

**San Jose State University**  
**Department of Computer Engineering**

## **CMPE 200 Lab Report**

---

### **Assignment 4 Report**

**Title:** Array Processing, Stack and Recursive Procedure

**Semester:** Fall 2023

**Date:** 10/03/2023

**by**

**Name:** Sai Kashyap Kurella

**SID:** 016018925

## Source Code

```
# $a0 = array base address
# $a1 = n
# $s0 = n!
Main:
    addi $a0, $0, 0x100 # array base address = 0x100
    addi $a1, $0, 0 # i = 0
    addi $t0, $0, 3
    addi $t1, $0, 50 # $t1 = 50
CreateArray_Loop:
    slt $t2, $a1, $t1 # i < 50?
    beq $t2, $0, Exit_Loop # if not then exit loop
    sll $t2, $a1, 2 # $t2 = i * 4 (byte offset)
    add $t2, $t2, $a0 # address of array[i]
    mult $a1, $t0
    mflo $t3 # $t3 = i * 3
    sw $t3, 0($t2) # save array[i]
    addi $a1, $a1, 1 # i = i + 1
    j CreateArray_Loop
Exit_Loop:
    # your code goes in here...
    # arithmetic calculation
    lw $s1, 356($t2)
    lw $s2, 376($t2)
    add $s2, $s2, $s1
    addi $s3, $zero, 30
    div $s2, $s3
    mflo $s4
    move $a1, $s4
    sw $a1, 0($0)
    # ...
    # factorial computation
    jal factorial # call procedure
    add $s0, $v0, $0 # return value
    sw $s0, 16($0)
    j exit
factorial:
    addi $sp, $sp, -8
    sw $ra, 4($sp)
    sw $a1, 0($sp)
    slti $t0, $a1, 2
    beq $t0, $zero, L1
    addi $v0, $zero, 1
    addi $sp, $sp, 8
    jr $ra
L1: addi $a1, $a1, -1
    jal factorial
    lw $a1, 0($sp)
    lw $ra, 4($sp)
    addi $sp, $sp, 8
    mul $v0, $a1, $v0
    jr $ra
exit:
```

## CMPE 200 Assignment 4 Test Log

**Programmer's Names:** Sai Kashyap Kurella

**Date:** 10/03/2023

Record the observed contents of registers and data memory after each instruction is executed.

Addr	MIPS Instruction	Machine Code	Registers				Memory Content	
			\$a1	\$sp	\$ra	\$v0	[0x00]	[0x10]
3034	lw \$s1,356(\$t2)	0x8d510164	0x00000032	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
3038	lw \$s2,376(\$t2)	0x8d520178	0x00000032	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
303c	add \$s2,\$s2,\$s1	0x02519020	0x00000032	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
3040	addi \$s3,\$zero,30	0x20130013	0x00000032	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
3044	div \$s2,\$s3	0x0253001a	0x00000032	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
3048	mflo \$s4	0x0000a012	0x00000032	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
304c	move \$a1,\$s4	0x00142821	0x00000005	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
3050	sw \$a1, 0(\$0)	0xac050000	0x00000005	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
3054	Jal factorial	0x0c000c19	0x00000005	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
3058	add \$s0,\$v0, \$0	0x00408020	0x00000005	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
305c	sw \$s0,16(\$0)	0xac100010	0x00000005	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
3060	J exit	0x08000c28	0x00000005	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
3064	addi \$sp, \$sp, -8	0x23bdfff8	0x00000005	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
3068	sw \$ra, 4(\$sp)	0xafbf0004	0x00000005	0x00002ff4	0x00000000	0x00000000	0x00000000	0x00000000
306c	sw \$a1, 0(\$sp)	0xafa50000	0x00000005	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
3070	slti \$t0,\$a1,2	0x28a80002	0x00000005	0x00002ffc	0x00000000	0x00000000	0x00000000	0x00000000
3074	beq \$t0,\$zero,L1	0x10000003	0x00000005	0x00002ffc	0x00000000	0x00000000	0x00000005	0x00000000
3078	addi \$v0,\$zero,1	0x20020001	0x00000005	0x00002ffc	0x00000000	0x00000078	0x00000005	0x00000078
307c	addi \$sp,\$sp,8	0x23bd0008	0x00000005	0x00002ffc	0x00003058	0x00000078	0x00000005	0x00000078
3080	jr \$ra	0x03e00008	0x00000005	0x00002ffc	0x00003058	0x00000078	0x00000005	0x00000078
3084	L1: addi \$a1,\$a1,-1	0x20a5ffff	0x00000005	0x00002ffc	0x00003058	0x00000078	0x00000005	0x00000078
3088	jal factorial	0x0c000019	0x00000005	0x00002ffc	0x00003058	0x00000078	0x00000005	0x00000078
308c	lw \$a1, 0(\$sp)	0x8fa5000	0x00000005	0x00002ffc	0x00003058	0x00000078	0x00000005	0x00000078
3090	lw \$ra, 4(\$sp)	0x8fbf0004	0x00000005	0x00002ffc	0x00003058	0x00000078	0x00000005	0x00000078
3094	addi \$sp, \$sp, 8	0x23bd0008	0x00000005	0x00002ffc	0x00003058	0x00000078	0x00000005	0x00000078
3098	mul \$v0,\$a1,\$v0	0x70a21002	0x00000005	0x00002ffc	0x00003058	0x00000078	0x00000005	0x00000078
309c	jr \$ra	0x03e00008	0x00000005	0x00002ffc	0x00003058	0x00000078	0x00000005	0x00000078

