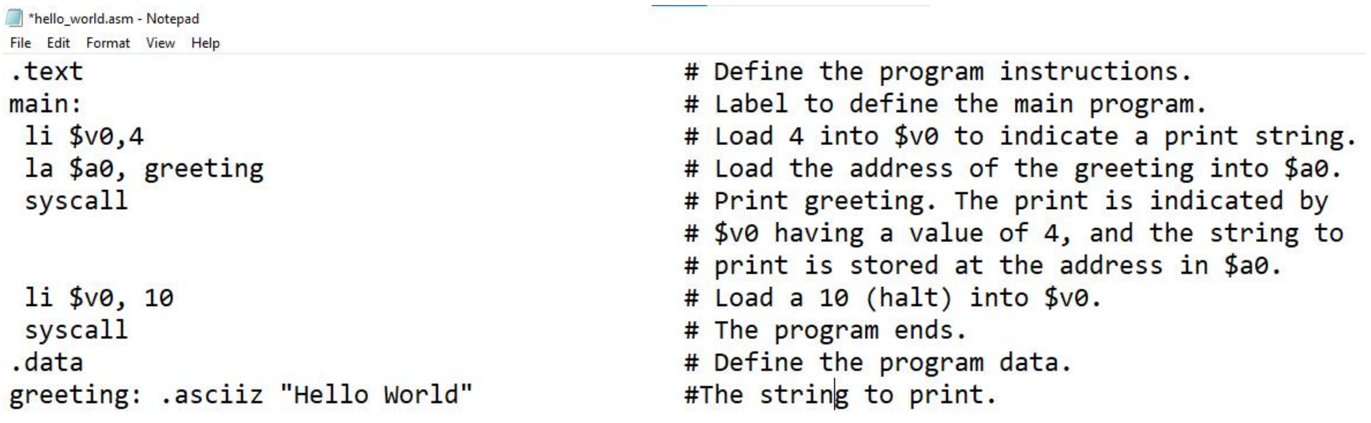
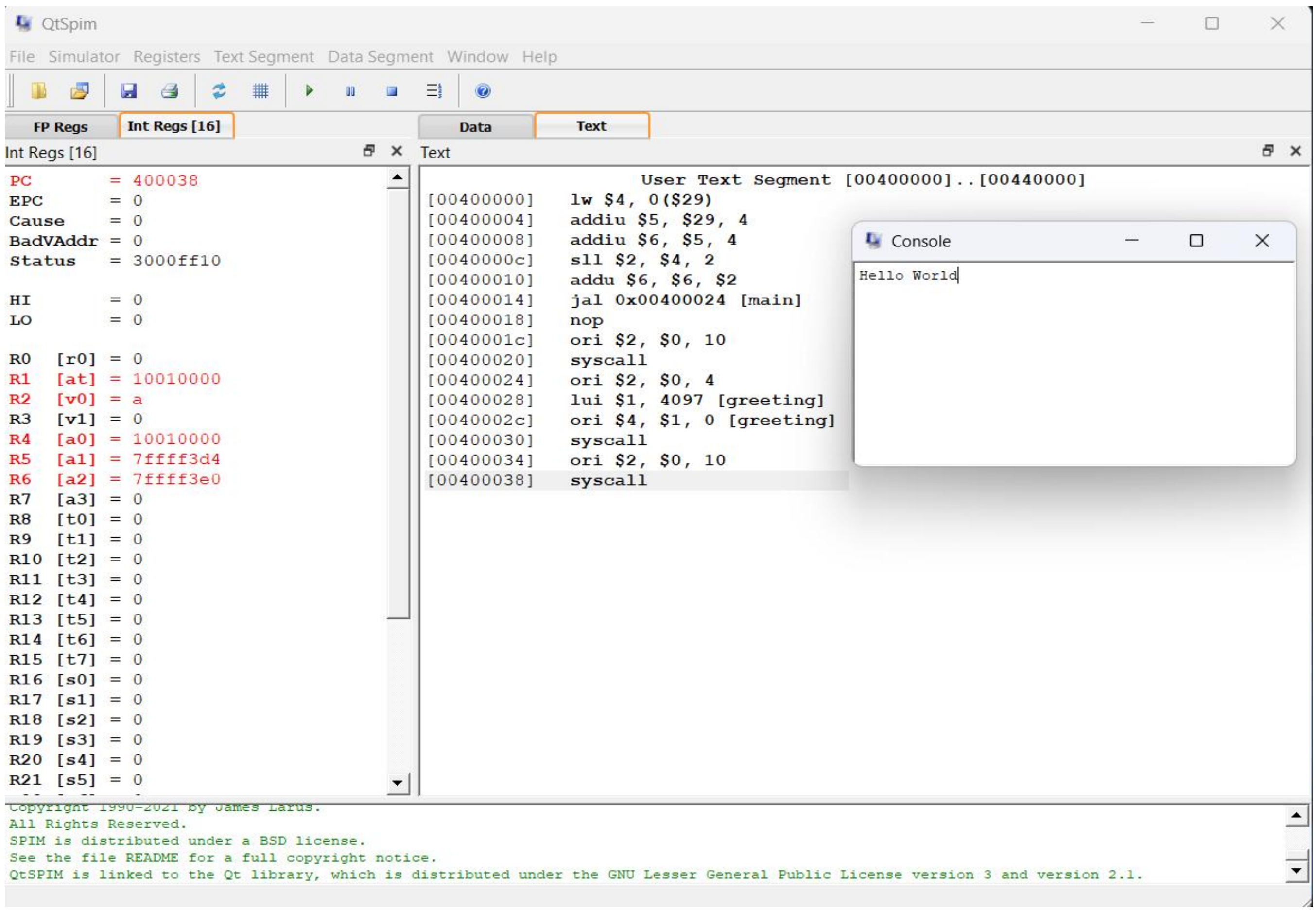
Lab Assignment - 03

Q1. Write a assembly program for Hello world and print it into console.

Ans :



```
.text                                   # Define the program instructions.
main:                                   # Label to define the main program.
 li $v0,4                               # Load 4 into $v0 to indicate a print string.
 la $a0, greeting                       # Load the address of the greeting into $a0.
 syscall                                # Print greeting. The print is indicated by
                                        # $v0 having a value of 4, and the string to
                                        # print is stored at the address in $a0.
 li $v0, 10                             # Load a 10 (halt) into $v0.
 syscall                                # The program ends.
.data                                   # Define the program data.
greeting: .asciiz "Hello World"         #The string to print.
```



QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
Text
Console

```
PC       = 400038
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 10010000
R5  [a1] = 7ffff3d4
R6  [a2] = 7ffff3e0
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] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
```

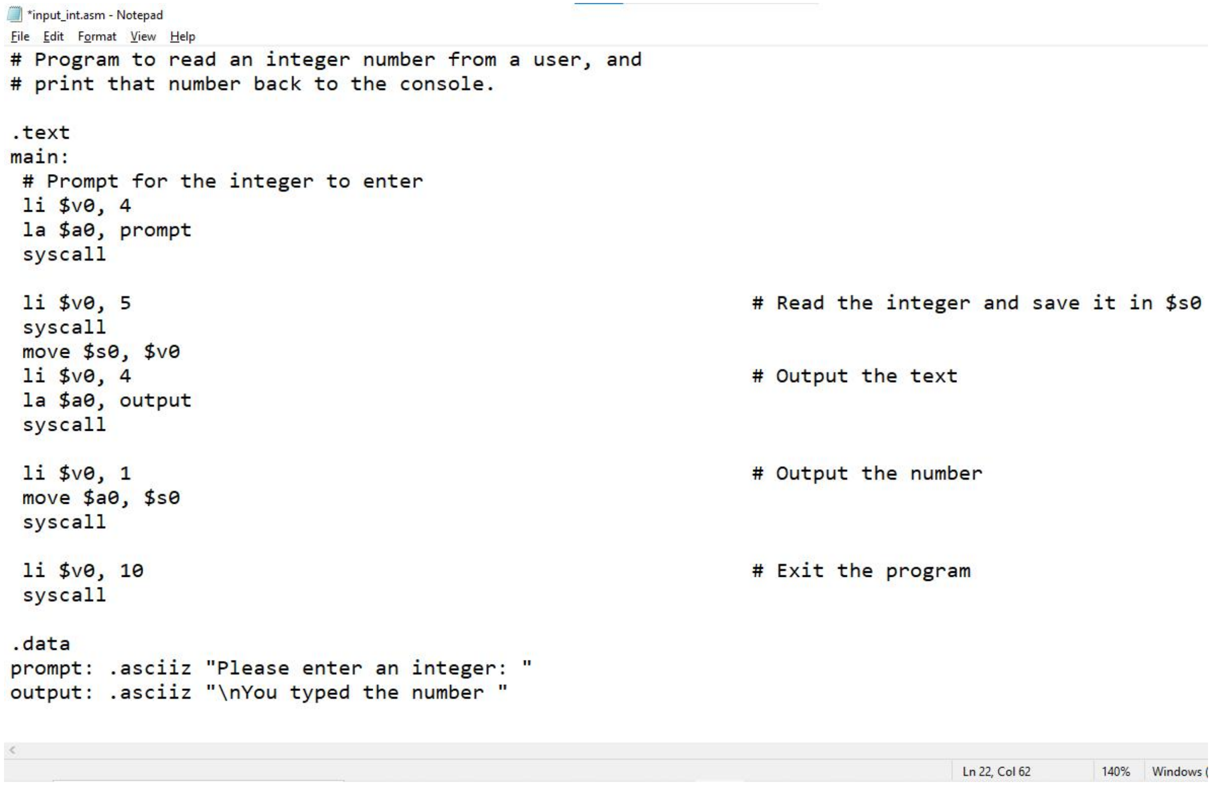
```
                User Text Segment [00400000]..[00440000]
[00400000]    lw $4, 0($29)
[00400004]    addiu $5, $29, 4
[00400008]    addiu $6, $5, 4
[0040000c]    sll $2, $4, 2
[00400010]    addu $6, $6, $2
[00400014]    jal 0x00400024 [main]
[00400018]    nop
[0040001c]    ori $2, $0, 10
[00400020]    syscall
[00400024]    ori $2, $0, 4
[00400028]    lui $1, 4097 [greeting]
[0040002c]    ori $4, $1, 0 [greeting]
[00400030]    syscall
[00400034]    ori $2, $0, 10
[00400038]    syscall
```

```
Hello World
```

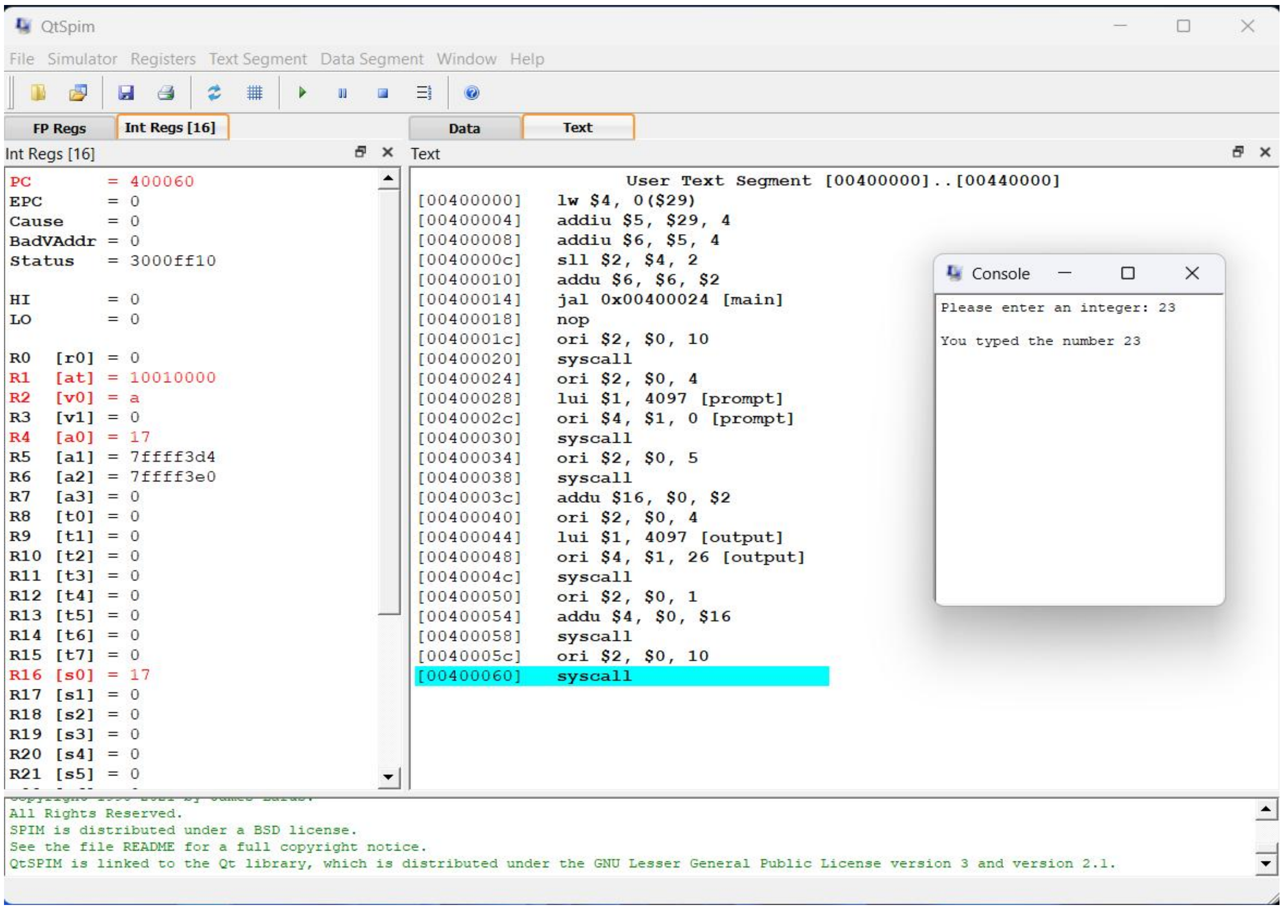
```
Copyright 1990-2021 by James Larus.
All Rights Reserved.
SPIM is distributed under a BSD license.
See the file README for a full copyright notice.
QtSPIM is linked to the Qt library, which is distributed under the GNU Lesser General Public License version 3 and version 2.1.
```

Q2. Write a program to read an integer number from user and print to the console.

ANS :



```
# Program to read an integer number from a user, and
# print that number back to the console.

.text
main:
 # Prompt for the integer to enter
 li $v0, 4
 la $a0, prompt
 syscall

 li $v0, 5                                                  # Read the integer and save it in $s0
 syscall
 move $s0, $v0
 li $v0, 4                                                  # Output the text
 la $a0, output
 syscall

 li $v0, 1                                                  # Output the number
 move $a0, $s0
 syscall

 li $v0, 10                                                 # Exit the program
 syscall

.data
prompt: .asciiz "Please enter an integer: "
output: .asciiz "\nYou typed the number "
```


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
Text
Console

```
PC       = 400060
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 17
R5  [a1] = 7ffff3d4
R6  [a2] = 7ffff3e0
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] = 17
R17 [s1] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
```

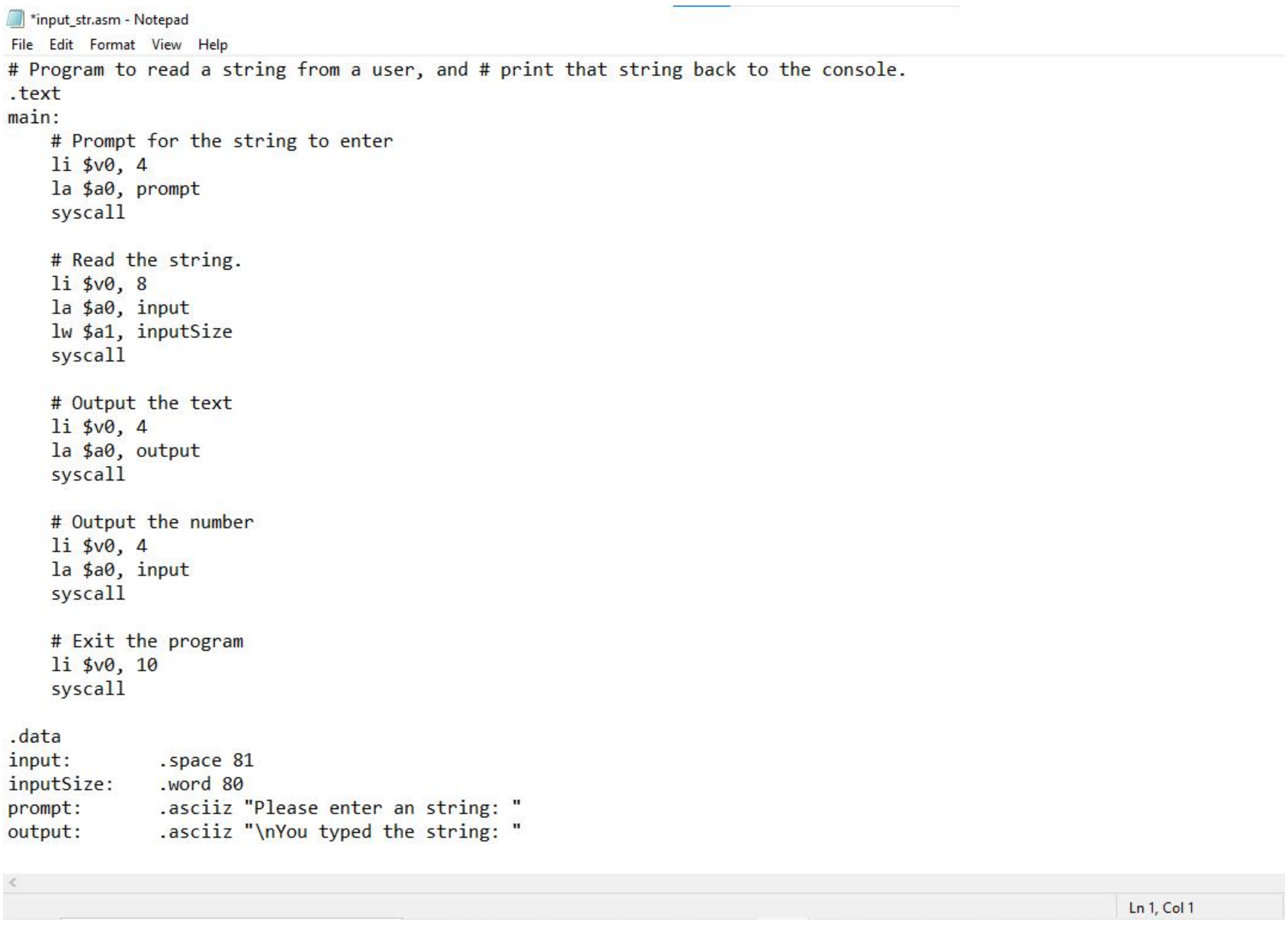
```
                User Text Segment [00400000]..[00440000]
[00400000]    lw $4, 0($29)
[00400004]    addiu $5, $29, 4
[00400008]    addiu $6, $5, 4
[0040000c]    sll $2, $4, 2
[00400010]    addu $6, $6, $2
[00400014]    jal 0x00400024 [main]
[00400018]    nop
[0040001c]    ori $2, $0, 10
[00400020]    syscall
[00400024]    ori $2, $0, 4
[00400028]    lui $1, 4097 [prompt]
[0040002c]    ori $4, $1, 0 [prompt]
[00400030]    syscall
[00400034]    ori $2, $0, 5
[00400038]    syscall
[0040003c]    addu $16, $0, $2
[00400040]    ori $2, $0, 4
[00400044]    lui $1, 4097 [output]
[00400048]    ori $4, $1, 26 [output]
[0040004c]    syscall
[00400050]    ori $2, $0, 1
[00400054]    addu $4, $0, $16
[00400058]    syscall
[0040005c]    ori $2, $0, 10
[00400060]    syscall
```

```
Please enter an integer: 23

You typed the number 23
```

```
All Rights Reserved.
SPIM is distributed under a BSD license.
See the file README for a full copyright notice.
QtSPIM is linked to the Qt library, which is distributed under the GNU Lesser General Public License version 3 and version 2.1.
```

Q3. Program to prompt and read a string from a user.

