CS 211

LAB 4 : Stack and Subroutines

Name – Gautam Kumar Mahar

Roll No. – 2103114

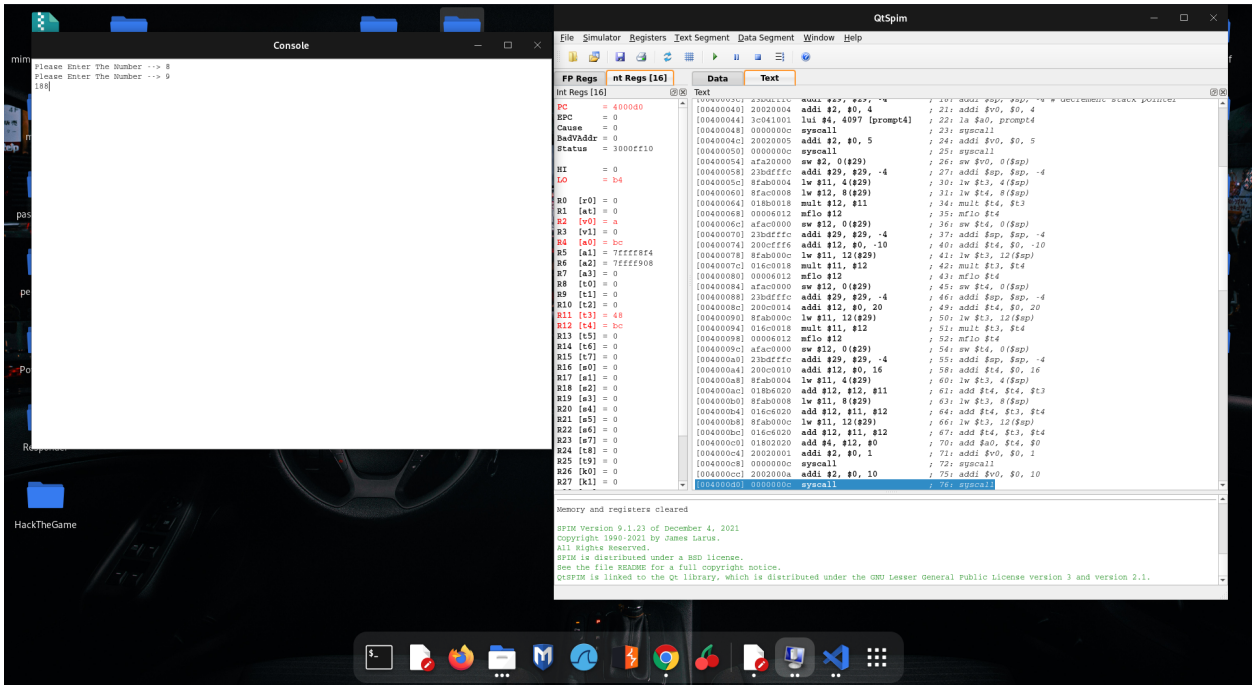
Branch – Computer Science Engineering

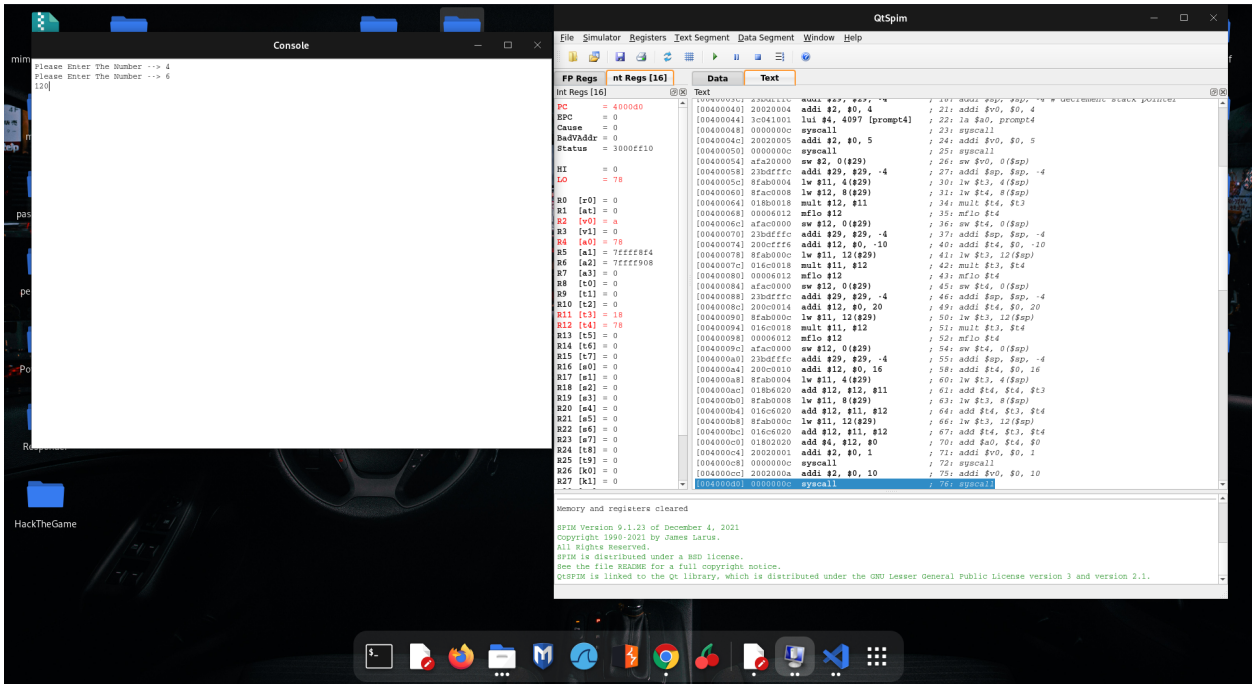
[PART B]

