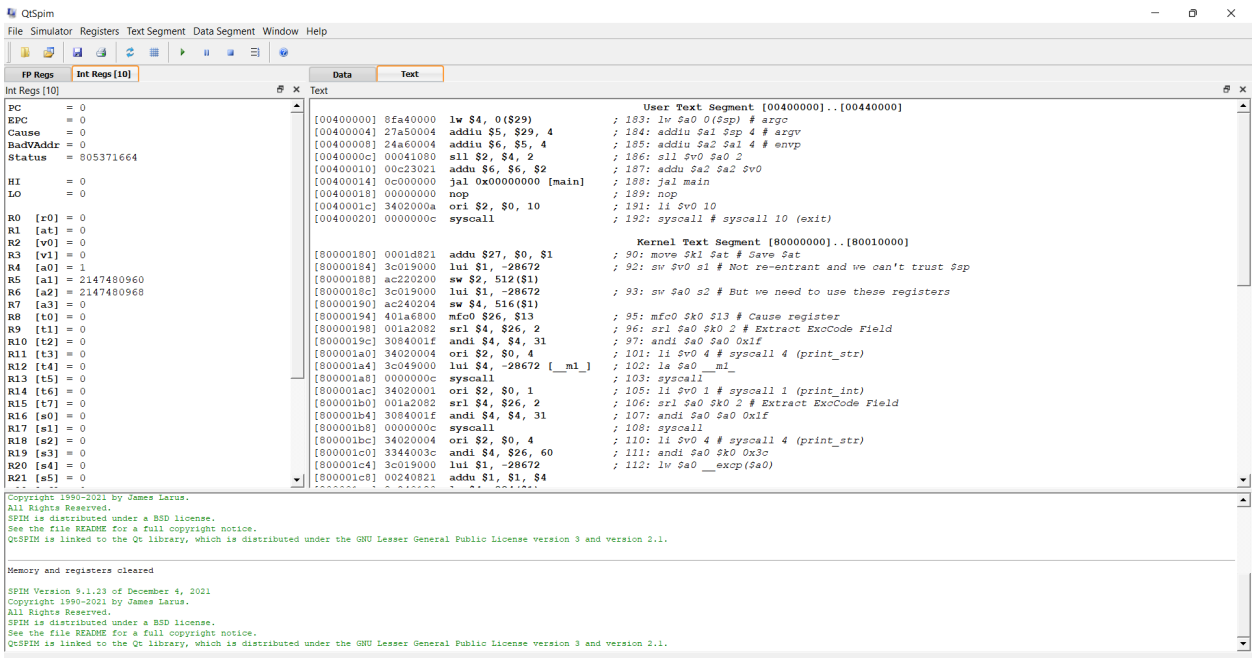
Lab4
CS211
Stack and Subroutines

Name-Pranav Tambe
Roll No-2106339



Interface before loading any file

1) Evaluate the expression 'ab-10a+20b+16'. Consider that only $t0 and $t1 are available to store temporary values. Store a=10 and b=20 in data section. Use stack for other memory requirements. Display the sum.

```
.data
a_var : .word 10
b_var : .word 20
result: .asciiz "The result of ab-10a+20b+16 is: "

.text
```

```
main:
  # calculate 10*a==a*a
  lw $t0,a_var # store a_var in $t0
  mult $t0,$t0
  mflo $t0
  # use stack for storing 10a
  subu $sp,$sp,4 # subtract 4 from  stack pointer
  sw $t0,($sp) # store 10*a in stack

 # calculate ab
  lw $t0,a_var # store a_var in $t0
  lw $t1,b_var  # store b_var in $t1
  mult $t0,$t1 # multiply $t0 and $t1
  mflo $t0 # Move from low register to $t0

   # use stack for storing ab
  subu $sp,$sp,4 # subtract 4 from  stack pointer
  sw $t0,($sp) # store ab in stack


  # calculate 20*b
  # $t1 contains b_var = 20
  mult $t1,$t1 # 20*20
  mflo  $t0 # Move from low register to $t0

  # use stack for storing 10a
  subu $sp,$sp,4 # subtract 4 from  stack pointer
  sw $t0,($sp) # store 20b in stack

  addi $t0,$0,0x10 # store 16 in $t0
  lw $t1,($sp) # load 20b from stack in $t1