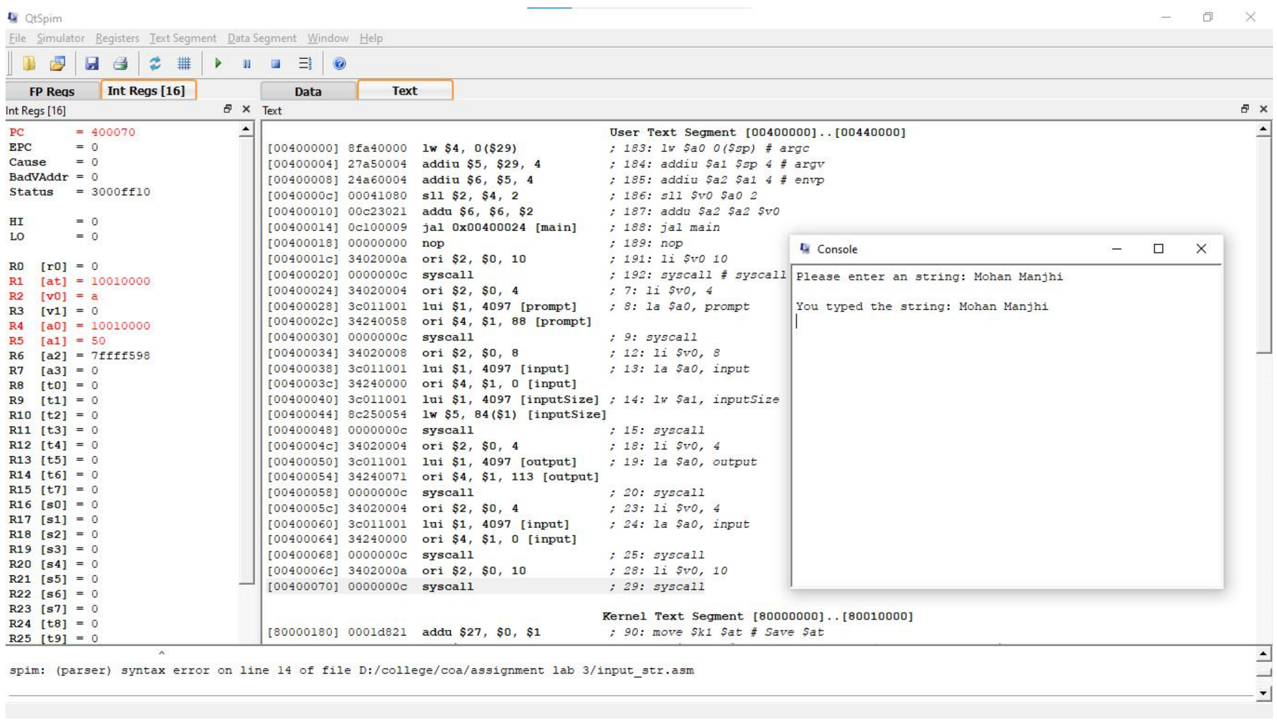
ANS :



*input_str.asm - Notepad

```
# Program to read a string from a user, and # print that string back to the console.
.text
main:
    # Prompt for the string to enter
    li $v0, 4
    la $a0, prompt
    syscall

    # Read the string.
    li $v0, 8
    la $a0, input
    lw $a1, inputSize
    syscall

    # Output the text
    li $v0, 4
    la $a0, output
    syscall

    # Output the number
    li $v0, 4
    la $a0, input
    syscall

    # Exit the program
    li $v0, 10
    syscall

.data
input:          .space 81
inputSize:      .word 80
prompt:         .asciiz "Please enter an string: "
output:         .asciiz "\nYou typed the string: "
```

Ln 1, Col 1



QtSpim

```
PC       = 400070
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 10010000
R5  [a1] = 50
R6  [a2] = 7ffff598
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] = 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
```

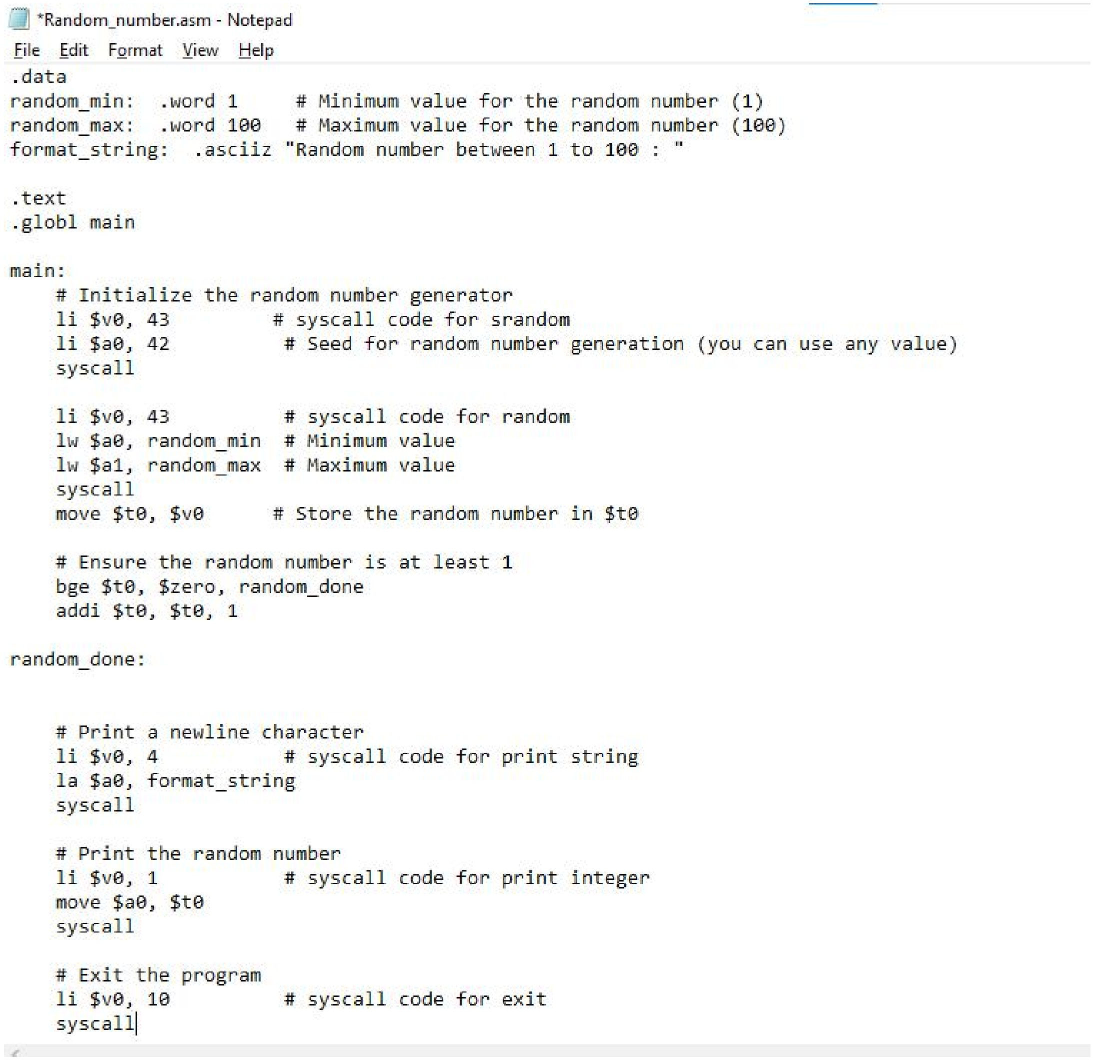
```
                                   User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)           ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4        ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4         ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2           ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2         ; 187: addu $a2 $a2 $v0
[00400014] 0c100009  jal 0x00400024 [main]   ; 188: jal main
[00400018] 00000000  nop                     ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10          ; 191: li $v0 10
[00400020] 0000000c  syscall                 ; 192: syscall # syscall
[00400024] 34020004  ori $2, $0, 4           ; 7: li $v0, 4
[00400028] 3c011001  lui $1, 4097 [prompt]   ; 8: la $a0, prompt
[0040002c] 34240058  ori $4, $1, 88 [prompt]
[00400030] 0000000c  syscall                 ; 9: syscall
[00400034] 34020008  ori $2, $0, 8           ; 12: li $v0, 8
[00400038] 3c011001  lui $1, 4097 [input]    ; 13: la $a0, input
[0040003c] 34240000  ori $4, $1, 0 [input]
[00400040] 3c011001  lui $1, 4097 [inputSize] ; 14: lw $a1, inputSize
[00400044] 8c250054  lw $5, 84($1) [inputSize]
[00400048] 0000000c  syscall                 ; 15: syscall
[0040004c] 34020004  ori $2, $0, 4           ; 18: li $v0, 4
[00400050] 3c011001  lui $1, 4097 [output]   ; 19: la $a0, output
[00400054] 34240071  ori $4, $1, 113 [output]
[00400058] 0000000c  syscall                 ; 20: syscall
[0040005c] 34020004  ori $2, $0, 4           ; 23: li $v0, 4
[00400060] 3c011001  lui $1, 4097 [input]    ; 24: la $a0, input
[00400064] 34240000  ori $4, $1, 0 [input]
[00400068] 0000000c  syscall                 ; 25: syscall
[0040006c] 3402000a  ori $2, $0, 10          ; 28: li $v0, 10
[00400070] 0000000c  syscall                 ; 29: syscall

                                  Kernel Text Segment [80000000]..[80010000]
[80000180] 0001d821  addu $27, $0, $1        ; 90: move $k1 $at # Save $at
```

Console

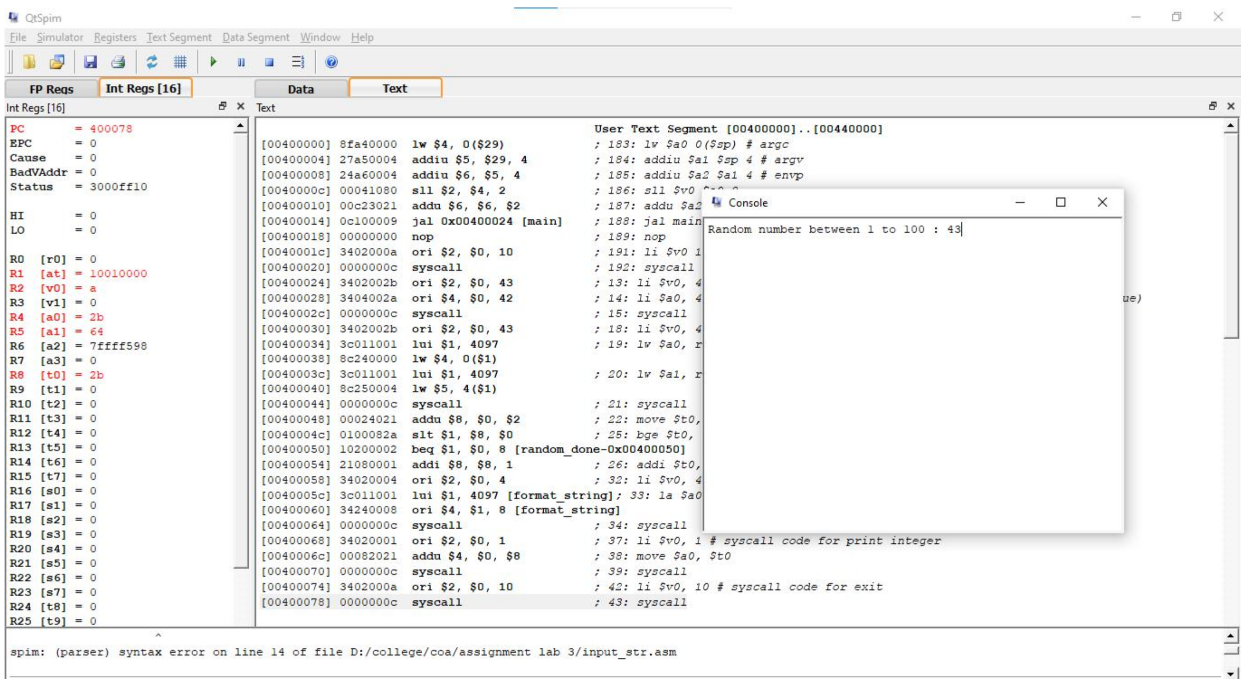
```
Please enter an string: Mohan Manjhi

You typed the string: Mohan Manjhi
```

```
spim: (parser) syntax error on line 14 of file D:/college/coa/assignment lab 3/input_str.asm
```

Q4. Write a program to print out a random number from 1..100.



```
.data
random_min:   .word 1     # Minimum value for the random number (1)
random_max:   .word 100   # Maximum value for the random number (100)
format_string:  .asciiz "Random number between 1 to 100 : "

.text
.globl main

main:
    # Initialize the random number generator
    li $v0, 43         # syscall code for srandom
    li $a0, 42          # Seed for random number generation (you can use any value)
    syscall

    li $v0, 43          # syscall code for random
    lw $a0, random_min  # Minimum value
    lw $a1, random_max  # Maximum value
    syscall
    move $t0, $v0      # Store the random number in $t0

    # Ensure the random number is at least 1
    bge $t0, $zero, random_done
    addi $t0, $t0, 1

random_done:


    # Print a newline character
    li $v0, 4           # syscall code for print string
    la $a0, format_string
    syscall

    # Print the random number
    li $v0, 1           # syscall code for print integer
    move $a0, $t0
    syscall

    # Exit the program
    li $v0, 10          # syscall code for exit
    syscall
```


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
Text
Console

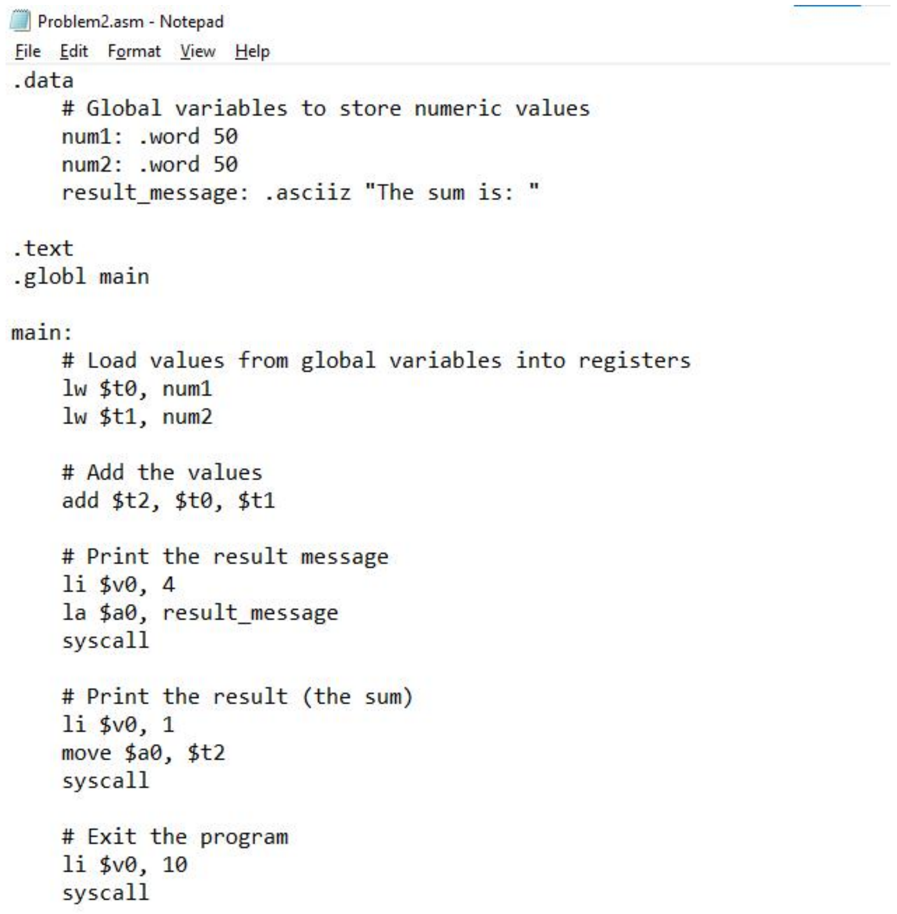
```
PC       = 400078
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 2b
R5  [a1] = 64
R6  [a2] = 7ffff598
R7  [a3] = 0
R8  [t0] = 2b
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] = 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
```

```
                                                User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)              ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4           ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4            ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2              ; 186: sll $v0
[00400010] 00c23021  addu $6, $6, $2            ; 187: addu $a2
[00400014] 0c100009  jal 0x00400024 [main]      ; 188: jal main
[00400018] 00000000  nop                        ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10             ; 191: li $v0 1
[00400020] 0000000c  syscall                    ; 192: syscall
[00400024] 3402002b  ori $2, $0, 43             ; 13: li $v0, 4
[00400028] 3404002a  ori $4, $0, 42             ; 14: li $a0, 4                                            ue)
[0040002c] 0000000c  syscall                    ; 15: syscall
[00400030] 3402002b  ori $2, $0, 43             ; 18: li $v0, 4
[00400034] 3c011001  lui $1, 4097               ; 19: lw $a0, r
[00400038] 8c240000  lw $4, 0($1)
[0040003c] 3c011001  lui $1, 4097               ; 20: lw $a1, r
[00400040] 8c250004  lw $5, 4($1)
[00400044] 0000000c  syscall                    ; 21: syscall
[00400048] 00024021  addu $8, $0, $2            ; 22: move $t0,
[0040004c] 0100082a  slt $1, $8, $0             ; 25: bge $t0,
[00400050] 10200002  beq $1, $0, 8 [random_done-0x00400050]
[00400054] 21080001  addi $8, $8, 1             ; 26: addi $t0,
[00400058] 34020004  ori $2, $0, 4              ; 32: li $v0, 4
[0040005c] 3c011001  lui $1, 4097 [format_string]; 33: la $a0
[00400060] 34240008  ori $4, $1, 8 [format_string]
[00400064] 0000000c  syscall                    ; 34: syscall
[00400068] 34020001  ori $2, $0, 1              ; 37: li $v0, 1 # syscall code for print integer
[0040006c] 00082021  addu $4, $0, $8            ; 38: move $a0, $t0
[00400070] 0000000c  syscall                    ; 39: syscall
[00400074] 3402000a  ori $2, $0, 10             ; 42: li $v0, 10 # syscall code for exit
[00400078] 0000000c  syscall                    ; 43: syscall
```

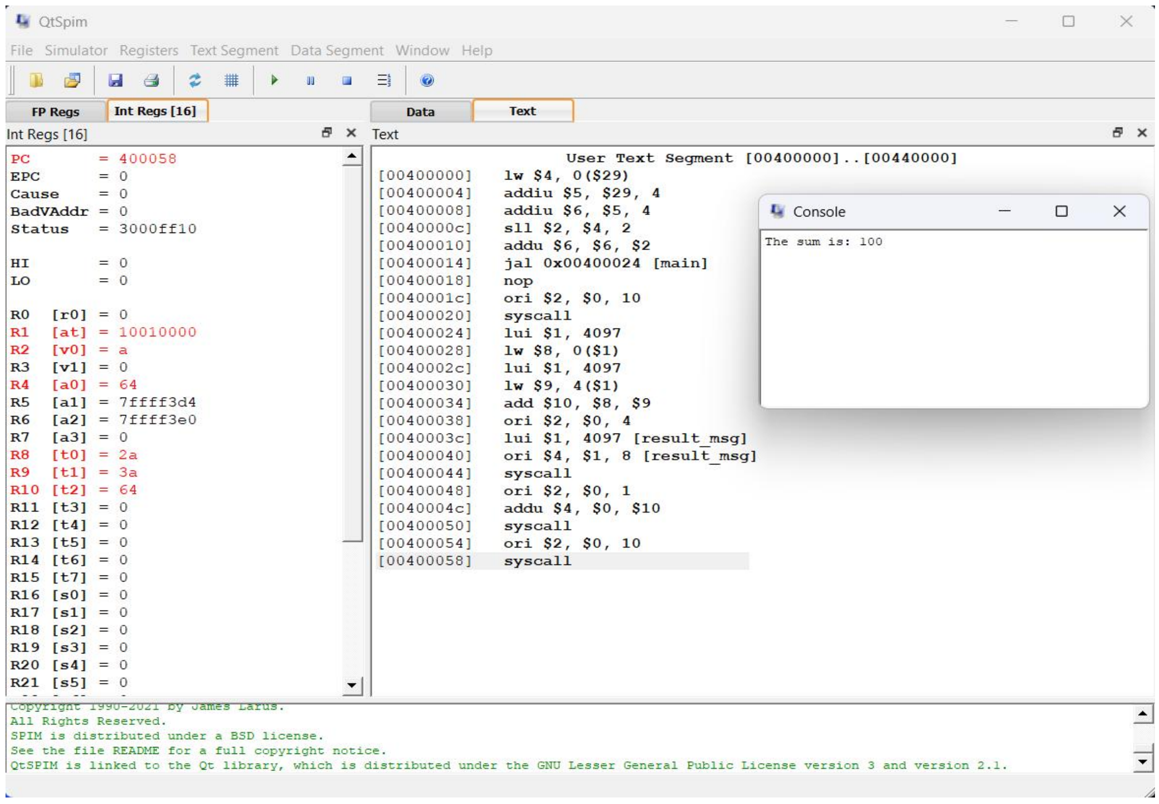
```
Random number between 1 to 100 : 43
```

```
spim: (parser) syntax error on line 14 of file D:/college/coa/assignment lab 3/input_str.asm
```

