CS7381 Project 3
MIPS Assembly Code Programming Using MARS Tool – Additional Coding Assignment

Name: Bingying Liang
ID: 48999397
Distance

February 19 2023

For this exercise, you will write another MIPS assembly code program. This time, you will create and run MIPS code for the following high-level language (HLL) code:

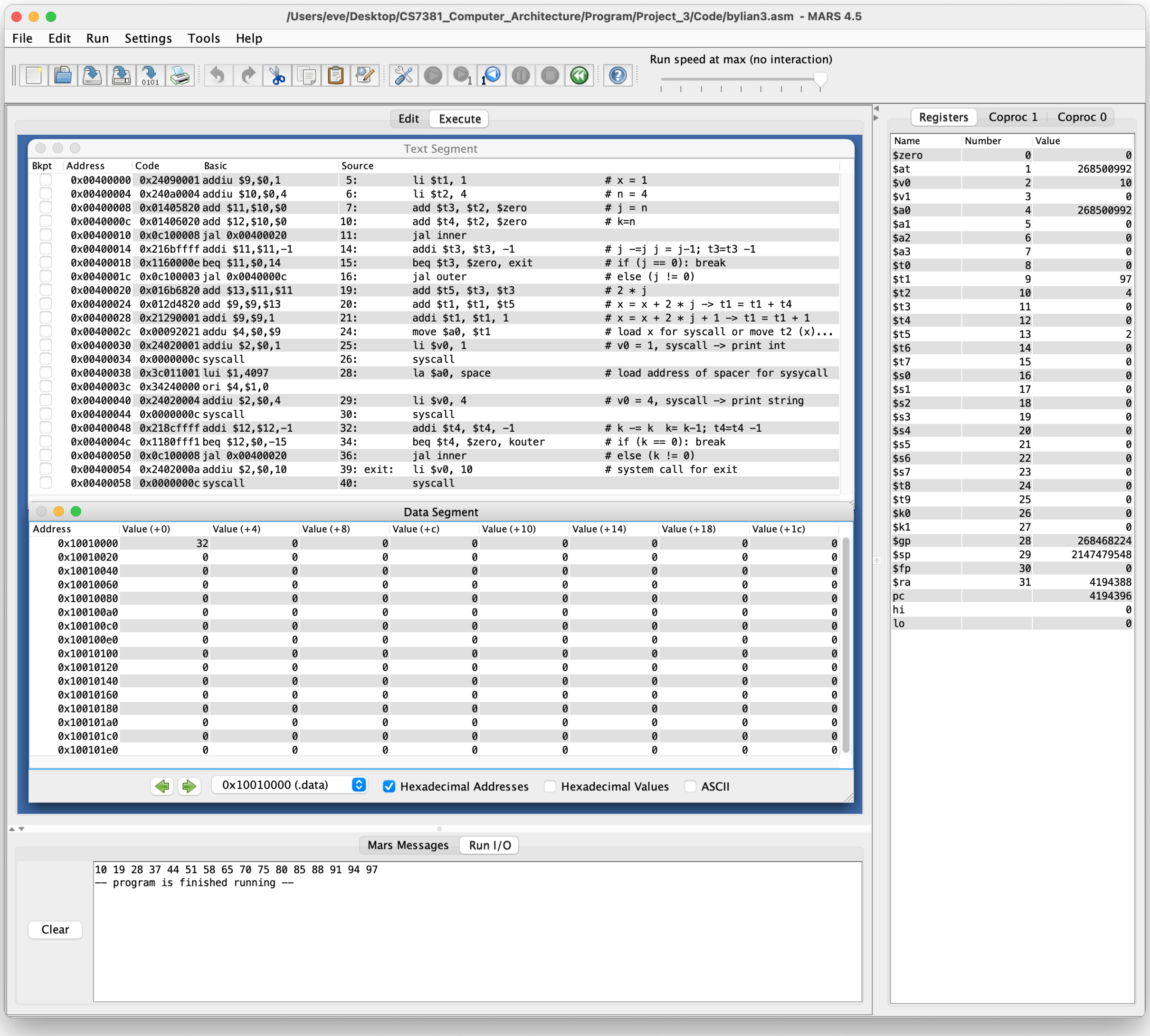
C++ Version

```
int j, k, n, x;
x = 1;
n = 4;
j = n;
// outer loop with j
do
{
    k = n;
    // inner loop with k
    do
    {
        x = x + 2 * j + 1;
        cout << x << " ";
        k--;
    } while (k > 0);
    j--;
} while (j > 0);
```

