw $t4, 12($s2)
            addi $s2, $s2, 16
            la $s3, ($s0)
            addi $t0, $t0, 4
            j CONTINUE

    LEFT:
        lw $t3, 4($s3)
        beq $t3,$zero, INSERTLEFT
        lw $s3, 4($s3)
        j CONTINUE
        INSERTLEFT:
            sw $t1, ($s2)
            la $t4,($s2)
            sw $t4, 4($s3)
            la $t4, ($s3)
            sw $t4, 12($s2)
            addi $s2, $s2, 16
            la $s3, ($s0)
            addi $t0, $t0, 4
            j CONTINUE

    ROOT:
        sw $t1, ($s0)
        addi $s2,$s2,16
        addi $t0,$t0,4
        j CONTINUE

    EQUAL:
        la $s3, ($s0)
        addi $t0, $t0, 4
        j CONTINUE
```

8, 7, 3 . . . . . -9999

INDEX OF BST (POINTER)

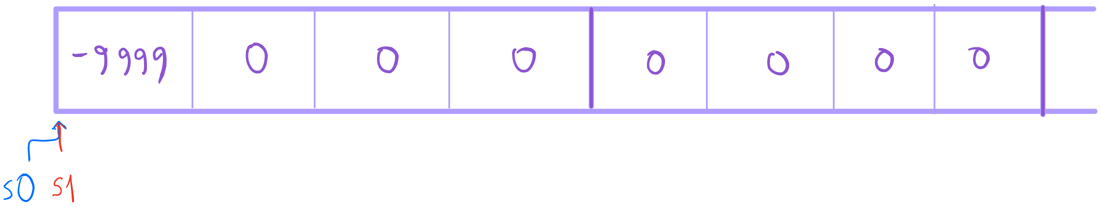
checks first value in the BST is equal -9999.
If it's equal than jump to ROOT and create first node in the BST.

$t1 is next value in list.
If equals -9999, then list ended.

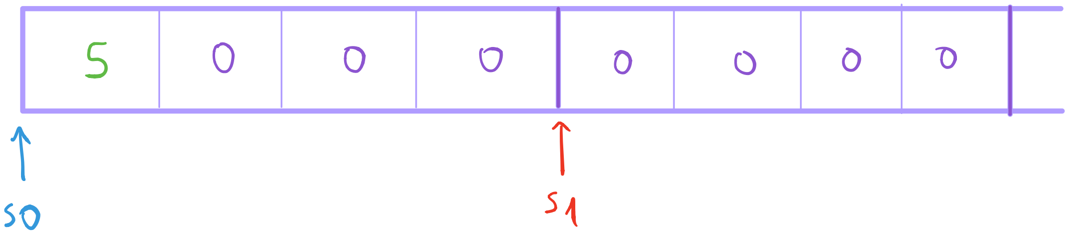
If greater than the root ($s3) than check left side,
If it is 0 than add node,
If not make. the node,
new root node ($s3)

Pass the list value.

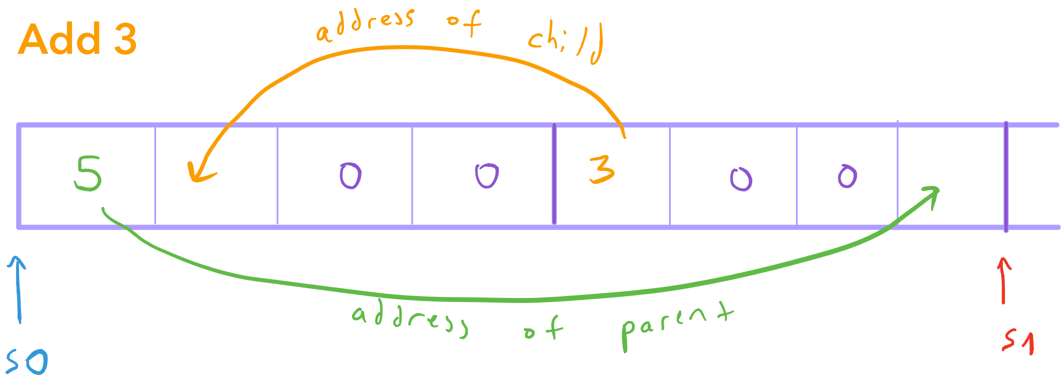
HOW WORKS BUILT AND INSERT ?



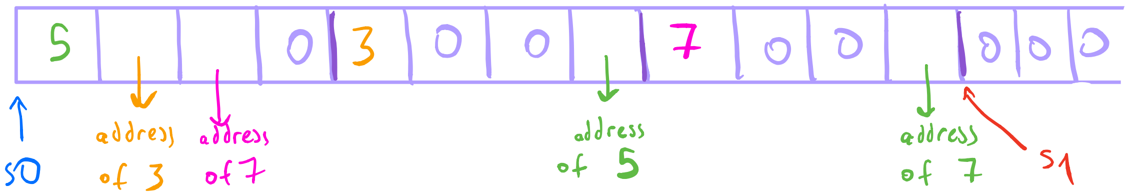
Add 5



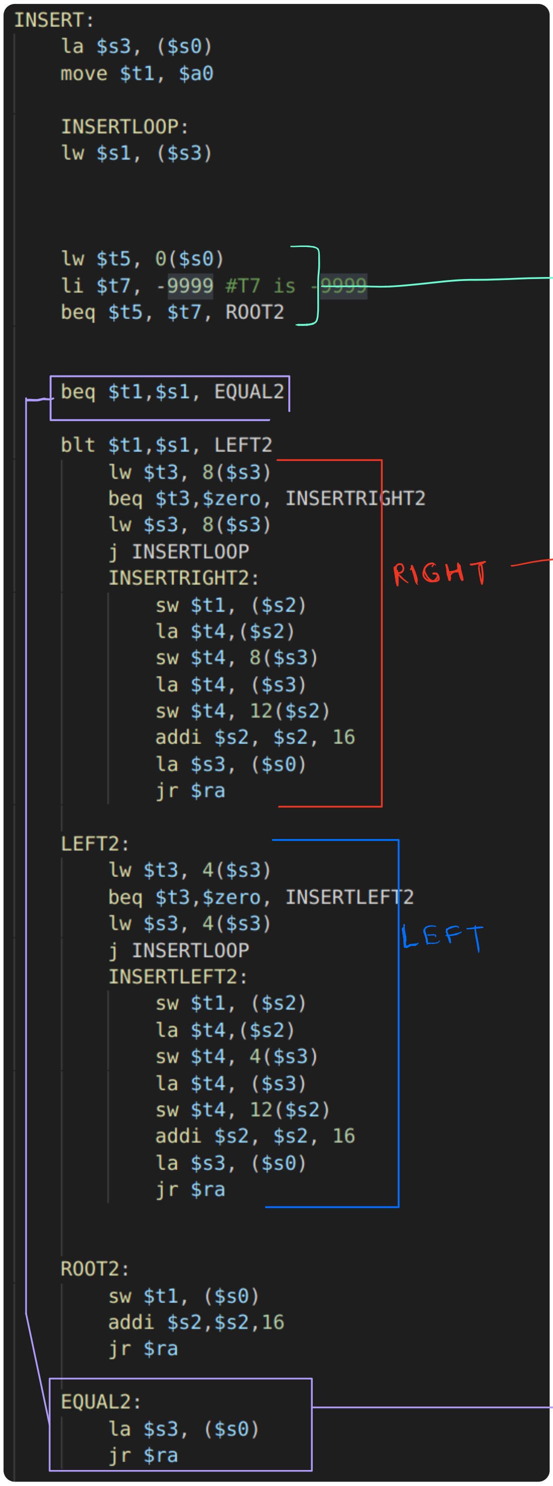
Add 3



Add 7



INSERT



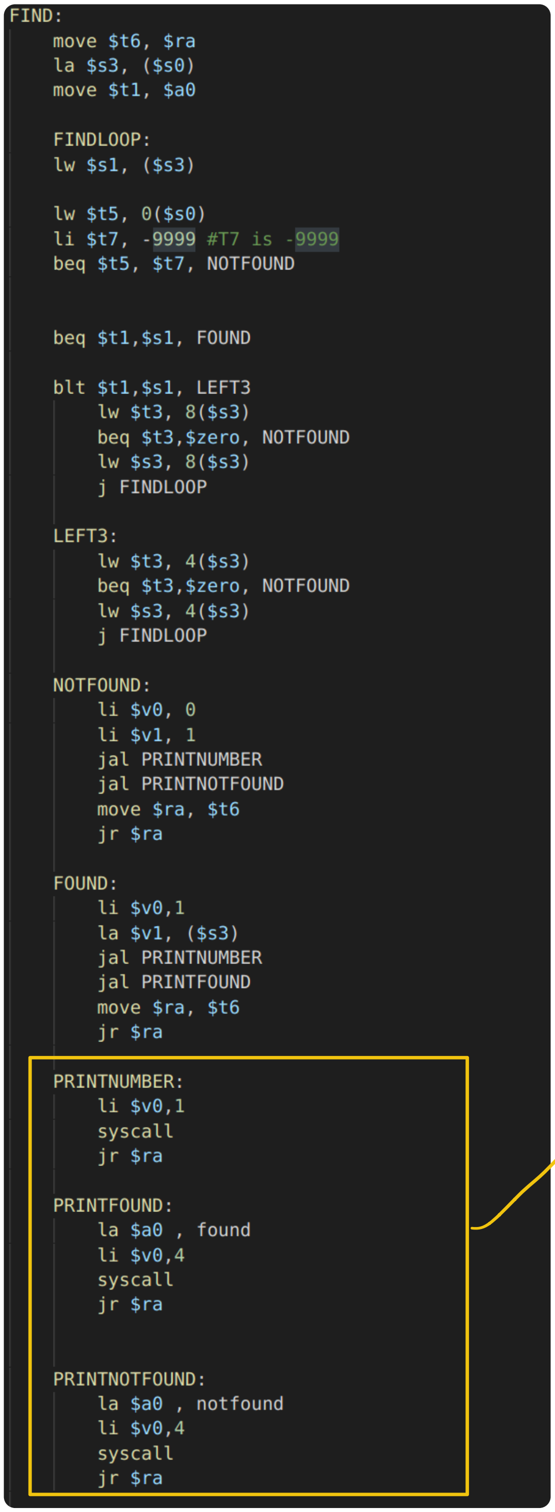
```
INSERT:
    la $s3, ($s0)
    move $t1, $a0