Q5. Write a MIPS assembly program called Problem2.asm. This program should have two global variables, which stores numeric values. The program itself should sum those two values and then print the result to the console.



```
.data
    # Global variables to store numeric values
    num1: .word 50
    num2: .word 50
    result_message: .asciiz "The sum is: "

.text
.globl main

main:
    # Load values from global variables into registers
    lw $t0, num1
    lw $t1, num2

    # Add the values
    add $t2, $t0, $t1

    # Print the result message
    li $v0, 4
    la $a0, result_message
    syscall

    # Print the result (the sum)
    li $v0, 1
    move $a0, $t2
    syscall

    # Exit the program
    li $v0, 10
    syscall
```


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
Text
Console

```
PC       = 400058
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 64
R5  [a1] = 7ffff3d4
R6  [a2] = 7ffff3e0
R7  [a3] = 0
R8  [t0] = 2a
R9  [t1] = 3a
R10 [t2] = 64
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
```

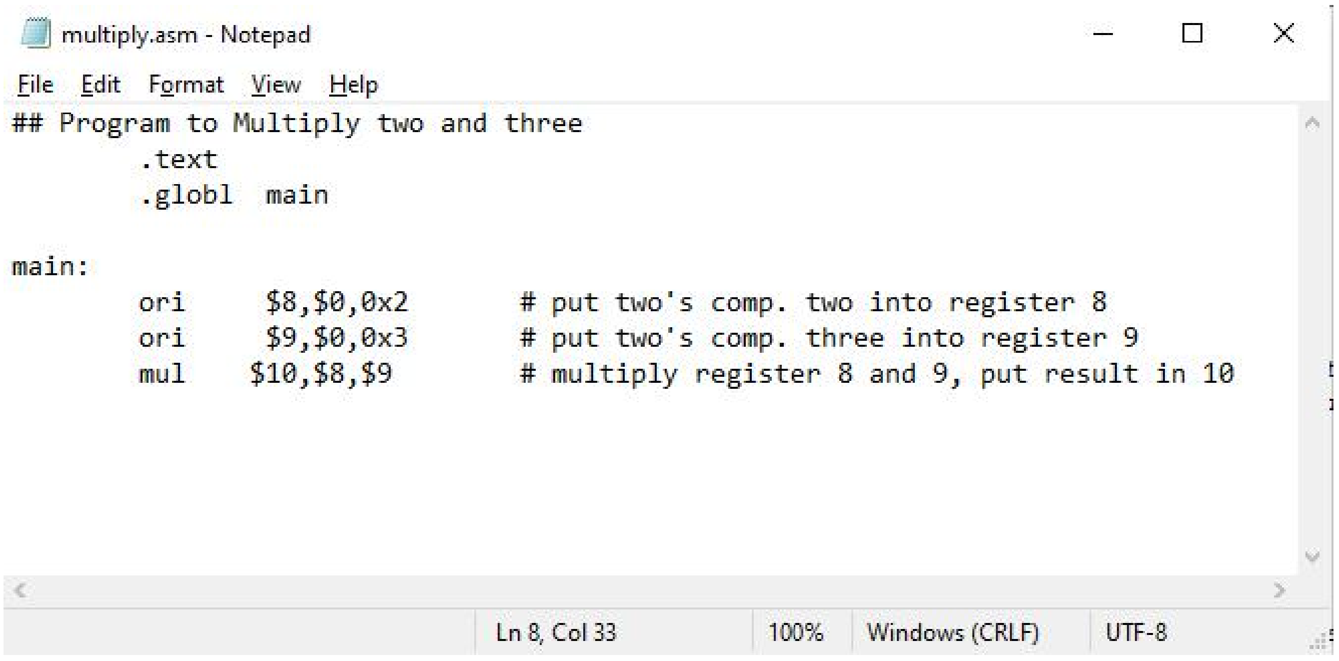
```
               User Text Segment [00400000]..[00440000]
[00400000]  lw $4, 0($29)
[00400004]  addiu $5, $29, 4
[00400008]  addiu $6, $5, 4
[0040000c]  sll $2, $4, 2
[00400010]  addu $6, $6, $2
[00400014]  jal 0x00400024 [main]
[00400018]  nop
[0040001c]  ori $2, $0, 10
[00400020]  syscall
[00400024]  lui $1, 4097
[00400028]  lw $8, 0($1)
[0040002c]  lui $1, 4097
[00400030]  lw $9, 4($1)
[00400034]  add $10, $8, $9
[00400038]  ori $2, $0, 4
[0040003c]  lui $1, 4097 [result_msg]
[00400040]  ori $4, $1, 8 [result_msg]
[00400044]  syscall
[00400048]  ori $2, $0, 1
[0040004c]  addu $4, $0, $10
[00400050]  syscall
[00400054]  ori $2, $0, 10
[00400058]  syscall
```

```
The sum is: 100
```

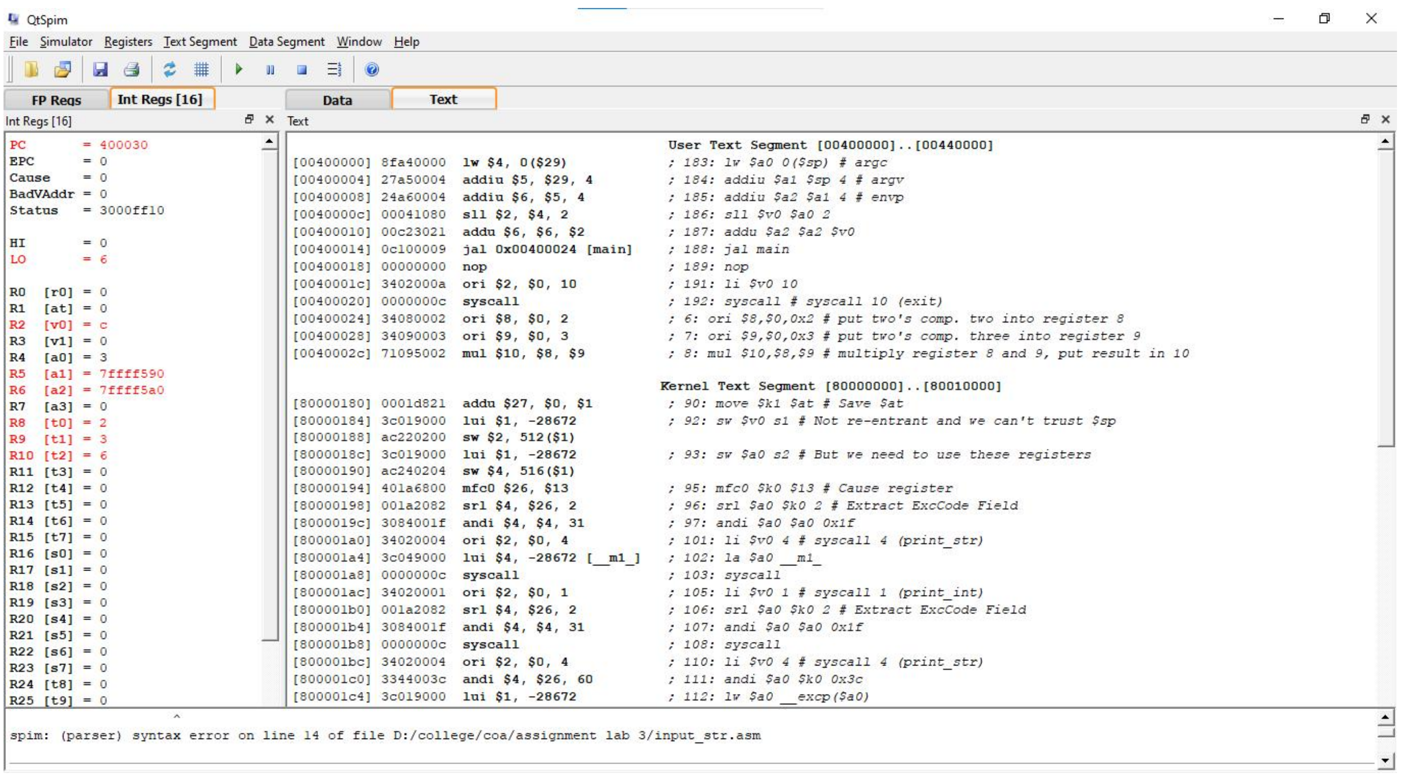
```
Copyright 1990-2021 by James Larus.
All Rights Reserved.
SPIM is distributed under a BSD license.
See the file README for a full copyright notice.
QtSPIM is linked to the Qt library, which is distributed under the GNU Lesser General Public License version 3 and version 2.1.
```

Q6. The program should have two temporary registers , which stores numeric values. The program must have multiply these two numbers and the result is store in the saved value registers.

ANS :



```
## Program to Multiply two and three
        .text
        .globl  main

main:
        ori     $8,$0,0x2       # put two's comp. two into register 8
        ori     $9,$0,0x3       # put two's comp. three into register 9
        mul    $10,$8,$9        # multiply register 8 and 9, put result in 10
```



QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text

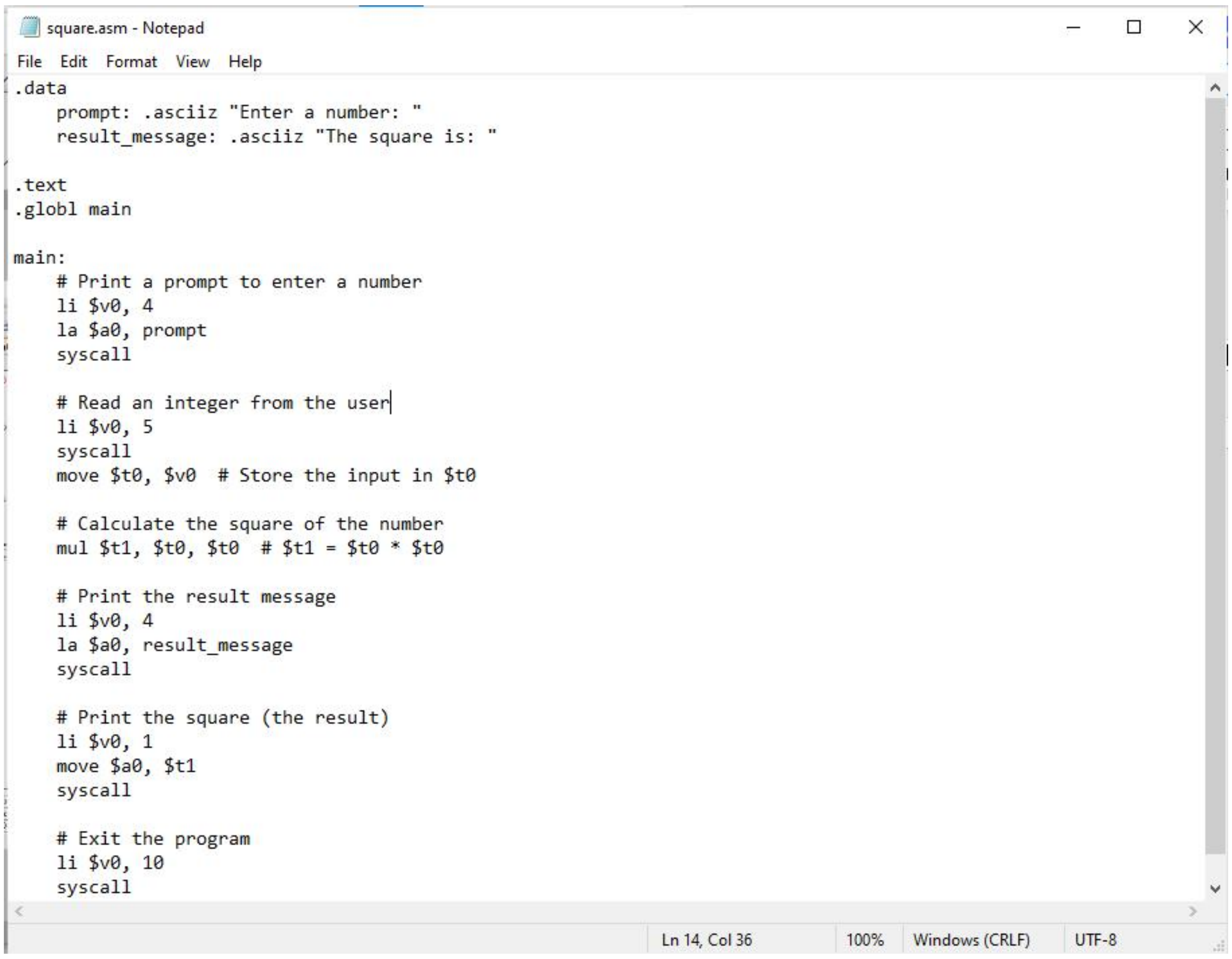
```
Int Regs [16]
PC       = 400030
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 6

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = c
R3  [v1] = 0
R4  [a0] = 3
R5  [a1] = 7ffff590
R6  [a2] = 7ffff5a0
R7  [a3] = 0
R8  [t0] = 2
R9  [t1] = 3
R10 [t2] = 6
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
```

```
Text
                                               User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)             ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4          ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4           ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2             ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2           ; 187: addu $a2 $a2 $v0
[00400014] 0c100009  jal 0x00400024 [main]     ; 188: jal main
[00400018] 00000000  nop                       ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10            ; 191: li $v0 10
[00400020] 0000000c  syscall                   ; 192: syscall # syscall 10 (exit)
[00400024] 34080002  ori $8, $0, 2             ; 6: ori $8,$0,0x2 # put two's comp. two into register 8
[00400028] 34090003  ori $9, $0, 3             ; 7: ori $9,$0,0x3 # put two's comp. three into register 9
[0040002c] 71095002  mul $10, $8, $9           ; 8: mul $10,$8,$9 # multiply register 8 and 9, put result in 10

                                              Kernel Text Segment [80000000]..[80010000]
[80000180] 0001d821  addu $27, $0, $1          ; 90: move $k1 $at # Save $at
[80000184] 3c019000  lui $1, -28672            ; 92: sw $v0 s1 # Not re-entrant and we can't trust $sp
[80000188] ac220200  sw $2, 512($1)
[8000018c] 3c019000  lui $1, -28672            ; 93: sw $a0 s2 # But we need to use these registers
[80000190] ac240204  sw $4, 516($1)
[80000194] 401a6800  mfc0 $26, $13             ; 95: mfc0 $k0 $13 # Cause register
[80000198] 001a2082  srl $4, $26, 2            ; 96: srl $a0 $k0 2 # Extract ExcCode Field
[8000019c] 3084001f  andi $4, $4, 31           ; 97: andi $a0 $a0 0x1f
[800001a0] 34020004  ori $2, $0, 4             ; 101: li $v0 4 # syscall 4 (print_str)
[800001a4] 3c049000  lui $4, -28672 [__m1_]    ; 102: la $a0 __m1_
[800001a8] 0000000c  syscall                   ; 103: syscall
[800001ac] 34020001  ori $2, $0, 1             ; 105: li $v0 1 # syscall 1 (print_int)
[800001b0] 001a2082  srl $4, $26, 2            ; 106: srl $a0 $k0 2 # Extract ExcCode Field
[800001b4] 3084001f  andi $4, $4, 31           ; 107: andi $a0 $a0 0x1f
[800001b8] 0000000c  syscall                   ; 108: syscall
[800001bc] 34020004  ori $2, $0, 4             ; 110: li $v0 4 # syscall 4 (print_str)
[800001c0] 3344003c  andi $4, $26, 60          ; 111: andi $a0 $k0 0x3c
[800001c4] 3c019000  lui $1, -28672            ; 112: lw $a0 __excp($a0)
```

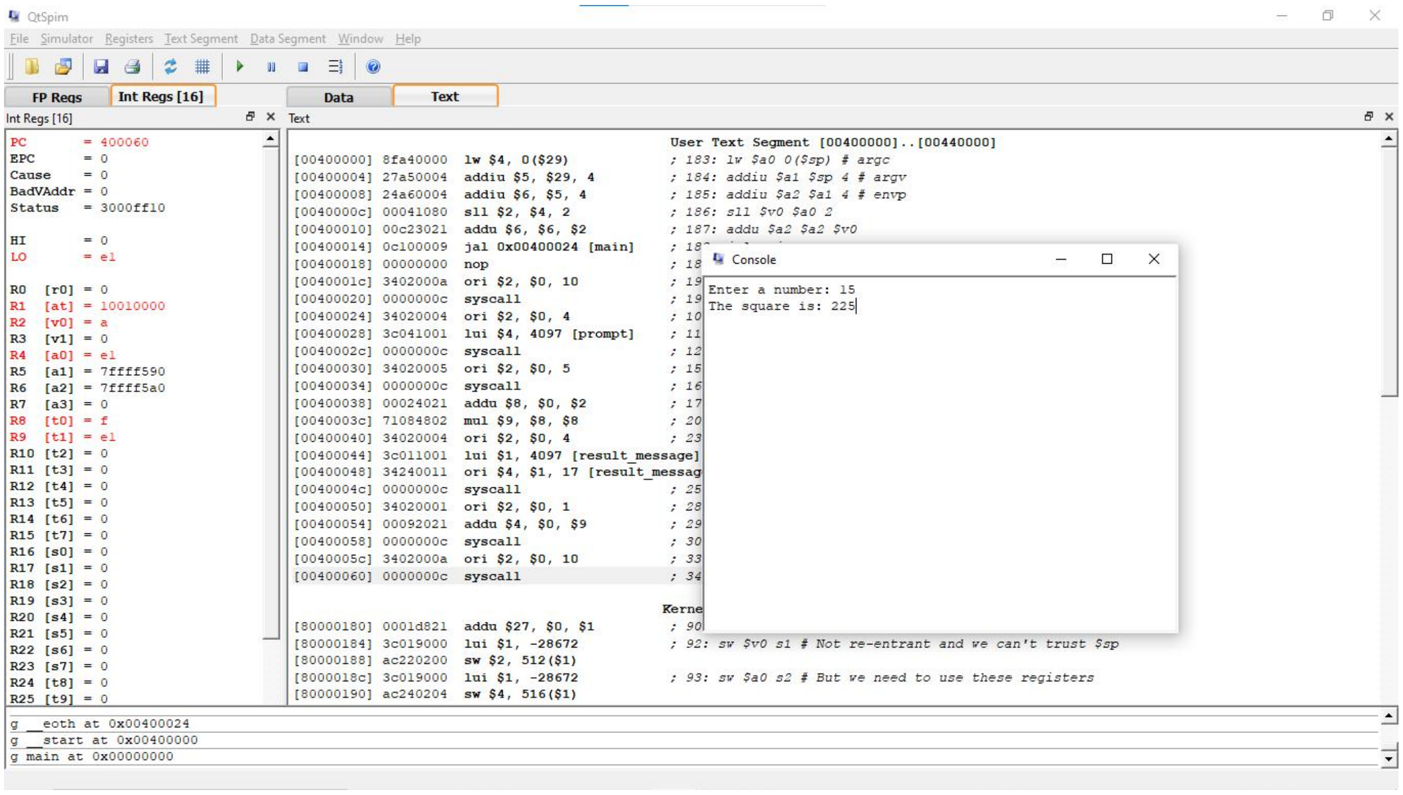
```
spim: (parser) syntax error on line 14 of file D:/college/coa/assignment lab 3/input_str.asm
```

Q7. Write a program to find out the square of an number and print the result to the console.

