

Laboratory 3

Load/ Store , Jump & Branch instructions

Đỗ Hải Dương - 20194528

Assignment 1

i = 4528 ; j = 1

Assignment1.asm

```
1 start:
2     addi    $s1, $zero, 4528      # i = 4528
3     addi    $s2, $zero, 1        # j = 1
4     slt     $t0, $s2, $s1        # j < i
5     bne     $t0, $zero, else      # branch to else if j < i
6     addi    $t1, $t1, 1          # then part: x = x+1
7     addi    $t3, $zero, 1        # z = 1
8     j       endif               # skip "else" part
9 else:     addi    $t2, $t2, -1    # begin else part: y = y-1
10    add     $t3, $t3, $t2        # z = 2 * z
11 endif:
```

Text Segment

Address	Code	Basic	Source
0x00400000	addi	\$t1, \$zero, 4528	# i = 4528
0x00400004	addi	\$t2, \$zero, 1	# j = 1
0x00400008	slt	\$t0, \$s2, \$s1	# j < i
0x0040000c	bne	\$t0, \$zero, else	# branch to else if j < i
0x00400010	addi	\$t1, \$t1, 1	# then part: x = x+1
0x00400014	addi	\$t3, \$zero, 1	# z = 1
0x00400018	j	endif	# skip "else" part
0x0040001c	addi	\$t2, \$t2, -1	# begin else part: y = y-1
0x00400020	add	\$t3, \$t3, \$t2	# z = 2 * z