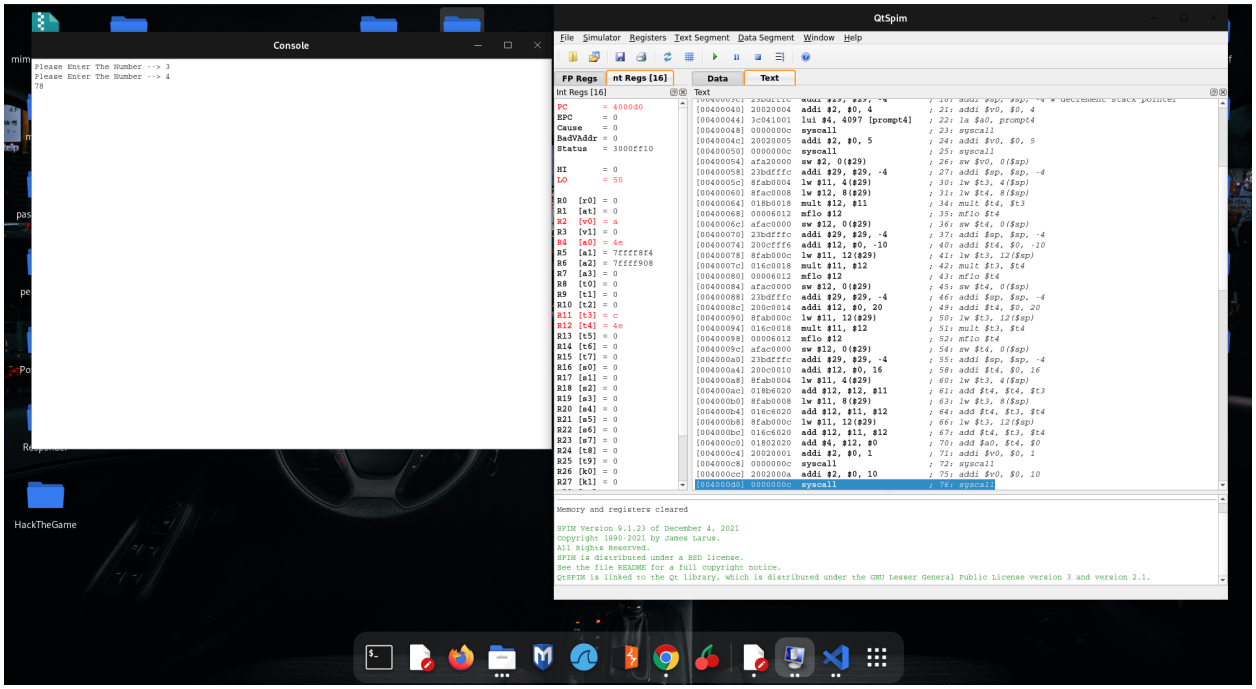
Write the following MIPS assembly language programs and run them using the single step method. This will help you understand the stack push and pop operations and also the flow of program in case of the subroutines.