Java Version

```
x = 1
n = 4
j = n
# outer loop with j
while True:
    k = n
    # inner loop with k
    while True:
        x = x + 2 * j + 1
        print(x, end=" ")
        k -= 1
        if (k == 0):
            break
    j -= 1
    if (j == 0):
        break
```

Screen shot of the MARS console


/Users/eve/Desktop/CS7381_Computer_Architecture/Program/Project_3/Code/bylian3.asm - MARS 4.5
File
Edit
Run
Settings
Tools
Help
Run speed at max (no interaction)
Edit
Execute
Text Segment
Data Segment
Registers
Coproc 1
Coproc 0
Mars Messages
Run I/O
Clear

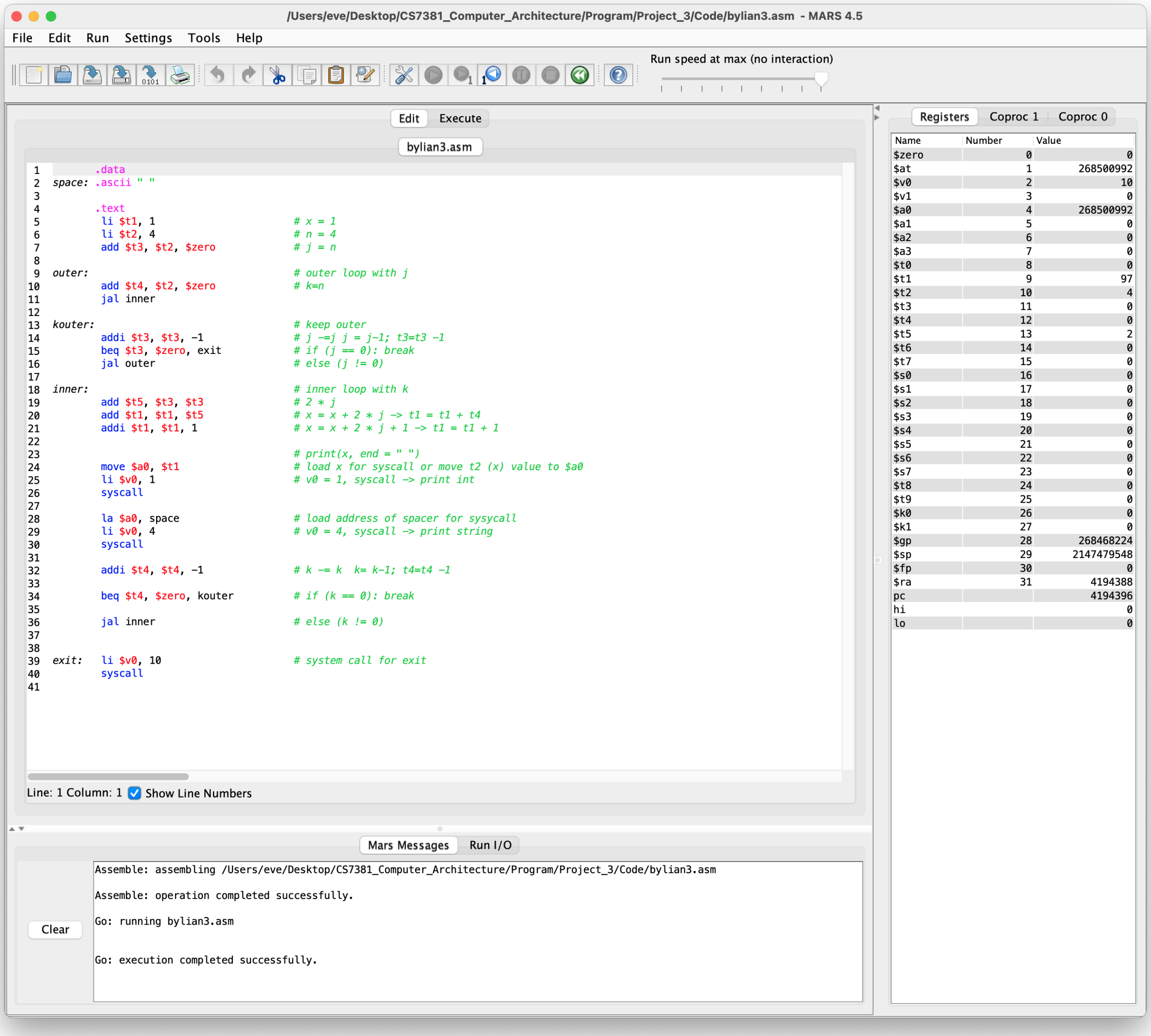
Bkpt	Address	Code	Basic	Source		
	0x00400000	0x24090001	addiu $9,$0,1	5:	li $t1, 1	# x = 1
	0x00400004	0x240a0004	addiu $10,$0,4	6:	li $t2, 4	# n = 4
	0x00400008	0x01405820	add $11,$10,$0	7:	add $t3, $t2, $zero	# j = n
	0x0040000c	0x01406020	add $12,$10,$0	10:	add $t4, $t2, $zero	# k=n
	0x00400010	0x0c100008	jal 0x00400020	11:	jal inner	
	0x00400014	0x216bffff	addi $11,$11,-1	14:	addi $t3, $t3, -1	# j -=j j = j-1; t3=t3 -1
	0x00400018	0x1160000e	beq $11,$0,14	15:	beq $t3, $zero, exit	# if (j == 0): break