ANS :



```
.data
    prompt: .asciiz "Enter a number: "
    result_message: .asciiz "The square is: "

.text
.globl main

main:
    # Print a prompt to enter a number
    li $v0, 4
    la $a0, prompt
    syscall

    # Read an integer from the user
    li $v0, 5
    syscall
    move $t0, $v0  # Store the input in $t0

    # Calculate the square of the number
    mul $t1, $t0, $t0  # $t1 = $t0 * $t0

    # Print the result message
    li $v0, 4
    la $a0, result_message
    syscall

    # Print the square (the result)
    li $v0, 1
    move $a0, $t1
    syscall

    # Exit the program
    li $v0, 10
    syscall
```


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
Text
Console

```
PC       = 400060
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = e1

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = e1
R5  [a1] = 7ffff590
R6  [a2] = 7ffff5a0
R7  [a3] = 0
R8  [t0] = f
R9  [t1] = e1
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
```

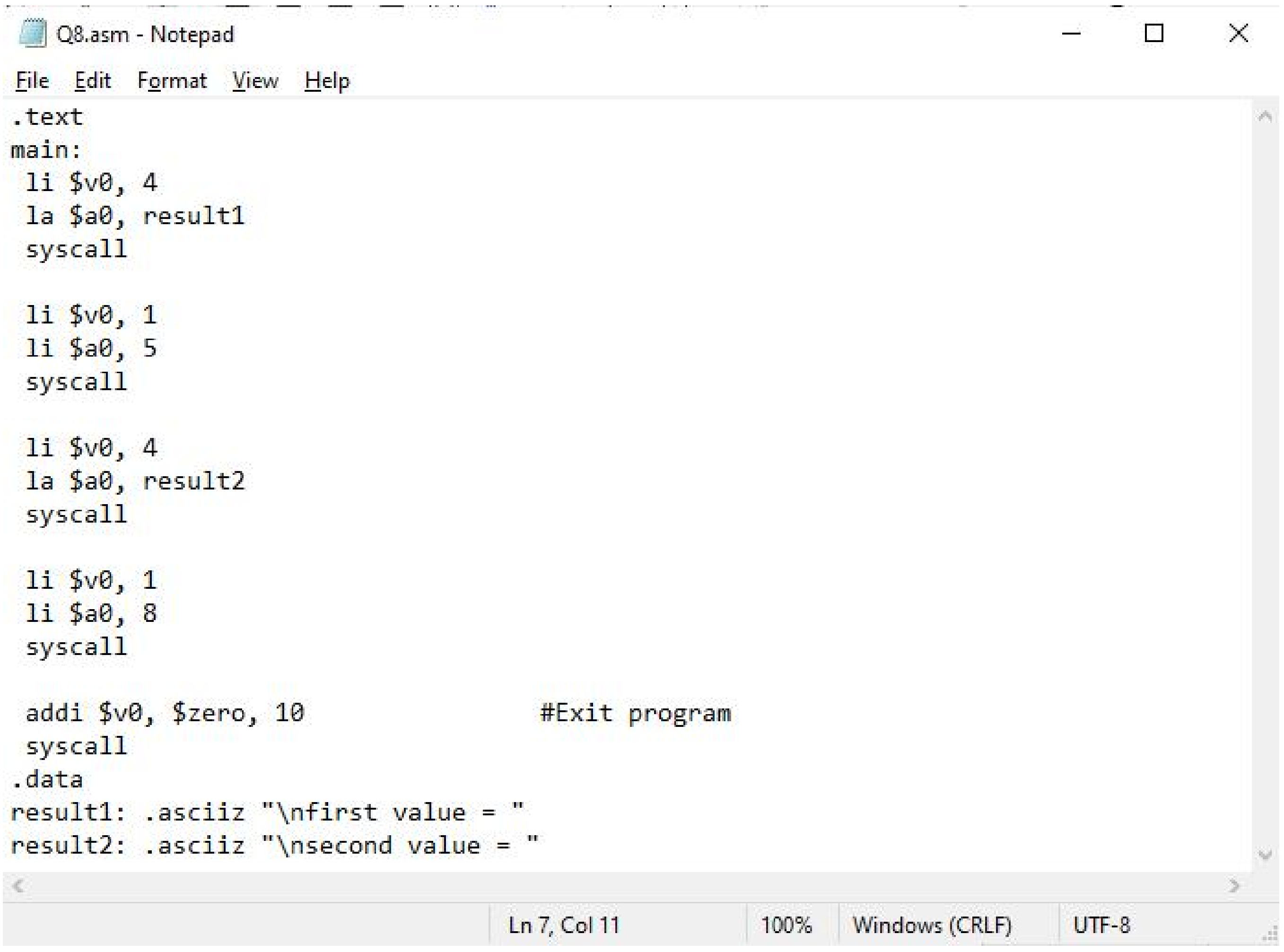
```
                                                  User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)                ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4             ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4              ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2                ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2              ; 187: addu $a2 $a2 $v0
[00400014] 0c100009  jal 0x00400024 [main]        ; 18
[00400018] 00000000  nop                          ; 18
[0040001c] 3402000a  ori $2, $0, 10               ; 19
[00400020] 0000000c  syscall                      ; 19
[00400024] 34020004  ori $2, $0, 4                ; 10
[00400028] 3c041001  lui $4, 4097 [prompt]        ; 11
[0040002c] 0000000c  syscall                      ; 12
[00400030] 34020005  ori $2, $0, 5                ; 15
[00400034] 0000000c  syscall                      ; 16
[00400038] 00024021  addu $8, $0, $2              ; 17
[0040003c] 71084802  mul $9, $8, $8               ; 20
[00400040] 34020004  ori $2, $0, 4                ; 23
[00400044] 3c011001  lui $1, 4097 [result_message]
[00400048] 34240011  ori $4, $1, 17 [result_messag
[0040004c] 0000000c  syscall                      ; 25
[00400050] 34020001  ori $2, $0, 1                ; 28
[00400054] 00092021  addu $4, $0, $9              ; 29
[00400058] 0000000c  syscall                      ; 30
[0040005c] 3402000a  ori $2, $0, 10               ; 33
[00400060] 0000000c  syscall                      ; 34

                                                 Kerne
[80000180] 0001d821  addu $27, $0, $1             ; 90
[80000184] 3c019000  lui $1, -28672               ; 92: sw $v0 s1 # Not re-entrant and we can't trust $sp
[80000188] ac220200  sw $2, 512($1)
[8000018c] 3c019000  lui $1, -28672               ; 93: sw $a0 s2 # But we need to use these registers
[80000190] ac240204  sw $4, 516($1)
```

```
Enter a number: 15
The square is: 225
```

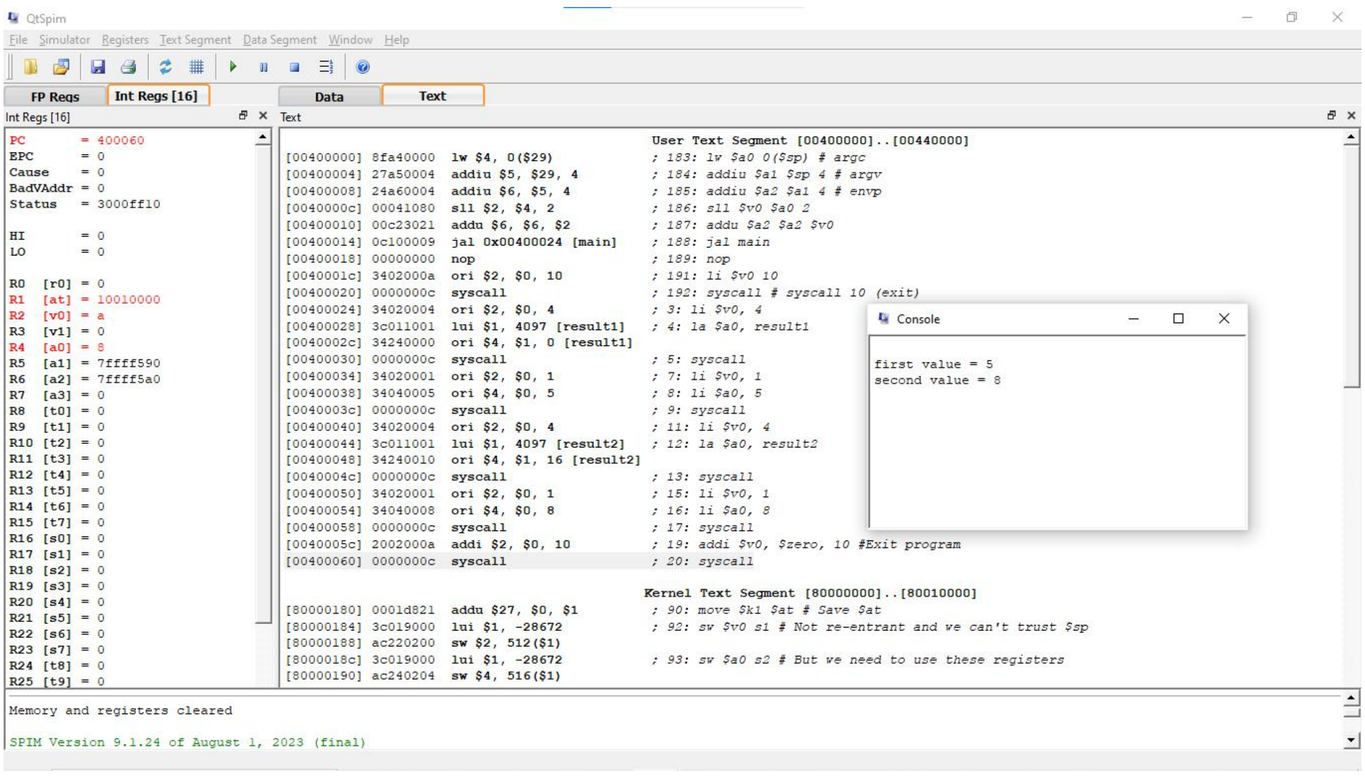
```
g __eoth at 0x00400024
g __start at 0x00400000
g main at 0x00000000
```

Q8. Convert the following program

ANS :



```
.text
main:
 li $v0, 4
 la $a0, result1
 syscall

 li $v0, 1
 li $a0, 5
 syscall

 li $v0, 4
 la $a0, result2
 syscall

 li $v0, 1
 li $a0, 8
 syscall

 addi $v0, $zero, 10            #Exit program
 syscall
.data
result1: .asciiz "\nfirst value = "
result2: .asciiz "\nsecond value = "
```



QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
Text
Console

```
PC       = 400060
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 8
R5  [a1] = 7ffff590
R6  [a2] = 7ffff5a0
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] = 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
```

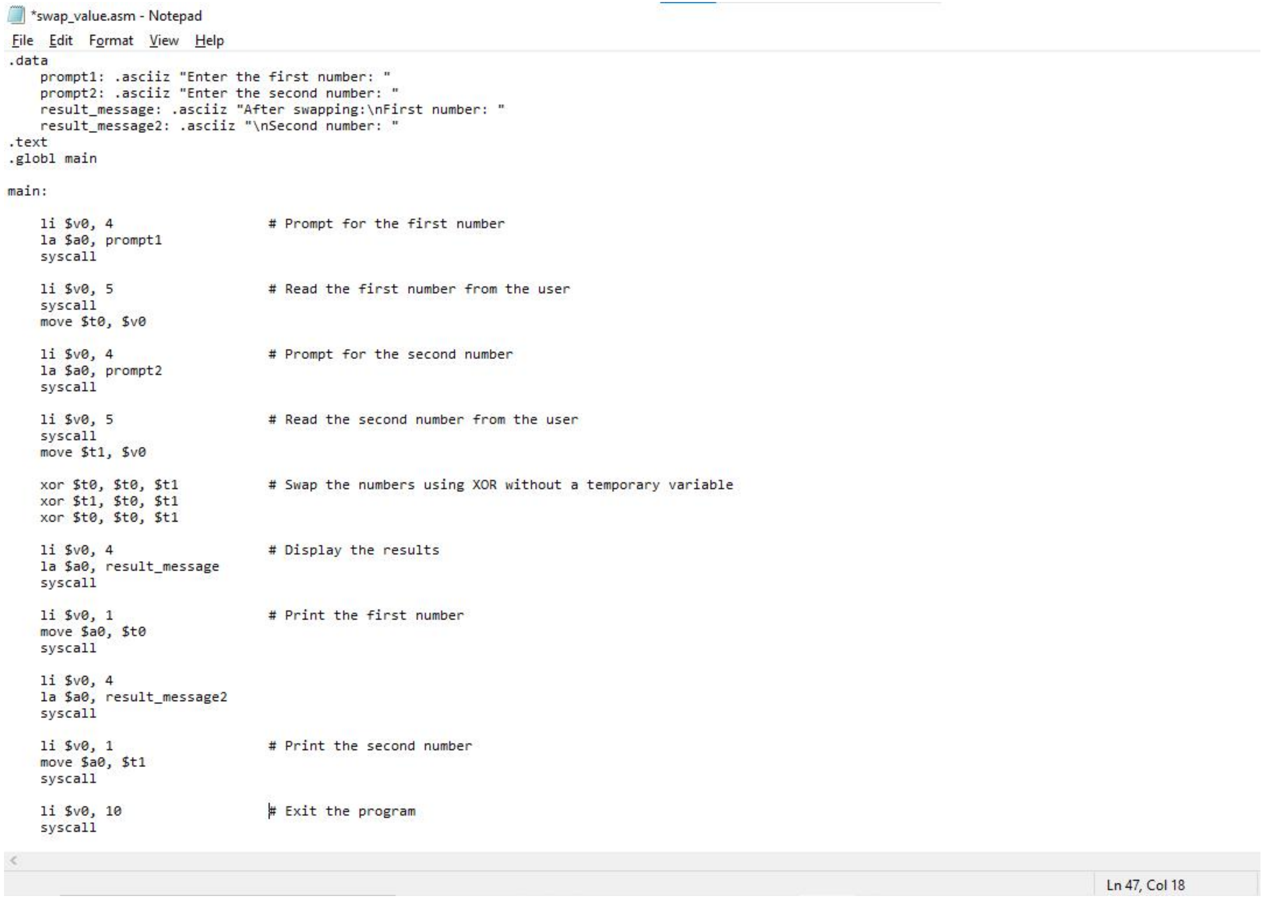
```
                                                User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)              ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4           ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4            ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2              ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2            ; 187: addu $a2 $a2 $v0
[00400014] 0c100009  jal 0x00400024 [main]      ; 188: jal main
[00400018] 00000000  nop                        ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10             ; 191: li $v0 10
[00400020] 0000000c  syscall                    ; 192: syscall # syscall 10 (exit)
[00400024] 34020004  ori $2, $0, 4              ; 3: li $v0, 4
[00400028] 3c011001  lui $1, 4097 [result1]     ; 4: la $a0, result1
[0040002c] 34240000  ori $4, $1, 0 [result1]
[00400030] 0000000c  syscall                    ; 5: syscall
[00400034] 34020001  ori $2, $0, 1              ; 7: li $v0, 1
[00400038] 34040005  ori $4, $0, 5              ; 8: li $a0, 5
[0040003c] 0000000c  syscall                    ; 9: syscall
[00400040] 34020004  ori $2, $0, 4              ; 11: li $v0, 4
[00400044] 3c011001  lui $1, 4097 [result2]     ; 12: la $a0, result2
[00400048] 34240010  ori $4, $1, 16 [result2]
[0040004c] 0000000c  syscall                    ; 13: syscall
[00400050] 34020001  ori $2, $0, 1              ; 15: li $v0, 1
[00400054] 34040008  ori $4, $0, 8              ; 16: li $a0, 8
[00400058] 0000000c  syscall                    ; 17: syscall
[0040005c] 2002000a  addi $2, $0, 10            ; 19: addi $v0, $zero, 10 #Exit program
[00400060] 0000000c  syscall                    ; 20: syscall

                                               Kernel Text Segment [80000000]..[80010000]
[80000180] 0001d821  addu $27, $0, $1           ; 90: move $k1 $at # Save $at
[80000184] 3c019000  lui $1, -28672             ; 92: sw $v0 s1 # Not re-entrant and we can't trust $sp
[80000188] ac220200  sw $2, 512($1)
[8000018c] 3c019000  lui $1, -28672             ; 93: sw $a0 s2 # But we need to use these registers
[80000190] ac240204  sw $4, 516($1)
```

```

first value = 5
second value = 8
```

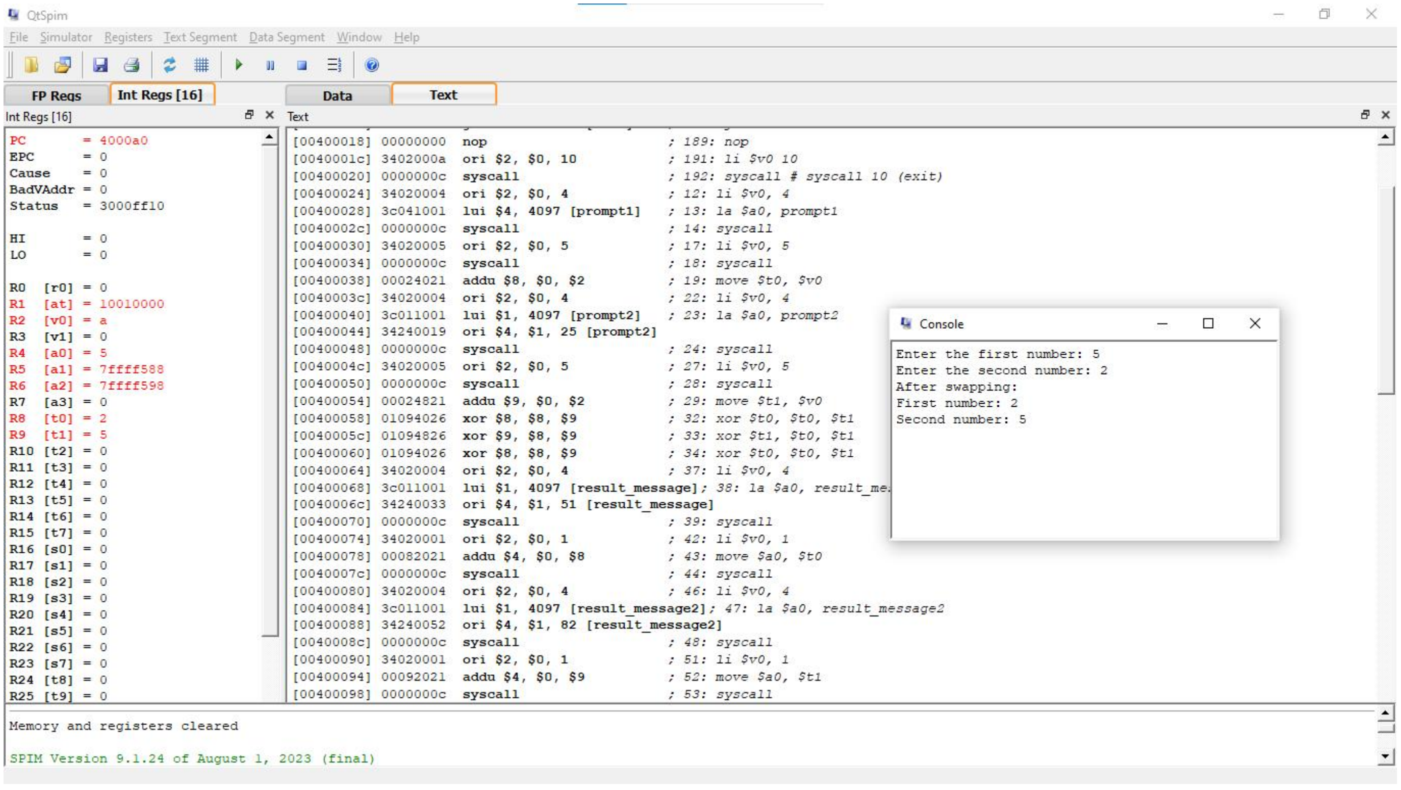
```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
```

Q9. Write a program to retrieve two numbers from a user, and swap those number using only the XOR operation. You should not use a temporary variable to store the numbers while swapping them. Your program should include a proper and useful prompt for input, and print the results in a meaningful manner.