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

Output -

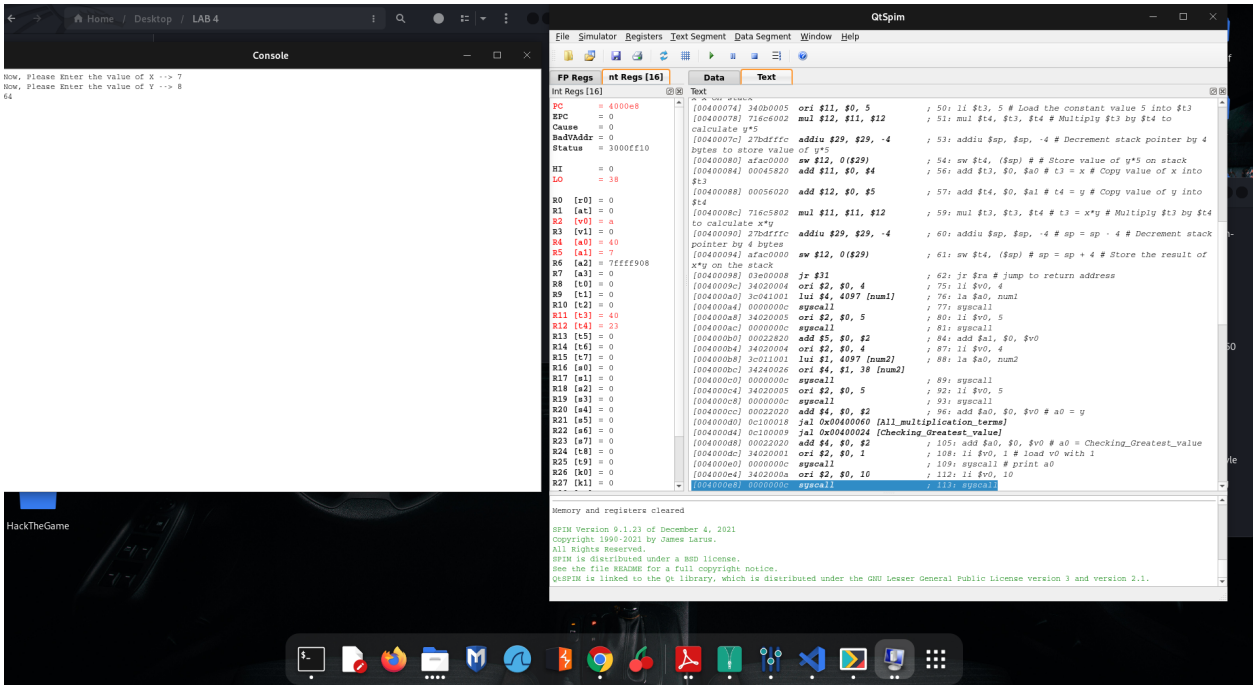


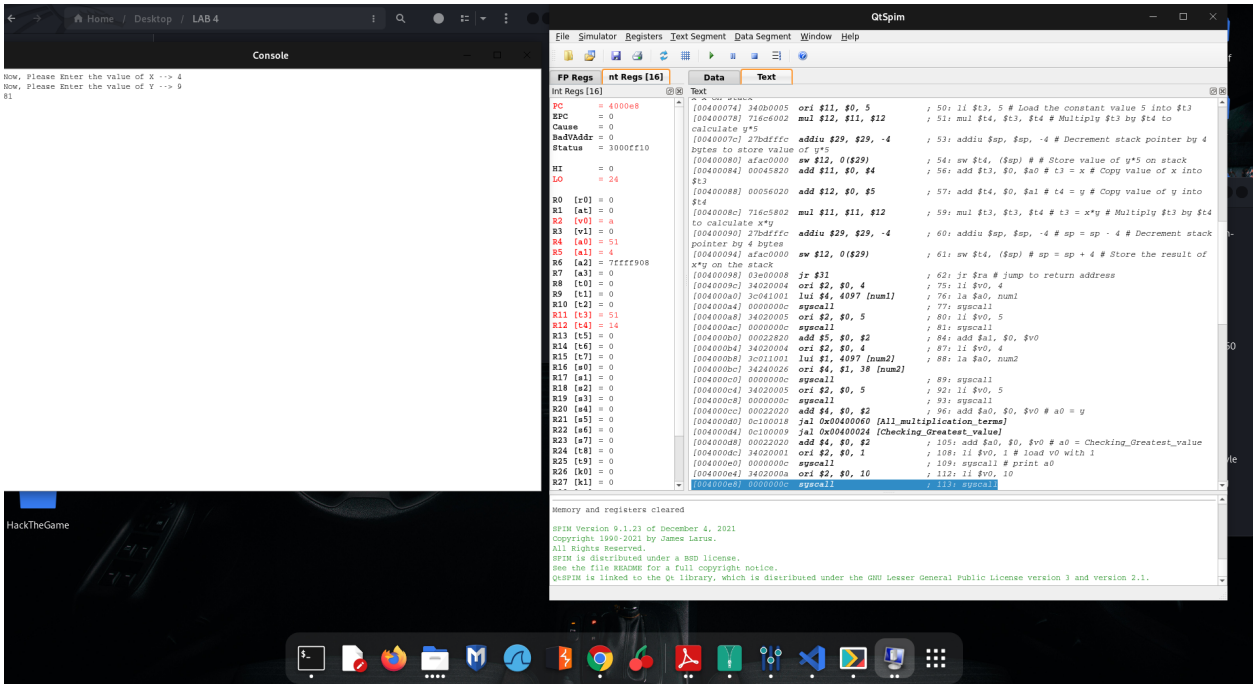

Console
Please Enter The Number --> 4
Please Enter The Number --> 6
120
QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
nt Regs [16]
Data
Text
Int Regs [16]
PC = 400040
EPC = 0
Cause = 0
BadVAddr = 0
Status = 3000ff10
HI = 0
LO = 78
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = a
R3 [v1] = 0
R4 [a0] = 78
R5 [a1] = 7fffff8f4
R6 [a2] = 7ffff908
R7 [a3] = 0
R8 [t0] = 0
R9 [t1] = 0
R10 [t2] = 0
R11 [t3] = 18
R12 [t4] = 78
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
[00400040] 20020004 addi $2, $0, 4 ; 21: addi $v0, $0, 4
[00400044] 3c041001 lui $4, 4097 [prompt4] ; 22: la $a0, prompt4
[00400048] 0000000c syscall ; 23: syscall
[0040004c] 20020005 addi $2, $0, 5 ; 24: addi $v0, $0, 5
[00400050] 0000000c syscall ; 25: syscall
[00400054] afa20000 sw $2, 0($29) ; 26: sw $v0, 0($sp)
[00400058] 23bdfffc addi $29, $29, -4 ; 27: addi $sp, $sp, -4
[0040005c] 8fab0004 lw $11, 4($29) ; 30: lw $t3, 4($sp)
[00400060] 8fac0008 lw $12, 8($29) ; 31: lw $t4, 8($sp)
[00400064] 018b0018 mult $12, $11 ; 34: mult $t4, $t3
[00400068] 00006012 mflo $12 ; 35: mflo $t4
[0040006c] afac0000 sw $12, 0($29) ; 36: sw $t4, 0($sp)
[00400070] 23bdfffc addi $29, $29, -4 ; 37: addi $sp, $sp, -4
[00400074] 200cfff6 addi $12, $0, -10 ; 40: addi $t4, $0, -10