## Snapshot before execution:

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00003000	0x20040100	addi \$4,\$0,0x00000100	9: addi \$a0, \$0, 0x100 # array base address = 0x100
	0x00003004	0x20050000	addi \$5,\$0,0x00000000	11: addi \$a1, \$0, 0 # i = 0
	0x00003008	0x20080003	addi \$8,\$0,0x00000003	13: addi \$t0, \$0, 3
	0x0000300c	0x20090032	addi \$9,\$0,0x00000032	15: addi \$t1, \$0, 50 # \$t1 = 50
	0x00003010	0x00a9502a	slt \$t0,\$5,\$9	19: slt \$t2, \$a1, \$t1 # i < 50?
	0x00003014	0x11400007	beq \$t0,\$0,0x00000007	21: beq \$t2, \$0, Exit_Loop # if not then exit loop
	0x00003018	0x00055000	sll \$t0,\$5,0x00000002	23: sll \$t2, \$a1, 2 # \$t2 = i * 4 (byte offset)
	0x0000301c	0x01445020	add \$t0,\$t0,\$4	25: add \$t2, \$t2, \$a0 # address of array[i]
	0x00003020	0x00a80018	mult \$5,\$8	27: mult \$a1, \$t0
	0x00003024	0x00005012	mflo \$t1	29: mflo \$t3 # \$t3 = i * 3
	0x00003028	0xad4b0000	sw \$t1,0x00000000(\$t0)	31: sw \$t3, 0(\$t2) # save array[i]
	0x0000302c	0x20a50001	addi \$5,\$5,0x00000001	33: addi \$a1, \$a1, 1 # i = i + 1

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000100	0x00000000	0x00000003	0x00000006	0x00000009	0x0000000c	0x0000000f	0x00000012	0x00000015

Registers Coproc 1 Coproc 0

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000100
\$a1	5	0x00000032
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000003
\$t1	9	0x00000032
\$t2	10	0x00000000
\$t3	11	0x00000093
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0x00000000
\$s2	18	0x00000000
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x00001800
\$sp	29	0x0002fffc
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00003034
hi		0x00000000
lo		0x00000093

Mars Messages Run I/O

— program is finished running (dropped off bottom) —

Clear

## Snapshot after execution:

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Text Segment

Bkpt	Address	Code	Basic	Source
	0x00003070	0x20a80002	slt \$t0,\$5,\$9	79: slt \$t0,\$a1,2
	0x00003074	0x11000003	beq \$t0,\$0,0x00000003	81: beq \$t0,\$zero,L1
	0x00003078	0x20020001	addi \$2,\$0,0x00000001	83: addi \$v0,\$zero,1
	0x0000307c	0x23bd0000	addi \$29,\$29,0x0000...	85: addi \$sp,\$sp,8
	0x00003080	0x03e00008	jr \$31	87: jr \$ra
	0x00003084	0x20a5ffff	addi \$5,\$5,0xfffffff	89: L1: addi \$a1,\$a1,-1
	0x00003088	0x0c000c19	jal 0x00003064	91: jal factorial
	0x0000308c	0x8fa50000	lw \$5,0x00000000(\$29)	93: lw \$a1, 0(\$sp)
	0x00003090	0x8fbf0004	lw \$31,0x00000004(\$29)	95: lw \$ra, 4(\$sp)
	0x00003094	0x23bd0008	addi \$29,\$29,0x0000...	97: addi \$sp,\$sp, 8
	0x00003098	0x70a21002	mul \$2,\$5,\$2	99: mul \$v0,\$a1,\$v0
	0x0000309c	0x03e00008	jr \$31	101: jr \$ra

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x00000000	0x00000005	0x00000000	0x00000000	0x00000000	0x00000078	0x00000000	0x00000000	0x00000000
0x00000020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x000000e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x00000100	0x00000000	0x00000003	0x00000006	0x00000009	0x0000000c	0x0000000f	0x00000012	0x00000015

Registers Coproc 1 Coproc 0

Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000078
\$v1	3	0x00000000
\$a0	4	0x00000100
\$a1	5	0x00000005
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000001
\$t1	9	0x00000032
\$t2	10	0x00000000
\$t3	11	0x00000093
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000078
\$s1	17	0x0000004b
\$s2	18	0x000000a5
\$s3	19	0x0000001e
\$s4	20	0x00000005
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x00001800
\$sp	29	0x0002fffc
\$fp	30	0x00000000
\$ra	31	0x00003058
pc		0x000030a0
hi		0x00000000
lo		0x00000078

Mars Messages Run I/O

— program is finished running (dropped off bottom) —

Clear

— program is finished running (dropped off bottom) —

— program is finished running (dropped off bottom) —

**Conclusion:**

In conclusion, this assignment has provided a valuable opportunity to delve into the world of MIPS assembly programming and gain a deeper understanding of various fundamental concepts within it. Building a 50-entry array at base address 0x100 and performing arithmetic calculations on it has allowed us to explore the implementation of arrays, stacks, procedures, and recursive procedures in MIPS.