	0x0040001c	0x0c100003	jal 0x0040000c	16:	jal outer	# else (j != 0)
	0x00400020	0x016b6820	add $13,$11,$11	19:	add $t5, $t3, $t3	# 2 * j
	0x00400024	0x012d4820	add $9,$9,$13	20:	add $t1, $t1, $t5	# x = x + 2 * j -> t1 = t1 + t4
	0x00400028	0x21290001	addi $9,$9,1	21:	addi $t1, $t1, 1	# x = x + 2 * j + 1 -> t1 = t1 + 1
	0x0040002c	0x00092021	addu $4,$0,$9	24:	move $a0, $t1	# load x for syscall or move t2 (x)...
	0x00400030	0x24020001	addiu $2,$0,1	25:	li $v0, 1	# v0 = 1, syscall -> print int
	0x00400034	0x0000000c	syscall	26:	syscall	
	0x00400038	0x3c011001	lui $1,4097	28:	la $a0, space	# load address of spacer for sysycall
	0x0040003c	0x34240000	ori $4,$1,0			
	0x00400040	0x24020004	addiu $2,$0,4	29:	li $v0, 4	# v0 = 4, syscall -> print string
	0x00400044	0x0000000c	syscall	30:	syscall	
	0x00400048	0x218cffff	addi $12,$12,-1	32:	addi $t4, $t4, -1	# k -= k k= k-1; t4=t4 -1
	0x0040004c	0x1180fff1	beq $12,$0,-15	34:	beq $t4, $zero, kouter	# if (k == 0): break
	0x00400050	0x0c100008	jal 0x00400020	36:	jal inner	# else (k != 0)
	0x00400054	0x2402000a	addiu $2,$0,10	39: exit:	li $v0, 10	# system call for exit
	0x00400058	0x0000000c	syscall	40:	syscall	

