POST LAB 7

NAME - RAJENDRA SINGH
ROLL NO. 111601017

Question(1) Compare two numbers A=0xFFFFFFFF, B=0x0000000F.

If A<B it should set the $t1 register, else it should reset $t1 register for the below cases:
(a) When A & B are considered as signed numbers.
(b) When A & B are considered as unsigned numbers.

The program should print a message ‘A is less than B’ if A<B, else if A>B, print ‘B is less than A’.

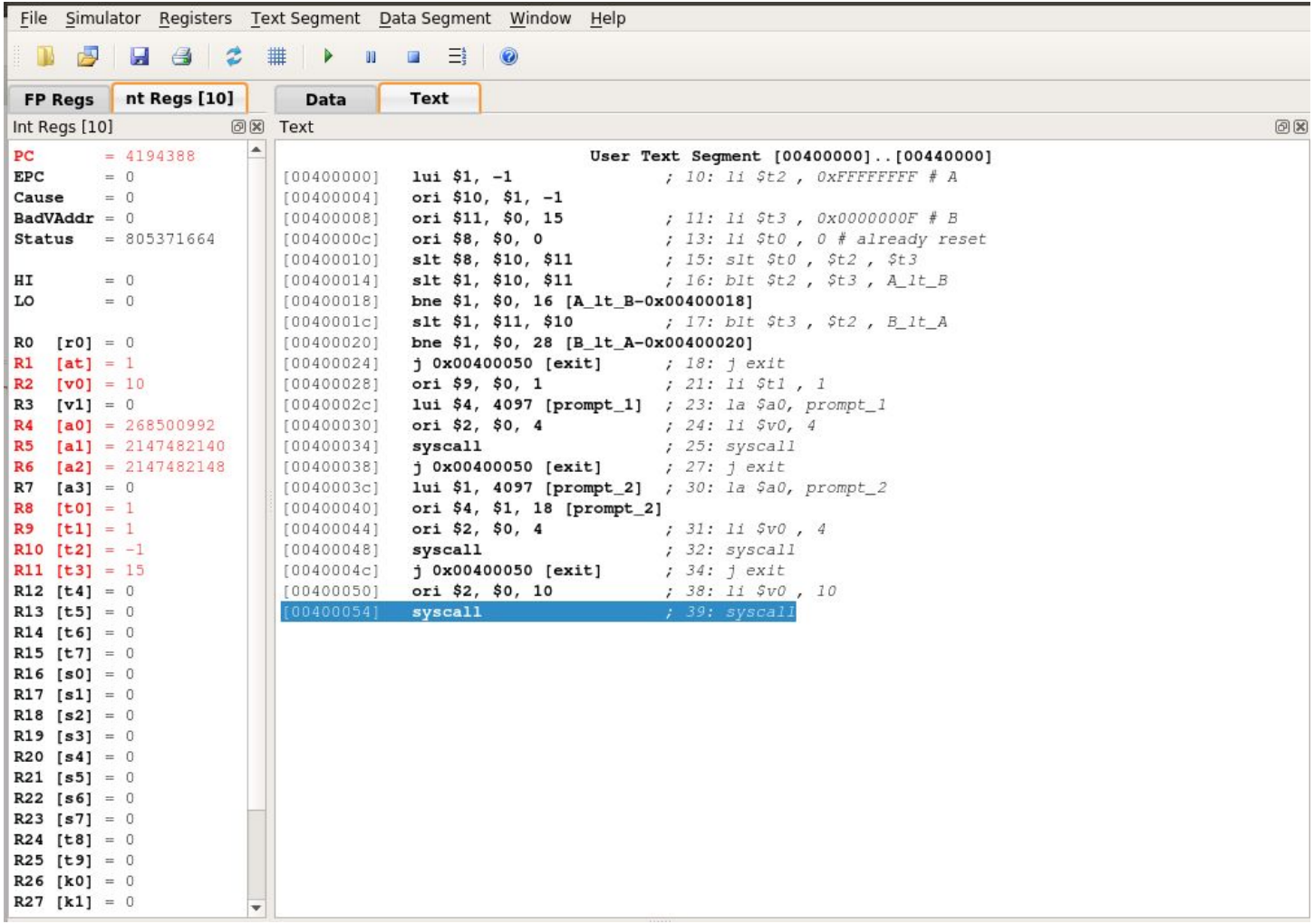
What did you observe in (a) & (b) and why?
Use compare and branch instructions.

CODE :

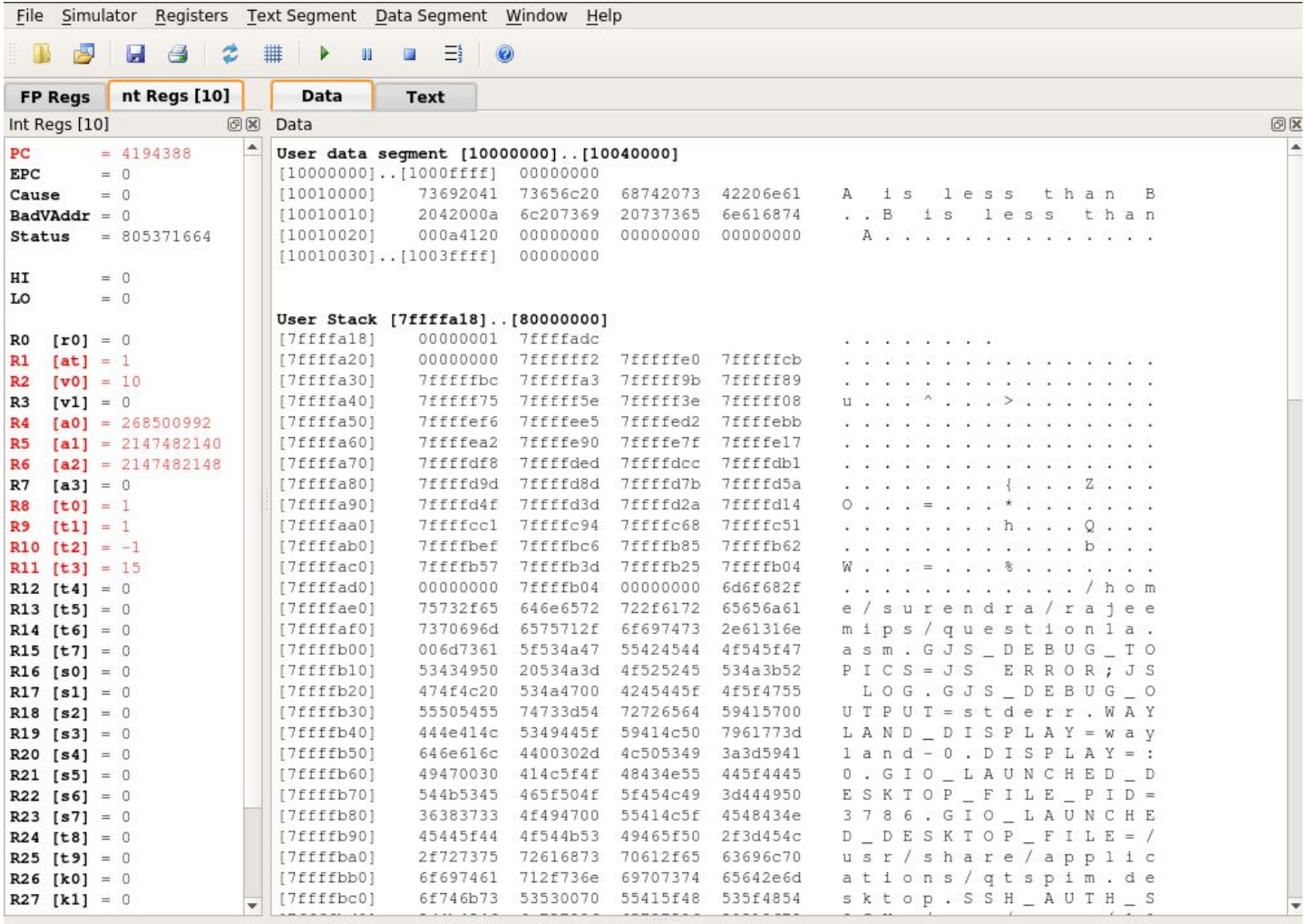
```
.data
        prompt_1 : .asciiz "A is less than B\n"
        prompt_2 : .asciiz "B is less than A\n"


.text

.globl main
        main:
                li $t3 , 0xFFFFFFFF  # A
                li $t2 , 0x0000000F  # B

                blt $t2 , $t3 , A_lt_B
                li $t0 , 0
                blt $t3 , $t2 , B_lt_A
                j exit

        A_lt_B:
                li $t1 , 1

                la $a0, prompt_1
                li $v0, 4
                syscall
                j exit
```

```
B_lt_A:
        la $a0, prompt_2
        li $v0 , 4
        syscall

        j exit


exit:
        li $v0 , 10
        syscall
```