  addu $sp,$sp,4 # pop out 20b by  incrementing  stack pointer
  # add 20b +16 and store it in $t0
  addu $t0,$t0,$t1

  lw $t1,($sp) # load ab from stack in $t1
  addu $sp,$sp,4 # pop out ab by  incrementing  stack pointer
   addu $t0,$t0,$t1  # add (20b +16)+ab and store it in $t0
   lw $t1,($sp) # load 10a in $t1
   addu $sp,$sp,4  # pop out 10a by  incrementing  stack pointer
```

```
   sub $t0,$t0,$t1 # subtract 10a from the value and we get (20b
+16)+ab-10a
    # Print result string
    li $v0, 4
    la $a0, result
    syscall
   # Load immediate value 1 into register $v0
   li $v0,1
   # Copy the contents of register $t0 to register $a0
   move $a0,$t0
   # Print the value stored in $a0 (which was copied from $t0)
   # using system call with service number 1 (print integer)
   syscall
  # Load immediate value 10 into register $v0
   li $v0,10
  # Exit the program using system call with service number 10 (exit)
   syscall
```

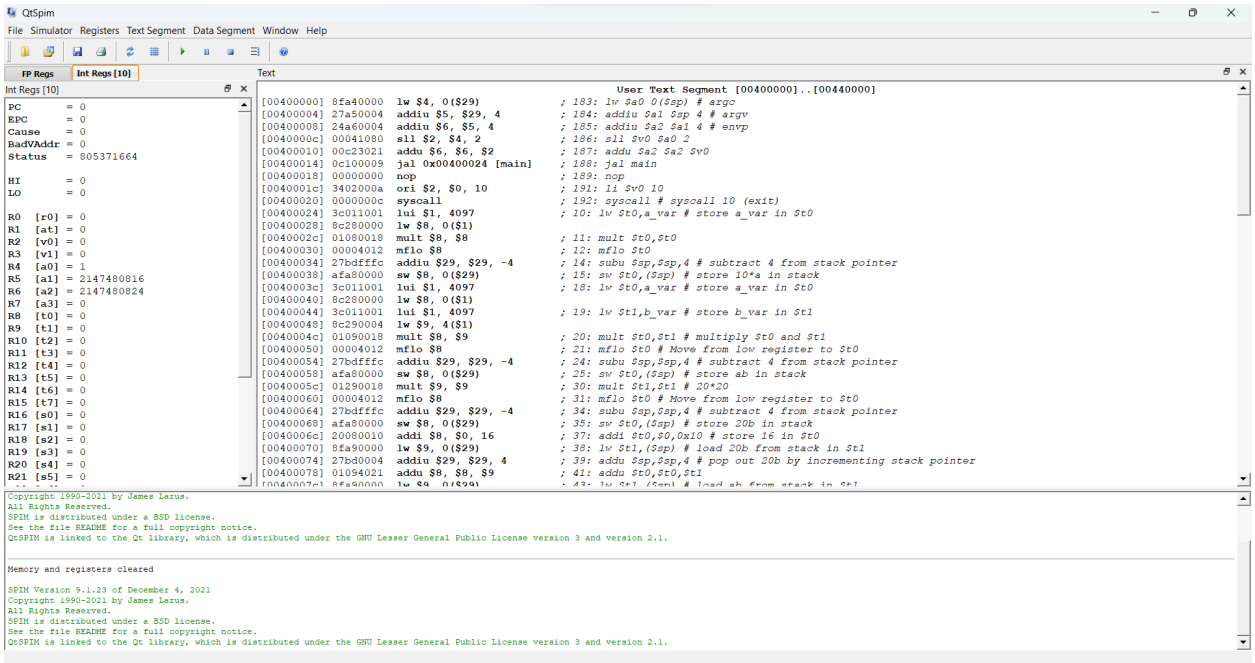
Brief overview of the code section

Code calculates the value of the expression ab-10a+20b+16 and prints the result.
First, it initializes two variables "a_var" and "b_var" with values 10 and 20 respectively, and a string variable "result" to hold the result message.
The "main" section of the code performs the following operations:

- Calculates the value of 10a and stores it on the stack.
- Calculates the value of ab and stores it on the stack.
- Calculates the value of 20b and stores it on the stack.
- Adds 20b+16 and stores it in $t0.
- Adds (20b+16)+ab and stores it in $t0.
- Subtracts 10a from the value and stores the final result in $t0.
- Prints the result message and the calculated value using system calls.

Overall, the code uses multiplication and addition/subtraction instructions along with stack manipulation to perform the required operations and output the final result.

After loading the file



Stack pointer before pushing any value

```
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 268468224
R29 [sp] = 2147480812
R30 [s8] = 0
R31 [ra] = 0
```

Stack pointer decremented by 4 and value is pushed into the stack

```
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 268468224
R29 [sp] = 2147480808
R30 [s8] = 0
R31 [ra] = 0
```

Stack pointer before popping any value

```
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 268468224
R29 [sp] = 2147480800
R30 [s8] = 0
R31 [ra] = 0
```

Stack pointer decremented by 4 and value is popped from the stack

```
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 268468224
R29 [sp] = 2147480804
R30 [s8] = 0
R31 [ra] = 0
```

Stack pointer after popping every value out

```
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 268468224
R29 [sp] = 2147480812
R30 [s8] = 0
R31 [ra] = 0
```

Value after performing step by step operations

Console

```
The result of ab-10a+20b+16 is: 516
```

a=10,b=20
a*b=200 , 10*a=100 , 20*b=400
200-100+400+16=516

Registers after execution of the code

```
R0   [r0] = 0
R1   [at] = 268500992
R2   [v0] = 10
R3   [v1] = 0
R4   [a0] = 516
R5   [a1] = 2147480816
R6   [a2] = 2147480824
R7   [a3] = 0
R8   [t0] = 516
R9   [t1] = 100
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
R28  [gp] = 268468224
R29  [sp] = 2147480812
```

2) Find the maximum of the three expressions: x*x ; x*y ; y*5. Take x and y as input from user.Write a global subroutine, in another file, to calculate values of these expressions. Write a subroutine to find a maximum of integers and use it to find the maximum of these three expressions. Display the result.

First file :-Pranav_2106339_4_2_Caller

```
.data
 string_one: .asciiz "Enter the number x:"
 string_two: .asciiz "Enter the number y:"
```

```
  result :.asciiz "  is the maximum of x*x , x*y and y*5."
.text
.globl next
# Load immediate value 10 into register $t0
addi $t0, $zero, 10

main:
# Load immediate value 4 into register $v0 (used for printing strings)
li $v0, 4
# Load address of string_one into register $a0 (used as an argument for
syscall to print string)
la $a0,string_one
# Print the string stored at address in $a0
syscall

# Load immediate value 5 into register $v0 (used for taking integer input
x)
li $v0, 5
# Read integer from input and store in register $v0
syscall
# Copy integer from $v0 to $t0
move $t0, $v0

# Load immediate value 4 into register $v0 (used for printing strings)
li $v0, 4
# Load address of string_two into register $a0 (used as an argument for
syscall to print string)
la $a0,string_two
# Print the string stored at address in $a0
syscall

# Load immediate value 5 into register $v0 (used for taking integer input
y)
li $v0, 5
# Read integer from input and store in register $v0
syscall
# Copy integer from $v0 to $t1
move $t1, $v0
# Move contents of $t0 to $a0 (used as an argument for jal to call max
function)
```

```
move $a0,$t0
# Move contents of $t1 to $a0 (used as an argument for jal to call max
function)
move $a1,$t1
# Jump to max function using jal instruction
jal max

next:
# max value is in $a0 itself
# just print the result
li $v0,1
syscall
# Print result string
# Load immediate value 4 into register $v0 (used for printing strings)
li $v0, 4
# Load address of string result into register $a0 (used as an argument for
syscall to print string)
la $a0, result
# Print the string stored at address in $a0
syscall
# Load immediate value 10 into register $v0 (used for terminating the
program)
li $v0, 10
# Terminate the program
syscall
```