[00400078] 8fab000c lw $11, 12($29) ; 41: lw $t3, 12($sp)
[0040007c] 016c0018 mult $11, $12 ; 42: mult $t3, $t4
[00400080] 00006012 mflo $12 ; 43: mflo $t4
[00400084] afac0000 sw $12, 0($29) ; 45: sw $t4, 0($sp)
[00400088] 23bdfffc addi $29, $29, -4 ; 46: addi $sp, $sp, -4
[0040008c] 200c0014 addi $12, $0, 20 ; 49: addi $t4, $0, 20
[00400090] 8fab000c lw $11, 12($29) ; 50: lw $t3, 12($sp)
[00400094] 016c0018 mult $11, $12 ; 51: mult $t3, $t4
[00400098] 00006012 mflo $12 ; 52: mflo $t4
[0040009c] afac0000 sw $12, 0($29) ; 54: sw $t4, 0($sp)
[004000a0] 23bdfffc addi $29, $29, -4 ; 55: addi $sp, $sp, -4
[004000a4] 200c0010 addi $12, $0, 16 ; 58: addi $t4, $0, 16
[004000a8] 8fab0004 lw $11, 4($29) ; 60: lw $t3, 4($sp)
[004000ac] 018b6020 add $12, $12, $11 ; 61: add $t4, $t4, $t3
[004000b0] 8fab0008 lw $11, 8($29) ; 63: lw $t3, 8($sp)
[004000b4] 016c6020 add $12, $11, $12 ; 64: add $t4, $t3, $t4
[004000b8] 8fab000c lw $11, 12($29) ; 66: lw $t3, 12($sp)
[004000bc] 016c6020 add $12, $11, $12 ; 67: add $t4, $t3, $t4
[004000c0] 01802020 add $4, $12, $0 ; 70: add $a0, $t4, $0
[004000c4] 20020001 addi $2, $0, 1 ; 71: addi $v0, $0, 1
[004000c8] 0000000c syscall ; 72: syscall
[004000cc] 2002000a addi $2, $0, 10 ; 75: addi $v0, $0, 10
[00400040] 0000000c syscall ; 76: syscall
Memory and registers cleared
SPIM Version 9.1.23 of December 4, 2021
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.
HackTheGame