    INSERTLOOP:
    lw $s1, ($s3)


    lw $t5, 0($s0)
    li $t7, -9999 #T7 is -9999
    beq $t5, $t7, ROOT2


    beq $t1,$s1, EQUAL2

    blt $t1,$s1, LEFT2
        lw $t3, 8($s3)
        beq $t3,$zero, INSERTRIGHT2
        lw $s3, 8($s3)
        j INSERTLOOP
        INSERTRIGHT2:
            sw $t1, ($s2)
            la $t4,($s2)
            sw $t4, 8($s3)
            la $t4, ($s3)
            sw $t4, 12($s2)
            addi $s2, $s2, 16
            la $s3, ($s0)
            jr $ra

    LEFT2:
        lw $t3, 4($s3)
        beq $t3,$zero, INSERTLEFT2
        lw $s3, 4($s3)
        j INSERTLOOP
        INSERTLEFT2:
            sw $t1, ($s2)
            la $t4,($s2)
            sw $t4, 4($s3)
            la $t4, ($s3)
            sw $t4, 12($s2)
            addi $s2, $s2, 16
            la $s3, ($s0)
            jr $ra


    ROOT2:
        sw $t1, ($s0)
        addi $s2,$s2,16
        jr $ra

    EQUAL2:
        la $s3, ($s0)
        jr $ra
```

Checks first value in the BST
is equal -9999.
If it's equal than jump to ROOT
and create first node in the BST.

If greater than the
root ($s3) than
check left side,
If it is 0 than add node,
If not make. the node,
new root node ($s3)

If value which
want to insert in BST is
already in BST then do
nothing,

FIND



```
FIND:
    move $t6, $ra
    la $s3, ($s0)
    move $t1, $a0

    FINDLOOP:
    lw $s1, ($s3)

    lw $t5, 0($s0)
    li $t7, -9999 #T7 is -9999
    beq $t5, $t7, NOTFOUND


    beq $t1,$s1, FOUND

    blt $t1,$s1, LEFT3
        lw $t3, 8($s3)
        beq $t3,$zero, NOTFOUND
        lw $s3, 8($s3)
        j FINDLOOP

    LEFT3:
        lw $t3, 4($s3)
        beq $t3,$zero, NOTFOUND
        lw $s3, 4($s3)
        j FINDLOOP

    NOTFOUND:
        li $v0, 0
        li $v1, 1
        jal PRINTNUMBER
        jal PRINTNOTFOUND
        move $ra, $t6
        jr $ra