Brief overview of the code section

code prompts the user to enter two integers, x and y, and then computes and prints the maximum value of xx, xy, and y*5.

The program starts by defining three strings in the .data section: string_one, string_two, and result. string_one and string_two are used as prompts for the user to enter the values of x and y, respectively, and result is used to print the final result.

In the .text section, the program defines a global function called "max" that takes two integer arguments, compares them, and returns the larger value in $a0.

In the main function, the program starts by setting $t0 to 10 using the addi instruction. Then, it prompts the user to enter the value of x using syscall 5 (read integer input) and stores it in $t0. Next, it prompts the user to enter the value of y using syscall 5 (read integer input) and stores it in $t1. It then calls the max function with x and y as arguments using the jal instruction.

The max function returns the maximum value in $a0, which is then printed to the console using syscall 1 (print integer) in the next function. Finally, the program prints the result string and terminates using syscalls 4 and 10, respectively.

Second file :-Pranav_2106339_4_2_Callee

```
.text
.globl max
max:
# Multiply contents of registers $t0 and $t0 and store result in $s0
mul $s0,$a0,$a0 # x*x
# Multiply contents of registers $t0 and $t1 and store result in $s1
mul $s1,$a0,$a1 # x*y

# Load immediate value 5 into register $t7
addi $t7,$0,5
# Multiply contents of registers $t1 and $t7 and store result in $lo
mult $a1,$t7 # 5*y

# Move contents of $s0 to $a0
move $a0,$s0 # x*x
# Move contents of $s1 to $a1
move $a1,$s1 # x*y
# Move contents of $lo to $a2
mflo $a2 # 5*y
# compare $a0 and $a2, store the result in $t2 (1 if $a2 < $a0, 0
otherwise)
slt $t2,$a2,$a0
# if $t2 is 0 (i.e., $a2 >= $a0), branch to check_a1_a2
beq $t2,0,check_a1_a2
# compare $a0 and $a1, store the result in $t2 (1 if $a1 < $a0, 0
otherwise)
slt $t2,$a1,$a0
 # if $t2 is 0 (i.e., $a1 >= $a0), branch to a1_is_max
beq $t2,0,a1_is_max
# otherwise, jump to next
jal next

check_a1_a2:
  # compare $a1 and $a2, store the result in $t2 (1 if $a2 < $a1, 0
otherwise)
  slt $t2,$a1,$a2
  # if $t2 is 0 (i.e., $a2 >= $a1), branch to a1_is_max
```