TEXT :


File Simulator Registers Text Segment Data Segment Window Help
FP Regs
nt Regs [10]
Data
Text
Int Regs [10]
PC = 4194388
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664
HI = 0
LO = 0
R0 [r0] = 0
R1 [at] = 1
R2 [v0] = 10
R3 [v1] = 0
R4 [a0] = 268500992
R5 [a1] = 2147482140
R6 [a2] = 2147482148
R7 [a3] = 0
R8 [t0] = 1
R9 [t1] = 1
R10 [t2] = -1
R11 [t3] = 15
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
Text
User Text Segment [00400000]..[00440000]
[00400000] lui $1, -1 ; 10: li $t2 , 0xFFFFFFFF # A
[00400004] ori $10, $1, -1
[00400008] ori $11, $0, 15 ; 11: li $t3 , 0x0000000F # B
[0040000c] ori $8, $0, 0 ; 13: li $t0 , 0 # already reset
[00400010] slt $8, $10, $11 ; 15: slt $t0 , $t2 , $t3
[00400014] slt $1, $10, $11 ; 16: blt $t2 , $t3 , A_lt_B
[00400018] bne $1, $0, 16 [A_lt_B-0x00400018]
[0040001c] slt $1, $11, $10 ; 17: blt $t3 , $t2 , B_lt_A
[00400020] bne $1, $0, 28 [B_lt_A-0x00400020]
[00400024] j 0x00400050 [exit] ; 18: j exit
[00400028] ori $9, $0, 1 ; 21: li $t1 , 1
[0040002c] lui $4, 4097 [prompt_1] ; 23: la $a0, prompt_1
[00400030] ori $2, $0, 4 ; 24: li $v0, 4
[00400034] syscall ; 25: syscall
[00400038] j 0x00400050 [exit] ; 27: j exit
[0040003c] lui $1, 4097 [prompt_2] ; 30: la $a0, prompt_2
[00400040] ori $4, $1, 18 [prompt_2]
[00400044] ori $2, $0, 4 ; 31: li $v0 , 4
[00400048] syscall ; 32: syscall
[0040004c] j 0x00400050 [exit] ; 34: j exit
[00400050] ori $2, $0, 10 ; 38: li $v0 , 10
[00400054] syscall ; 39: syscall

DATA :