    FOUND:
        li $v0,1
        la $v1, ($s3)
        jal PRINTNUMBER
        jal PRINTFOUND
        move $ra, $t6
        jr $ra

    PRINTNUMBER:
        li $v0,1
        syscall
        jr $ra

    PRINTFOUND:
        la $a0 , found
        li $v0,4
        syscall
        jr $ra


    PRINTNOTFOUND:
        la $a0 , notfound
        li $v0,4
        syscall
        jr $ra
```

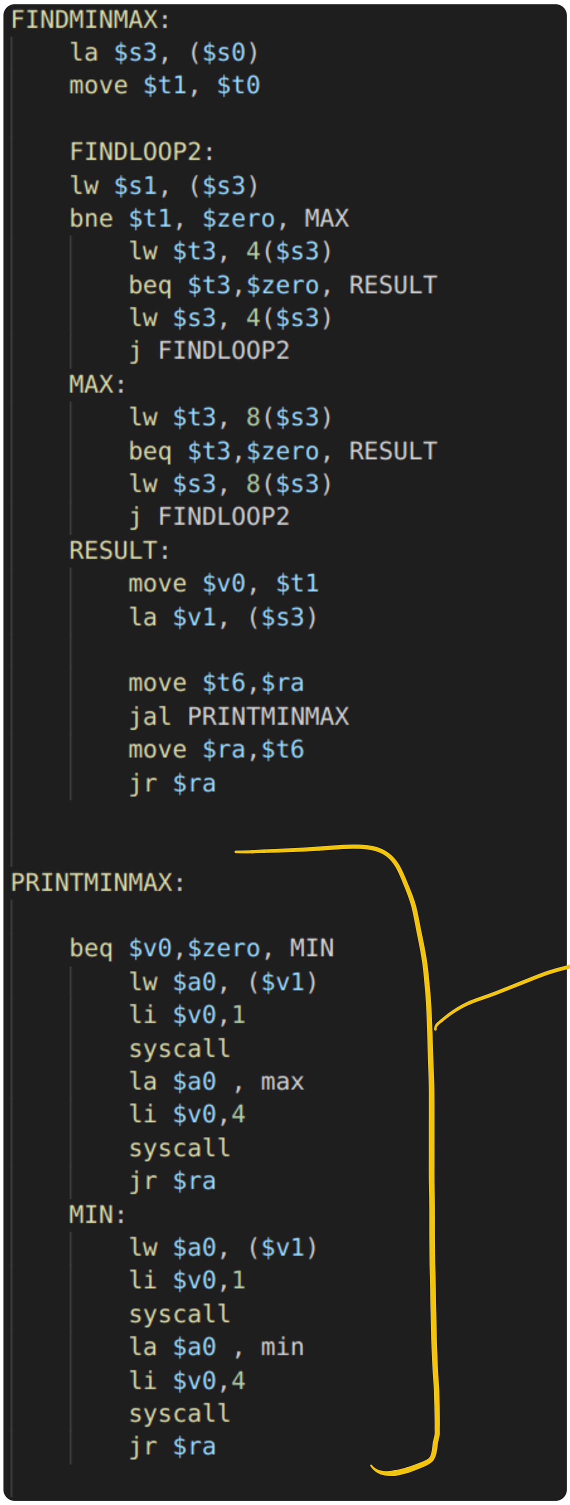
printing functions

If function found the number (in $a0) then loads 1 to $v0 and loads address of value to $v1 .

Else,

loads 0 to $v0 and loads 1 to $v1.

FINDMINMAX



```
FINDMINMAX:
    la $s3, ($s0)
    move $t1, $t0

    FINDLOOP2:
    lw $s1, ($s3)
    bne $t1, $zero, MAX
        lw $t3, 4($s3)
        beq $t3,$zero, RESULT
        lw $s3, 4($s3)
        j FINDLOOP2
    MAX:
        lw $t3, 8($s3)
        beq $t3,$zero, RESULT
        lw $s3, 8($s3)
        j FINDLOOP2
    RESULT:
        move $v0, $t1
        la $v1, ($s3)

        move $t6,$ra
        jal PRINTMINMAX
        move $ra,$t6
        jr $ra


PRINTMINMAX:

    beq $v0,$zero, MIN
        lw $a0, ($v1)
        li $v0,1
        syscall
        la $a0 , max
        li $v0,4
        syscall
        jr $ra
    MIN:
        lw $a0, ($v1)
        li $v0,1
        syscall
        la $a0 , min
        li $v0,4
        syscall
        jr $ra
```

If $t0 is 0 find minimum, and if it is 1 find maximum.

When min

Loads 0 to $v0 and loads address of min value to $v1 .

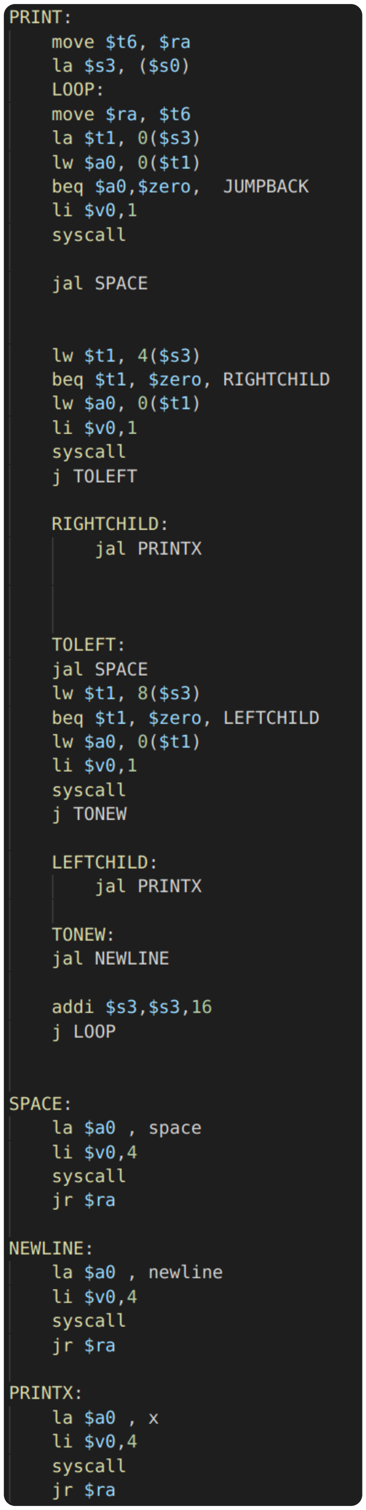
When max

loads 1 to $v0 and loads loads address of min value to $v1.

print function

for findminmax

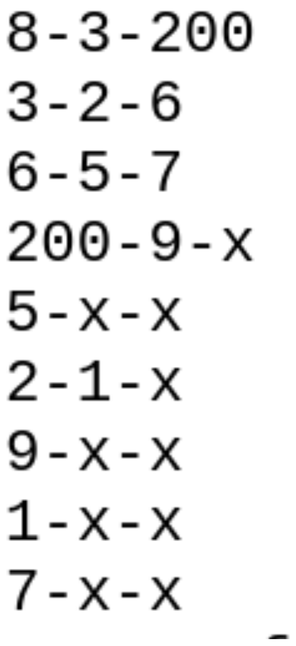
PRINT



```
PRINT:
    move $t6, $ra
    la $s3, ($s0)
    LOOP:
    move $ra, $t6
    la $t1, 0($s3)
    lw $a0, 0($t1)
    beq $a0,$zero,  JUMPBACK
    li $v0,1
    syscall

    jal SPACE


    lw $t1, 4($s3)
    beq $t1, $zero, RIGHTCHILD
    lw $a0, 0($t1)
    li $v0,1
    syscall
    j TOLEFT

    RIGHTCHILD:
        jal PRINTX



    TOLEFT:
    jal SPACE
    lw $t1, 8($s3)
    beq $t1, $zero, LEFTCHILD
    lw $a0, 0($t1)
    li $v0,1
    syscall
    j TONEW

    LEFTCHILD:
        jal PRINTX

    TONEW:
    jal NEWLINE

    addi $s3,$s3,16
    j LOOP


SPACE:
    la $a0 , space
    li $v0,4
    syscall
    jr $ra

NEWLINE:
    la $a0 , newline
    li $v0,4
    syscall
    jr $ra

PRINTX:
    la $a0 , x
    li $v0,4
    syscall
    jr $ra
```

It is not working as desired.

Works like that



```
8-3-200
3-2-6
6-5-7
200-9-x
5-x-x
2-1-x
9-x-x
1-x-x
7-x-x
```

(Show all nodes and childs)