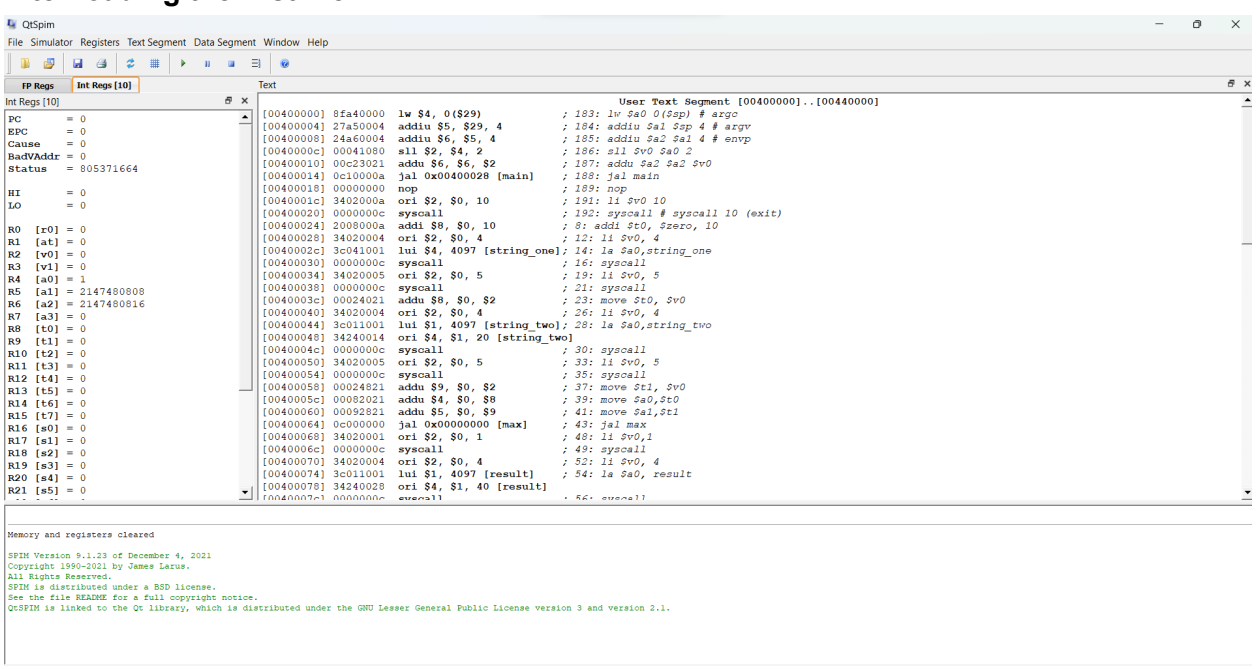
```
  beq $t2,0,a1_is_max
  # otherwise, move the value in $a2 to $a0 and jump to next
  move $a0,$a2
  jal next

a1_is_max:
  # move the value in $a1 to $a0 and jump to next
  move $a0,$a1
  jal next
```

Brief overview of the code section

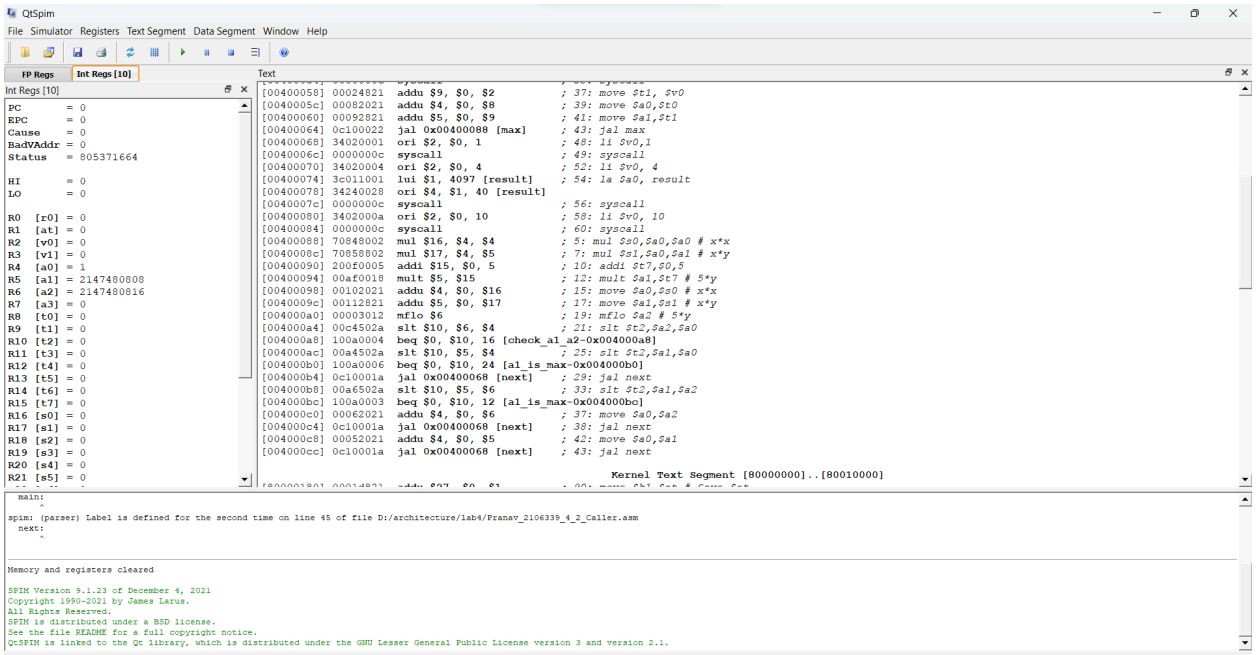
Code defines a function called "max" that computes the maximum value among xx, xy, and y*5. The code first multiplies x by itself and stores the result in $s0, and then multiplies x by y and stores the result in $s1. It also multiplies y by 5 using the "mult" instruction and stores the result in $lo.