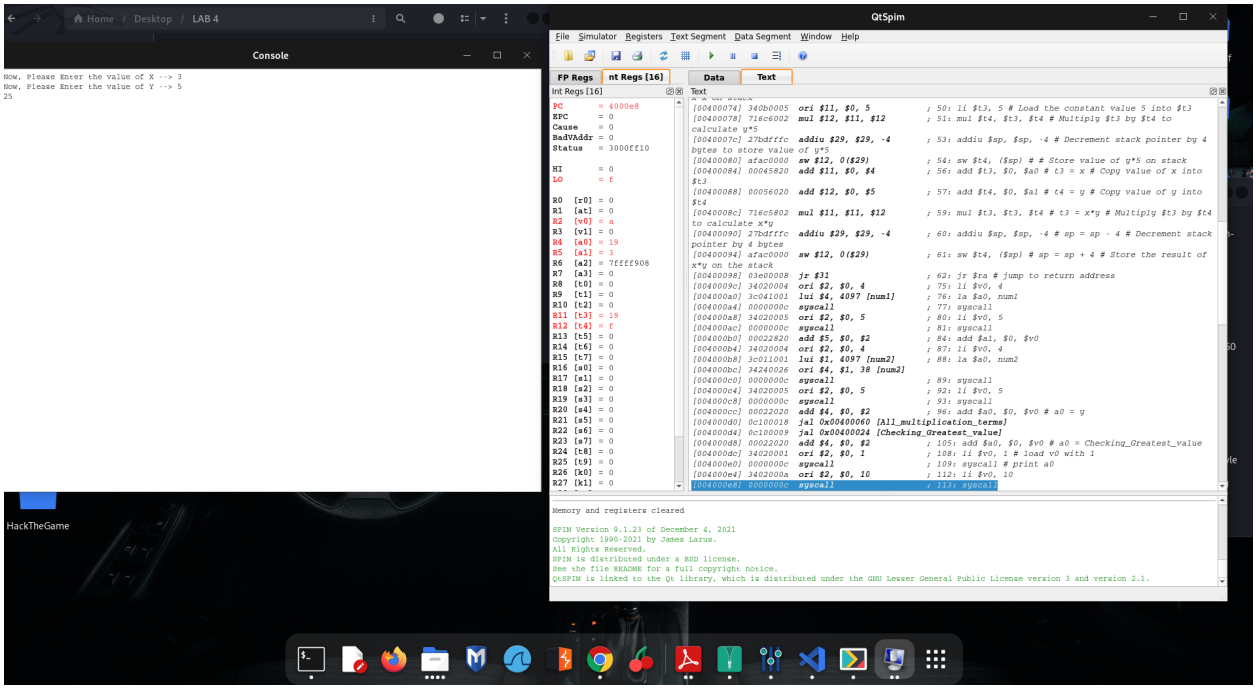

Console
Please Enter The Number --> 3
Please Enter The Number --> 4
78
QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
nt Regs [16]
Data
Text
Int Regs [16]
PC = 400040
EPC = 0
Cause = 0
BadVAddr = 0
Status = 3000ff10
HI = 0
LO = 50
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = a
R3 [v1] = 0
R4 [a0] = 4e
R5 [a1] = 7fffff8f4
R6 [a2] = 7ffff908
R7 [a3] = 0
R8 [t0] = 0
R9 [t1] = 0
R10 [t2] = 0
R11 [t3] = c
R12 [t4] = 4e
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
[00400040] 20020004 addi $2, $0, 4 ; 21: addi $v0, $0, 4
[00400044] 3c041001 lui $4, 4097 [prompt4] ; 22: la $a0, prompt4
[00400048] 0000000c syscall ; 23: syscall
[0040004c] 20020005 addi $2, $0, 5 ; 24: addi $v0, $0, 5
[00400050] 0000000c syscall ; 25: syscall
[00400054] afa20000 sw $2, 0($29) ; 26: sw $v0, 0($sp)
[00400058] 23bdfffc addi $29, $29, -4 ; 27: addi $sp, $sp, -4
[0040005c] 8fab0004 lw $11, 4($29) ; 30: lw $t3, 4($sp)
[00400060] 8fac0008 lw $12, 8($29) ; 31: lw $t4, 8($sp)
[00400064] 018b0018 mult $12, $11 ; 34: mult $t4, $t3
[00400068] 00006012 mflo $12 ; 35: mflo $t4
[0040006c] afac0000 sw $12, 0($29) ; 36: sw $t4, 0($sp)
[00400070] 23bdfffc addi $29, $29, -4 ; 37: addi $sp, $sp, -4
[00400074] 200cfff6 addi $12, $0, -10 ; 40: addi $t4, $0, -10
[00400078] 8fab000c lw $11, 12($29) ; 41: lw $t3, 12($sp)
[0040007c] 016c0018 mult $11, $12 ; 42: mult $t3, $t4
[00400080] 00006012 mflo $12 ; 43: mflo $t4