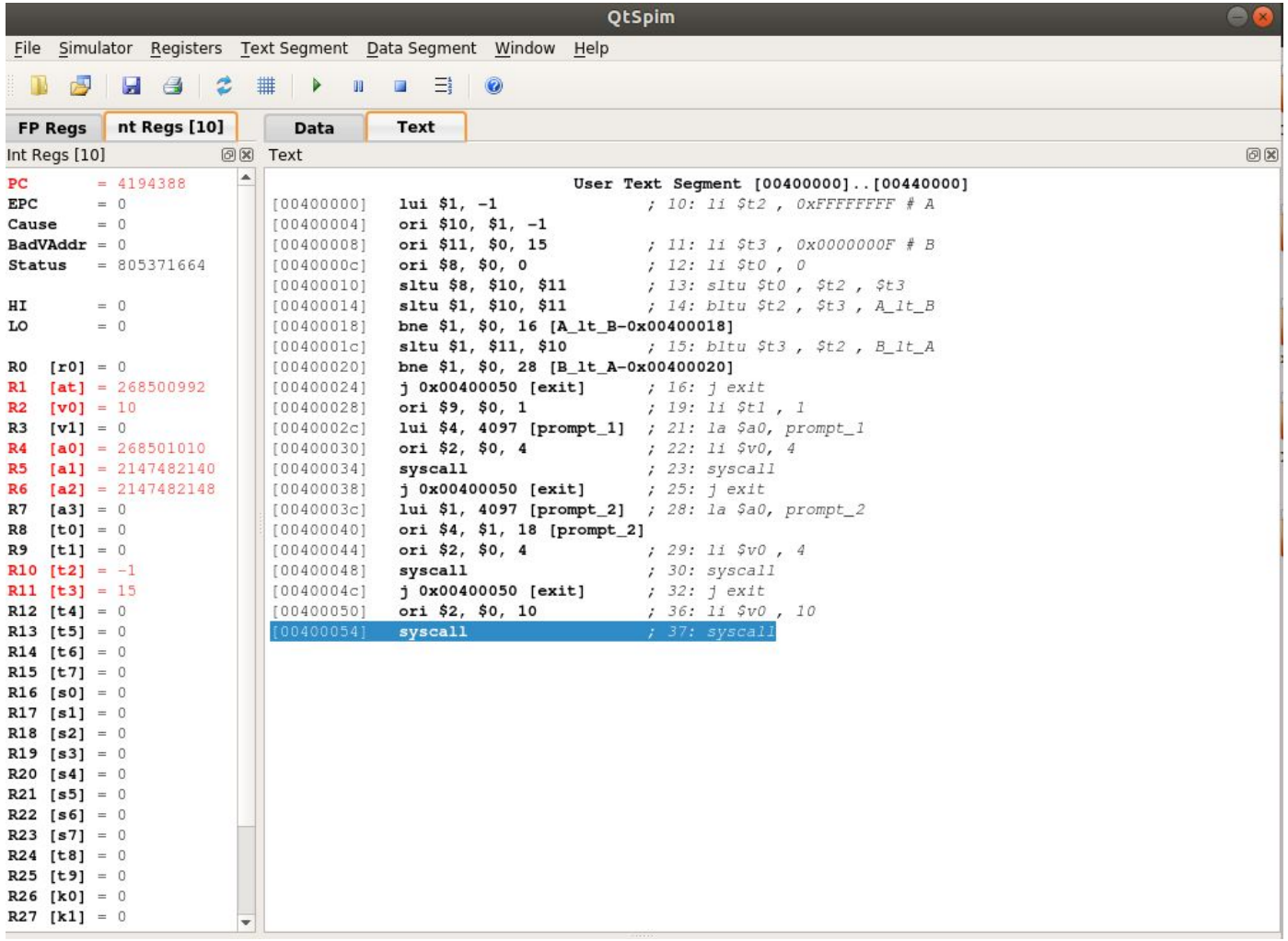

File Simulator Registers Text Segment Data Segment Window Help
FP Regs
nt Regs [10]
Data
Text
Int Regs [10]
PC = 4194388
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664
HI = 0
LO = 0
R0 [r0] = 0
R1 [at] = 1
R2 [v0] = 10
R3 [v1] = 0
R4 [a0] = 268500992
R5 [a1] = 2147482140
R6 [a2] = 2147482148
R7 [a3] = 0
R8 [t0] = 1
R9 [t1] = 1
R10 [t2] = -1
R11 [t3] = 15
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
Data
User data segment [10000000]..[10040000]
[10000000]..[1000ffff] 00000000
[10010000] 73692041 73656c20 68742073 42206e61 A is less than B
[10010010] 2042000a 6c207369 20737365 6e616874 . . B is less than
[10010020] 000a4120 00000000 00000000 00000000 A . . . . . .
[10010030]..[1003ffff] 00000000
User Stack [7ffffa18]..[80000000]
[7ffffa18] 00000001 7ffffadc
[7ffffa20] 00000000 7fffff2 7fffffe0 7fffffcb
[7ffffa30] 7fffffbc 7fffffa3 7fffff9b 7fffff89
[7ffffa40] 7fffff75 7fffff5e 7fffff3e 7fffff08 u . . . ^ . . . > . . . . .
[7ffffa50] 7ffffef6 7ffffee5 7ffffed2 7ffffebb
[7ffffa60] 7ffffea2 7ffffe90 7ffffe7f 7ffffe17
[7ffffa70] 7ffffdf8 7ffffded 7ffffdcc 7ffffdb1
[7ffffa80] 7ffffd9d 7ffffd8d 7ffffd7b 7ffffd5a . . . . . { . . . Z . . .
[7ffffa90] 7ffffd4f 7ffffd3d 7ffffd2a 7ffffd14 O . . . = . . . * . . . . .
[7ffffaa0] 7ffffcc1 7ffffc94 7ffffc68 7ffffc51 . . . . . h . . . Q . . .
[7ffffab0] 7ffffbef 7ffffbc6 7ffffb85 7ffffb62 . . . . . b . . .
[7ffffac0] 7ffffb57 7ffffb3d 7ffffb25 7ffffb04 W . . . = . . . % . . . . .
[7ffffad0] 00000000 7ffffb04 00000000 6d6f682f . . . . . / h o m
[7ffffae0] 75732f65 646e6572 722f6172 65656a61 e / s u r e n d r a / r a j e e
[7ffffaf0] 7370696d 6575712f 6f697473 2e61316e m i p s / q u e s t i o n l a .
[7ffffb00] 006d7361 5f534a47 55424544 4f545f47 a s m . G J S _ D E B U G _ T O
[7ffffb10] 53434950 20534a3d 4f525245 534a3b52 P I C S = J S E R R O R ; J S
[7ffffb20] 474f4c20 534a4700 4245445f 4f5f4755 L O G . G J S _ D E B U G _ O
[7ffffb30] 55505455 74733d54 72726564 59415700 U T P U T = s t d e r r . W A Y
[7ffffb40] 444e414c 5349445f 59414c50 7961773d L A N D _ D I S P L A Y = w a y
[7ffffb50] 646e616c 4400302d 4c505349 3a3d5941 l a n d - 0 . D I S P L A Y = :
[7ffffb60] 49470030 414c5f4f 48434e55 445f4445 0 . G I O _ L A U N C H E D _ D
[7ffffb70] 544b5345 465f504f 5f454c49 3d444950 E S K T O P _ F I L E _ P I D =
[7ffffb80] 36383733 4f494700 55414c5f 4548434e 3 7 8 6 . G I O _ L A U N C H E
[7ffffb90] 45445f44 4f544b53 49465f50 2f3d454c D _ D E S K T O P _ F I L E = /
[7ffffba0] 2f727375 72616873 70612f65 63696c70 u s r / s h a r e / a p p l i c
[7ffffbb0] 6f697461 712f736e 69707374 65642e6d a t i o n s / q t s p i m . d e
[7ffffbc0] 6f746b73 53530070 55415f48 535f4854 s k t o p . S S H _ A U T H _ S

CODE :

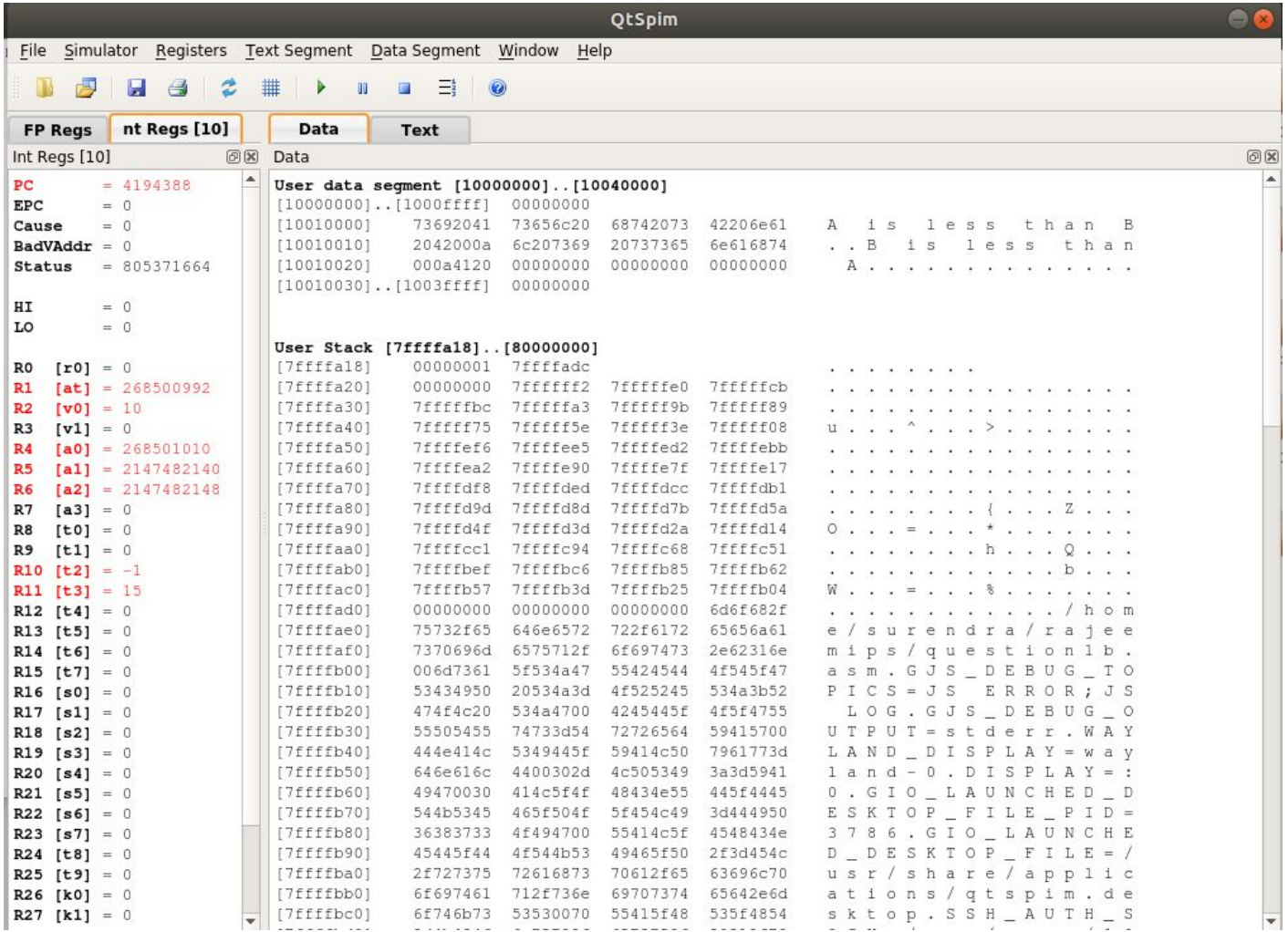
```
.data
        prompt_1 : .asciiz "A is less than B\n"
        prompt_2 : .asciiz "B is less than A\n"


.text

.globl main
        main:
                li $t2 , 0xFFFFFFFF  # A
                li $t3 , 0x0000000F  # B
                li $t0 , 0
                sltu $t0 , $t2 , $t3
                bltu $t2 , $t3 , A_lt_B
                bltu $t3 , $t2 , B_lt_A
                j exit

        A_lt_B:
                li $t1 , 1

                la $a0, prompt_1
                li $v0, 4
                syscall
                j exit
```

```
B_lt_A:
        la $a0, prompt_2
        li $v0 , 4
        syscall

        j exit


exit:
        li $v0 , 10
        syscall
```

TEXT :


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
nt Regs [10]
Data
Text
Int Regs [10]
PC = 4194388
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664
HI = 0
LO = 0
R0 [r0] = 0
R1 [at] = 268500992
R2 [v0] = 10
R3 [v1] = 0
R4 [a0] = 268501010
R5 [a1] = 2147482140
R6 [a2] = 2147482148
R7 [a3] = 0
R8 [t0] = 0
R9 [t1] = 0
R10 [t2] = -1
R11 [t3] = 15
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
Text
User Text Segment [00400000]..[00440000]
[00400000] lui $1, -1 ; 10: li $t2 , 0xFFFFFFFF # A
[00400004] ori $10, $1, -1
[00400008] ori $11, $0, 15 ; 11: li $t3 , 0x0000000F # B
[0040000c] ori $8, $0, 0 ; 12: li $t0 , 0
[00400010] sltu $8, $10, $11 ; 13: sltu $t0 , $t2 , $t3
[00400014] sltu $1, $10, $11 ; 14: bltu $t2 , $t3 , A_lt_B
[00400018] bne $1, $0, 16 [A_lt_B-0x00400018]
[0040001c] sltu $1, $11, $10 ; 15: bltu $t3 , $t2 , B_lt_A
[00400020] bne $1, $0, 28 [B_lt_A-0x00400020]
[00400024] j 0x00400050 [exit] ; 16: j exit
[00400028] ori $9, $0, 1 ; 19: li $t1 , 1
[0040002c] lui $4, 4097 [prompt_1] ; 21: la $a0, prompt_1
[00400030] ori $2, $0, 4 ; 22: li $v0, 4
[00400034] syscall ; 23: syscall
[00400038] j 0x00400050 [exit] ; 25: j exit
[0040003c] lui $1, 4097 [prompt_2] ; 28: la $a0, prompt_2
[00400040] ori $4, $1, 18 [prompt_2]
[00400044] ori $2, $0, 4 ; 29: li $v0 , 4
[00400048] syscall ; 30: syscall
[0040004c] j 0x00400050 [exit] ; 32: j exit
[00400050] ori $2, $0, 10 ; 36: li $v0 , 10
[00400054] syscall ; 37: syscall