The code then compares the results of xx and y*5, and if y*5 is greater, it moves the value in $a2 (which holds the result of y*5) to $a0 (which will be used to print the result). If x*x is greater or equal to y*5, the code compares x*x and x*y, and if x*y is greater, it moves the value in $a1 (which holds the result of x*y) to $a0. Otherwise, it moves the value in $s0 (which holds the result of x*x) to $a0. Finally, the code jumps to a label called "next," which prints the result and terminates the program.

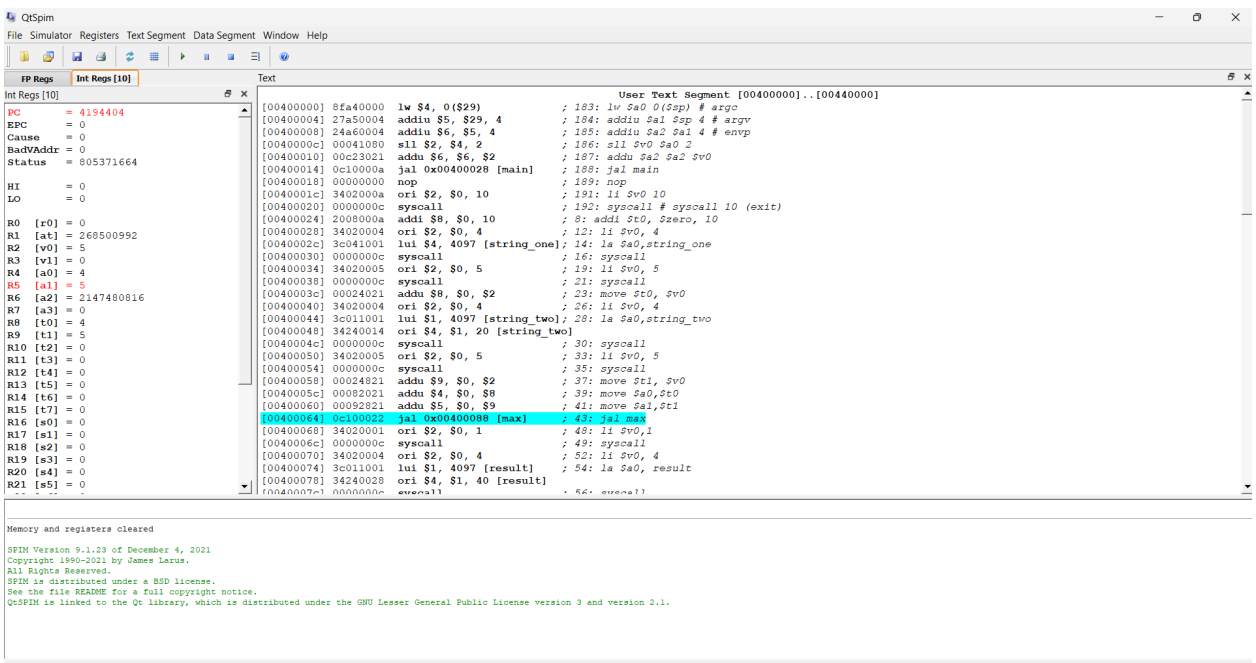
After loading the first file



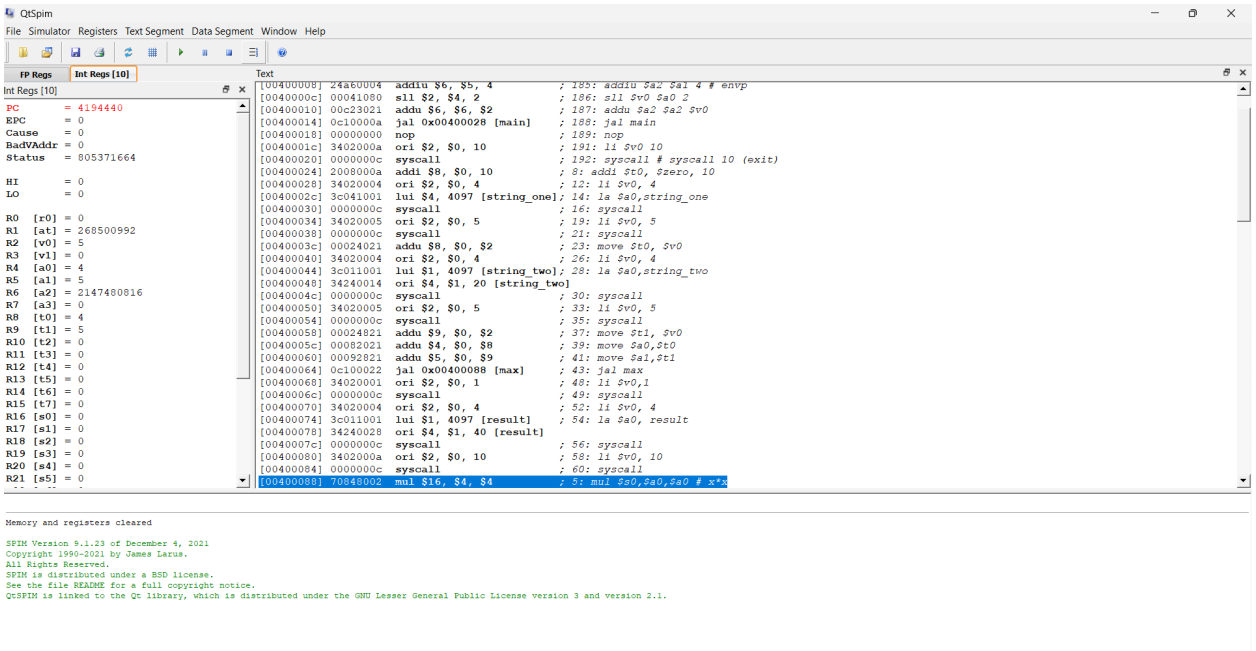
After loading the second file