[00400084] afac0000 sw $12, 0($29) ; 45: sw $t4, 0($sp)
[00400088] 23bdfffc addi $29, $29, -4 ; 46: addi $sp, $sp, -4
[0040008c] 200c0014 addi $12, $0, 20 ; 49: addi $t4, $0, 20
[00400090] 8fab000c lw $11, 12($29) ; 50: lw $t3, 12($sp)
[00400094] 016c0018 mult $11, $12 ; 51: mult $t3, $t4
[00400098] 00006012 mflo $12 ; 52: mflo $t4
[0040009c] afac0000 sw $12, 0($29) ; 54: sw $t4, 0($sp)
[004000a0] 23bdfffc addi $29, $29, -4 ; 55: addi $sp, $sp, -4
[004000a4] 200c0010 addi $12, $0, 16 ; 58: addi $t4, $0, 16
[004000a8] 8fab0004 lw $11, 4($29) ; 60: lw $t3, 4($sp)
[004000ac] 018b6020 add $12, $12, $11 ; 61: add $t4, $t4, $t3
[004000b0] 8fab0008 lw $11, 8($29) ; 63: lw $t3, 8($sp)
[004000b4] 016c6020 add $12, $11, $12 ; 64: add $t4, $t3, $t4
[004000b8] 8fab000c lw $11, 12($29) ; 66: lw $t3, 12($sp)
[004000bc] 016c6020 add $12, $11, $12 ; 67: add $t4, $t3, $t4
[004000c0] 01802020 add $4, $12, $0 ; 70: add $a0, $t4, $0
[004000c4] 20020001 addi $2, $0, 1 ; 71: addi $v0, $0, 1
[004000c8] 0000000c syscall ; 72: syscall
[004000cc] 2002000a addi $2, $0, 10 ; 75: addi $v0, $0, 10
[00400040] 0000000c syscall ; 76: syscall
Memory and registers cleared
SPIM Version 9.1.23 of December 4, 2021
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.
HackTheGame

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 maximum of two integers and use it to find the maximum of these three expressions. Display the result.

