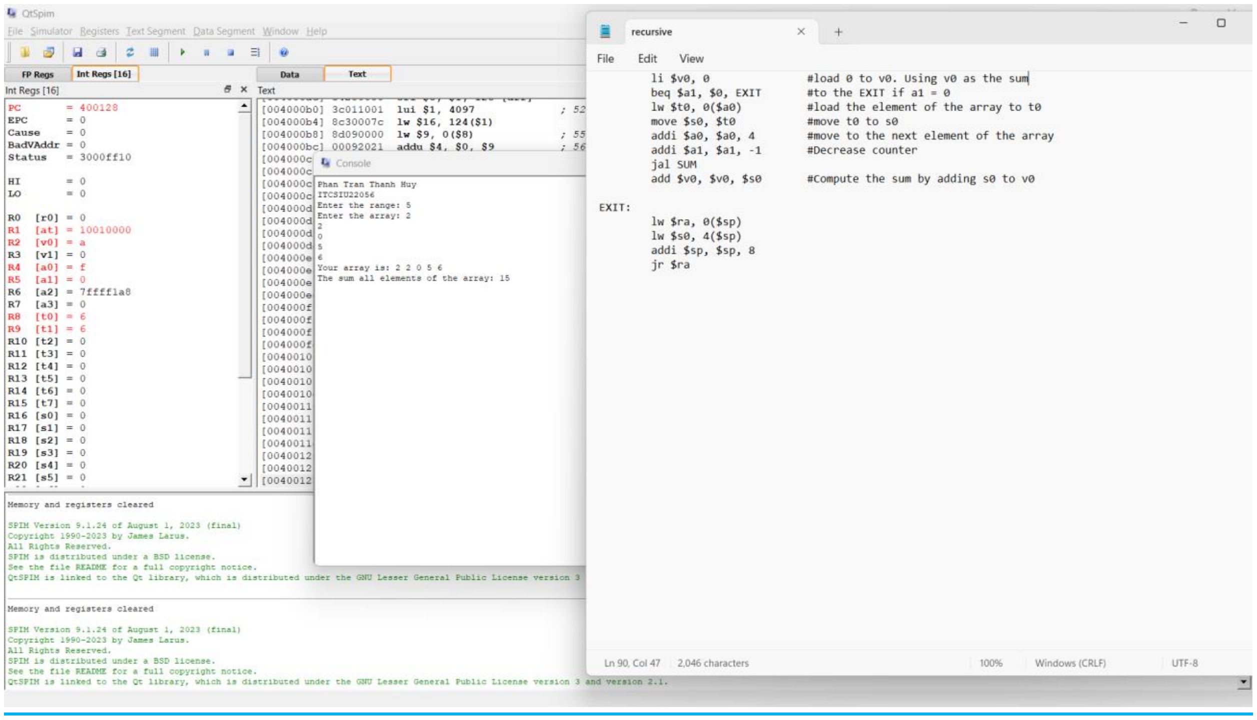
Name: Phan Tran Thanh Huy

ID: ITCSIU22056

Lab 6



```
PC       = 400128
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] = f
R5  [a1] = 0
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 6
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
```

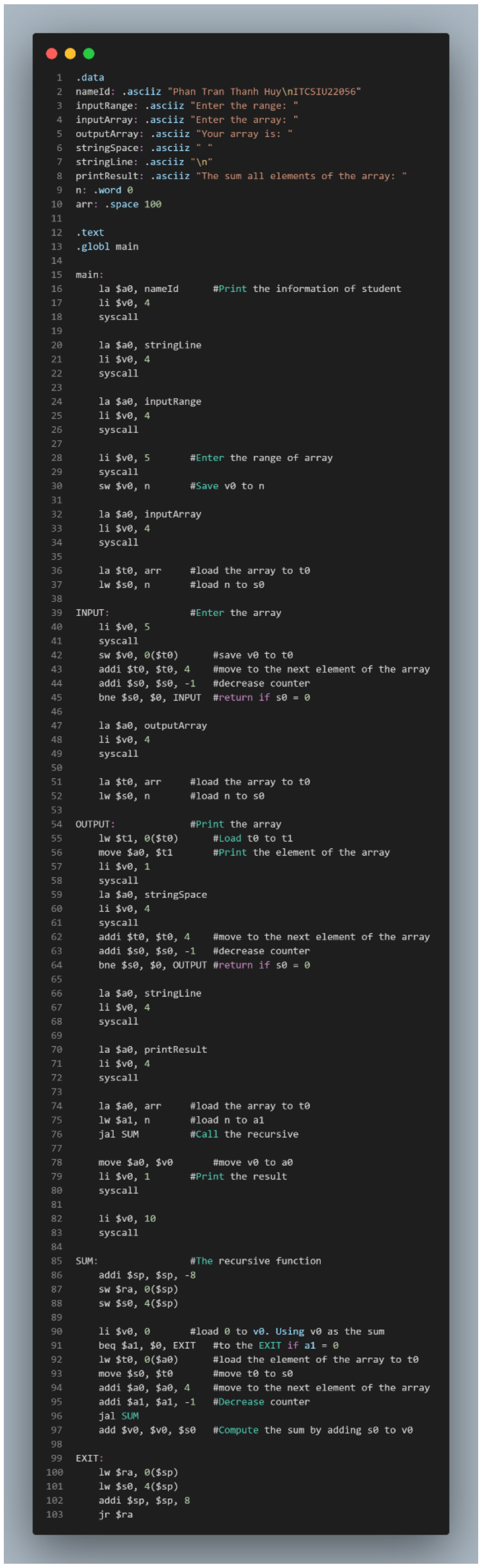
```
[004000b0] 3c011001  lui $1, 4097          ; 52
[004000b4] 8c30007c  lw $16, 124($1)
[004000b8] 8d090000  lw $9, 0($8)          ; 55
[004000bc] 00092021  addu $4, $0, $9       ; 56
```

```
Phan Tran Thanh Huy
ITCSIU22056
Enter the range: 5
Enter the array: 2
2
0
5
6
Your array is: 2 2 0 5 6
The sum all elements of the array: 15
```

```
    li $v0, 0                   #load 0 to v0. Using v0 as the sum
    beq $a1, $0, EXIT           #to the EXIT if a1 = 0
    lw $t0, 0($a0)              #load the element of the array to t0
    move $s0, $t0               #move t0 to s0
    addi $a0, $a0, 4            #move to the next element of the array
    addi $a1, $a1, -1           #Decrease counter
    jal SUM
    add $v0, $v0, $s0           #Compute the sum by adding s0 to v0

EXIT:
    lw $ra, 0($sp)
    lw $s0, 4($sp)
    addi $sp, $sp, 8
    jr $ra
```