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	32	0	0	0	0	0	0	0
0x10010020	0	0	0	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0
0x10010140	0	0	0	0	0	0	0	0
0x10010160	0	0	0	0	0	0	0	0
0x10010180	0	0	0	0	0	0	0	0
0x100101a0	0	0	0	0	0	0	0	0
0x100101c0	0	0	0	0	0	0	0	0
0x100101e0	0	0	0	0	0	0	0	0

0x10010000 (.data) ✓ Hexadecimal Addresses ☐ Hexadecimal Values ☐ ASCII

Name	Number	Value
$zero	0	0
$at	1	268500992
$v0	2	10
$v1	3	0
$a0	4	268500992
$a1	5	0
$a2	6	0
$a3	7	0
$t0	8	0
$t1	9	97
$t2	10	4
$t3	11	0
$t4	12	0
$t5	13	2
$t6	14	0
$t7	15	0
$s0	16	0
$s1	17	0
$s2	18	0
$s3	19	0
$s4	20	0
$s5	21	0
$s6	22	0
$s7	23	0
$t8	24	0
$t9	25	0
$k0	26	0
$k1	27	0
$gp	28	268468224
$sp	29	2147479548
$fp	30	0
$ra	31	4194388
pc		4194396
hi		0
lo		0

```
10 19 28 37 44 51 58 65 70 75 80 85 88 91 94 97
-- program is finished running --
```


/Users/eve/Desktop/CS7381_Computer_Architecture/Program/Project_3/Code/bylian3.asm - MARS 4.5
File Edit Run Settings Tools Help
Run speed at max (no interaction)
Edit Execute
bylian3.asm
Registers Coproc 1 Coproc 0
Mars Messages Run I/O
Clear

```
        .data
space: .ascii " "

        .text
         li $t1, 1                     # x = 1
         li $t2, 4                     # n = 4
         add $t3, $t2, $zero           # j = n

outer:                                 # outer loop with j
         add $t4, $t2, $zero           # k=n
         jal inner

kouter:                                # keep outer
         addi $t3, $t3, -1             # j -=j j = j-1; t3=t3 -1
         beq $t3, $zero, exit          # if (j == 0): break
         jal outer                     # else (j != 0)

inner:                                 # inner loop with k
         add $t5, $t3, $t3             # 2 * j
         add $t1, $t1, $t5             # x = x + 2 * j -> t1 = t1 + t4
         addi $t1, $t1, 1              # x = x + 2 * j + 1 -> t1 = t1 + 1

                                       # print(x, end = " ")
         move $a0, $t1                 # load x for syscall or move t2 (x) value to $a0
         li $v0, 1                     # v0 = 1, syscall -> print int
         syscall

         la $a0, space                 # load address of spacer for sysycall
         li $v0, 4                     # v0 = 4, syscall -> print string
         syscall

         addi $t4, $t4, -1             # k -= k  k= k-1; t4=t4 -1

         beq $t4, $zero, kouter        # if (k == 0): break

         jal inner                     # else (k != 0)


exit:    li $v0, 10                    # system call for exit
         syscall

```

Line: 1 Column: 1 ✓ Show Line Numbers

Name	Number	Value
$zero	0	0
$at	1	268500992
$v0	2	10
$v1	3	0
$a0	4	268500992
$a1	5	0
$a2	6	0
$a3	7	0
$t0	8	0
$t1	9	97
$t2	10	4
$t3	11	0
$t4	12	0
$t5	13	2
$t6	14	0
$t7	15	0
$s0	16	0
$s1	17	0
$s2	18	0
$s3	19	0
$s4	20	0
$s5	21	0
$s6	22	0
$s7	23	0
$t8	24	0
$t9	25	0
$k0	26	0
$k1	27	0
$gp	28	268468224
$sp	29	2147479548
$fp	30	0
$ra	31	4194388
pc		4194396
hi		0
lo		0

```
Assemble: assembling /Users/eve/Desktop/CS7381_Computer_Architecture/Program/Project_3/Code/bylian3.asm

Assemble: operation completed successfully.

Go: running bylian3.asm


Go: execution completed successfully.
```