Output -




Home / Desktop / LAB 4
Console
Now, Please Enter the value of X --> 4
Now, Please Enter the value of Y --> 9
81
HackTheGame
QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
nt Regs [16]
Data
Text
Int Regs [16]
PC = 4000e8
EPC = 0
Cause = 0
BadVAddr = 0
Status = 3000ff10
HI = 0
LO = 24
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = a
R3 [v1] = 0
R4 [a0] = 51
R5 [a1] = 4
R6 [a2] = 7fff908
R7 [a3] = 0
R8 [t0] = 0
R9 [t1] = 0
R10 [t2] = 0
R11 [t3] = 51
R12 [t4] = 14
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
[00400074] 340b0005 ori $11, $0, 5 ; 50: li $t3, 5 # Load the constant value 5 into $t3
[00400078] 716c6002 mul $12, $11, $12 ; 51: mul $t4, $t3, $t4 # Multiply $t3 by $t4 to calculate y*5
[0040007c] 27bdfffc addiu $29, $29, -4 ; 53: addiu $sp, $sp, -4 # Decrement stack pointer by 4 bytes to store value of y*5
[00400080] afac0000 sw $12, 0($29) ; 54: sw $t4, ($sp) # # Store value of y*5 on stack
[00400084] 00045820 add $11, $0, $4 ; 56: add $t3, $0, $a0 # t3 = x # Copy value of x into $t3
[00400088] 00056020 add $12, $0, $5 ; 57: add $t4, $0, $a1 # t4 = y # Copy value of y into $t4
[0040008c] 716c5802 mul $11, $11, $12 ; 59: mul $t3, $t3, $t4 # t3 = x*y # Multiply $t3 by $t4 to calculate x*y
[00400090] 27bdfffc addiu $29, $29, -4 ; 60: addiu $sp, $sp, -4 # sp = sp - 4 # Decrement stack pointer by 4 bytes
[00400094] afac0000 sw $12, 0($29) ; 61: sw $t4, ($sp) # sp = sp + 4 # Store the result of x*y on the stack
[00400098] 03e00008 jr $31 ; 62: jr $ra # jump to return address
[0040009c] 34020004 ori $2, $0, 4 ; 75: li $v0, 4
[004000a0] 3c041001 lui $4, 4097 [num1] ; 76: la $a0, num1
[004000a4] 0000000c syscall ; 77: syscall
[004000a8] 34020005 ori $2, $0, 5 ; 80: li $v0, 5
[004000ac] 0000000c syscall ; 81: syscall
[004000b0] 00022820 add $5, $0, $2 ; 84: add $a1, $0, $v0
[004000b4] 34020004 ori $2, $0, 4 ; 87: li $v0, 4
[004000b8] 3c011001 lui $1, 4097 [num2] ; 88: la $a0, num2
[004000bc] 34240026 ori $4, $1, 38 [num2]
[004000c0] 0000000c syscall ; 89: syscall
[004000c4] 34020005 ori $2, $0, 5 ; 92: li $v0, 5
[004000c8] 0000000c syscall ; 93: syscall
[004000cc] 00022020 add $4, $0, $2 ; 96: add $a0, $0, $v0 # a0 = y
[004000d0] 0c100018 jal 0x00400060 [All_multiplication_terms]
[004000d4] 0c100009 jal 0x00400024 [Checking_Greatest_value]
[004000d8] 00022020 add $4, $0, $2 ; 105: add $a0, $0, $v0 # a0 = Checking_Greatest_value
[004000dc] 34020001 ori $2, $0, 1 ; 108: li $v0, 1 # load v0 with 1
[004000e0] 0000000c syscall ; 109: syscall # print a0
[004000e4] 3402000a ori $2, $0, 10 ; 112: li $v0, 10
[004000e8] 0000000c syscall ; 113: syscall
Memory and registers cleared
SPIM Version 9.1.23 of December 4, 2021
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.