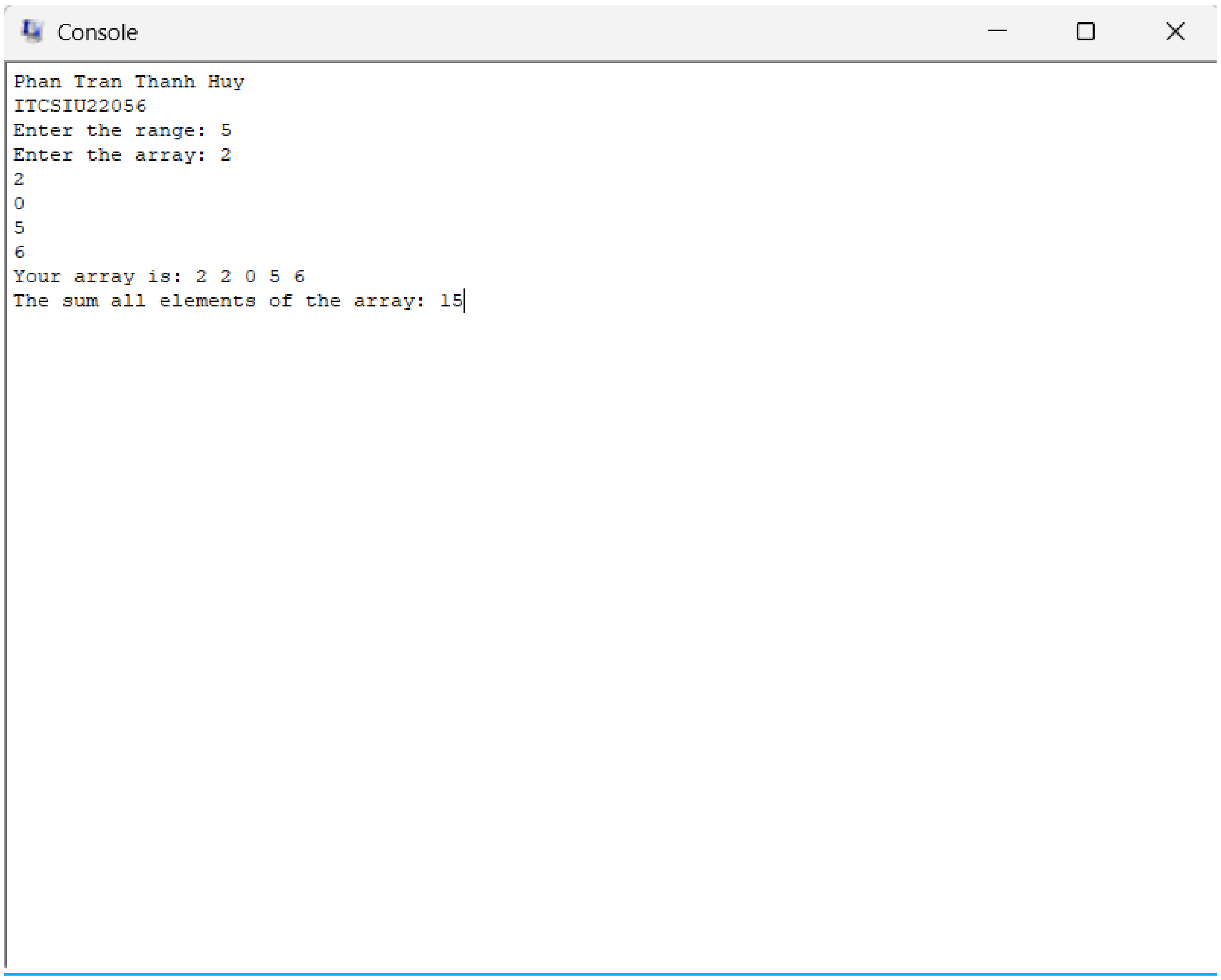
```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
Copyright 1990-2023 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

Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
Copyright 1990-2023 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.
```

Code:



```
.data
nameId: .asciiz "Phan Tran Thanh Huy\nITCSIU22056"
inputRange: .asciiz "Enter the range: "
inputArray: .asciiz "Enter the array: "
outputArray: .asciiz "Your array is: "
stringSpace: .asciiz " "
stringLine: .asciiz "\n"
printResult: .asciiz "The sum all elements of the array: "
n: .word 0
arr: .space 100

.text
.globl main

main:
    la $a0, nameId      #Print the information of student
    li $v0, 4
    syscall

    la $a0, stringLine
    li $v0, 4
    syscall

    la $a0, inputRange
    li $v0, 4
    syscall

    li $v0, 5       #Enter the range of array
    syscall
    sw $v0, n       #Save v0 to n

    la $a0, inputArray
    li $v0, 4
    syscall

    la $t0, arr     #load the array to t0
    lw $s0, n       #load n to s0

INPUT:              #Enter the array
    li $v0, 5
    syscall
    sw $v0, 0($t0)      #save v0 to t0
    addi $t0, $t0, 4    #move to the next element of the array
    addi $s0, $s0, -1   #decrease counter
    bne $s0, $0, INPUT  #return if s0 = 0

    la $a0, outputArray
    li $v0, 4
    syscall

    la $t0, arr     #load the array to t0
    lw $s0, n       #load n to s0

OUTPUT:             #Print the array
    lw $t1, 0($t0)      #Load t0 to t1
    move $a0, $t1       #Print the element of the array
    li $v0, 1
    syscall
    la $a0, stringSpace
    li $v0, 4
    syscall
    addi $t0, $t0, 4    #move to the next element of the array
    addi $s0, $s0, -1   #decrease counter
    bne $s0, $0, OUTPUT #return if s0 = 0

    la $a0, stringLine
    li $v0, 4
    syscall

    la $a0, printResult
    li $v0, 4
    syscall

    la $a0, arr     #load the array to t0
    lw $a1, n       #load n to a1
    jal SUM         #Call the recursive

    move $a0, $v0       #move v0 to a0
    li $v0, 1       #Print the result
    syscall

    li $v0, 10
    syscall

SUM:                #The recursive function
    addi $sp, $sp, -8
    sw $ra, 0($sp)
    sw $s0, 4($sp)

    li $v0, 0       #load 0 to v0. Using v0 as the sum
    beq $a1, $0, EXIT   #to the EXIT if a1 = 0
    lw $t0, 0($a0)      #load the element of the array to t0
    move $s0, $t0       #move t0 to s0
    addi $a0, $a0, 4    #move to the next element of the array
    addi $a1, $a1, -1   #Decrease counter
    jal SUM
    add $v0, $v0, $s0   #Compute the sum by adding s0 to v0

EXIT:
    lw $ra, 0($sp)
    lw $s0, 4($sp)
    addi $sp, $sp, 8
    jr $ra
```

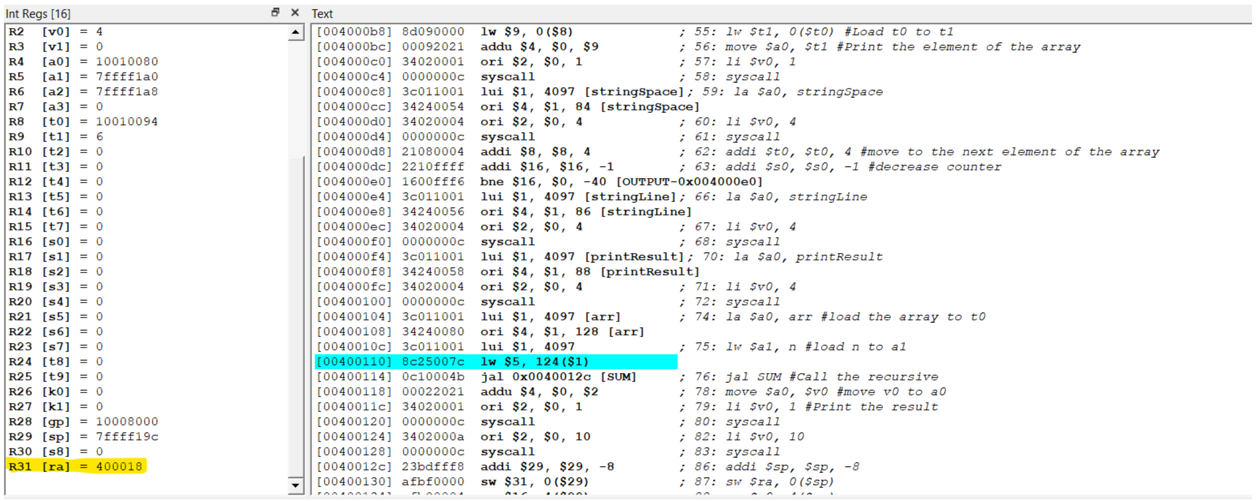
Result:



```
Phan Tran Thanh Huy
ITCSIU22056
Enter the range: 5
Enter the array: 2
2
0
5
6
Your array is: 2 2 0 5 6
The sum all elements of the array: 15
```

Explaination of the address:

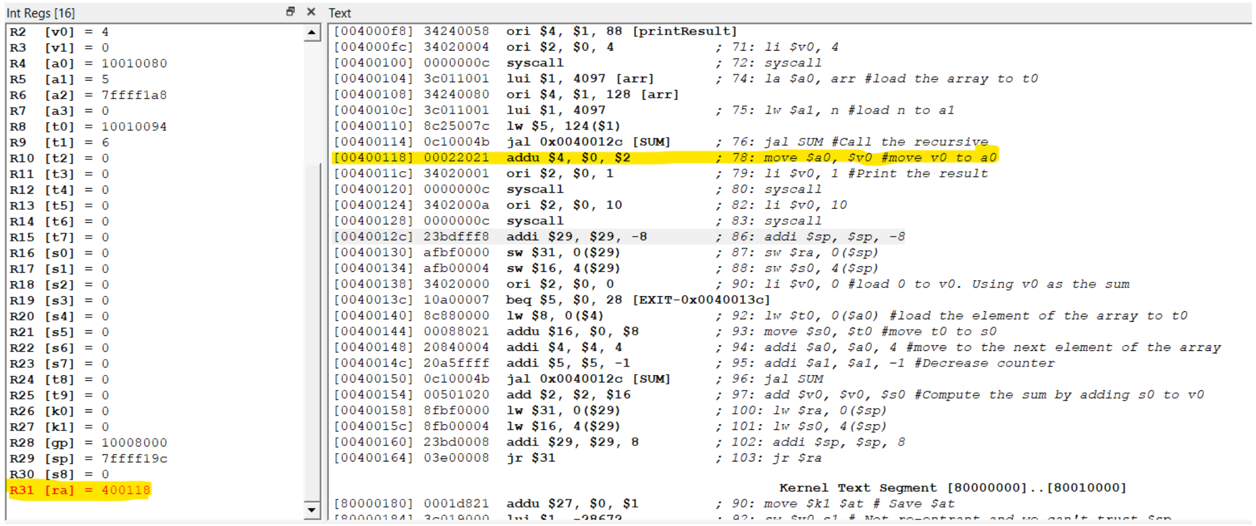
Before calling the procedure SUM, the initial return address is stored in memory address [00400018] and the initial stack pointer address is stored in memory address [7ffff19c].



```
Int Regs [16]
R2  [v0] = 4
R3  [v1] = 0
R4  [a0] = 10010080
R5  [a1] = 7ffff1a0
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 10010094
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
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 10008000
R29 [sp] = 7ffff19c
R30 [s8] = 0
R31 [ra] = 400018
```

```
Text
[004000b8] 8d090000  lw $9, 0($8)              ; 55: lw $t1, 0($t0) #Load t0 to t1
[004000bc] 00092021  addu $4, $0, $9           ; 56: move $a0, $t1 #Print the element of the array
[004000c0] 34020001  ori $2, $0, 1             ; 57: li $v0, 1
[004000c4] 0000000c  syscall                   ; 58: syscall
[004000c8] 3c011001  lui $1, 4097 [stringSpace]; 59: la $a0, stringSpace
[004000cc] 34240054  ori $4, $1, 84 [stringSpace]
[004000d0] 34020004  ori $2, $0, 4             ; 60: li $v0, 4
[004000d4] 0000000c  syscall                   ; 61: syscall
[004000d8] 21080004  addi $8, $8, 4            ; 62: addi $t0, $t0, 4 #move to the next element of the array
[004000dc] 2210ffff  addi $16, $16, -1         ; 63: addi $s0, $s0, -1 #decrease counter
[004000e0] 1600fff6  bne $16, $0, -40 [OUTPUT-0x004000e0]
[004000e4] 3c011001  lui $1, 4097 [stringLine]; 66: la $a0, stringLine
[004000e8] 34240056  ori $4, $1, 86 [stringLine]
[004000ec] 34020004  ori $2, $0, 4             ; 67: li $v0, 4
[004000f0] 0000000c  syscall                   ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4             ; 71: li $v0, 4
[00400100] 0000000c  syscall                   ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]        ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097              ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]      ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2           ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1             ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                   ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10            ; 82: li $v0, 10
[00400128] 0000000c  syscall                   ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8         ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)            ; 87: sw $ra, 0($sp)
```

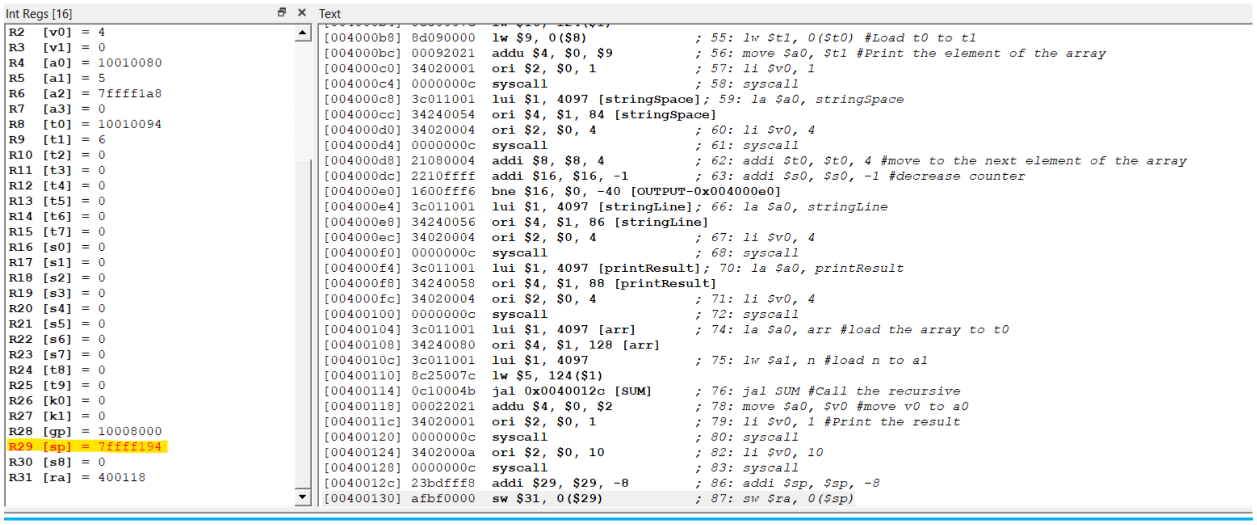
After calling the procedure SUM, the return address is stored in memory address [00400118], which is the return address of the callee to the main function.



```
Int Regs [16]
R2  [v0] = 4
R3  [v1] = 0
R4  [a0] = 10010080
R5  [a1] = 5
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 10010094
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
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 10008000
R29 [sp] = 7ffff19c
R30 [s8] = 0
R31 [ra] = 400118
```

```
Text
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4             ; 71: li $v0, 4
[00400100] 0000000c  syscall                   ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]        ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097              ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]      ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2           ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1             ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                   ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10            ; 82: li $v0, 10
[00400128] 0000000c  syscall                   ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8         ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)            ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)            ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0             ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)              ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8          ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4            ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1           ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]      ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16           ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)            ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)            ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8          ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                    ; 103: jr $ra

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

And the stack pointer is stored in memory address [7ffff194]



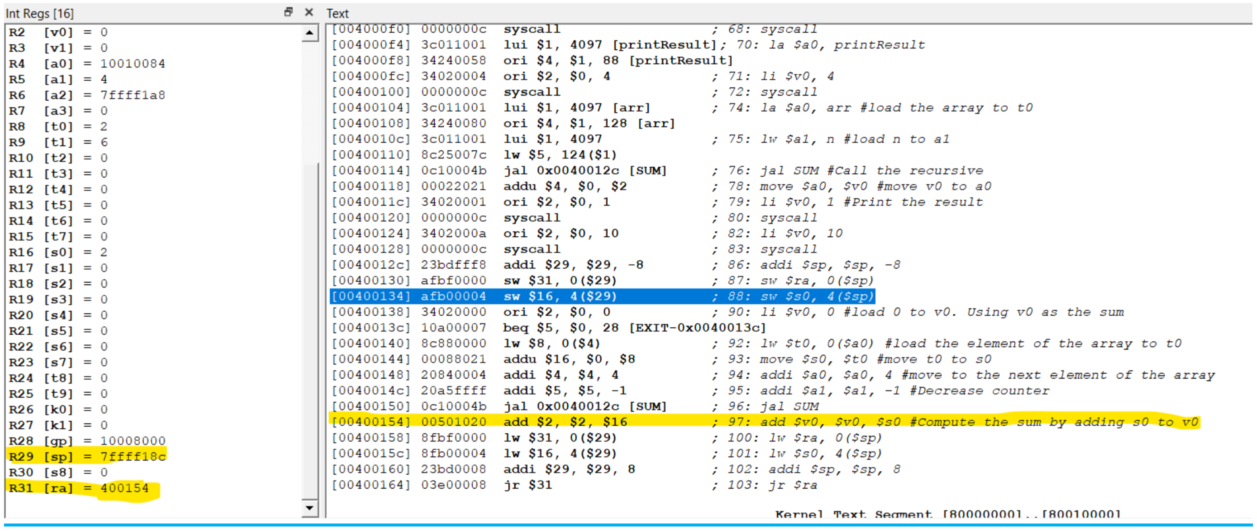
Int Regs [16]