Data Segment

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x10010000	0	0	0	0	0	0	0	0
0x10010004	0	0	0	0	0	0	0	0
0x10010008	0	0	0	0	0	0	0	0
0x1001000c	0	0	0	0	0	0	0	0
0x10010010	0	0	0	0	0	0	0	0
0x10010014	0	0	0	0	0	0	0	0
0x10010018	0	0	0	0	0	0	0	0
0x1001001c	0	0	0	0	0	0	0	0
0x10010020	0	0	0	0	0	0	0	0
0x10010024	0	0	0	0	0	0	0	0
0x10010028	0	0	0	0	0	0	0	0
0x1001002c	0	0	0	0	0	0	0	0
0x10010030	0	0	0	0	0	0	0	0
0x10010034	0	0	0	0	0	0	0	0
0x10010038	0	0	0	0	0	0	0	0
0x1001003c	0	0	0	0	0	0	0	0
0x10010040	0	0	0	0	0	0	0	0
0x10010044	0	0	0	0	0	0	0	0
0x10010048	0	0	0	0	0	0	0	0
0x1001004c	0	0	0	0	0	0	0	0
0x10010050	0	0	0	0	0	0	0	0
0x10010054	0	0	0	0	0	0	0	0
0x10010058	0	0	0	0	0	0	0	0
0x1001005c	0	0	0	0	0	0	0	0
0x10010060	0	0	0	0	0	0	0	0
0x10010064	0	0	0	0	0	0	0	0
0x10010068	0	0	0	0	0	0	0	0
0x1001006c	0	0	0	0	0	0	0	0
0x10010070	0	0	0	0	0	0	0	0
0x10010074	0	0	0	0	0	0	0	0
0x10010078	0	0	0	0	0	0	0	0
0x1001007c	0	0	0	0	0	0	0	0
0x10010080	0	0	0	0	0	0	0	0
0x10010084	0	0	0	0	0	0	0	0
0x10010088	0	0	0	0	0	0	0	0
0x1001008c	0	0	0	0	0	0	0	0
0x10010090	0	0	0	0	0	0	0	0
0x10010094	0	0	0	0	0	0	0	0
0x10010098	0	0	0	0	0	0	0	0
0x1001009c	0	0	0	0	0	0	0	0
0x100100a0	0	0	0	0	0	0	0	0
0x100100a4	0	0	0	0	0	0	0	0
0x100100a8	0	0	0	0	0	0	0	0
0x100100ac	0	0	0	0	0	0	0	0
0x100100b0	0	0	0	0	0	0	0	0
0x100100b4	0	0	0	0	0	0	0	0
0x100100b8	0	0	0	0	0	0	0	0
0x100100bc	0	0	0	0	0	0	0	0
0x100100c0	0	0	0	0	0	0	0	0
0x100100c4	0	0	0	0	0	0	0	0
0x100100c8	0	0	0	0	0	0	0	0
0x100100cc	0	0	0	0	0	0	0	0
0x100100d0	0	0	0	0	0	0	0	0
0x100100d4	0	0	0	0	0	0	0	0
0x100100d8	0	0	0	0	0	0	0	0
0x100100dc	0	0	0	0	0	0	0	0
0x100100e0	0	0	0	0	0	0	0	0
0x100100e4	0	0	0	0	0	0	0	0
0x100100e8	0	0	0	0	0	0	0	0
0x100100ec	0	0	0	0	0	0	0	0
0x100100f0	0	0	0	0	0	0	0	0
0x100100f4	0	0	0	0	0	0	0	0
0x100100f8	0	0	0	0	0	0	0	0
0x100100fc	0	0	0	0	0	0	0	0
0x10010100	0	0	0	0	0	0	0	0
0x10010104	0	0	0	0	0	0	0	0
0x10010108	0	0	0	0	0	0	0	0
0x1001010c	0	0	0	0	0	0	0	0
0x10010110	0	0	0	0	0	0	0	0
0x10010114	0	0	0	0	0	0	0	0
0x10010118	0	0	0	0	0	0	0	0
0x1001011c	0	0	0	0	0	0	0	0
0x10010120	0	0	0	0	0	0	0	0
0x10010124	0	0	0	0	0	0	0	0
0x10010128	0	0	0	0	0	0	0	0
0x1001012c	0	0	0	0	0	0	0	0
0x10010130	0	0	0	0	0	0	0	0
0x10010134	0	0	0	0	0	0	0	0
0x10010138	0	0	0	0	0	0	0	0
0x1001013c	0	0	0	0	0	0	0	0
0x10010140	0	0	0	0	0	0	0	0
0x10010144	0	0	0	0	0	0	0	0
0x10010148	0	0	0	0	0	0	0	0
0x1001014c	0	0	0	0	0	0	0	0
0x10010150	0	0	0	0	0	0	0	0
0x10010154	0	0	0	0	0	0	0	0
0x10010158	0	0	0	0	0	0	0	0
0x1001015c	0	0	0	0	0	0	0	0
0x10010160	0	0	0	0	0	0	0	0
0x10010164	0	0	0	0	0	0	0	0
0x10010168	0	0	0	0	0	0	0	0
0x1001016c	0	0	0	0	0	0	0	0
0x10010170	0	0	0	0	0	0	0	0
0x10010174	0	0	0	0	0	0	0	0
0x10010178	0	0	0	0	0	0	0	0
0x1001017c	0	0	0	0	0	0	0	0
0x10010180	0	0	0	0	0	0	0	0
0x10010184	0	0	0	0	0	0	0	0
0x10010188	0	0	0	0	0	0	0	0
0x1001018c	0	0	0	0	0	0	0	0
0x10010190	0	0	0	0	0	0	0	0
0x10010194	0	0	0	0	0	0	0	0
0x10010198	0	0	0	0	0	0	0	0
0x1001019c	0	0	0	0	0	0	0	0
0x100101a0	0	0	0	0	0	0	0	0
0x100101a4	0	0	0	0	0	0	0	0
0x100101a8	0	0	0	0	0	0	0	0
0x100101ac	0	0	0	0	0	0	0	0
0x100101b0	0	0	0	0	0	0	0	0
0x100101b4	0	0	0	0	0	0	0	0
0x100101b8	0	0	0	0	0	0	0	0
0x100101bc	0	0	0	0	0	0	0	0
0x100101c0	0	0	0	0	0	0	0	0
0x100101c4	0	0	0	0	0	0	0	0
0x100101c8	0	0	0	0	0	0	0	0
0x100101cc	0	0	0	0	0	0	0	0
0x100101d0	0	0	0	0	0	0	0	0
0x100101d4	0	0	0	0	0	0	0	0
0x100101d8	0	0	0	0	0	0	0	0
0x100101dc	0	0	0	0	0	0	0	0
0x100101e0	0	0	0	0	0	0	0	0
0x100101e4	0	0	0	0	0	0	0	0
0x100101e8	0	0	0	0	0	0	0	0
0x100101ec	0	0	0	0	0	0	0	0
0x100101f0	0	0	0	0	0	0	0	0
0x100101f4	0	0	0	0	0	0	0	0
0x100101f8	0	0	0	0	0	0	0	0
0x100101fc	0	0	0	0	0	0	0	0
0x10010200	0	0	0	0	0	0	0	0
0x10010204	0	0	0	0	0	0	0	0
0x10010208	0	0	0	0	0	0	0	0
0x1001020c	0	0	0	0	0	0	0	0
0x10010210	0	0	0	0	0	0	0	0
0x10010214	0	0	0	0	0	0	0	0
0x10010218	0	0	0	0	0	0	0	0
0x1001021c	0	0	0	0	0	0	0	0
0x10010220	0	0	0	0	0	0	0	0
0x10010224	0	0	0	0	0	0	0	0
0x10010228	0	0	0	0	0	0	0	0
0x1001022c	0	0	0	0	0	0	0	0
0x10010230	0	0	0	0	0	0	0	0
0x10010234	0	0	0	0	0	0	0	0
0x10010238	0	0	0	0	0	0	0	0
0x1001023c	0	0	0	0	0	0	0	0
0x10010240	0	0	0	0	0	0	0	0
0x10010244	0	0	0	0	0	0	0	0
0x10010248	0	0	0	0	0	0	0	0
0x1001024c	0	0	0	0	0	0	0	0
0x10010250	0	0	0	0	0	0	0	0
0x10010254	0	0	0	0	0	0	0	0
0x10010258	0	0	0	0	0	0	0	0
0x1001025c	0	0	0	0	0	0	0	0
0x10010260	0	0	0	0	0	0	0	0
0x10010264	0	0	0	0	0	0	0	0
0x10010268	0	0	0	0	0	0	0	0
0x1001026c	0	0	0	0	0	0	0	0
0x10010270	0	0	0	0	0	0	0	0
0x10010274	0	0	0	0	0	0	0	0
0x10010278	0	0	0	0	0	0	0	0
0x1001027c	0	0	0	0	0	0	0	0
0x10010280	0	0	0	0	0	0	0	0
0x10010284	0	0	0	0	0	0	0	0
0x10010288	0	0	0	0	0	0	0	0
0x1001028c	0	0	0	0	0	0	0	0
0x10010290	0	0	0	0	0	0	0	0
0x10010294	0	0	0	0	0	0	0	0
0x10010298	0	0	0	0	0	0	0	0
0x1001029c	0	0	0	0	0</			