DATA :


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
nt Regs [10]
Data
Text
Int Regs [10]
PC = 4194388
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664
HI = 0
LO = 0
R0 [r0] = 0
R1 [at] = 268500992
R2 [v0] = 10
R3 [v1] = 0
R4 [a0] = 268501010
R5 [a1] = 2147482140
R6 [a2] = 2147482148
R7 [a3] = 0
R8 [t0] = 0
R9 [t1] = 0
R10 [t2] = -1
R11 [t3] = 15
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
Data
User data segment [10000000]..[10040000]
[10000000]..[1000ffff] 00000000
[10010000] 73692041 73656c20 68742073 42206e61 A is less than B
[10010010] 2042000a 6c207369 20737365 6e616874 . . B is less than
[10010020] 000a4120 00000000 00000000 00000000 A . . . . . .
[10010030]..[1003ffff] 00000000
User Stack [7ffffa18]..[80000000]
[7ffffa18] 00000001 7ffffadc
[7ffffa20] 00000000 7fffffe2 7fffffe0 7fffffcb
[7ffffa30] 7fffffbc 7fffffa3 7fffff9b 7fffff89
[7ffffa40] 7fffff75 7fffff5e 7fffff3e 7fffff08 u . . . ^ . . . > . . . . .
[7ffffa50] 7ffffef6 7ffffee5 7ffffed2 7ffffebb
[7ffffa60] 7ffffea2 7ffffe90 7ffffe7f 7ffffe17
[7ffffa70] 7ffffdf8 7ffffded 7ffffdcc 7ffffdb1
[7ffffa80] 7ffffd9d 7ffffd8d 7ffffd7b 7ffffd5a { . . . Z . . .
[7ffffa90] 7ffffd4f 7ffffd3d 7ffffd2a 7ffffd14 O . . . = . . . * . . . . .
[7ffffaa0] 7ffffcc1 7ffffc94 7ffffc68 7ffffc51 h . . . Q . . .
[7ffffab0] 7ffffbef 7ffffbc6 7ffffb85 7ffffb62 b . . .
[7ffffac0] 7ffffb57 7ffffb3d 7ffffb25 7ffffb04 W . . . = . . . % . . . .
[7ffffad0] 00000000 00000000 00000000 6d6f682f / h o m
[7ffffae0] 75732f65 646e6572 722f6172 65656a61 e / s u r e n d r a / r a j e e
[7ffffaf0] 7370696d 6575712f 6f697473 2e62316e m i p s / q u e s t i o n 1 b .
[7ffffb00] 006d7361 5f534a47 55424544 4f545f47 a s m . G J S _ D E B U G _ T O
[7ffffb10] 53434950 20534a3d 4f525245 534a3b52 P I C S = J S E R R O R ; J S
[7ffffb20] 474f4c20 534a4700 4245445f 4f5f4755 L O G . G J S _ D E B U G _ O
[7ffffb30] 55505455 74733d54 72726564 59415700 U T P U T = s t d e r r . W A Y
[7ffffb40] 444e414c 5349445f 59414c50 7961773d L A N D _ D I S P L A Y = w a y
[7ffffb50] 646e616c 4400302d 4c505349 3a3d5941 l a n d - 0 . D I S P L A Y = :
[7ffffb60] 49470030 414c5f4f 48434e55 445f4445 0 . G I O _ L A U N C H E D _ D
[7ffffb70] 544b5345 465f504f 5f454c49 3d444950 E S K T O P _ F I L E _ P I D =
[7ffffb80] 36383733 4f494700 55414c5f 4548434e 3 7 8 6 . G I O _ L A U N C H E
[7ffffb90] 45445f44 4f544b53 49465f50 2f3d454c D _ D E S K T O P _ F I L E = /
[7ffffba0] 2f727375 72616873 70612f65 63696c70 u s r / s h a r e / a p p l i c
[7ffffbb0] 6f697461 712f736e 69707374 65642e6d a t i o n s / q t s p i m . d e
[7ffffbc0] 6f746b73 53530070 55415f48 535f4854 s k t o p . S S H _ A U T H _ S

Question(2) **Introduction to arrays:**

The marks of a student in 4 subjects are stored in registers s0, s1, s2, s3. Store the marks in an array named ‘marks’. Display the elements of the array in Console window. Each element should appear on a new line. Assume that all marks are integers. Use the concept of loops to display the array elements. Register t0 should be used to keep track of the offset from the starting address of the array which is initially 0.

```
.data
        marks : .word  0,0,0,0   # array declaration
        length: .word  4
        newline: .asciiz "\n"              # new-line character

.text
.globl main
main:

# ------- MARKS STORED IN THE REGISTERS ------  #
        li $s0 , 10
        li $s1 , 7
        li $s2 , 8
        li $s3 , 9
# ------- STORING THOSE IN THE ARRAYS --------- #
        la $t0 , marks
        lw $t1 , length
        li $t3 , 1              # loop index - (1 to length)

store_in_array:
        sw $s0 , ($t0)
        sw $s1 , 4($t0)
        sw $s2 , 8($t0)
        sw $s3 , 12($t0)
```

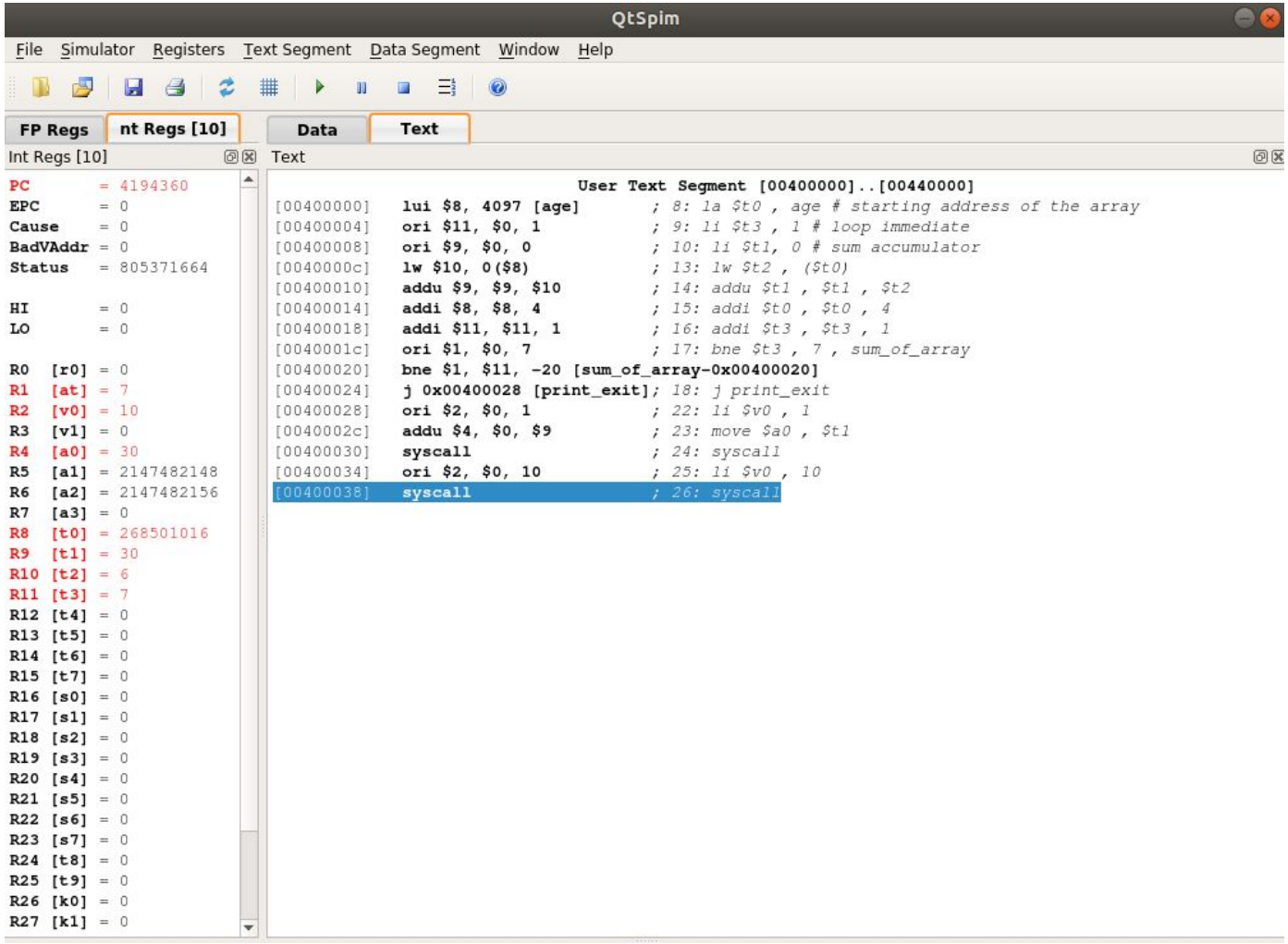
```
print_the_array:
        li $v0 , 1
        lw $a0 , ($t0)
        syscall
        li $v0 , 4
        la $a0 , newline
        syscall
        addi $t0 , $t0 , 4
        addi $t3 , $t3 , 1
        bne $t3 , 5 , print_the_array
        j exit

# ------------ TERMINATE THE PROGRAM ----------- #
exit:
        li $v0 , 10
        syscall
.end main
```