```
R2  [v0] = 4
R3  [v1] = 0
R4  [a0] = 10010080
R5  [a1] = 5
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 10010094
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
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 10008000
R29 [sp] = 7ffff194
R30 [s8] = 0
R31 [ra] = 400118
```

Text

```
[004000b8] 8d090000  lw $9, 0($8)              ; 55: lw $t1, 0($t0) #Load t0 to t1
[004000bc] 00092021  addu $4, $0, $9           ; 56: move $a0, $t1 #Print the element of the array
[004000c0] 34020001  ori $2, $0, 1             ; 57: li $v0, 1
[004000c4] 0000000c  syscall                   ; 58: syscall
[004000c8] 3c011001  lui $1, 4097 [stringSpace]; 59: la $a0, stringSpace
[004000cc] 34240054  ori $4, $1, 84 [stringSpace]
[004000d0] 34020004  ori $2, $0, 4             ; 60: li $v0, 4
[004000d4] 0000000c  syscall                   ; 61: syscall
[004000d8] 21080004  addi $8, $8, 4            ; 62: addi $t0, $t0, 4 #move to the next element of the array
[004000dc] 2210ffff  addi $16, $16, -1         ; 63: addi $s0, $s0, -1 #decrease counter
[004000e0] 1600fff6  bne $16, $0, -40 [OUTPUT-0x004000e0]
[004000e4] 3c011001  lui $1, 4097 [stringLine]; 66: la $a0, stringLine
[004000e8] 34240056  ori $4, $1, 86 [stringLine]
[004000ec] 34020004  ori $2, $0, 4             ; 67: li $v0, 4
[004000f0] 0000000c  syscall                   ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4             ; 71: li $v0, 4
[00400100] 0000000c  syscall                   ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]        ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097              ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]      ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2           ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1             ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                   ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10            ; 82: li $v0, 10
[00400128] 0000000c  syscall                   ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8         ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)            ; 87: sw $ra, 0($sp)
```

In the first save, the stack pointer is stored in memory address [7ffff18c] and the return address is stored in memory address [00400154], which is the return address of the add operation.



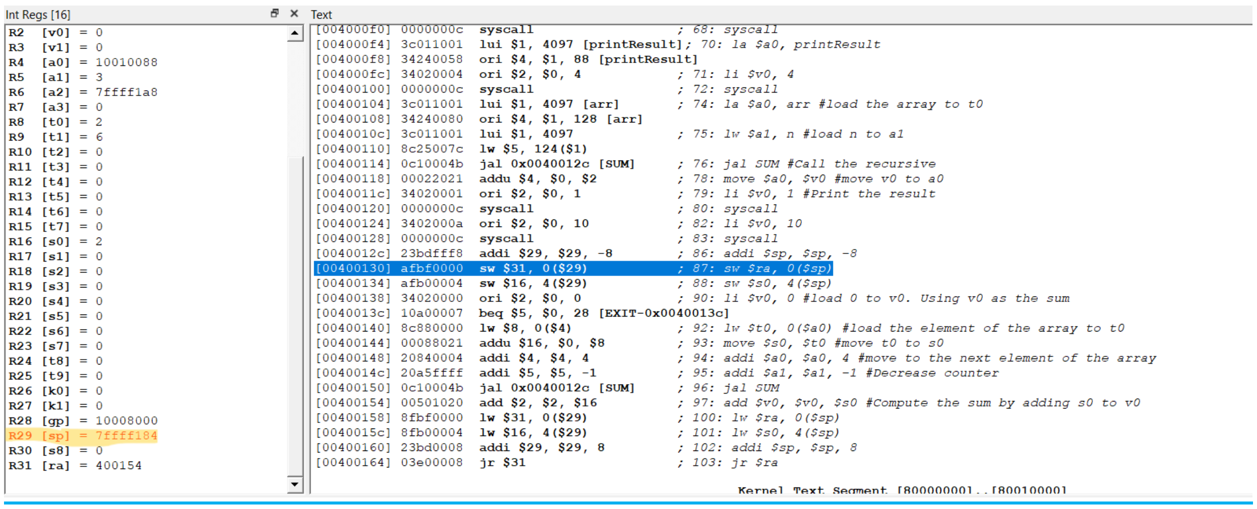
Int Regs [16]

```
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 10010084
R5  [a1] = 4
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 2
R9  [t1] = 6
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 2
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] = 10008000
R29 [sp] = 7ffff18c
R30 [s8] = 0
R31 [ra] = 400154
```

Text

```
[004000f0] 0000000c  syscall                   ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4             ; 71: li $v0, 4
[00400100] 0000000c  syscall                   ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]        ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097              ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]      ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2           ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1             ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                   ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10            ; 82: li $v0, 10
[00400128] 0000000c  syscall                   ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8         ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)            ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)            ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0             ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)              ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8          ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4            ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1           ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]      ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16           ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)            ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)            ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8          ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                    ; 103: jr $ra

                        Kernel Text Segment [80000000]..[80010000]
```

In the second save, the stack pointer is stored in memory address [7ffff184]



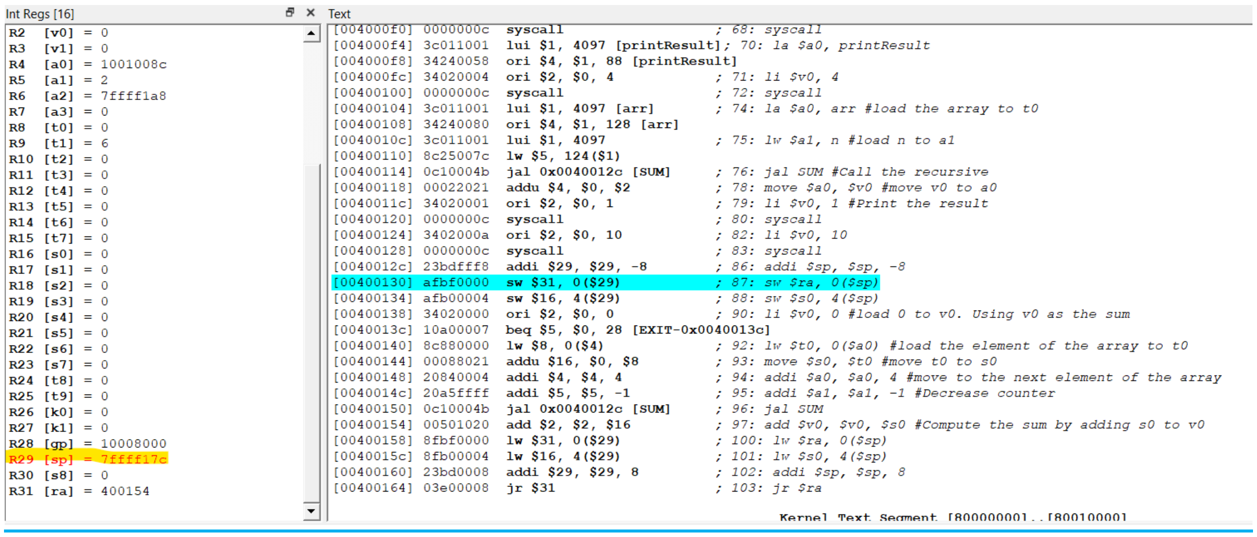
Int Regs [16]

```
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 10010088
R5  [a1] = 3
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 2
R9  [t1] = 6
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 2
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] = 10008000
R29 [sp] = 7ffff184
R30 [s8] = 0
R31 [ra] = 400154
```

Text

```
[004000f0] 0000000c  syscall                   ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4             ; 71: li $v0, 4
[00400100] 0000000c  syscall                   ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]        ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097              ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]      ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2           ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1             ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                   ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10            ; 82: li $v0, 10
[00400128] 0000000c  syscall                   ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8         ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)            ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)            ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0             ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)              ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8          ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4            ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1           ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]      ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16           ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)            ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)            ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8          ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                    ; 103: jr $ra