ANS :



```
.data
    prompt1: .asciiz "Enter the first number: "
    prompt2: .asciiz "Enter the second number: "
    result_message: .asciiz "After swapping:\nFirst number: "
    result_message2: .asciiz "\nSecond number: "
.text
.globl main

main:

    li $v0, 4                   # Prompt for the first number
    la $a0, prompt1
    syscall

    li $v0, 5                   # Read the first number from the user
    syscall
    move $t0, $v0

    li $v0, 4                   # Prompt for the second number
    la $a0, prompt2
    syscall

    li $v0, 5                   # Read the second number from the user
    syscall
    move $t1, $v0

    xor $t0, $t0, $t1           # Swap the numbers using XOR without a temporary variable
    xor $t1, $t0, $t1
    xor $t0, $t0, $t1

    li $v0, 4                   # Display the results
    la $a0, result_message
    syscall

    li $v0, 1                   # Print the first number
    move $a0, $t0
    syscall

    li $v0, 4
    la $a0, result_message2
    syscall

    li $v0, 1                   # Print the second number
    move $a0, $t1
    syscall

    li $v0, 10                  # Exit the program
    syscall
```


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Console

```
PC       = 4000a0
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 5
R5  [a1] = 7ffff588
R6  [a2] = 7ffff598
R7  [a3] = 0
R8  [t0] = 2
R9  [t1] = 5
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
```

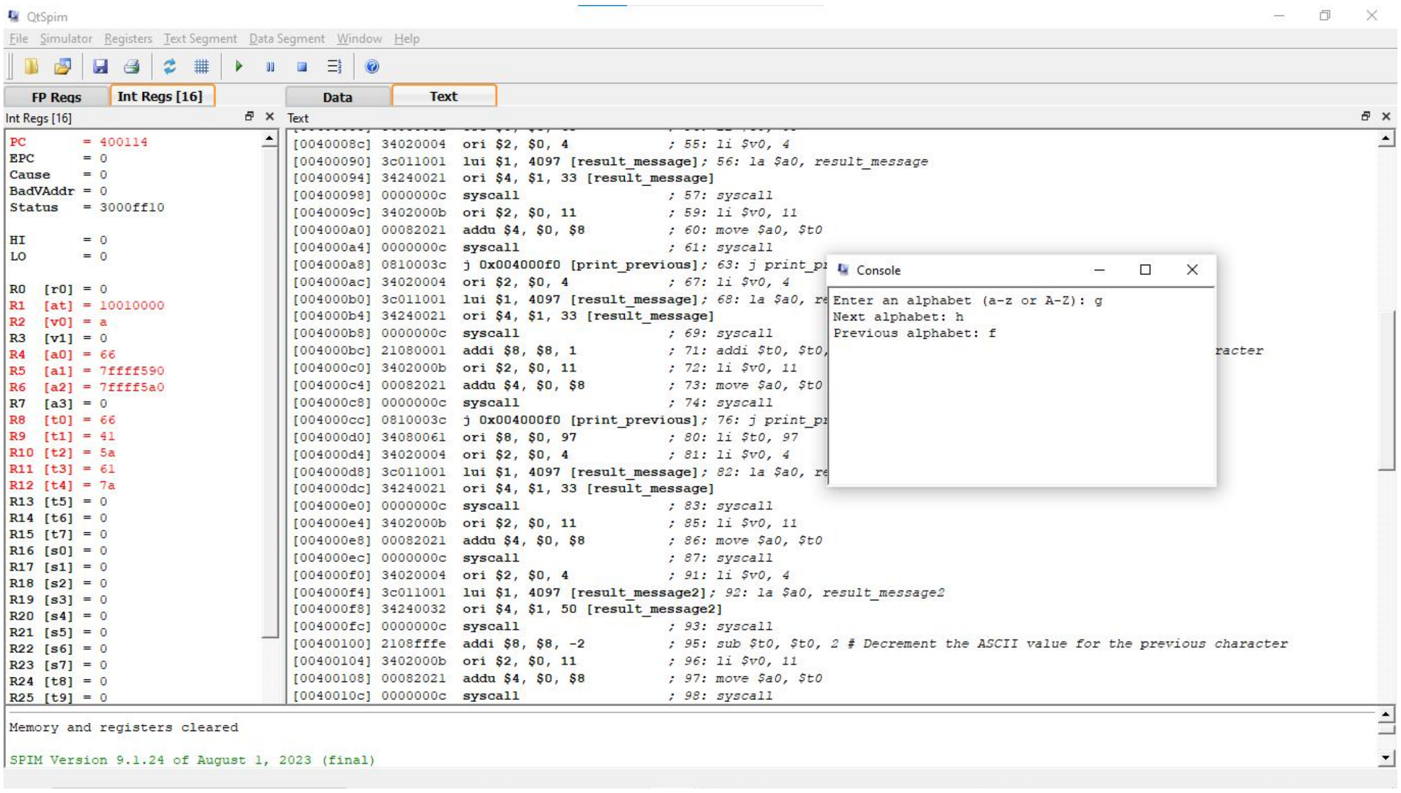
```
[00400018] 00000000  nop                        ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10             ; 191: li $v0 10
[00400020] 0000000c  syscall                    ; 192: syscall # syscall 10 (exit)
[00400024] 34020004  ori $2, $0, 4              ; 12: li $v0, 4
[00400028] 3c041001  lui $4, 4097 [prompt1]     ; 13: la $a0, prompt1
[0040002c] 0000000c  syscall                    ; 14: syscall
[00400030] 34020005  ori $2, $0, 5              ; 17: li $v0, 5
[00400034] 0000000c  syscall                    ; 18: syscall
[00400038] 00024021  addu $8, $0, $2            ; 19: move $t0, $v0
[0040003c] 34020004  ori $2, $0, 4              ; 22: li $v0, 4
[00400040] 3c011001  lui $1, 4097 [prompt2]     ; 23: la $a0, prompt2
[00400044] 34240019  ori $4, $1, 25 [prompt2]
[00400048] 0000000c  syscall                    ; 24: syscall
[0040004c] 34020005  ori $2, $0, 5              ; 27: li $v0, 5
[00400050] 0000000c  syscall                    ; 28: syscall
[00400054] 00024821  addu $9, $0, $2            ; 29: move $t1, $v0
[00400058] 01094026  xor $8, $8, $9             ; 32: xor $t0, $t0, $t1
[0040005c] 01094826  xor $9, $8, $9             ; 33: xor $t1, $t0, $t1
[00400060] 01094026  xor $8, $8, $9             ; 34: xor $t0, $t0, $t1
[00400064] 34020004  ori $2, $0, 4              ; 37: li $v0, 4
[00400068] 3c011001  lui $1, 4097 [result_message]; 38: la $a0, result_me
[0040006c] 34240033  ori $4, $1, 51 [result_message]
[00400070] 0000000c  syscall                    ; 39: syscall
[00400074] 34020001  ori $2, $0, 1              ; 42: li $v0, 1
[00400078] 00082021  addu $4, $0, $8            ; 43: move $a0, $t0
[0040007c] 0000000c  syscall                    ; 44: syscall
[00400080] 34020004  ori $2, $0, 4              ; 46: li $v0, 4
[00400084] 3c011001  lui $1, 4097 [result_message2]; 47: la $a0, result_message2
[00400088] 34240052  ori $4, $1, 82 [result_message2]
[0040008c] 0000000c  syscall                    ; 48: syscall
[00400090] 34020001  ori $2, $0, 1              ; 51: li $v0, 1
[00400094] 00092021  addu $4, $0, $9            ; 52: move $a0, $t1
[00400098] 0000000c  syscall                    ; 53: syscall
```

```
Enter the first number: 5
Enter the second number: 2
After swapping:
First number: 2
Second number: 5
```

```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
```

Q10. Write an Assembly code which take an alphabet from user and print the next and previous alphabets on the screen.

ANS :



QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
Text
Console

```
PC       = 400114
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 66
R5  [a1] = 7ffff590
R6  [a2] = 7ffff5a0
R7  [a3] = 0
R8  [t0] = 66
R9  [t1] = 41
R10 [t2] = 5a
R11 [t3] = 61
R12 [t4] = 7a
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
```

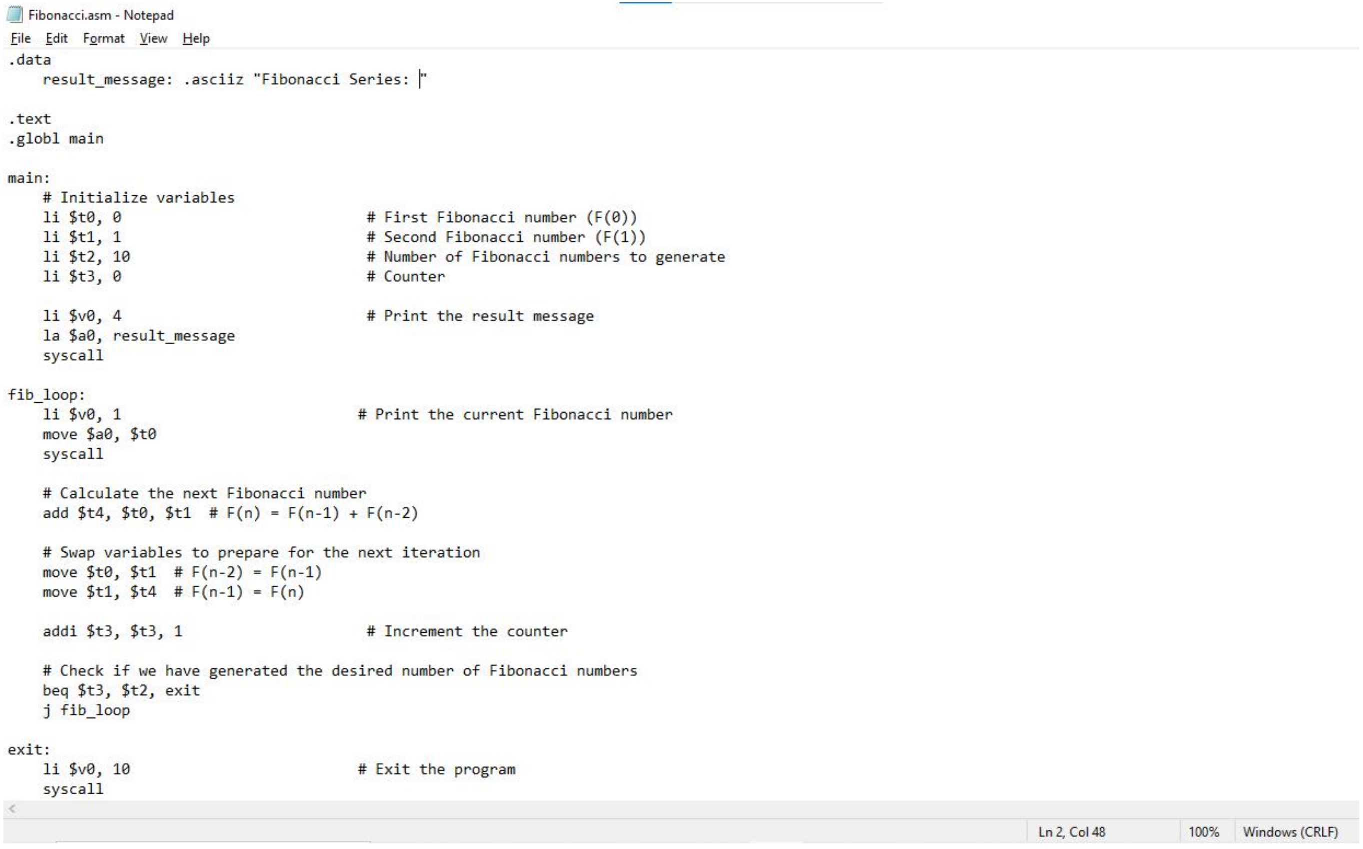
```
[0040008c] 34020004  ori $2, $0, 4            ; 55: li $v0, 4
[00400090] 3c011001  lui $1, 4097 [result_message]; 56: la $a0, result_message
[00400094] 34240021  ori $4, $1, 33 [result_message]
[00400098] 0000000c  syscall                  ; 57: syscall
[0040009c] 3402000b  ori $2, $0, 11           ; 59: li $v0, 11
[004000a0] 00082021  addu $4, $0, $8          ; 60: move $a0, $t0
[004000a4] 0000000c  syscall                  ; 61: syscall
[004000a8] 0810003c  j 0x004000f0 [print_previous]; 63: j print_pr
[004000ac] 34020004  ori $2, $0, 4            ; 67: li $v0, 4
[004000b0] 3c011001  lui $1, 4097 [result_message]; 68: la $a0, re
[004000b4] 34240021  ori $4, $1, 33 [result_message]
[004000b8] 0000000c  syscall                  ; 69: syscall
[004000bc] 21080001  addi $8, $8, 1           ; 71: addi $t0, $t0,                         racter
[004000c0] 3402000b  ori $2, $0, 11           ; 72: li $v0, 11
[004000c4] 00082021  addu $4, $0, $8          ; 73: move $a0, $t0
[004000c8] 0000000c  syscall                  ; 74: syscall
[004000cc] 0810003c  j 0x004000f0 [print_previous]; 76: j print_pr
[004000d0] 34080061  ori $8, $0, 97           ; 80: li $t0, 97
[004000d4] 34020004  ori $2, $0, 4            ; 81: li $v0, 4
[004000d8] 3c011001  lui $1, 4097 [result_message]; 82: la $a0, re
[004000dc] 34240021  ori $4, $1, 33 [result_message]
[004000e0] 0000000c  syscall                  ; 83: syscall
[004000e4] 3402000b  ori $2, $0, 11           ; 85: li $v0, 11
[004000e8] 00082021  addu $4, $0, $8          ; 86: move $a0, $t0
[004000ec] 0000000c  syscall                  ; 87: syscall
[004000f0] 34020004  ori $2, $0, 4            ; 91: li $v0, 4
[004000f4] 3c011001  lui $1, 4097 [result_message2]; 92: la $a0, result_message2
[004000f8] 34240032  ori $4, $1, 50 [result_message2]
[004000fc] 0000000c  syscall                  ; 93: syscall
[00400100] 2108fffe  addi $8, $8, -2          ; 95: sub $t0, $t0, 2 # Decrement the ASCII value for the previous character
[00400104] 3402000b  ori $2, $0, 11           ; 96: li $v0, 11
[00400108] 00082021  addu $4, $0, $8          ; 97: move $a0, $t0
[0040010c] 0000000c  syscall                  ; 98: syscall
```

```
Enter an alphabet (a-z or A-Z): g
Next alphabet: h
Previous alphabet: f
```

```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
```

Q11. Implementation of Fibonacci series program In MIPS.

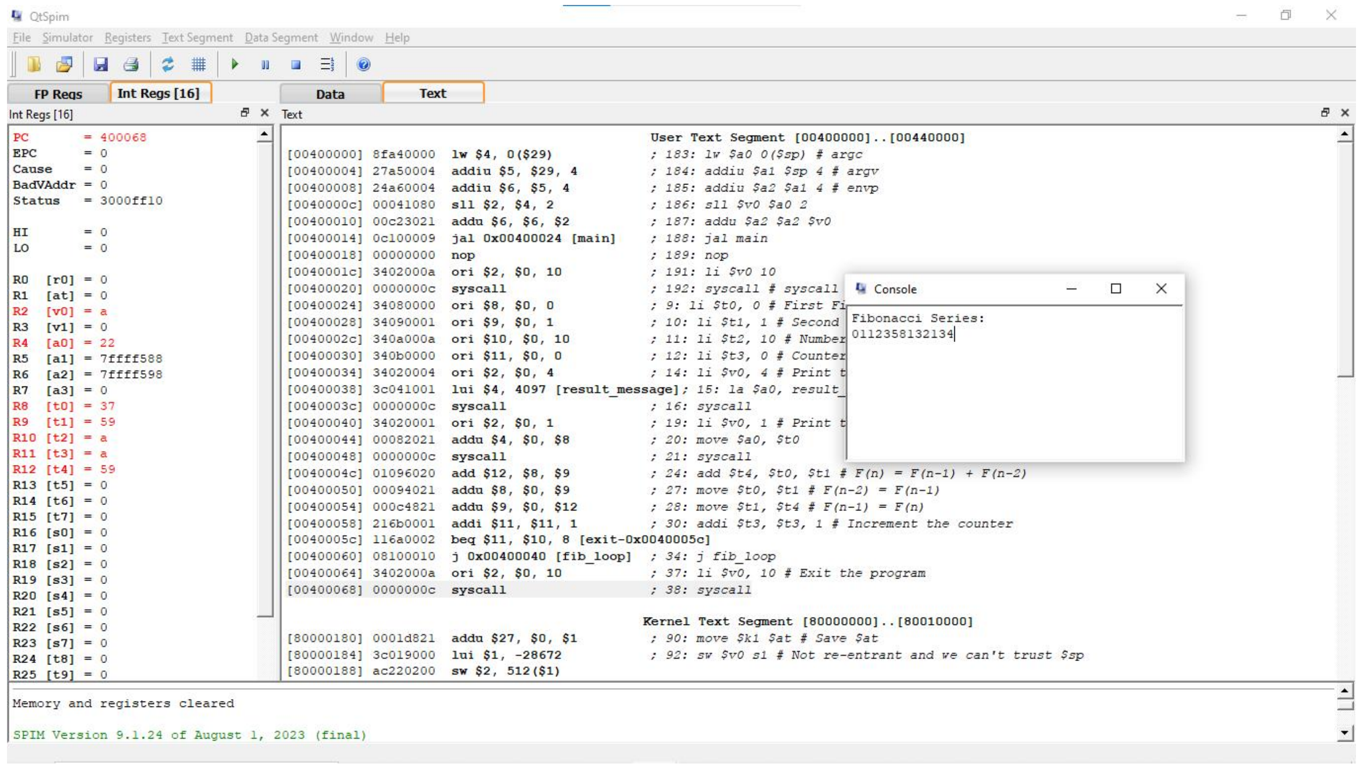
ANS :



Fibonacci.asm - Notepad

```
.data
    result_message: .asciiz "Fibonacci Series: "

.text
.globl main

main:
    # Initialize variables
    li $t0, 0                       # First Fibonacci number (F(0))
    li $t1, 1                       # Second Fibonacci number (F(1))
    li $t2, 10                      # Number of Fibonacci numbers to generate
    li $t3, 0                       # Counter

    li $v0, 4                       # Print the result message
    la $a0, result_message
    syscall

fib_loop:
    li $v0, 1                      # Print the current Fibonacci number
    move $a0, $t0
    syscall

    # Calculate the next Fibonacci number
    add $t4, $t0, $t1  # F(n) = F(n-1) + F(n-2)

    # Swap variables to prepare for the next iteration
    move $t0, $t1  # F(n-2) = F(n-1)
    move $t1, $t4  # F(n-1) = F(n)

    addi $t3, $t3, 1                # Increment the counter

    # Check if we have generated the desired number of Fibonacci numbers
    beq $t3, $t2, exit
    j fib_loop

exit:
    li $v0, 10                     # Exit the program
    syscall
```

Ln 2, Col 48   100%   Windows (CRLF)