Question(3) The ages of 6 members in a family are stored in an array. Create a code sequence that computes the total age in the array. Register t0 should be used to keep track of the offset from the starting address of the array which is initially 0. Register t1 should used to store the Sum.

```
.data
        age : .word  10 ,2 ,3, 4, 5, 6   # array declaration
        length: .word  6

.text
.globl main
main:
        la $t0 , age # starting address of the array
        li $t3 , 1       # loop immediate
        li $t1, 0        # sum accumulator

sum_of_array:
        lw $t2 , ($t0)
        addu $t1 , $t1 , $t2
        addi $t0 , $t0 , 4
        addi $t3 , $t3 , 1
        bne $t3 , 7 , sum_of_array
        j print_exit
```

```
# ------------ TERMINATE THE PROGRAM ----------- #
print_exit:
        li $v0 , 1
        move $a0 , $t1
        syscall
        li $v0 , 10
        syscall
.end main
```

TEXT:


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
nt Regs [10]
Data
Text
Int Regs [10]
PC = 4194360
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664
HI = 0
LO = 0
R0 [r0] = 0
R1 [at] = 7
R2 [v0] = 10
R3 [v1] = 0
R4 [a0] = 30
R5 [a1] = 2147482148
R6 [a2] = 2147482156
R7 [a3] = 0
R8 [t0] = 268501016
R9 [t1] = 30
R10 [t2] = 6
R11 [t3] = 7
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
Text
User Text Segment [00400000]..[00440000]
[00400000] lui $8, 4097 [age] ; 8: la $t0 , age # starting address of the array
[00400004] ori $11, $0, 1 ; 9: li $t3 , 1 # loop immediate
[00400008] ori $9, $0, 0 ; 10: li $t1, 0 # sum accumulator
[0040000c] lw $10, 0($8) ; 13: lw $t2 , ($t0)
[00400010] addu $9, $9, $10 ; 14: addu $t1 , $t1 , $t2
[00400014] addi $8, $8, 4 ; 15: addi $t0 , $t0 , 4
[00400018] addi $11, $11, 1 ; 16: addi $t3 , $t3 , 1
[0040001c] ori $1, $0, 7 ; 17: bne $t3 , 7 , sum_of_array
[00400020] bne $1, $11, -20 [sum_of_array-0x00400020]
[00400024] j 0x00400028 [print_exit] ; 18: j print_exit
[00400028] ori $2, $0, 1 ; 22: li $v0 , 1
[0040002c] addu $4, $0, $9 ; 23: move $a0 , $t1
[00400030] syscall ; 24: syscall
[00400034] ori $2, $0, 10 ; 25: li $v0 , 10
[00400038] syscall ; 26: syscall

CONSOLE :

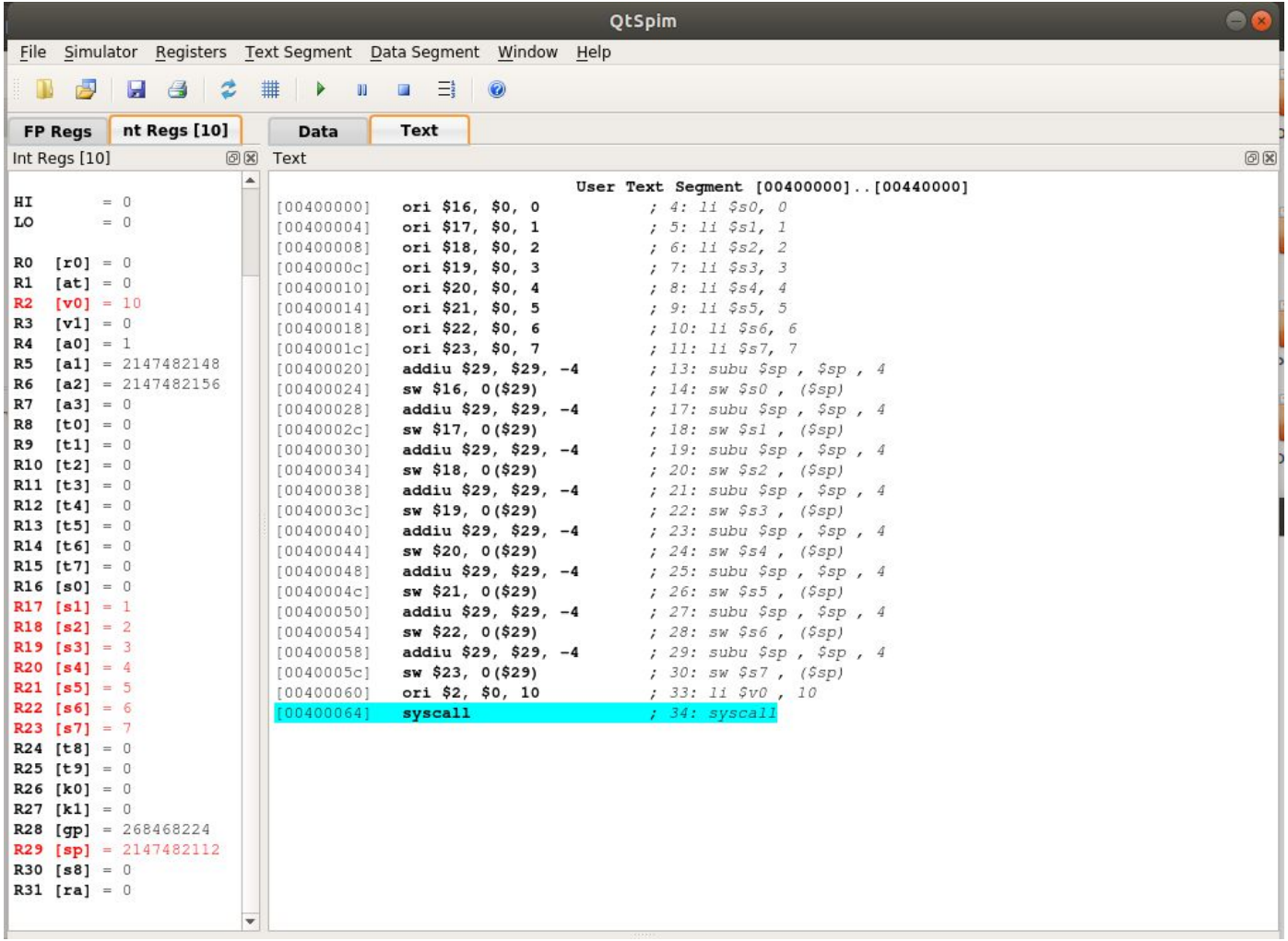
Question(4) Introduction to stack operations:

Store the saved registers s0 to s7 with values from 1 to 7. Increment the values of all these registers by 1. Illustrate how you add the present values (ie. the incremented values) in the ‘s’ registers with the corresponding previous values in these registers using the concept of stack. You are not supposed to move the previous values into any temporary registers directly. The sum after addition should be stored in registers t0 to t7.