                        Kernel Text Segment [80000000]..[80010000]
```

In the third save, the stack pointer is stored in memory address [7ffff17c]



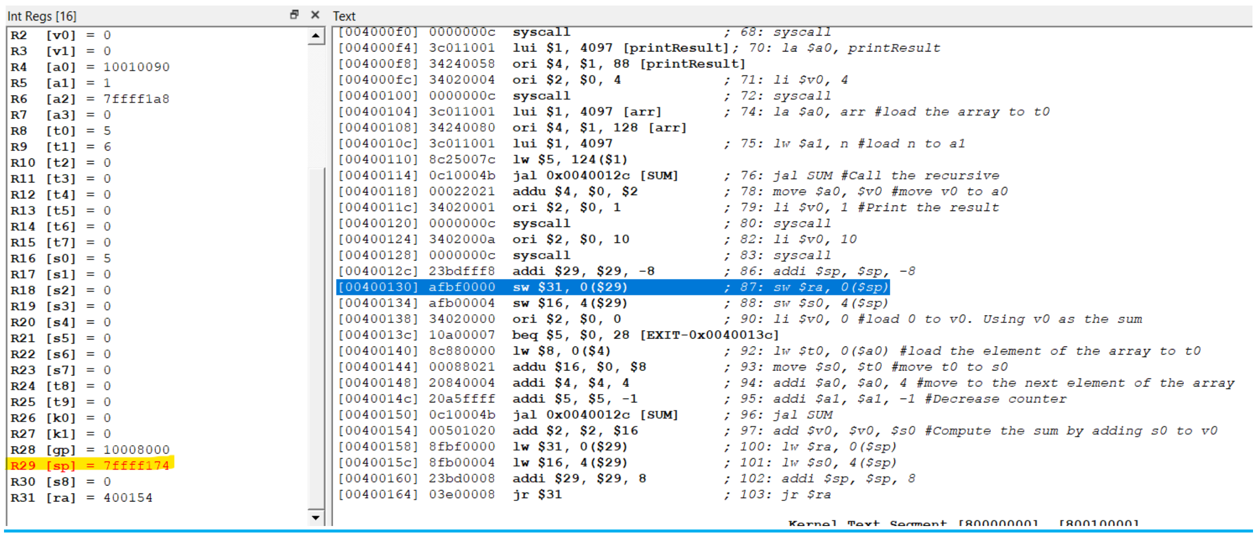
Int Regs [16]

```
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 1001008c
R5  [a1] = 2
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 0
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
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 10008000
R29 [sp] = 7ffff17c
R30 [s8] = 0
R31 [ra] = 400154
```

Text

```
[004000f0] 0000000c  syscall                   ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4             ; 71: li $v0, 4
[00400100] 0000000c  syscall                   ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]        ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097              ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]      ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2           ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1             ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                   ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10            ; 82: li $v0, 10
[00400128] 0000000c  syscall                   ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8         ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)            ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)            ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0             ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)              ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8          ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4            ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1           ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]      ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16           ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)            ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)            ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8          ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                    ; 103: jr $ra

                        Kernel Text Segment [80000000]..[80010000]
```

In the fourth save, the stack pointer is stored in memory address [7ffff174]

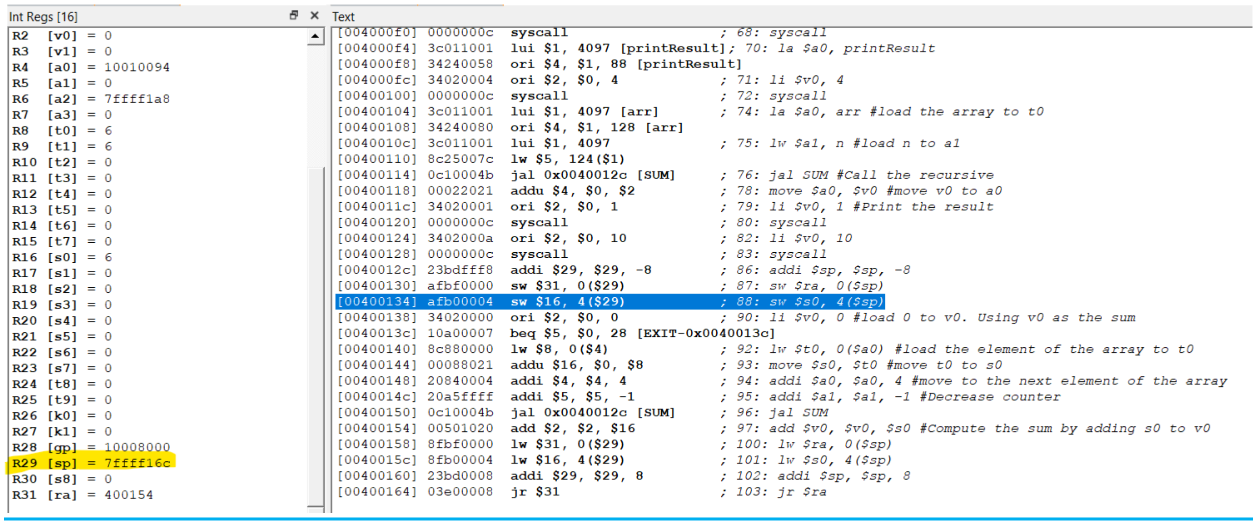

Int Regs [16]

```
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 10010090
R5  [a1] = 1
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 5
R9  [t1] = 6
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 5
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] = 10008000
R29 [sp] = 7ffff174
R30 [s8] = 0
R31 [ra] = 400154
```

Text

```
[004000f0] 0000000c  syscall                    ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4              ; 71: li $v0, 4
[00400100] 0000000c  syscall                    ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]         ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097               ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]       ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2            ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1              ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                    ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10             ; 82: li $v0, 10
[00400128] 0000000c  syscall                    ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8          ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)             ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)             ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0              ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)               ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8           ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4             ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1            ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]       ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16            ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)             ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)             ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8           ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                     ; 103: jr $ra

                                   Kernel Text Segment [80000000]..[80010000]
```

In the fifth save, the stack pointer is stored in memory address [7ffff16c]

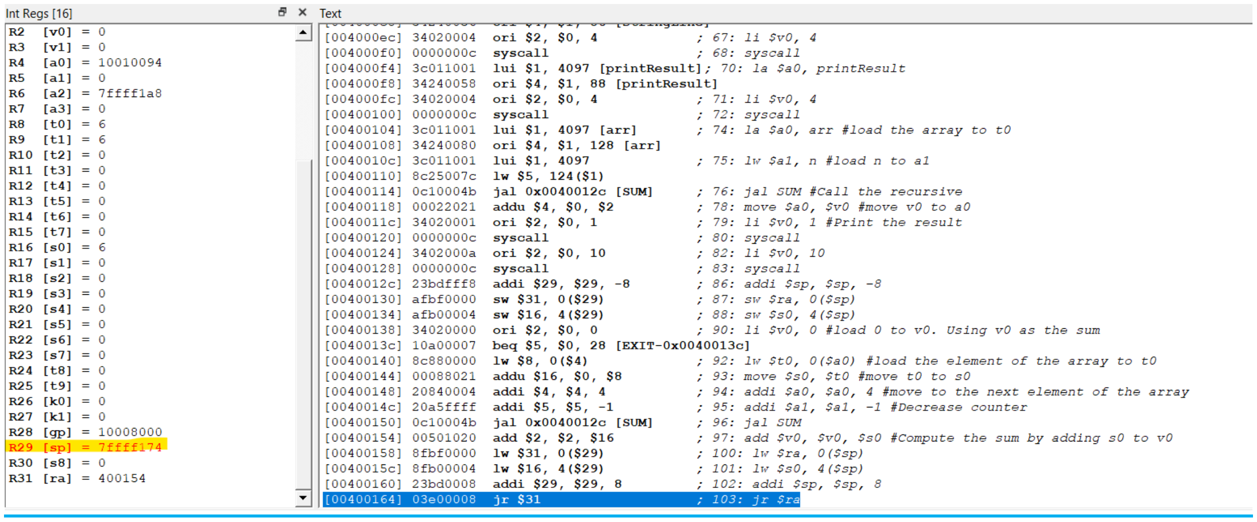

Int Regs [16]

```
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 10010094
R5  [a1] = 0
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 6
R9  [t1] = 6
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 6
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] = 10008000
R29 [sp] = 7ffff16c
R30 [s8] = 0
R31 [ra] = 400154
```

Text