QtSpim

```
PC       = 400068
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 22
R5  [a1] = 7ffff588
R6  [a2] = 7ffff598
R7  [a3] = 0
R8  [t0] = 37
R9  [t1] = 59
R10 [t2] = a
R11 [t3] = a
R12 [t4] = 59
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
```

```
                                          User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)           ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4        ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4         ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2           ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2         ; 187: addu $a2 $a2 $v0
[00400014] 0c100009  jal 0x00400024 [main]   ; 188: jal main
[00400018] 00000000  nop                     ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10          ; 191: li $v0 10
[00400020] 0000000c  syscall                 ; 192: syscall # syscall
[00400024] 34080000  ori $8, $0, 0           ; 9: li $t0, 0 # First Fi
[00400028] 34090001  ori $9, $0, 1           ; 10: li $t1, 1 # Second
[0040002c] 340a000a  ori $10, $0, 10         ; 11: li $t2, 10 # Number
[00400030] 340b0000  ori $11, $0, 0          ; 12: li $t3, 0 # Counter
[00400034] 34020004  ori $2, $0, 4           ; 14: li $v0, 4 # Print t
[00400038] 3c041001  lui $4, 4097 [result_message]; 15: la $a0, result_
[0040003c] 0000000c  syscall                 ; 16: syscall
[00400040] 34020001  ori $2, $0, 1           ; 19: li $v0, 1 # Print t
[00400044] 00082021  addu $4, $0, $8         ; 20: move $a0, $t0
[00400048] 0000000c  syscall                 ; 21: syscall
[0040004c] 01096020  add $12, $8, $9         ; 24: add $t4, $t0, $t1 # F(n) = F(n-1) + F(n-2)
[00400050] 00094021  addu $8, $0, $9         ; 27: move $t0, $t1 # F(n-2) = F(n-1)
[00400054] 000c4821  addu $9, $0, $12        ; 28: move $t1, $t4 # F(n-1) = F(n)
[00400058] 216b0001  addi $11, $11, 1        ; 30: addi $t3, $t3, 1 # Increment the counter
[0040005c] 116a0002  beq $11, $10, 8 [exit-0x0040005c]
[00400060] 08100010  j 0x00400040 [fib_loop]  ; 34: j fib_loop
[00400064] 3402000a  ori $2, $0, 10          ; 37: li $v0, 10 # Exit the program
[00400068] 0000000c  syscall                 ; 38: syscall

                                          Kernel Text Segment [80000000]..[80010000]
[80000180] 0001d821  addu $27, $0, $1        ; 90: move $k1 $at # Save $at
[80000184] 3c019000  lui $1, -28672          ; 92: sw $v0 s1 # Not re-entrant and we can't trust $sp
[80000188] ac220200  sw $2, 512($1)
```

Console

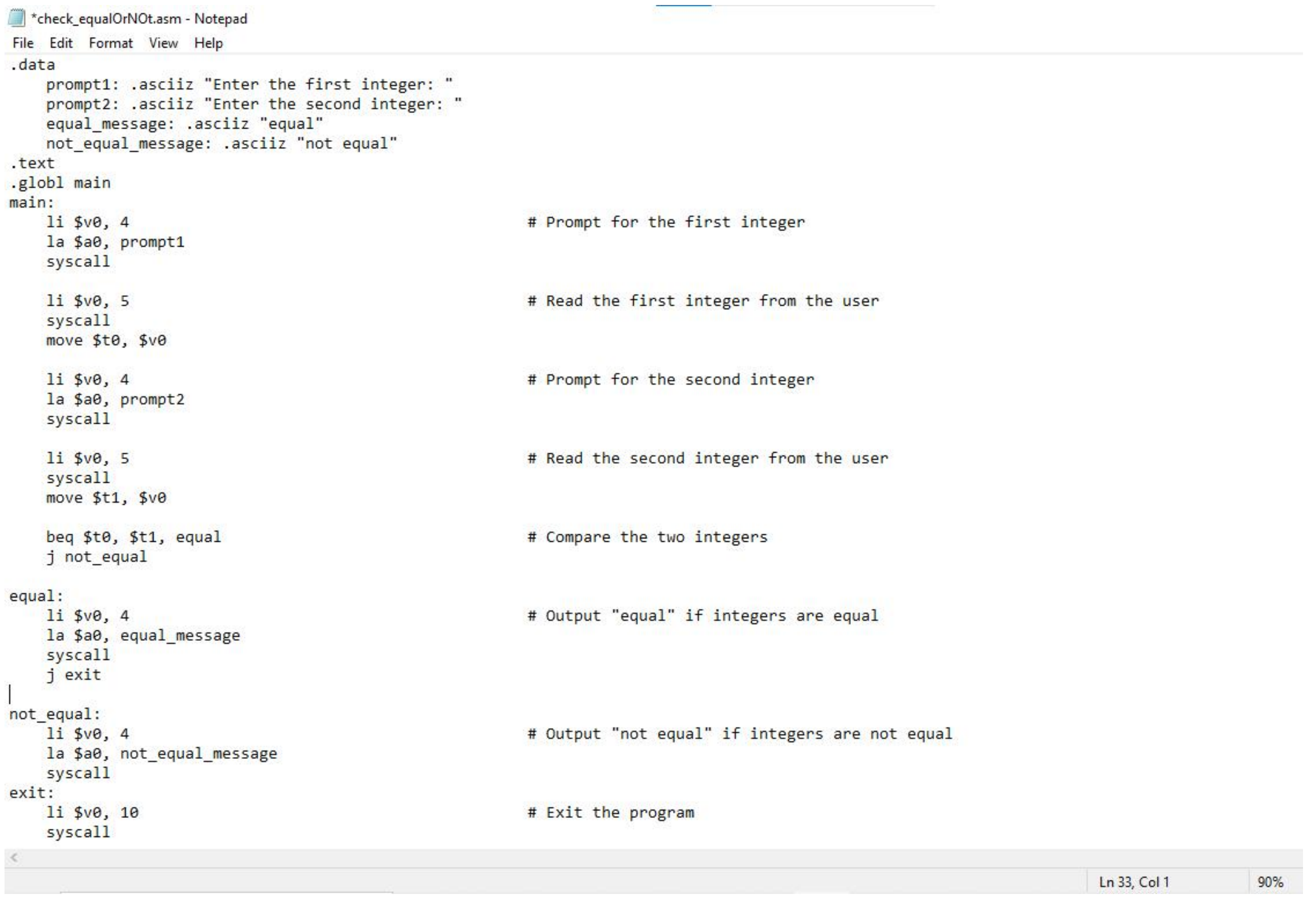
```
Fibonacci Series:
0112358132134
```

```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
```

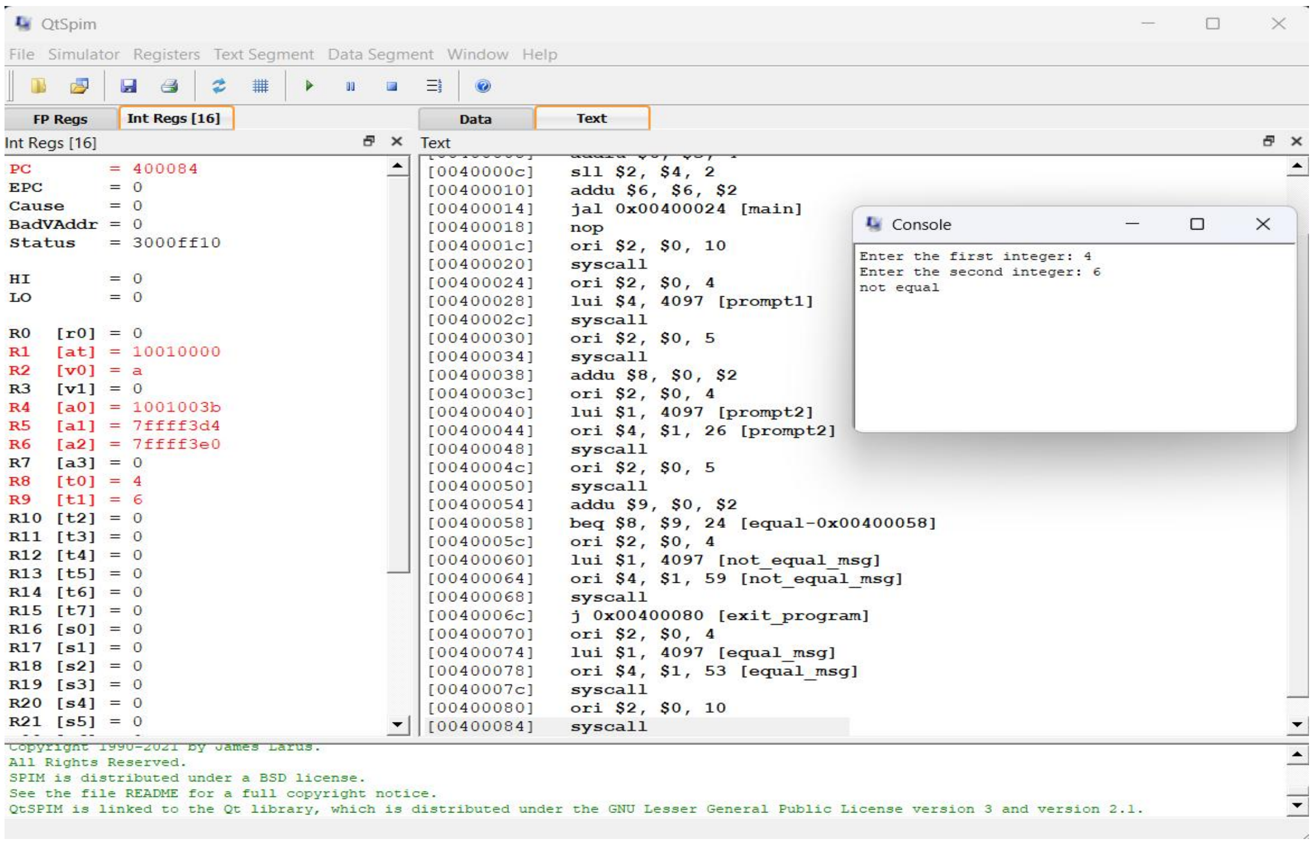
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text

Q12. Write a MIPS program that inputs two integer values. The program should output equal if the two integers are equal. Otherwise, it should output not equal. Use the branch instruction to check for equality.

ANS :



```
.data
    prompt1: .asciiz "Enter the first integer: "
    prompt2: .asciiz "Enter the second integer: "
    equal_message: .asciiz "equal"
    not_equal_message: .asciiz "not equal"
.text
.globl main
main:
    li $v0, 4                                    # Prompt for the first integer
    la $a0, prompt1
    syscall

    li $v0, 5                                    # Read the first integer from the user
    syscall
    move $t0, $v0

    li $v0, 4                                    # Prompt for the second integer
    la $a0, prompt2
    syscall

    li $v0, 5                                    # Read the second integer from the user
    syscall
    move $t1, $v0

    beq $t0, $t1, equal                          # Compare the two integers
    j not_equal

equal:
    li $v0, 4                                    # Output "equal" if integers are equal
    la $a0, equal_message
    syscall
    j exit

not_equal:
    li $v0, 4                                    # Output "not equal" if integers are not equal
    la $a0, not_equal_message
    syscall
exit:
    li $v0, 10                                   # Exit the program
    syscall
```


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
Text
Console

```
PC       = 400084
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 1001003b
R5  [a1] = 7ffff3d4
R6  [a2] = 7ffff3e0
R7  [a3] = 0
R8  [t0] = 4
R9  [t1] = 6
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
```

```
[0040000c]   sll $2, $4, 2
[00400010]   addu $6, $6, $2
[00400014]   jal 0x00400024 [main]
[00400018]   nop
[0040001c]   ori $2, $0, 10
[00400020]   syscall
[00400024]   ori $2, $0, 4
[00400028]   lui $4, 4097 [prompt1]
[0040002c]   syscall
[00400030]   ori $2, $0, 5
[00400034]   syscall
[00400038]   addu $8, $0, $2
[0040003c]   ori $2, $0, 4
[00400040]   lui $1, 4097 [prompt2]
[00400044]   ori $4, $1, 26 [prompt2]
[00400048]   syscall
[0040004c]   ori $2, $0, 5
[00400050]   syscall
[00400054]   addu $9, $0, $2
[00400058]   beq $8, $9, 24 [equal-0x00400058]
[0040005c]   ori $2, $0, 4
[00400060]   lui $1, 4097 [not_equal_msg]
[00400064]   ori $4, $1, 59 [not_equal_msg]
[00400068]   syscall
[0040006c]   j 0x00400080 [exit_program]
[00400070]   ori $2, $0, 4
[00400074]   lui $1, 4097 [equal_msg]
[00400078]   ori $4, $1, 53 [equal_msg]
[0040007c]   syscall
[00400080]   ori $2, $0, 10
[00400084]   syscall
```

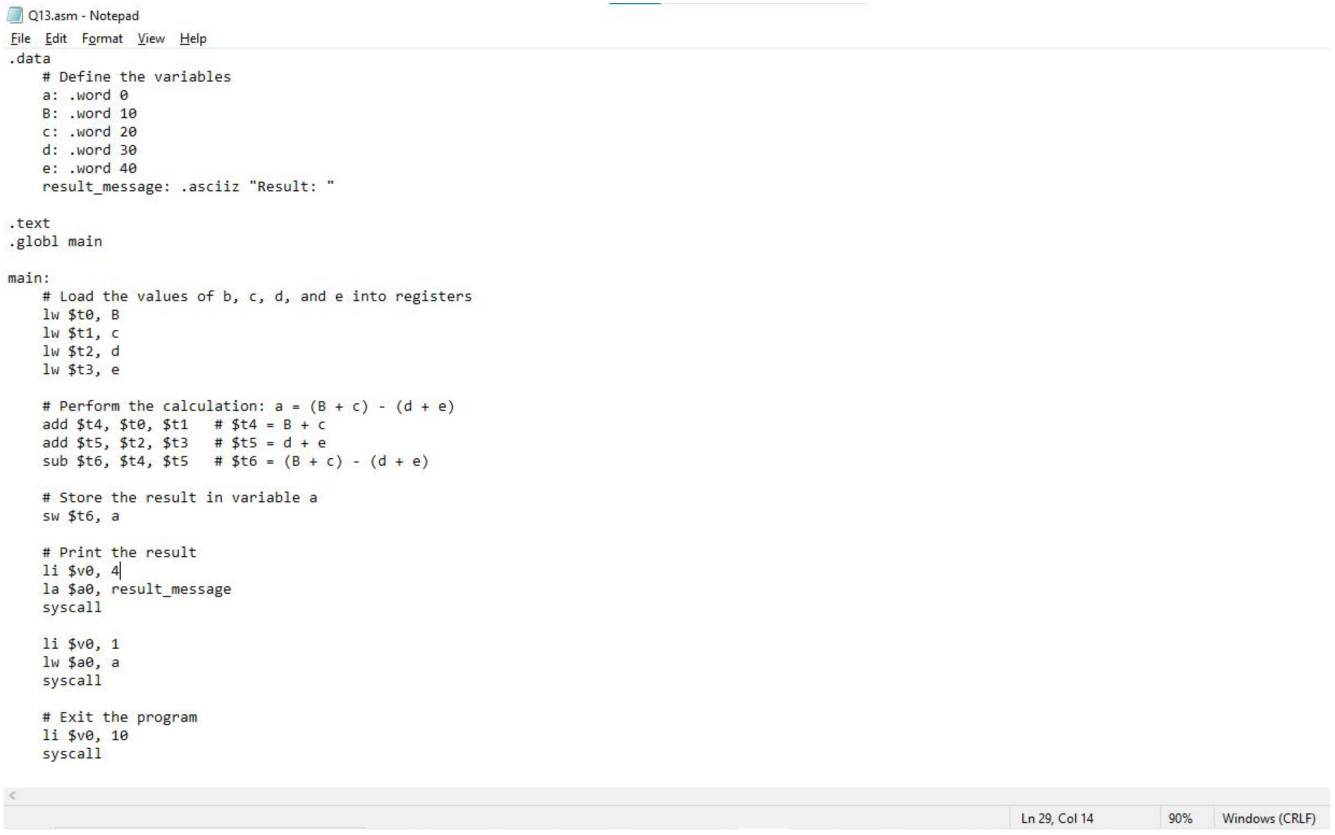
```
Enter the first integer: 4
Enter the second integer: 6
not equal
```

```
Copyright 1990-2021 by James Larus.
All Rights Reserved.
SPIM is distributed under a BSD license.
See the file README for a full copyright notice.
QtSPIM is linked to the Qt library, which is distributed under the GNU Lesser General Public License version 3 and version 2.1.
```

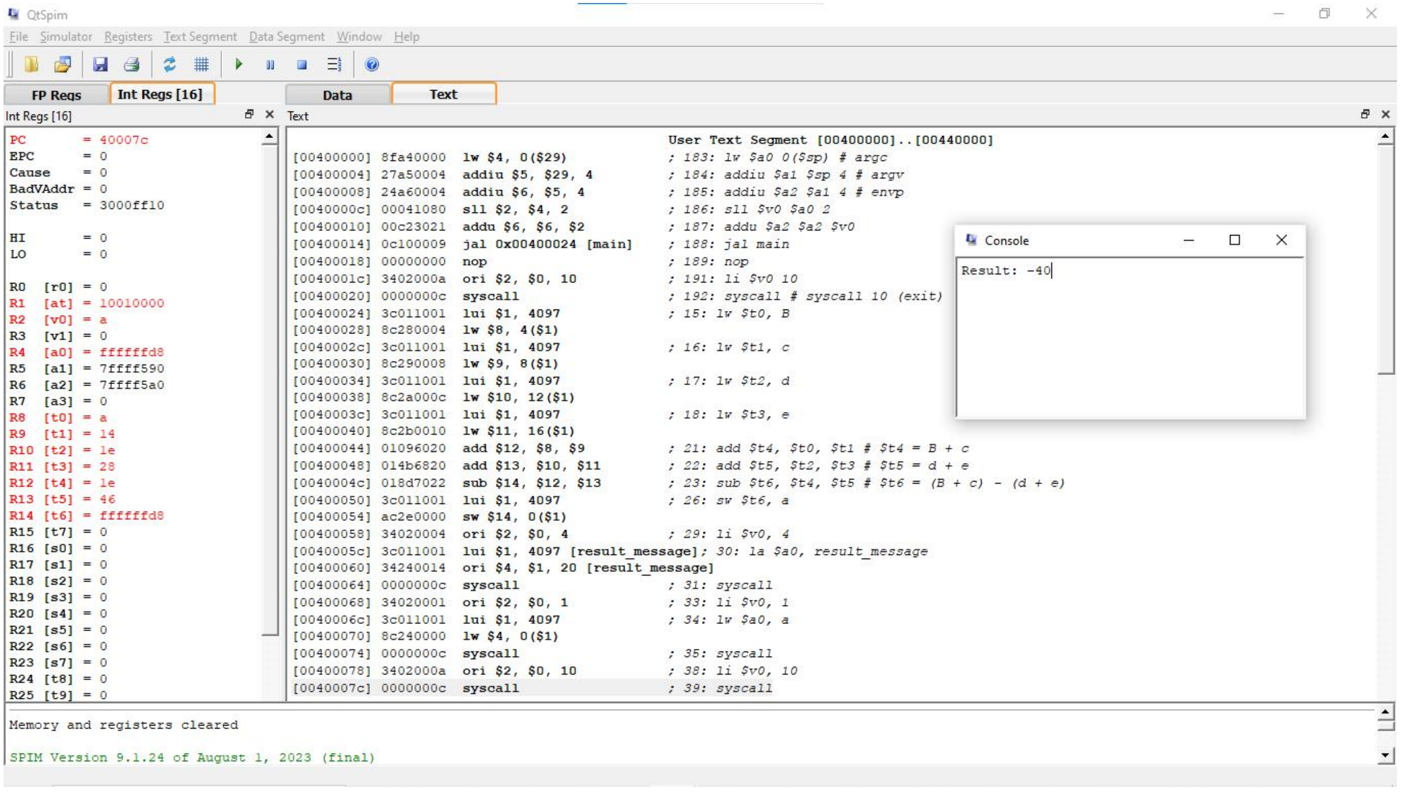
Q13. Variables are of type unsigned integers and word sized. Also the variables b, c, d and e are initialized to values 10, 20, 30 and 40 respectively.

a = (b + c) - (d + e)

ANS :



```
.data
    # Define the variables
    a: .word 0
    B: .word 10
    c: .word 20
    d: .word 30
    e: .word 40
    result_message: .asciiz "Result: "

.text
.globl main

main:
    # Load the values of b, c, d, and e into registers
    lw $t0, B
    lw $t1, c
    lw $t2, d
    lw $t3, e

    # Perform the calculation: a = (B + c) - (d + e)
    add $t4, $t0, $t1   # $t4 = B + c
    add $t5, $t2, $t3   # $t5 = d + e
    sub $t6, $t4, $t5   # $t6 = (B + c) - (d + e)

    # Store the result in variable a
    sw $t6, a

    # Print the result
    li $v0, 4
    la $a0, result_message
    syscall

    li $v0, 1
    lw $a0, a
    syscall

    # Exit the program
    li $v0, 10
    syscall
```



QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
Text
Console

```
PC       = 40007c
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = ffffffd8
R5  [a1] = 7ffff590
R6  [a2] = 7ffff5a0
R7  [a3] = 0
R8  [t0] = a
R9  [t1] = 14
R10 [t2] = 1e
R11 [t3] = 28
R12 [t4] = 1e
R13 [t5] = 46
R14 [t6] = ffffffd8
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
```