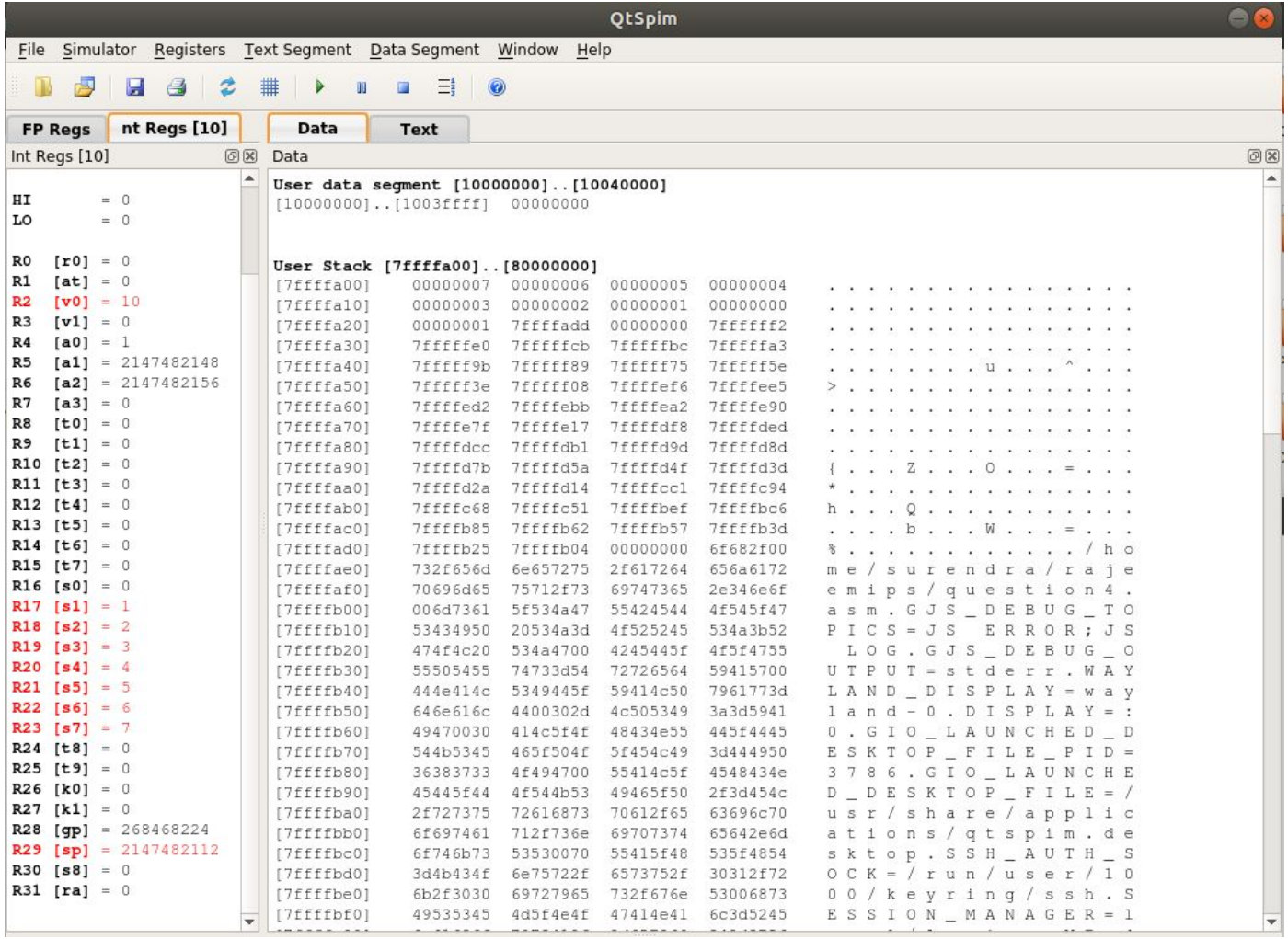
```
.text
.globl main
main:
        li $s0, 0
        li $s1, 1
        li $s2, 2
        li $s3, 3
        li $s4, 4
        li $s5, 5
        li $s6, 6
        li $s7, 7
# ----------- ADDING THE STORED REGISTERS ------ #
        subu $sp , $sp , 4
        sw          $s0 , ($sp)
```

```
        subu $sp , $sp , 4
        sw          $s1 , ($sp)
        subu $sp , $sp , 4
        sw          $s2 , ($sp)
        subu $sp , $sp , 4
        sw          $s3 , ($sp)
        subu $sp , $sp , 4
        sw          $s4 , ($sp)
        subu $sp , $sp , 4
        sw          $s5 , ($sp)
        subu $sp , $sp , 4
        sw          $s6 , ($sp)
        subu $sp , $sp , 4
        sw          $s7 , ($sp)

# ------------ TERMINATE THE PROGRAM ----------- #
        li $v0 , 10
        syscall
.end main
```

TEXT :


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
nt Regs [10]
Data
Text
Int Regs [10]
HI = 0
LO = 0
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = 10
R3 [v1] = 0
R4 [a0] = 1
R5 [a1] = 2147482148
R6 [a2] = 2147482156
R7 [a3] = 0
R8 [t0] = 0
R9 [t1] = 0
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 1
R18 [s2] = 2
R19 [s3] = 3
R20 [s4] = 4
R21 [s5] = 5
R22 [s6] = 6
R23 [s7] = 7
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 268468224
R29 [sp] = 2147482112
R30 [s8] = 0
R31 [ra] = 0
Text
User Text Segment [00400000]..[00440000]
[00400000] ori $16, $0, 0 ; 4: li $s0, 0
[00400004] ori $17, $0, 1 ; 5: li $s1, 1
[00400008] ori $18, $0, 2 ; 6: li $s2, 2
[0040000c] ori $19, $0, 3 ; 7: li $s3, 3
[00400010] ori $20, $0, 4 ; 8: li $s4, 4
[00400014] ori $21, $0, 5 ; 9: li $s5, 5
[00400018] ori $22, $0, 6 ; 10: li $s6, 6
[0040001c] ori $23, $0, 7 ; 11: li $s7, 7
[00400020] addiu $29, $29, -4 ; 13: subu $sp , $sp , 4
[00400024] sw $16, 0($29) ; 14: sw $s0 , ($sp)
[00400028] addiu $29, $29, -4 ; 17: subu $sp , $sp , 4
[0040002c] sw $17, 0($29) ; 18: sw $s1 , ($sp)
[00400030] addiu $29, $29, -4 ; 19: subu $sp , $sp , 4
[00400034] sw $18, 0($29) ; 20: sw $s2 , ($sp)
[00400038] addiu $29, $29, -4 ; 21: subu $sp , $sp , 4
[0040003c] sw $19, 0($29) ; 22: sw $s3 , ($sp)
[00400040] addiu $29, $29, -4 ; 23: subu $sp , $sp , 4
[00400044] sw $20, 0($29) ; 24: sw $s4 , ($sp)
[00400048] addiu $29, $29, -4 ; 25: subu $sp , $sp , 4
[0040004c] sw $21, 0($29) ; 26: sw $s5 , ($sp)
[00400050] addiu $29, $29, -4 ; 27: subu $sp , $sp , 4
[00400054] sw $22, 0($29) ; 28: sw $s6 , ($sp)
[00400058] addiu $29, $29, -4 ; 29: subu $sp , $sp , 4
[0040005c] sw $23, 0($29) ; 30: sw $s7 , ($sp)
[00400060] ori $2, $0, 10 ; 33: li $v0 , 10
[00400064] syscall ; 34: syscall

DATA :