```
[004000f0] 0000000c  syscall                    ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4              ; 71: li $v0, 4
[00400100] 0000000c  syscall                    ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]         ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097               ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]       ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2            ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1              ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                    ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10             ; 82: li $v0, 10
[00400128] 0000000c  syscall                    ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8          ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)             ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)             ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0              ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)               ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8           ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4             ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1            ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]       ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16            ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)             ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)             ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8           ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                     ; 103: jr $ra
```

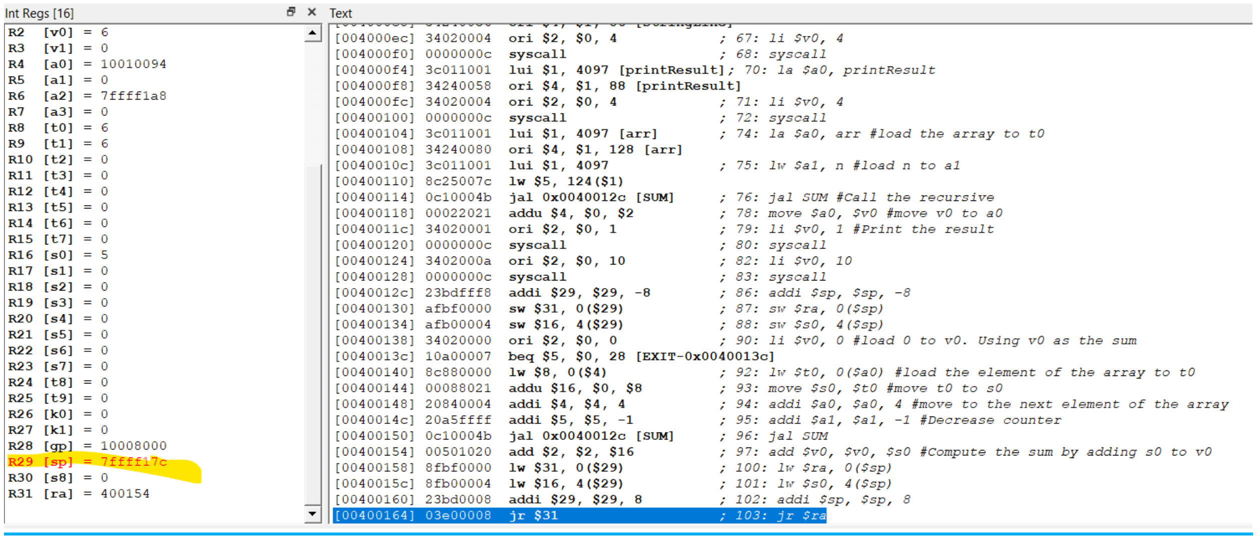
Now using all the value contain in the stack. The stack pointer is stored back to memory address [7ffff174]



```
Int Regs [16]
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 10010094
R5  [a1] = 0
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 6
R9  [t1] = 6
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 6
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] = 10008000
R29 [sp] = 7ffff174
R30 [s8] = 0
R31 [ra] = 400154
```

```
Text
[004000ec] 34020004  ori $2, $0, 4            ; 67: li $v0, 4
[004000f0] 0000000c  syscall                  ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4            ; 71: li $v0, 4
[00400100] 0000000c  syscall                  ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]       ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097             ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]     ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2          ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1            ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                  ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10           ; 82: li $v0, 10
[00400128] 0000000c  syscall                  ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8        ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)           ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)           ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0            ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)             ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8         ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4           ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1          ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]     ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16          ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)           ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)           ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8         ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                   ; 103: jr $ra
```

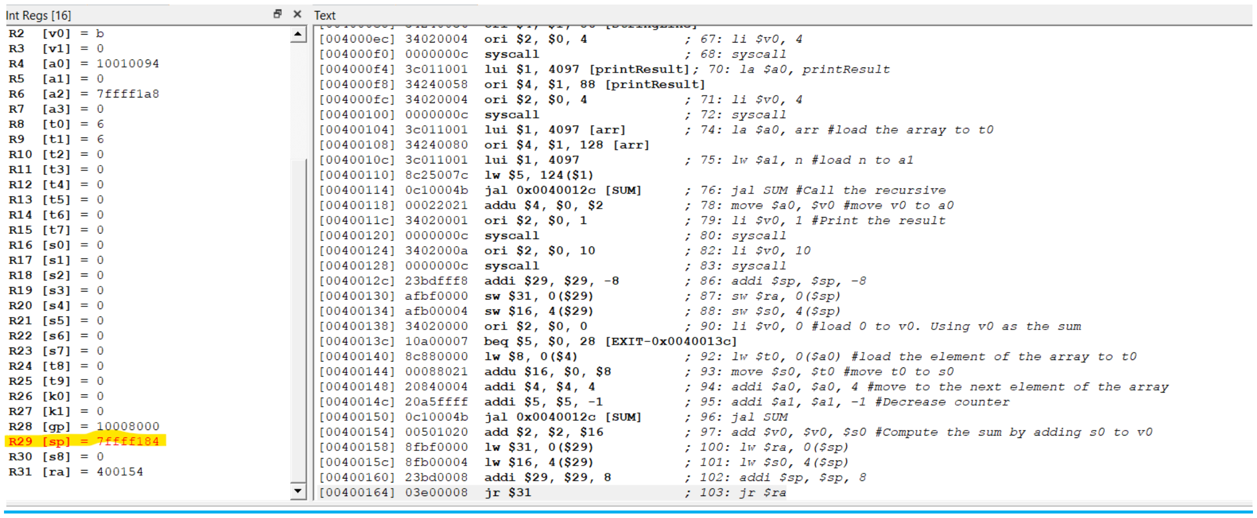
After the add operation, the stack pointer is stored back to memory address [7ffff17c]



```
Int Regs [16]
R2  [v0] = 6
R3  [v1] = 0
R4  [a0] = 10010094
R5  [a1] = 0
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 6
R9  [t1] = 6
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 5
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] = 10008000
R29 [sp] = 7ffff17c
R30 [s8] = 0
R31 [ra] = 400154
```

```
Text
[004000ec] 34020004  ori $2, $0, 4            ; 67: li $v0, 4
[004000f0] 0000000c  syscall                  ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4            ; 71: li $v0, 4
[00400100] 0000000c  syscall                  ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]       ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097             ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]     ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2          ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1            ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                  ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10           ; 82: li $v0, 10
[00400128] 0000000c  syscall                  ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8        ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)           ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)           ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0            ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)             ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8         ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4           ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1          ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]     ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16          ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)           ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)           ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8         ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                   ; 103: jr $ra
```

After the add operation, the stack pointer is stored back to memory address [7ffff184]

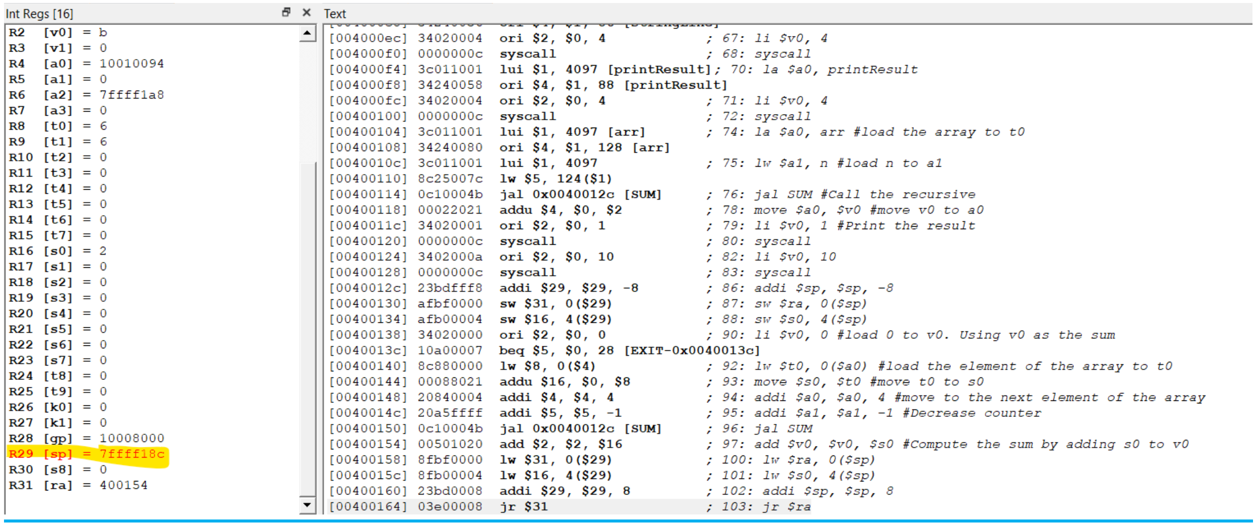

Int Regs [16]