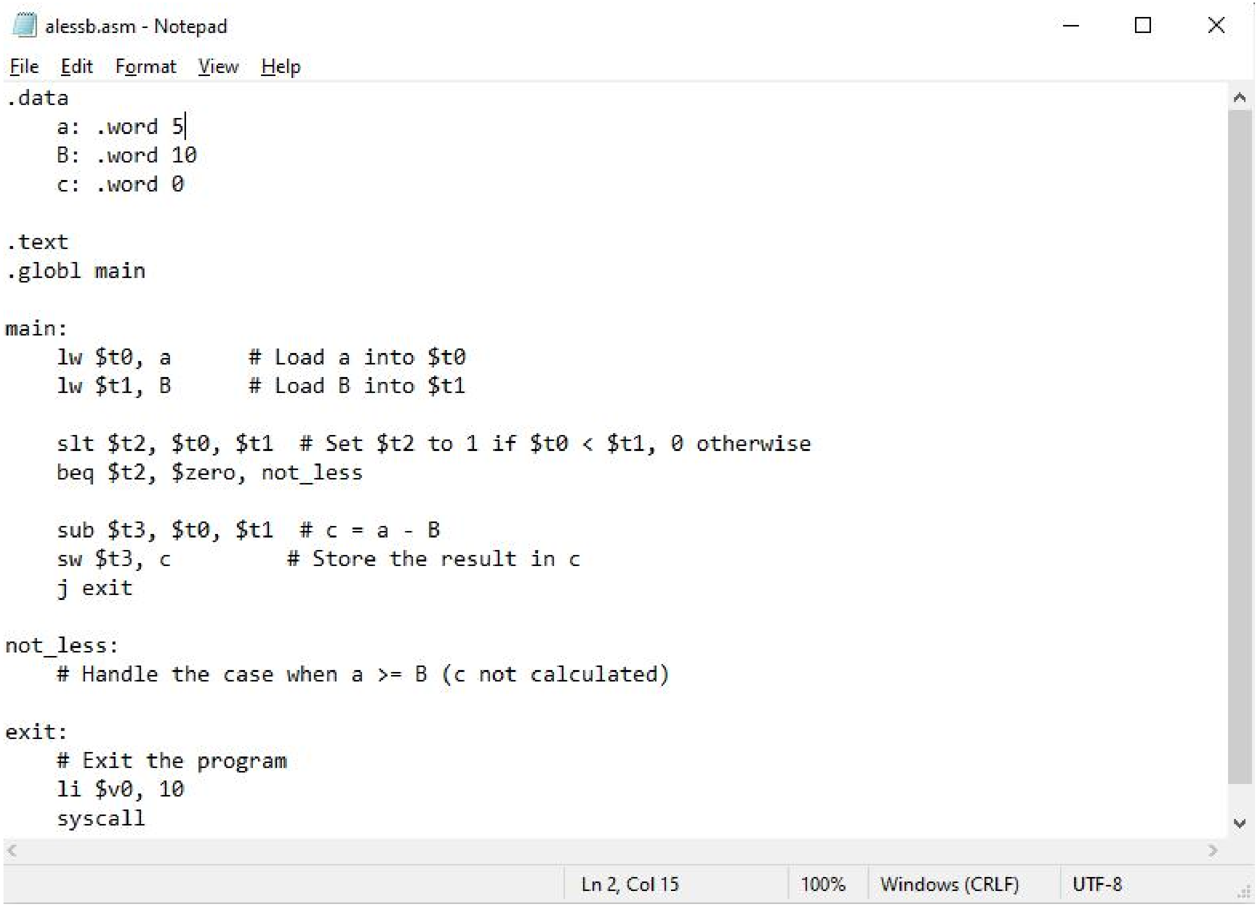
```
                                                User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)              ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4           ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4            ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2              ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2            ; 187: addu $a2 $a2 $v0
[00400014] 0c100009  jal 0x00400024 [main]      ; 188: jal main
[00400018] 00000000  nop                        ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10             ; 191: li $v0 10
[00400020] 0000000c  syscall                    ; 192: syscall # syscall 10 (exit)
[00400024] 3c011001  lui $1, 4097               ; 15: lw $t0, B
[00400028] 8c280004  lw $8, 4($1)
[0040002c] 3c011001  lui $1, 4097               ; 16: lw $t1, c
[00400030] 8c290008  lw $9, 8($1)
[00400034] 3c011001  lui $1, 4097               ; 17: lw $t2, d
[00400038] 8c2a000c  lw $10, 12($1)
[0040003c] 3c011001  lui $1, 4097               ; 18: lw $t3, e
[00400040] 8c2b0010  lw $11, 16($1)
[00400044] 01096020  add $12, $8, $9            ; 21: add $t4, $t0, $t1 # $t4 = B + c
[00400048] 014b6820  add $13, $10, $11          ; 22: add $t5, $t2, $t3 # $t5 = d + e
[0040004c] 018d7022  sub $14, $12, $13          ; 23: sub $t6, $t4, $t5 # $t6 = (B + c) - (d + e)
[00400050] 3c011001  lui $1, 4097               ; 26: sw $t6, a
[00400054] ac2e0000  sw $14, 0($1)
[00400058] 34020004  ori $2, $0, 4              ; 29: li $v0, 4
[0040005c] 3c011001  lui $1, 4097 [result_message]; 30: la $a0, result_message
[00400060] 34240014  ori $4, $1, 20 [result_message]
[00400064] 0000000c  syscall                    ; 31: syscall
[00400068] 34020001  ori $2, $0, 1              ; 33: li $v0, 1
[0040006c] 3c011001  lui $1, 4097               ; 34: lw $a0, a
[00400070] 8c240000  lw $4, 0($1)
[00400074] 0000000c  syscall                    ; 35: syscall
[00400078] 3402000a  ori $2, $0, 10             ; 38: li $v0, 10
[0040007c] 0000000c  syscall                    ; 39: syscall
```

```
Result: -40
```

```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
```

Q14. Write MIPS Assembly Language Programs equivalent to the following C-code fragments.

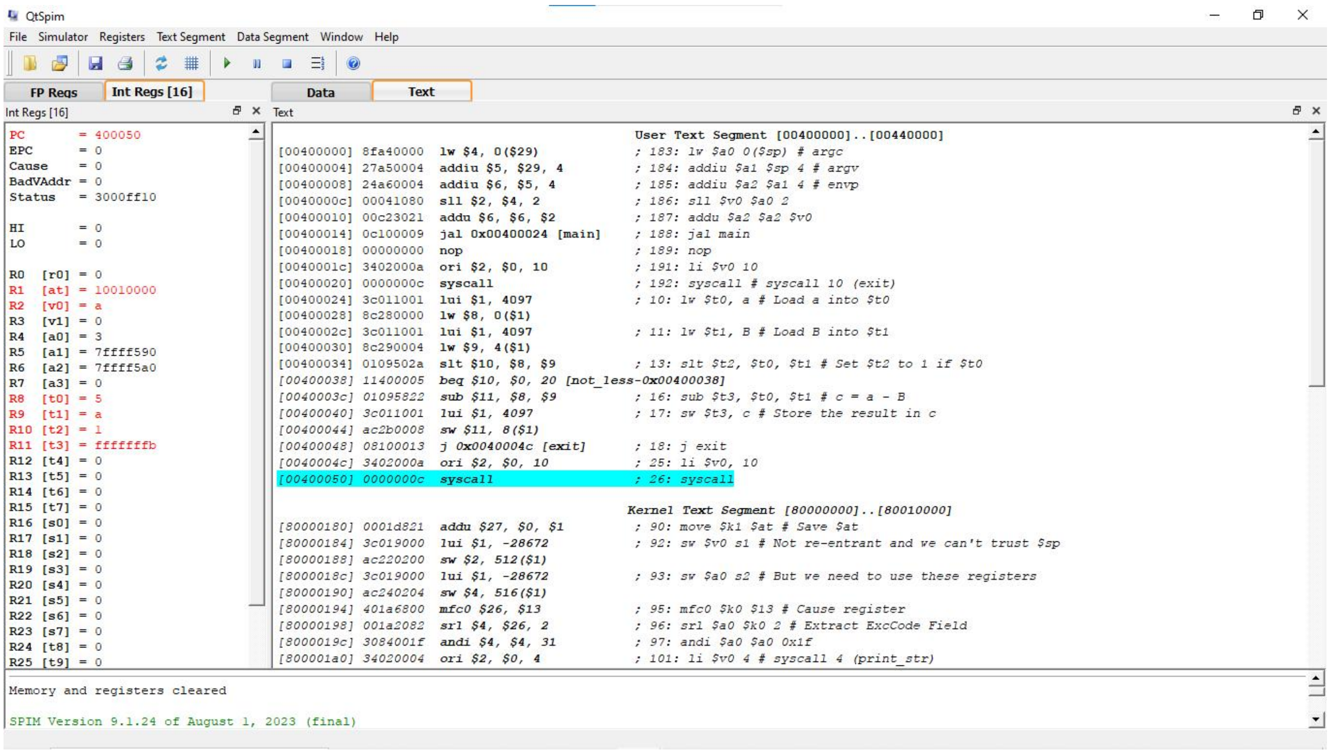
1.
```
if(a<b)
{                                   //a and b are signed integers
    c = a - b;
}
```



alessb.asm - Notepad

```
.data
    a: .word 5
    B: .word 10
    c: .word 0

.text
.globl main

main:
    lw $t0, a      # Load a into $t0
    lw $t1, B      # Load B into $t1

    slt $t2, $t0, $t1  # Set $t2 to 1 if $t0 < $t1, 0 otherwise
    beq $t2, $zero, not_less

    sub $t3, $t0, $t1  # c = a - B
    sw $t3, c         # Store the result in c
    j exit

not_less:
    # Handle the case when a >= B (c not calculated)

exit:
    # Exit the program
    li $v0, 10
    syscall
```

Ln 2, Col 15 | 100% | Windows (CRLF) | UTF-8



QtSpim

Int Regs [16]

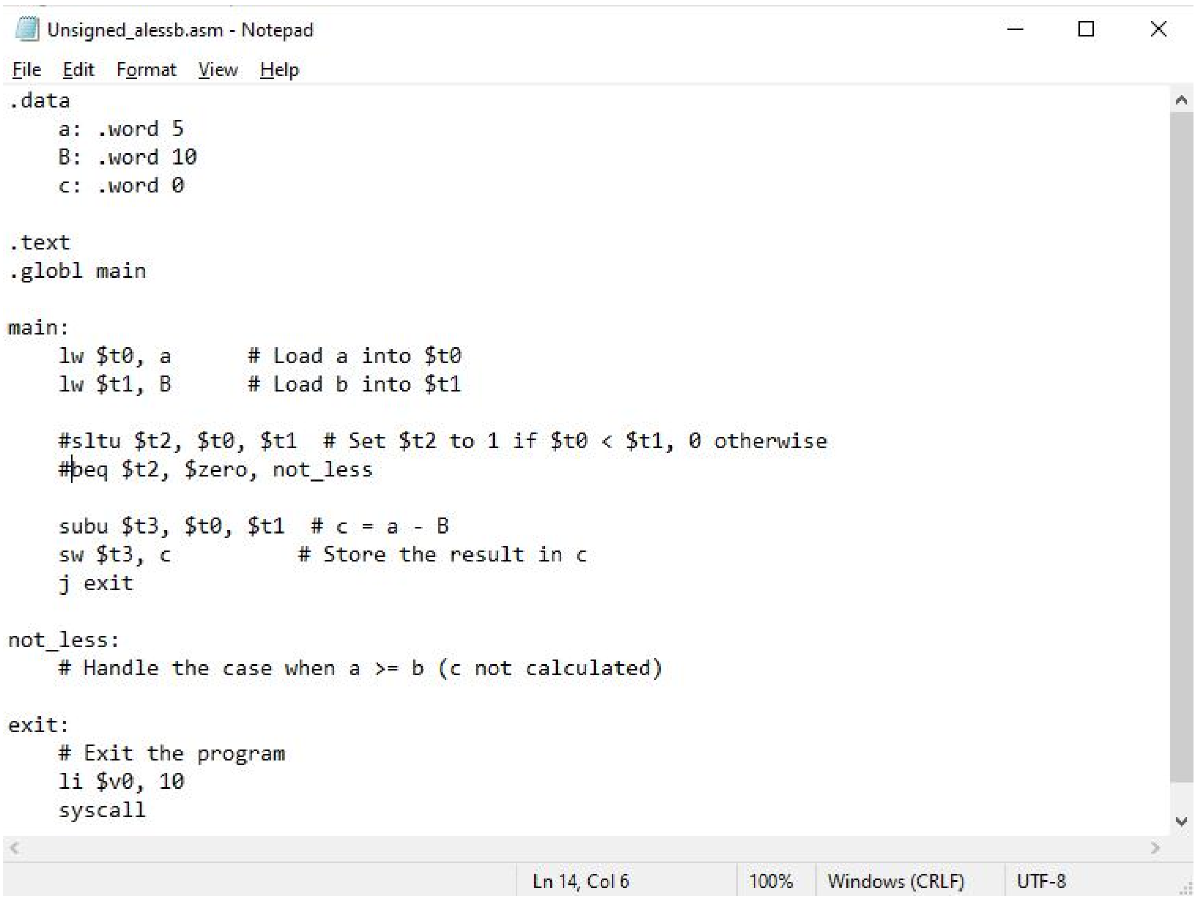
```
PC       = 400050
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 3
R5  [a1] = 7ffff590
R6  [a2] = 7ffff5a0
R7  [a3] = 0
R8  [t0] = 5
R9  [t1] = a
R10 [t2] = 1
R11 [t3] = fffffffb
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
```

Text

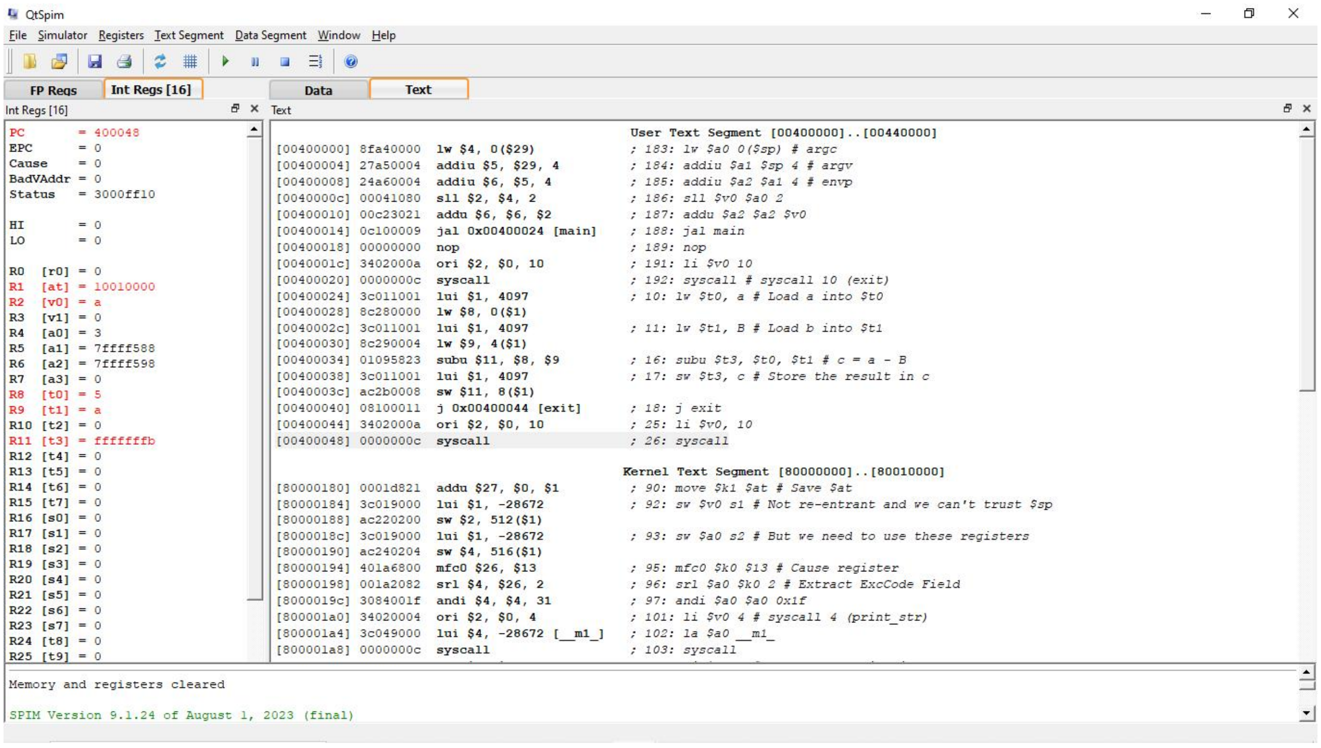
```
                                         User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)           ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4        ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4         ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2           ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2         ; 187: addu $a2 $a2 $v0
[00400014] 0c100009  jal 0x00400024 [main]   ; 188: jal main
[00400018] 00000000  nop                     ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10          ; 191: li $v0 10
[00400020] 0000000c  syscall                 ; 192: syscall # syscall 10 (exit)
[00400024] 3c011001  lui $1, 4097            ; 10: lw $t0, a # Load a into $t0
[00400028] 8c280000  lw $8, 0($1)
[0040002c] 3c011001  lui $1, 4097            ; 11: lw $t1, B # Load B into $t1
[00400030] 8c290004  lw $9, 4($1)
[00400034] 0109502a  slt $10, $8, $9         ; 13: slt $t2, $t0, $t1 # Set $t2 to 1 if $t0
[00400038] 11400005  beq $10, $0, 20 [not_less-0x00400038]
[0040003c] 01095822  sub $11, $8, $9         ; 16: sub $t3, $t0, $t1 # c = a - B
[00400040] 3c011001  lui $1, 4097            ; 17: sw $t3, c # Store the result in c
[00400044] ac2b0008  sw $11, 8($1)
[00400048] 08100013  j 0x0040004c [exit]     ; 18: j exit
[0040004c] 3402000a  ori $2, $0, 10          ; 25: li $v0, 10
[00400050] 0000000c  syscall                 ; 26: syscall

                                        Kernel Text Segment [80000000]..[80010000]
[80000180] 0001d821  addu $27, $0, $1        ; 90: move $k1 $at # Save $at
[80000184] 3c019000  lui $1, -28672          ; 92: sw $v0 s1 # Not re-entrant and we can't trust $sp
[80000188] ac220200  sw $2, 512($1)
[8000018c] 3c019000  lui $1, -28672          ; 93: sw $a0 s2 # But we need to use these registers
[80000190] ac240204  sw $4, 516($1)
[80000194] 401a6800  mfc0 $26, $13           ; 95: mfc0 $k0 $13 # Cause register
[80000198] 001a2082  srl $4, $26, 2          ; 96: srl $a0 $k0 2 # Extract ExcCode Field
[8000019c] 3084001f  andi $4, $4, 31         ; 97: andi $a0 $a0 0x1f
[800001a0] 34020004  ori $2, $0, 4           ; 101: li $v0 4 # syscall 4 (print_str)
```

```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
```

2.
   ```
   if(a<b)
   {                                    // a and b are unsigned integers
       c=a-b;
   }
   ```



```
.data
    a: .word 5
    B: .word 10
    c: .word 0

.text
.globl main

main:
    lw $t0, a      # Load a into $t0
    lw $t1, B      # Load b into $t1

    #sltu $t2, $t0, $t1  # Set $t2 to 1 if $t0 < $t1, 0 otherwise
    #beq $t2, $zero, not_less

    subu $t3, $t0, $t1  # c = a - B
    sw $t3, c           # Store the result in c
    j exit

not_less:
    # Handle the case when a >= b (c not calculated)

exit:
    # Exit the program
    li $v0, 10
    syscall
```