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
nt Regs [10]
Data
Text
Int Regs [10]
HI = 0
LO = 0
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = 10
R3 [v1] = 0
R4 [a0] = 1
R5 [a1] = 2147482148
R6 [a2] = 2147482156
R7 [a3] = 0
R8 [t0] = 0
R9 [t1] = 0
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 1
R18 [s2] = 2
R19 [s3] = 3
R20 [s4] = 4
R21 [s5] = 5
R22 [s6] = 6
R23 [s7] = 7
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 268468224
R29 [sp] = 2147482112
R30 [s8] = 0
R31 [ra] = 0
Data
User data segment [10000000]..[10040000]
[10000000]..[1003ffff] 00000000
User Stack [7ffffa00]..[80000000]
[7ffffa00] 00000007 00000006 00000005 00000004 . . . . . .
[7ffffa10] 00000003 00000002 00000001 00000000 . . . . . .
[7ffffa20] 00000001 7ffffadd 00000000 7fffff2 . . . . . .
[7ffffa30] 7fffffe0 7fffffcb 7fffffbc 7fffffa3 . . . . . .
[7ffffa40] 7fffff9b 7fffff89 7fffff75 7fffff5e . . . . . u . . . ^ . . .
[7ffffa50] 7fffff3e 7fffff08 7ffffef6 7ffffee5 > . . . . . .
[7ffffa60] 7ffffed2 7ffffebb 7ffffea2 7ffffe90 . . . . . .
[7ffffa70] 7ffffe7f 7ffffe17 7ffffdf8 7ffffded . . . . . .
[7ffffa80] 7ffffdcc 7ffffdb1 7ffffd9d 7ffffd8d . . . . . .
[7ffffa90] 7ffffd7b 7ffffd5a 7ffffd4f 7ffffd3d { . . . Z . . . O . . . = . . .
[7ffffaa0] 7ffffd2a 7ffffd14 7ffffcc1 7ffffc94 * . . . . . .
[7ffffab0] 7ffffc68 7ffffc51 7ffffbef 7ffffbc6 h . . . Q . . . . . .
[7ffffac0] 7ffffb85 7ffffb62 7ffffb57 7ffffb3d . . . . b . . . W . . . = . . .
[7ffffad0] 7ffffb25 7ffffb04 00000000 6f682f00 % . . . . . / h o
[7ffffae0] 732f656d 6e657275 2f617264 656a6172 m e / s u r e n d r a / r a j e
[7ffffaf0] 70696d65 75712f73 69747365 2e346e6f e m i p s / q u e s t i o n 4 .
[7ffffb00] 006d7361 5f534a47 55424544 4f545f47 a s m . G J S _ D E B U G _ T O
[7ffffb10] 53434950 20534a3d 4f525245 534a3b52 P I C S = J S E R R O R ; J S
[7ffffb20] 474f4c20 534a4700 4245445f 4f5f4755 L O G . G J S _ D E B U G _ O
[7ffffb30] 55505455 74733d54 72726564 59415700 U T P U T = s t d e r r . W A Y
[7ffffb40] 444e414c 5349445f 59414c50 7961773d L A N D _ D I S P L A Y = w a y
[7ffffb50] 646e616c 4400302d 4c505349 3a3d5941 l a n d - 0 . D I S P L A Y = :
[7ffffb60] 49470030 414c5f4f 48434e55 445f4445 0 . G I O _ L A U N C H E D _ D
[7ffffb70] 544b5345 465f504f 5f454c49 3d444950 E S K T O P _ F I L E _ P I D =
[7ffffb80] 36383733 4f494700 55414c5f 4548434e 3 7 8 6 . G I O _ L A U N C H E
[7ffffb90] 45445f44 4f544b53 49465f50 2f3d454c D _ D E S K T O P _ F I L E = /
[7ffffba0] 2f727375 72616873 70612f65 63696c70 u s r / s h a r e / a p p l i c
[7ffffbb0] 6f697461 712f736e 69707374 65642e6d a t i o n s / q t s p i m . d e
[7ffffbc0] 6f746b73 53530070 55415f48 535f4854 s k t o p . S S H _ A U T H _ S
[7ffffbd0] 3d4b434f 6e75722f 6573752f 30312f72 O C K = / r u n / u s e r / 1 0
[7ffffbe0] 6b2f3030 69727965 732f676e 53006873 0 0 / k e y r i n g / s s h . S
[7ffffbf0] 49535345 4d5f4e4f 47414e41 6c3d5245 E S S I O N _ M A N A G E R = l

Question(5) **Floating point arithmetic: A student has stored his SGPAs for 4 semesters as an array. The SGPAs are floating point numbers. Display the sum, product, average, minimum and maximum, of the entered SGPAs. Use loops wherever possible.**

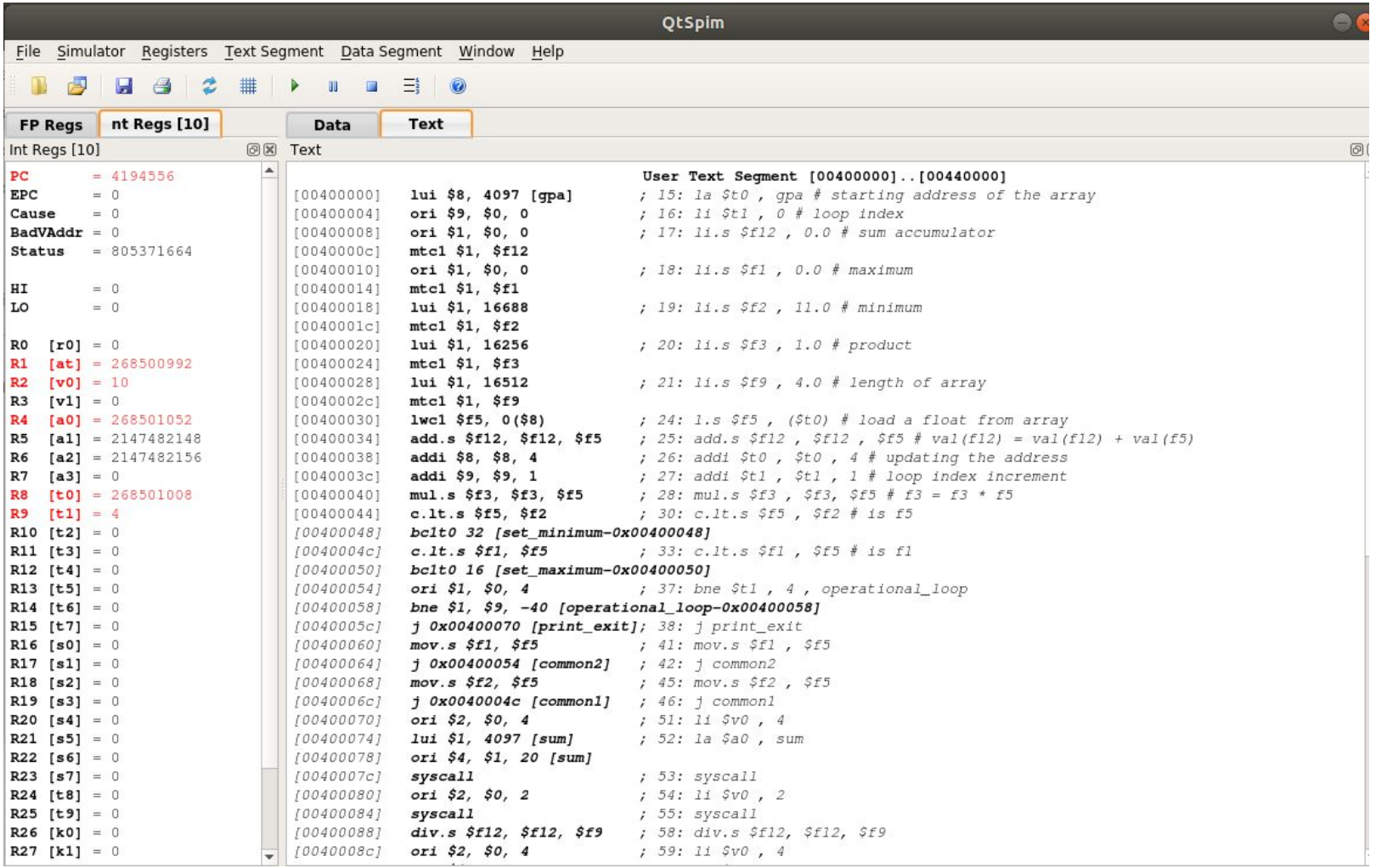
```
.data
        gpa:   .float  10.0 8.23 8.10 7.0
        length:        .word 4

        sum:  .asciiz "\nSum: "
        product: .asciiz "\nProduct: "
        average: .asciiz "\nAverage: "
        max: .asciiz "\nMaximum: "
        min: .asciiz "\nMinimum: "

# sum, product, average, minimum and maximum,
.text
.globl main
main:
        la $t0 , gpa          # starting address of the array
        li $t1 , 0            # loop index
        li.s $f12 , 0.0       # sum accumulator
        li.s $f1 , 0.0# maximum
        li.s $f2 , 11.0       # minimum
        li.s $f3 , 1.0# product
        li.s $f9 , 4.0# length of array
```

```
operational_loop:
        l.s $f5 , ($t0)                 # load a float from array
        add.s $f12 , $f12 , $f5  # val(f12) = val(f12) + val(f5)
        addi $t0 , $t0 , 4          # updating the address
        addi $t1 , $t1 , 1          # loop index increment
        mul.s $f3 , $f3, $f5       # f3 = f3 * f5

        c.lt.s $f5 , $f2             # is f5<f2?
        bc1t set_minimum
common1:
        c.lt.s $f1 , $f5             # is f1<f5?
        bc1t set_maximum

common2:
        bne $t1 , 4 , operational_loop
        j print_exit

set_maximum:
        mov.s $f1 , $f5
        j common2

set_minimum:
        mov.s $f2 , $f5
        j common1
```

```
# ------------ TERMINATE THE PROGRAM ----------- #
print_exit:
# ------------ PRINT SUM ----------------- #
        li $v0 , 4
        la $a0 , sum
        syscall
        li $v0 , 2
        syscall

# ------------ AVERAGE ------------------- #
        div.s  $f12, $f12, $f9
        li $v0 , 4
        la $a0 , average
        syscall
        li $v0 , 2
        syscall

# ------------ PRODUCT ------------------- #
        li $v0 , 4
        la $a0 , product
        syscall
        li $v0 , 2
        mov.s $f12 , $f3
        syscall
```

```
# ------------ MAXIMUM -------------------- #
        li $v0 , 4
        la $a0 , max
        syscall
        li $v0 , 2
        mov.s $f12 , $f1
        syscall

# ------------ MINIMUM -------------------- #
        li $v0 , 4
        la $a0 , min
        syscall
        li $v0 , 2
        mov.s $f12 , $f2
        syscall


# ------------- EXIT ----------------- #
        li $v0 , 10
        syscall
.end main
```