```
R2  [v0] = b
R3  [v1] = 0
R4  [a0] = 10010094
R5  [a1] = 0
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 6
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
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 10008000
R29 [sp] = 7ffff184
R30 [s8] = 0
R31 [ra] = 400154
```

Text

```
[004000ec] 34020004  ori $2, $0, 4            ; 67: li $v0, 4
[004000f0] 0000000c  syscall                  ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4            ; 71: li $v0, 4
[00400100] 0000000c  syscall                  ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]       ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097             ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]     ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2          ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1            ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                  ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10           ; 82: li $v0, 10
[00400128] 0000000c  syscall                  ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8        ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)           ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)           ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0            ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)             ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8         ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4           ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1          ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]     ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16          ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)           ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)           ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8         ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                   ; 103: jr $ra
```

After the add operation, the stack pointer is stored back to memory address [7ffff18c]

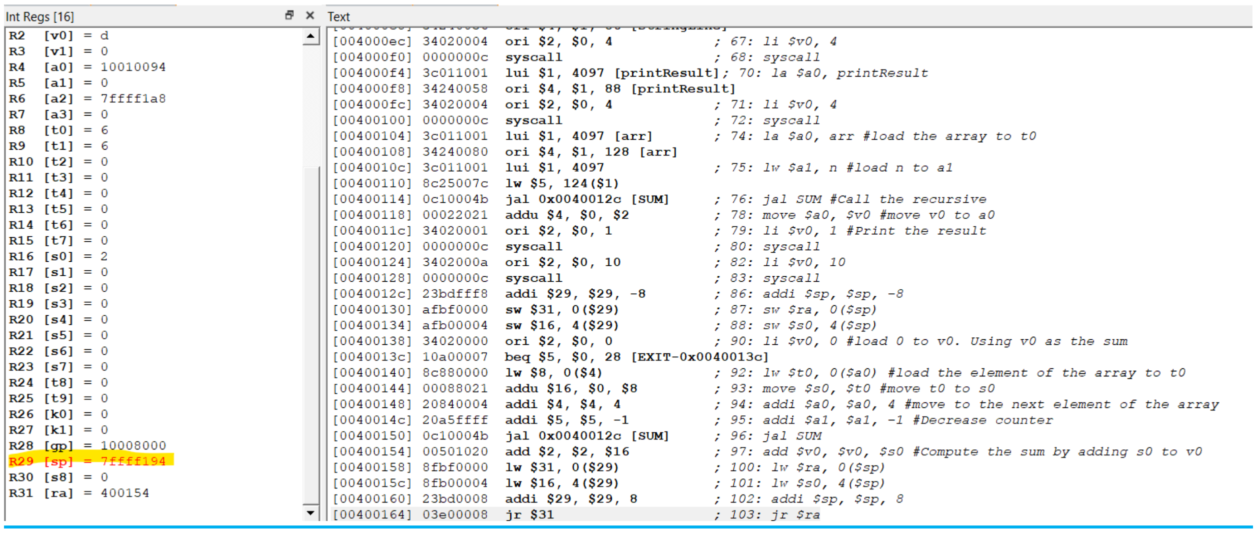

Int Regs [16]

```
R2  [v0] = b
R3  [v1] = 0
R4  [a0] = 10010094
R5  [a1] = 0
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 6
R9  [t1] = 6
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 2
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] = 10008000
R29 [sp] = 7ffff18c
R30 [s8] = 0
R31 [ra] = 400154
```

Text

```
[004000ec] 34020004  ori $2, $0, 4            ; 67: li $v0, 4
[004000f0] 0000000c  syscall                  ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4            ; 71: li $v0, 4
[00400100] 0000000c  syscall                  ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]       ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097             ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]     ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2          ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1            ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                  ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10           ; 82: li $v0, 10
[00400128] 0000000c  syscall                  ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8        ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)           ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)           ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0            ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)             ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8         ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4           ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1          ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]     ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16          ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)           ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)           ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8         ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                   ; 103: jr $ra
```

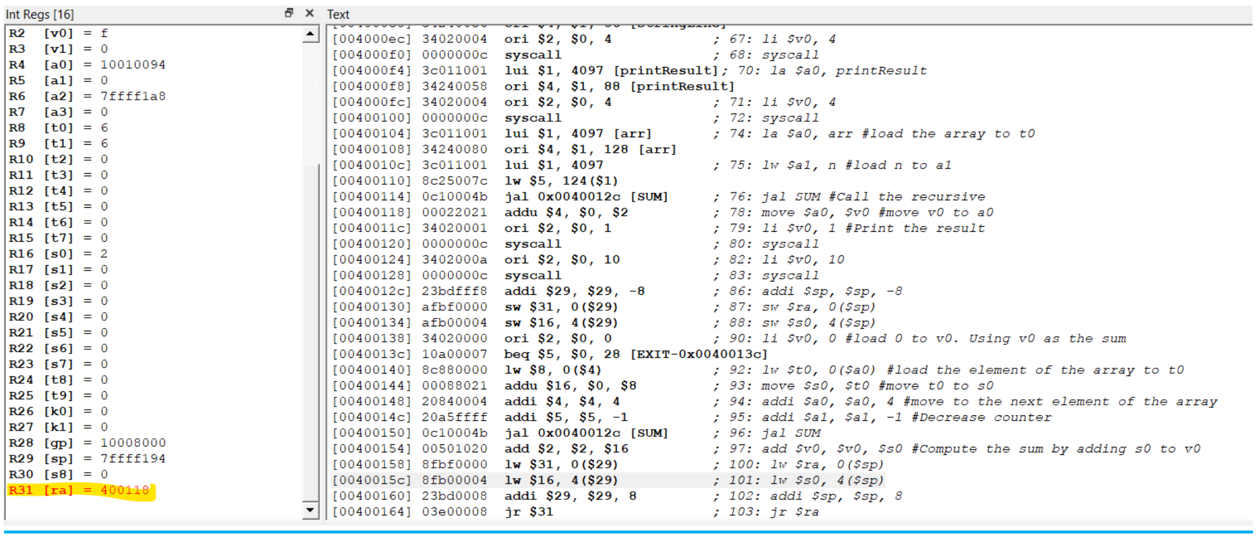
After the add operation, the stack pointer is stored back to memory address [7ffff194]



```
R2  [v0] = d
R3  [v1] = 0
R4  [a0] = 10010094
R5  [a1] = 0
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 6
R9  [t1] = 6
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 2
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] = 10008000
R29 [sp] = 7ffff194
R30 [s8] = 0
R31 [ra] = 400154
```

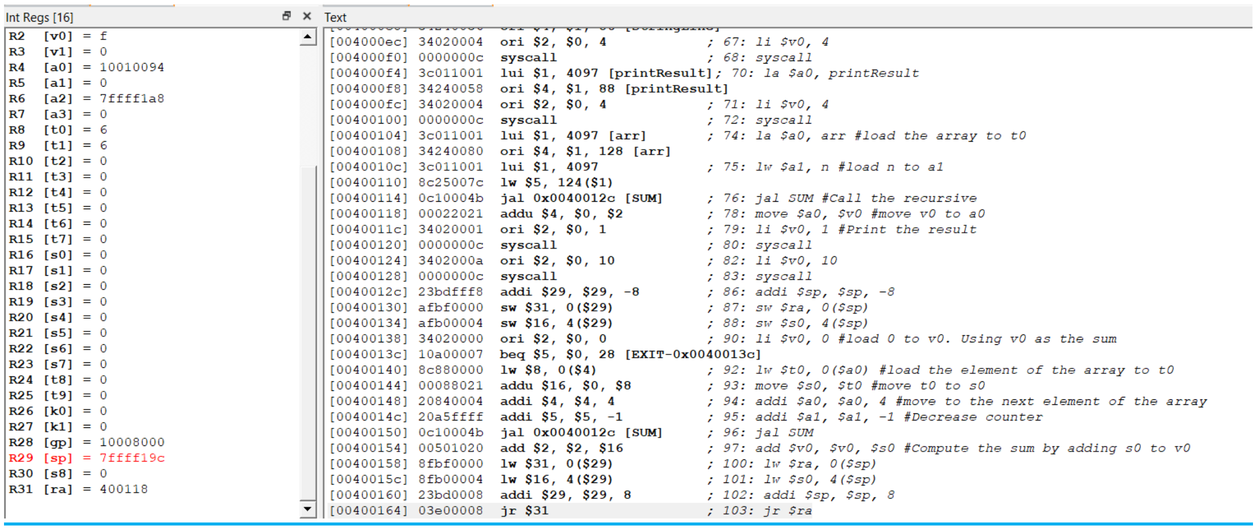
```
[004000ec] 34020004  ori $2, $0, 4             ; 67: li $v0, 4
[004000f0] 0000000c  syscall                   ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4             ; 71: li $v0, 4
[00400100] 0000000c  syscall                   ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]        ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097              ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]      ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2           ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1             ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                   ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10            ; 82: li $v0, 10
[00400128] 0000000c  syscall                   ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8         ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)            ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)            ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0             ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)              ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8          ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4            ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1           ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]      ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16           ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)            ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)            ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8          ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                    ; 103: jr $ra
```

After the add operation, the stack pointer is stored back to memory address [7ffff19c] and the return address is stored back to memory address [00400118] as callee of the main function.