Home / Desktop / LAB 4
Console
Now, Please Enter the value of X --> 3
Now, Please Enter the value of Y --> 5
25
HackTheGame
QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
nt Regs [16]
Data
Text
Int Regs [16]
PC = 4000e8
EPC = 0
Cause = 0
BadVAddr = 0
Status = 3000ff10
HI = 0
LO = f
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = a
R3 [v1] = 0
R4 [a0] = 19
R5 [a1] = 3
R6 [a2] = 7fff908
R7 [a3] = 0
R8 [t0] = 0
R9 [t1] = 0
R10 [t2] = 0
R11 [t3] = 19
R12 [t4] = f
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
[00400074] 340b0005 ori $11, $0, 5 ; 50: li $t3, 5 # Load the constant value 5 into $t3
[00400078] 716c6002 mul $12, $11, $12 ; 51: mul $t4, $t3, $t4 # Multiply $t3 by $t4 to calculate y*5
[0040007c] 27bdfffc addiu $29, $29, -4 ; 53: addiu $sp, $sp, -4 # Decrement stack pointer by 4 bytes to store value of y*5
[00400080] afac0000 sw $12, 0($29) ; 54: sw $t4, ($sp) # # Store value of y*5 on stack
[00400084] 00045820 add $11, $0, $4 ; 56: add $t3, $0, $a0 # t3 = x # Copy value of x into $t3
[00400088] 00056020 add $12, $0, $5 ; 57: add $t4, $0, $a1 # t4 = y # Copy value of y into $t4
[0040008c] 716c5802 mul $11, $11, $12 ; 59: mul $t3, $t3, $t4 # t3 = x*y # Multiply $t3 by $t4 to calculate x*y
[00400090] 27bdfffc addiu $29, $29, -4 ; 60: addiu $sp, $sp, -4 # sp = sp - 4 # Decrement stack pointer by 4 bytes
[00400094] afac0000 sw $12, 0($29) ; 61: sw $t4, ($sp) # sp = sp + 4 # Store the result of x*y on the stack
[00400098] 03e00008 jr $31 ; 62: jr $ra # jump to return address
[0040009c] 34020004 ori $2, $0, 4 ; 75: li $v0, 4
[004000a0] 3c041001 lui $4, 4097 [num1] ; 76: la $a0, num1
[004000a4] 0000000c syscall ; 77: syscall
[004000a8] 34020005 ori $2, $0, 5 ; 80: li $v0, 5
[004000ac] 0000000c syscall ; 81: syscall
[004000b0] 00022820 add $5, $0, $2 ; 84: add $a1, $0, $v0
[004000b4] 34020004 ori $2, $0, 4 ; 87: li $v0, 4
[004000b8] 3c011001 lui $1, 4097 [num2] ; 88: la $a0, num2
[004000bc] 34240026 ori $4, $1, 38 [num2]
[004000c0] 0000000c syscall ; 89: syscall
[004000c4] 34020005 ori $2, $0, 5 ; 92: li $v0, 5
[004000c8] 0000000c syscall ; 93: syscall
[004000cc] 00022020 add $4, $0, $2 ; 96: add $a0, $0, $v0 # a0 = y
[004000d0] 0c100018 jal 0x00400060 [All_multiplication_terms]
[004000d4] 0c100009 jal 0x00400024 [Checking_Greatest_value]
[004000d8] 00022020 add $4, $0, $2 ; 105: add $a0, $0, $v0 # a0 = Checking_Greatest_value
[004000dc] 34020001 ori $2, $0, 1 ; 108: li $v0, 1 # load v0 with 1
[004000e0] 0000000c syscall ; 109: syscall # print a0
[004000e4] 3402000a ori $2, $0, 10 ; 112: li $v0, 10
[004000e8] 0000000c syscall ; 113: syscall
Memory and registers cleared
SPIM Version 9.1.23 of December 4, 2021
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.

Thank You