TEXT :


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
nt Regs [10]
Data
Text
Int Regs [10]
Text
PC = 4194556
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664
HI = 0
LO = 0
R0 [r0] = 0
R1 [at] = 268500992
R2 [v0] = 10
R3 [v1] = 0
R4 [a0] = 268501052
R5 [a1] = 2147482148
R6 [a2] = 2147482156
R7 [a3] = 0
R8 [t0] = 268501008
R9 [t1] = 4
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
User Text Segment [00400000]..[00440000]
[00400000] lui $8, 4097 [gpa] ; 15: la $t0 , gpa # starting address of the array
[00400004] ori $9, $0, 0 ; 16: li $t1 , 0 # loop index
[00400008] ori $1, $0, 0 ; 17: li.s $f12 , 0.0 # sum accumulator
[0040000c] mtc1 $1, $f12
[00400010] ori $1, $0, 0 ; 18: li.s $f1 , 0.0 # maximum
[00400014] mtc1 $1, $f1
[00400018] lui $1, 16688 ; 19: li.s $f2 , 11.0 # minimum
[0040001c] mtc1 $1, $f2
[00400020] lui $1, 16256 ; 20: li.s $f3 , 1.0 # product
[00400024] mtc1 $1, $f3
[00400028] lui $1, 16512 ; 21: li.s $f9 , 4.0 # length of array
[0040002c] mtc1 $1, $f9
[00400030] lwc1 $f5, 0($8) ; 24: l.s $f5 , ($t0) # load a float from array
[00400034] add.s $f12, $f12, $f5 ; 25: add.s $f12 , $f12 , $f5 # val(f12) = val(f12) + val(f5)
[00400038] addi $8, $8, 4 ; 26: addi $t0 , $t0 , 4 # updating the address
[0040003c] addi $9, $9, 1 ; 27: addi $t1 , $t1 , 1 # loop index increment
[00400040] mul.s $f3, $f3, $f5 ; 28: mul.s $f3 , $f3, $f5 # f3 = f3 * f5
[00400044] c.lt.s $f5, $f2 ; 30: c.lt.s $f5 , $f2 # is f5
[00400048] bc1t0 32 [set_minimum-0x00400048]
[0040004c] c.lt.s $f1, $f5 ; 33: c.lt.s $f1 , $f5 # is f1
[00400050] bc1t0 16 [set_maximum-0x00400050]
[00400054] ori $1, $0, 4 ; 37: bne $t1 , 4 , operational_loop
[00400058] bne $1, $9, -40 [operational_loop-0x00400058]
[0040005c] j 0x00400070 [print_exit] ; 38: j print_exit
[00400060] mov.s $f1, $f5 ; 41: mov.s $f1 , $f5
[00400064] j 0x00400054 [common2] ; 42: j common2
[00400068] mov.s $f2, $f5 ; 45: mov.s $f2 , $f5
[0040006c] j 0x0040004c [common1] ; 46: j common1
[00400070] ori $2, $0, 4 ; 51: li $v0 , 4
[00400074] lui $1, 4097 [sum] ; 52: la $a0 , sum
[00400078] ori $4, $1, 20 [sum]
[0040007c] syscall ; 53: syscall
[00400080] ori $2, $0, 2 ; 54: li $v0 , 2
[00400084] syscall ; 55: syscall
[00400088] div.s $f12, $f12, $f9 ; 58: div.s $f12, $f12, $f9
[0040008c] ori $2, $0, 4 ; 59: li $v0 , 4

```
HI        = 0
LO        = 0

R0  [r0]  = 0
R1  [at]  = 268500992
R2  [v0]  = 10
R3  [v1]  = 0
R4  [a0]  = 268501052
R5  [a1]  = 2147482148
R6  [a2]  = 2147482156
R7  [a3]  = 0
R8  [t0]  = 268501008
R9  [t1]  = 4
R10 [t2]  = 0
R11 [t3]  = 0
R12 [t4]  = 0
R13 [t5]  = 0
R14 [t6]  = 0
R15 [t7]  = 0
R16 [s0]  = 0
R17 [s1]  = 0
R18 [s2]  = 0
R19 [s3]  = 0
R20 [s4]  = 0
R21 [s5]  = 0
R22 [s6]  = 0
R23 [s7]  = 0
R24 [t8]  = 0
R25 [t9]  = 0
R26 [k0]  = 0
R27 [k1]  = 0
```

```
[00400088]   div.s $f12, $f12, $f9     ; 58: div.s $f12, $f12, $f9
[0040008c]   ori $2, $0, 4             ; 59: li $v0 , 4
[00400090]   lui $1, 4097 [average]    ; 60: la $a0 , average
[00400094]   ori $4, $1, 38 [average]
[00400098]   syscall                   ; 61: syscall
[0040009c]   ori $2, $0, 2             ; 62: li $v0 , 2
[004000a0]   syscall                   ; 63: syscall
[004000a4]   ori $2, $0, 4             ; 66: li $v0 , 4
[004000a8]   lui $1, 4097 [product]    ; 67: la $a0 , product
[004000ac]   ori $4, $1, 27 [product]
[004000b0]   syscall                   ; 68: syscall
[004000b4]   ori $2, $0, 2             ; 69: li $v0 , 2
[004000b8]   mov.s $f12, $f3           ; 70: mov.s $f12 , $f3
[004000bc]   syscall                   ; 71: syscall
[004000c0]   ori $2, $0, 4             ; 74: li $v0 , 4
[004000c4]   lui $1, 4097 [max]        ; 75: la $a0 , max
[004000c8]   ori $4, $1, 49 [max]
[004000cc]   syscall                   ; 76: syscall
[004000d0]   ori $2, $0, 2             ; 77: li $v0 , 2
[004000d4]   mov.s $f12, $f1           ; 78: mov.s $f12 , $f1
[004000d8]   syscall                   ; 79: syscall
[004000dc]   ori $2, $0, 4             ; 82: li $v0 , 4
[004000e0]   lui $1, 4097 [min]        ; 83: la $a0 , min
[004000e4]   ori $4, $1, 60 [min]
[004000e8]   syscall                   ; 84: syscall
[004000ec]   ori $2, $0, 2             ; 85: li $v0 , 2
[004000f0]   mov.s $f12, $f2           ; 86: mov.s $f12 , $f2
[004000f4]   syscall                   ; 87: syscall
[004000f8]   ori $2, $0, 10            ; 91: li $v0 , 10
[004000fc]   syscall                   ; 92: syscall
```

CONSOLE :


Console
Sum: 33.33000183
Average: 8.33250046
Product: 4666.41015625
Maximum: 10.00000000
Minimum: 7.00000000