```
R2  [v0] = f
R3  [v1] = 0
R4  [a0] = 10010094
R5  [a1] = 0
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 6
R9  [t1] = 6
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 2
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] = 10008000
R29 [sp] = 7ffff194
R30 [s8] = 0
R31 [ra] = 400118
```

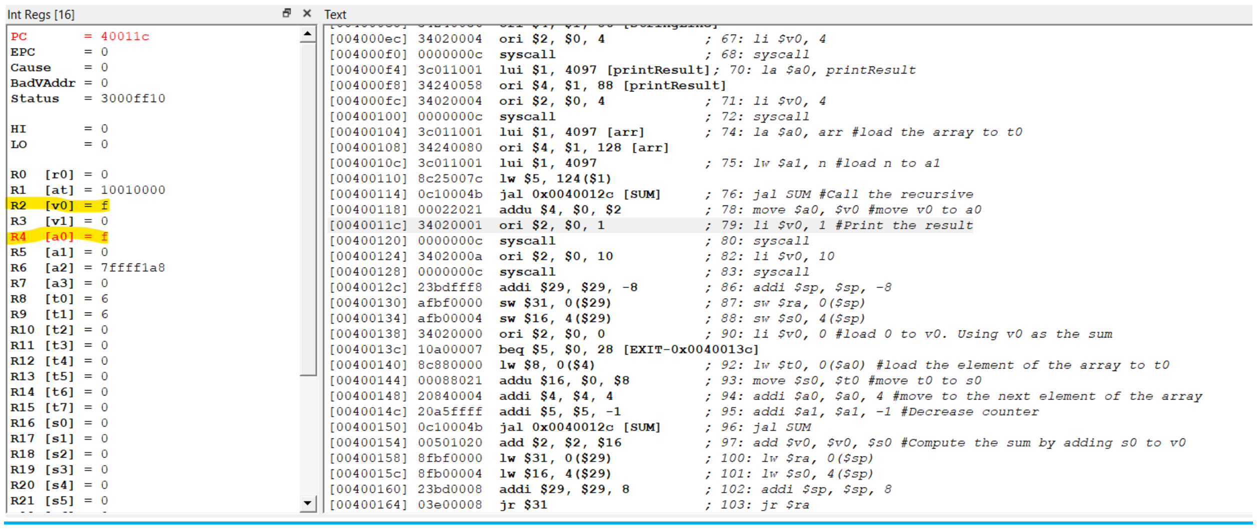
```
[004000ec] 34020004  ori $2, $0, 4             ; 67: li $v0, 4
[004000f0] 0000000c  syscall                   ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4             ; 71: li $v0, 4
[00400100] 0000000c  syscall                   ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]        ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097              ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]      ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2           ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1             ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                   ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10            ; 82: li $v0, 10
[00400128] 0000000c  syscall                   ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8         ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)            ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)            ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0             ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)              ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8          ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4            ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1           ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]      ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16           ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)            ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)            ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8          ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                    ; 103: jr $ra
```



```
R2  [v0] = f
R3  [v1] = 0
R4  [a0] = 10010094
R5  [a1] = 0
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 6
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
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 10008000
R29 [sp] = 7ffff19c
R30 [s8] = 0
R31 [ra] = 400118
```

```
[004000ec] 34020004  ori $2, $0, 4              ; 67: li $v0, 4
[004000f0] 0000000c  syscall                    ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4              ; 71: li $v0, 4
[00400100] 0000000c  syscall                    ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]         ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097               ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]       ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2            ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1              ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                    ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10             ; 82: li $v0, 10
[00400128] 0000000c  syscall                    ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8          ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)             ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)             ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0              ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)               ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8           ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4             ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1            ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]       ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16            ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)             ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)             ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8           ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                     ; 103: jr $ra
```

The register v0 and a0 now contain the result



```
PC       = 40011c
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = f
R3  [v1] = 0
R4  [a0] = f
R5  [a1] = 0
R6  [a2] = 7ffff1a8
R7  [a3] = 0
R8  [t0] = 6
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
```

```
[004000ec] 34020004  ori $2, $0, 4              ; 67: li $v0, 4
[004000f0] 0000000c  syscall                    ; 68: syscall
[004000f4] 3c011001  lui $1, 4097 [printResult]; 70: la $a0, printResult
[004000f8] 34240058  ori $4, $1, 88 [printResult]
[004000fc] 34020004  ori $2, $0, 4              ; 71: li $v0, 4
[00400100] 0000000c  syscall                    ; 72: syscall
[00400104] 3c011001  lui $1, 4097 [arr]         ; 74: la $a0, arr #load the array to t0
[00400108] 34240080  ori $4, $1, 128 [arr]
[0040010c] 3c011001  lui $1, 4097               ; 75: lw $a1, n #load n to a1
[00400110] 8c25007c  lw $5, 124($1)
[00400114] 0c10004b  jal 0x0040012c [SUM]       ; 76: jal SUM #Call the recursive
[00400118] 00022021  addu $4, $0, $2            ; 78: move $a0, $v0 #move v0 to a0
[0040011c] 34020001  ori $2, $0, 1              ; 79: li $v0, 1 #Print the result
[00400120] 0000000c  syscall                    ; 80: syscall
[00400124] 3402000a  ori $2, $0, 10             ; 82: li $v0, 10
[00400128] 0000000c  syscall                    ; 83: syscall
[0040012c] 23bdfff8  addi $29, $29, -8          ; 86: addi $sp, $sp, -8
[00400130] afbf0000  sw $31, 0($29)             ; 87: sw $ra, 0($sp)
[00400134] afb00004  sw $16, 4($29)             ; 88: sw $s0, 4($sp)
[00400138] 34020000  ori $2, $0, 0              ; 90: li $v0, 0 #load 0 to v0. Using v0 as the sum
[0040013c] 10a00007  beq $5, $0, 28 [EXIT-0x0040013c]
[00400140] 8c880000  lw $8, 0($4)               ; 92: lw $t0, 0($a0) #load the element of the array to t0
[00400144] 00088021  addu $16, $0, $8           ; 93: move $s0, $t0 #move t0 to s0
[00400148] 20840004  addi $4, $4, 4             ; 94: addi $a0, $a0, 4 #move to the next element of the array
[0040014c] 20a5ffff  addi $5, $5, -1            ; 95: addi $a1, $a1, -1 #Decrease counter
[00400150] 0c10004b  jal 0x0040012c [SUM]       ; 96: jal SUM
[00400154] 00501020  add $2, $2, $16            ; 97: add $v0, $v0, $s0 #Compute the sum by adding s0 to v0
[00400158] 8fbf0000  lw $31, 0($29)             ; 100: lw $ra, 0($sp)
[0040015c] 8fb00004  lw $16, 4($29)             ; 101: lw $s0, 4($sp)
[00400160] 23bd0008  addi $29, $29, 8           ; 102: addi $sp, $sp, 8
[00400164] 03e00008  jr $31                     ; 103: jr $ra
```