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
Text

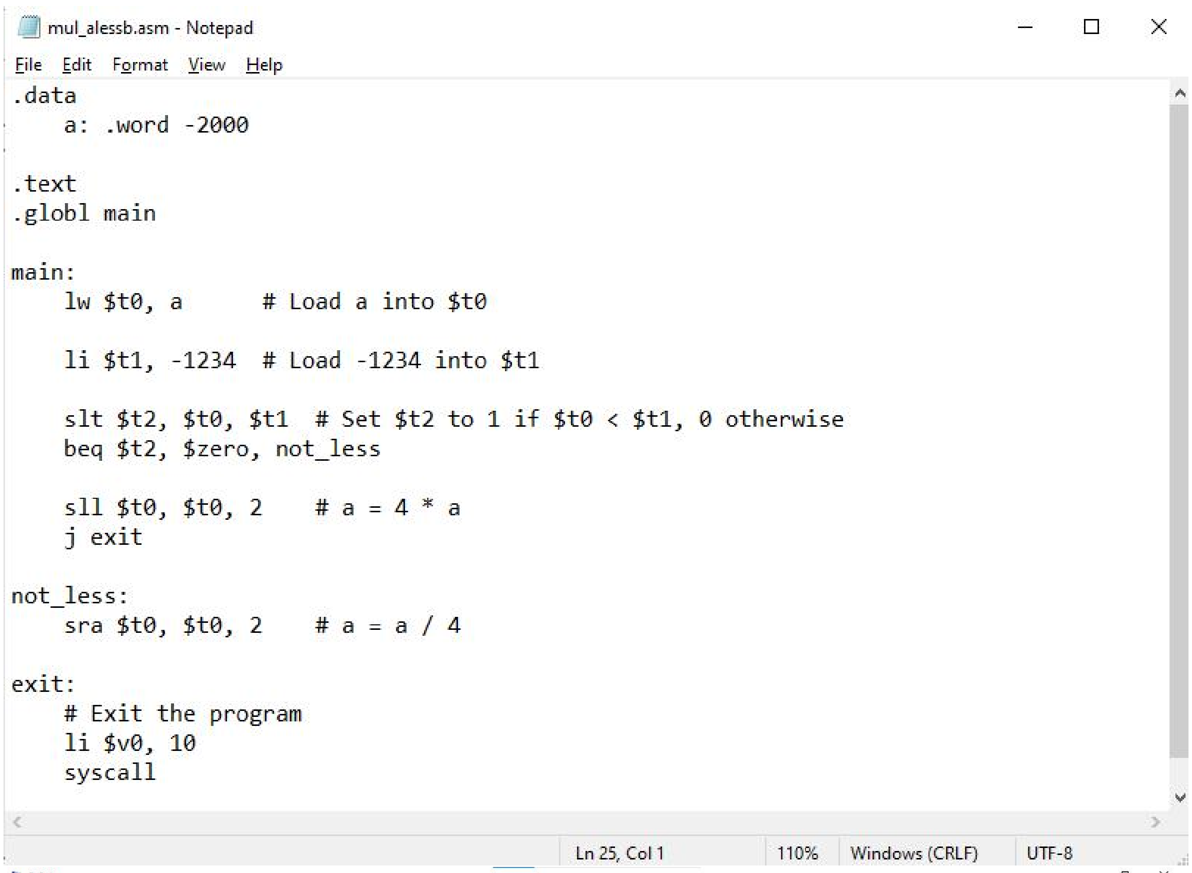
```
PC       = 400048
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 3
R5  [a1] = 7ffff588
R6  [a2] = 7ffff598
R7  [a3] = 0
R8  [t0] = 5
R9  [t1] = a
R10 [t2] = 0
R11 [t3] = fffffffb
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
```

```
                                          User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)          ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4       ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4        ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2          ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2        ; 187: addu $a2 $a2 $v0
[00400014] 0c100009  jal 0x00400024 [main]  ; 188: jal main
[00400018] 00000000  nop                    ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10         ; 191: li $v0 10
[00400020] 0000000c  syscall                ; 192: syscall # syscall 10 (exit)
[00400024] 3c011001  lui $1, 4097           ; 10: lw $t0, a # Load a into $t0
[00400028] 8c280000  lw $8, 0($1)
[0040002c] 3c011001  lui $1, 4097           ; 11: lw $t1, B # Load b into $t1
[00400030] 8c290004  lw $9, 4($1)
[00400034] 01095823  subu $11, $8, $9       ; 16: subu $t3, $t0, $t1 # c = a - B
[00400038] 3c011001  lui $1, 4097           ; 17: sw $t3, c # Store the result in c
[0040003c] ac2b0008  sw $11, 8($1)
[00400040] 08100011  j 0x00400044 [exit]    ; 18: j exit
[00400044] 3402000a  ori $2, $0, 10         ; 25: li $v0, 10
[00400048] 0000000c  syscall                ; 26: syscall

                                         Kernel Text Segment [80000000]..[80010000]
[80000180] 0001d821  addu $27, $0, $1       ; 90: move $k1 $at # Save $at
[80000184] 3c019000  lui $1, -28672         ; 92: sw $v0 s1 # Not re-entrant and we can't trust $sp
[80000188] ac220200  sw $2, 512($1)
[8000018c] 3c019000  lui $1, -28672         ; 93: sw $a0 s2 # But we need to use these registers
[80000190] ac240204  sw $4, 516($1)
[80000194] 401a6800  mfc0 $26, $13          ; 95: mfc0 $k0 $13 # Cause register
[80000198] 001a2082  srl $4, $26, 2         ; 96: srl $a0 $k0 2 # Extract ExcCode Field
[8000019c] 3084001f  andi $4, $4, 31        ; 97: andi $a0 $a0 0x1f
[800001a0] 34020004  ori $2, $0, 4          ; 101: li $v0 4 # syscall 4 (print_str)
[800001a4] 3c049000  lui $4, -28672 [__m1_] ; 102: la $a0 __m1_
[800001a8] 0000000c  syscall                ; 103: syscall
```

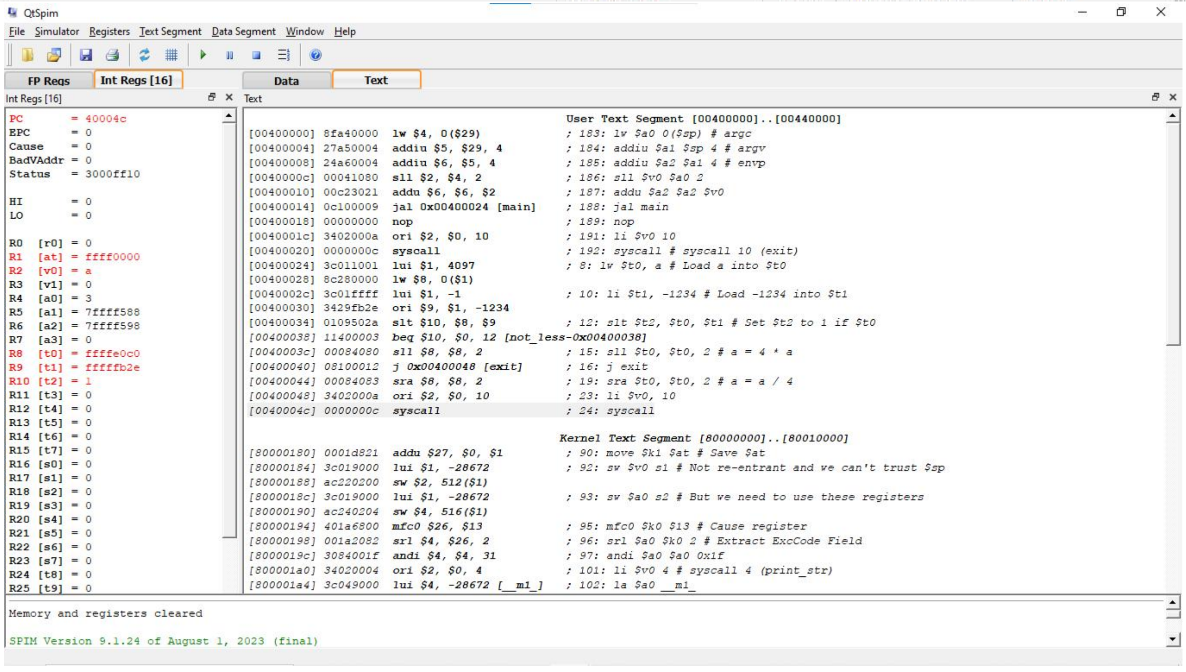
```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
```

3. if (a < -1234)

```
if ( a < -1234)
 {                              // a is a signed integer
    a = 4 * a ;
}
else
{
     a = a/4;
}
```



```
.data
    a: .word -2000

.text
.globl main

main:
    lw $t0, a       # Load a into $t0

    li $t1, -1234  # Load -1234 into $t1

    slt $t2, $t0, $t1  # Set $t2 to 1 if $t0 < $t1, 0 otherwise
    beq $t2, $zero, not_less

    sll $t0, $t0, 2    # a = 4 * a
    j exit

not_less:
    sra $t0, $t0, 2    # a = a / 4

exit:
    # Exit the program
    li $v0, 10
    syscall
```


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
Text

```
PC       = 40004c
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = ffff0000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 3
R5  [a1] = 7ffff588
R6  [a2] = 7ffff598
R7  [a3] = 0
R8  [t0] = fffffe0c0
R9  [t1] = fffffb2e
R10 [t2] = 1
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
```

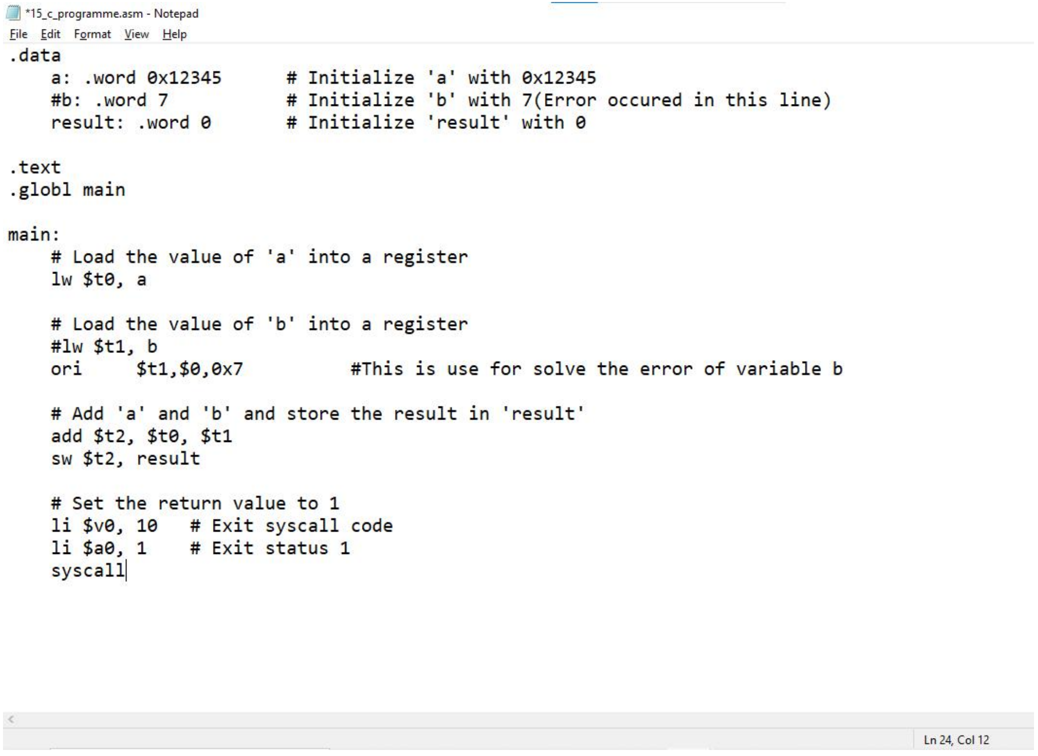
```
                                         User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)            ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4         ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4          ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2            ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2          ; 187: addu $a2 $a2 $v0
[00400014] 0c100009  jal 0x00400024 [main]    ; 188: jal main
[00400018] 00000000  nop                      ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10           ; 191: li $v0 10
[00400020] 0000000c  syscall                  ; 192: syscall # syscall 10 (exit)
[00400024] 3c011001  lui $1, 4097             ; 8: lw $t0, a # Load a into $t0
[00400028] 8c280000  lw $8, 0($1)
[0040002c] 3c01ffff  lui $1, -1               ; 10: li $t1, -1234 # Load -1234 into $t1
[00400030] 3429fb2e  ori $9, $1, -1234
[00400034] 0109502a  slt $10, $8, $9          ; 12: slt $t2, $t0, $t1 # Set $t2 to 1 if $t0
[00400038] 11400003  beq $10, $0, 12 [not_less-0x00400038]
[0040003c] 00084080  sll $8, $8, 2            ; 15: sll $t0, $t0, 2 # a = 4 * a
[00400040] 08100012  j 0x00400048 [exit]      ; 16: j exit
[00400044] 00084083  sra $8, $8, 2            ; 19: sra $t0, $t0, 2 # a = a / 4
[00400048] 3402000a  ori $2, $0, 10           ; 23: li $v0, 10
[0040004c] 0000000c  syscall                  ; 24: syscall

                                        Kernel Text Segment [80000000]..[80010000]
[80000180] 0001d821  addu $27, $0, $1         ; 90: move $k1 $at # Save $at
[80000184] 3c019000  lui $1, -28672           ; 92: sw $v0 s1 # Not re-entrant and we can't trust $sp
[80000188] ac220200  sw $2, 512($1)
[8000018c] 3c019000  lui $1, -28672           ; 93: sw $a0 s2 # But we need to use these registers
[80000190] ac240204  sw $4, 516($1)
[80000194] 401a6800  mfc0 $26, $13            ; 95: mfc0 $k0 $13 # Cause register
[80000198] 001a2082  srl $4, $26, 2           ; 96: srl $a0 $k0 2 # Extract ExcCode Field
[8000019c] 3084001f  andi $4, $4, 31          ; 97: andi $a0 $a0 0x1f
[800001a0] 34020004  ori $2, $0, 4            ; 101: li $v0 4 # syscall 4 (print_str)
[800001a4] 3c049000  lui $4, -28672 [__m1_]   ; 102: la $a0 __m1_
```

```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
```

15. Write a MIPS assembly for the following C codes :

```c
int a, b, result

int main(){

 a = 0x12345;

 b = 7;

 result = a + b;

 return 1;

}
```

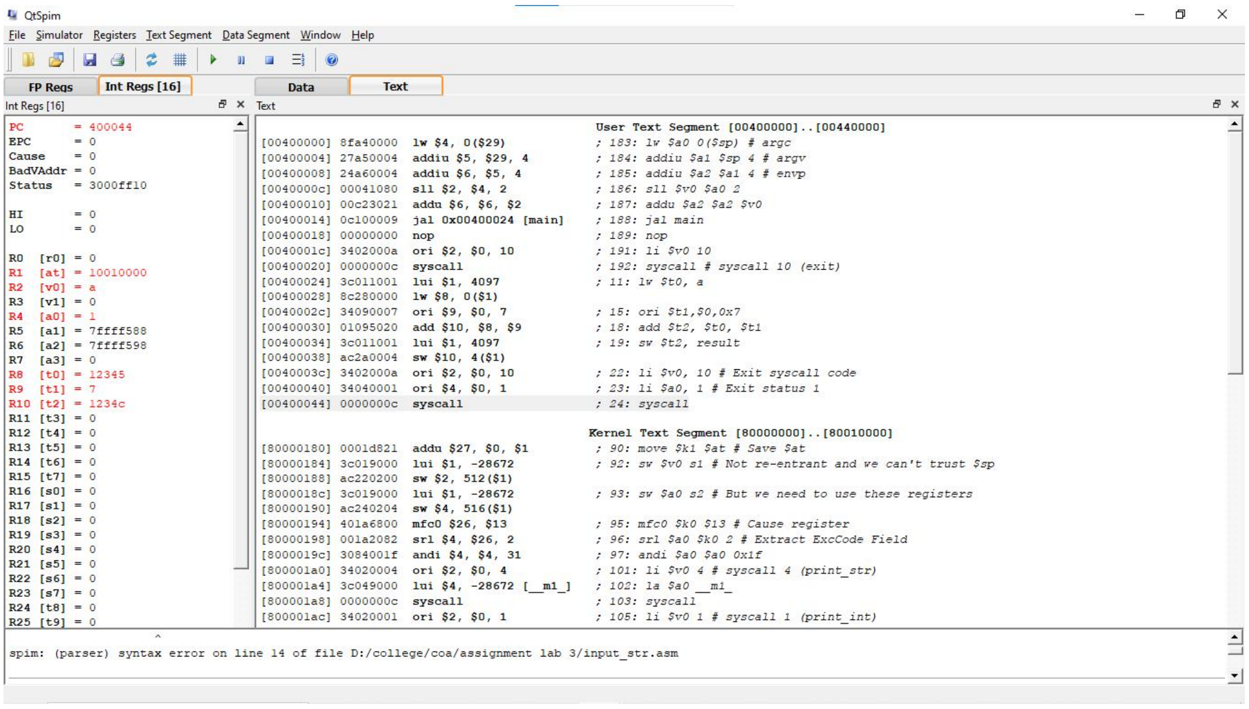
ANS :



*15_c_programme.asm - Notepad

```
.data
    a: .word 0x12345       # Initialize 'a' with 0x12345
    #b: .word 7            # Initialize 'b' with 7(Error occured in this line)
    result: .word 0        # Initialize 'result' with 0

.text
.globl main

main:
    # Load the value of 'a' into a register
    lw $t0, a

    # Load the value of 'b' into a register
    #lw $t1, b
    ori     $t1,$0,0x7          #This is use for solve the error of variable b

    # Add 'a' and 'b' and store the result in 'result'
    add $t2, $t0, $t1
    sw $t2, result

    # Set the return value to 1
    li $v0, 10    # Exit syscall code
    li $a0, 1     # Exit status 1
    syscall
```

Ln 24, Col 12



```
PC       = 400044
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 1
R5  [a1] = 7ffff588
R6  [a2] = 7ffff598
R7  [a3] = 0
R8  [t0] = 12345
R9  [t1] = 7
R10 [t2] = 1234c
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
```

```
                                              User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)            ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4         ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4          ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2            ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2          ; 187: addu $a2 $a2 $v0
[00400014] 0c100009  jal 0x00400024 [main]    ; 188: jal main
[00400018] 00000000  nop                      ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10           ; 191: li $v0 10
[00400020] 0000000c  syscall                  ; 192: syscall # syscall 10 (exit)
[00400024] 3c011001  lui $1, 4097             ; 11: lw $t0, a
[00400028] 8c280000  lw $8, 0($1)
[0040002c] 34090007  ori $9, $0, 7            ; 15: ori $t1,$0,0x7
[00400030] 01095020  add $10, $8, $9          ; 18: add $t2, $t0, $t1
[00400034] 3c011001  lui $1, 4097             ; 19: sw $t2, result
[00400038] ac2a0004  sw $10, 4($1)
[0040003c] 3402000a  ori $2, $0, 10           ; 22: li $v0, 10 # Exit syscall code
[00400040] 34040001  ori $4, $0, 1            ; 23: li $a0, 1 # Exit status 1
[00400044] 0000000c  syscall                  ; 24: syscall

                                             Kernel Text Segment [80000000]..[80010000]
[80000180] 0001d821  addu $27, $0, $1         ; 90: move $k1 $at # Save $at
[80000184] 3c019000  lui $1, -28672           ; 92: sw $v0 s1 # Not re-entrant and we can't trust $sp
[80000188] ac220200  sw $2, 512($1)
[8000018c] 3c019000  lui $1, -28672           ; 93: sw $a0 s2 # But we need to use these registers
[80000190] ac240204  sw $4, 516($1)
[80000194] 401a6800  mfc0 $26, $13            ; 95: mfc0 $k0 $13 # Cause register
[80000198] 001a2082  srl $4, $26, 2           ; 96: srl $a0 $k0 2 # Extract ExcCode Field
[8000019c] 3084001f  andi $4, $4, 31          ; 97: andi $a0 $a0 0x1f
[800001a0] 34020004  ori $2, $0, 4            ; 101: li $v0 4 # syscall 4 (print_str)
[800001a4] 3c049000  lui $4, -28672 [__m1_]   ; 102: la $a0 __m1_
[800001a8] 0000000c  syscall                  ; 103: syscall
[800001ac] 34020001  ori $2, $0, 1            ; 105: li $v0 1 # syscall 1 (print_int)
```

```
spim: (parser) syntax error on line 14 of file D:/college/coa/assignment lab 3/input_str.asm
```

QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
Text