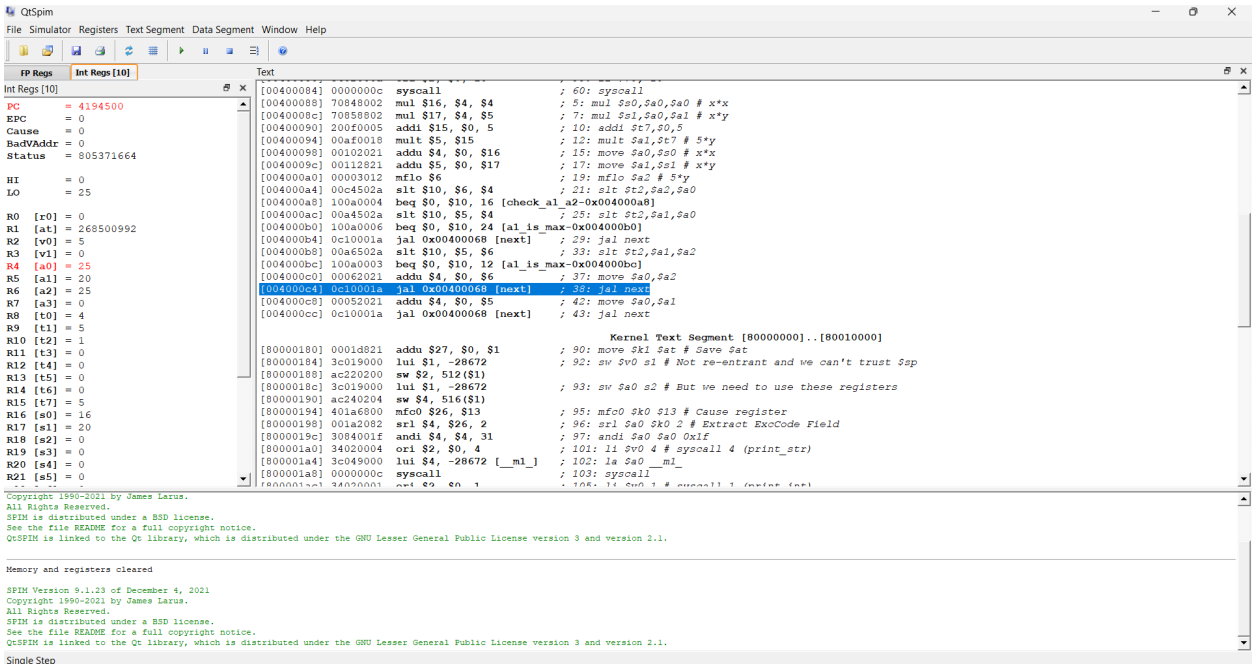
Jump and link to max from caller to callee file
$a0 , $a1 contains the value of x , y and passed to the max function as parameters



Execution flow moved to max function in another file



Execution flow moved to next block in the caller file



Console images

```
Console
Enter the number x:4
Enter the number y:
```

```
Console
Enter the number x:4
Enter the number y:5
25  is the maximum of x*x , x*y and y*5.
```

max(4*4,4*5,5*5)=max(16,20,25)=25

```
Console
Enter the number x:7
Enter the number y:10
70  is the maximum of x*x , x*y and y*5.
```

max(7*7,7*10,10*5)=max(49,70,50)=70

```
Console
Enter the number x:15
Enter the number y:6
225  is the maximum of x*x , x*y and y*5.
```

max(15*15,15*6,6*5)=max(225,90,30)=225

Registers after execution of the code

Int Regs [10]

```
PC        = 4194436
EPC       = 0
Cause     = 0
BadVAddr  = 0
Status    = 805371664

HI        = 0
LO        = 25

R0   [r0] = 0
R1   [at] = 268500992
R2   [v0] = 10
R3   [v1] = 0
R4   [a0] = 268501032
R5   [a1] = 20
R6   [a2] = 25
R7   [a3] = 0
R8   [t0] = 4
R9   [t1] = 5
R10  [t2] = 1
R11  [t3] = 0
R12  [t4] = 0
R13  [t5] = 0
R14  [t6] = 0
R15  [t7] = 5
R16  [s0] = 16
R17  [s1] = 20
R18  [s2] = 0
R19  [s3] = 0
R20  [s4] = 0
R21  [s5] = 0
```