- Nếu \$t0 != 0 thì Step 8: giảm \$t2 đi 1

Step 9: nhân đôi \$t3

Assignment 2

```
1  .data
2  A: .space 16 # Khai bao mang A co 4 phan tu
3
4  .text
5      li      $s1, -1      # gan i = -1
6      la      $s2, A        # $2 = &A
7
8      li      $t8, 2019
9      sw      $t8, 0($s2)   # A[0] = 2019
10     li      $t8, 4528
11     sw      $t8, 4($s2)   # A[1] = 4528
12     li      $t8, 3
13     sw      $t8, 8($s2)   # A[2] = 3
14     li      $t8, 4
15     sw      $t8, 12($s2)  # A[3] = 4
16
17     li      $s3, 4        # n = 4
18     li      $s4, 1        # step = 1
19     li      $s5, 0        # sum = 0
20
21 loop: add     $s1, $s1, $s4  #i = i+step
22       add     $t1, $s1, $s1  #t1 = 2 * s1
23       add     $t1, $t1, $t1  #t1 = 4 * s1
24       add     $t1, $t1, $s2  #t1 store the address of A[i]
25       lw      $t0, 0($t1)    #load value of A[i] in $t0
26       add     $s5, $s5, $t0  #sum = sum + A[i]
27       bne     $s1, $s3, loop #if i != n, goto loop
28
```

```

1  .data
2  test: .word 0
3  .text
4      la    $s0, test           #load the address of test variable
5      lw    $s1, 0($s0)        #load the value of test to register $t1
6      li    $t0, 0             #load value for test case
7      li    $t1, 1
8      li    $t2, 2
9      li    $s2, 14            # gán $s2 = 14
10     li    $s3, 5             # gán $s3 = 5
11     beq    $s1, $t0, case_0
12     beq    $s1, $t1, case_1
13     beq    $s1, $t2, case_2
14     j      default
15
16 case_0: addi    $s2, $s2, 1     # a = a + 1
17         j      continue
18 case_1: sub     $s2, $s2, $t1   # a = a - 1
19         j      continue
20 case_2: add     $s3, $s3, $s3   # b = 2 * b
21         j      continue
22 default:
23 continue:
24

```

Text Segment						Name		
Offset	Address	Code	Basic	4:	1a	asm, text	Number	Value
0	0x04000000	0x00100000	lui \$t1, 4097	4:	1a	asm, text	0	0
4	0x04000004	0x04300000	ori \$t1, \$t1, 0	5:	1w	\$t1, 0(\$t0) #load the address of test variable	1	268500992
8	0x04000008	0x04100000	lw \$t2, 0(\$t1)	6:	1l	\$t2, 0	2	0
12	0x0400000c	0x04000000	addiu \$t3, \$t3, 0	7:	1l	\$t3, 0	3	0
16	0x04000010	0x04000000	addiu \$t3, \$t3, 1	8:	1l	\$t3, 1	4	0
20	0x04000014	0x04000000	addiu \$t3, \$t3, 2	9:	1l	\$t3, 2	5	0
24	0x04000018	0x04100000	addiu \$t3, \$t3, 14	10:	1l	\$t3, 14	6	0
28	0x0400001c	0x04100000	addiu \$t3, \$t3, 5	11:	1l	\$t3, 5	7	0
32	0x04000020	0x04200000	beq \$t1, \$t2, case_0	12:	1l	\$t1, \$t2, case_0	8	0
36	0x04000024	0x04200000	beq \$t1, \$t2, case_1	13:	1l	\$t1, \$t2, case_1	9	1
40	0x04000028	0x04200000	beq \$t1, \$t2, case_2	14:	1l	\$t1, \$t2, case_2	10	2
44	0x0400002c	0x04100010	addi \$t3, \$t3, 1	15:	1l	\$t3, \$t3, 1	11	0
48	0x04000030	0x04100010	addi \$t3, \$t3, 1	16:	1l	\$t3, \$t3, 1	12	0
52	0x04000034	0x04100010	addi \$t3, \$t3, 1	17:	1l	\$t3, \$t3, 1	13	0
56	0x04000038	0x04100010	addi \$t3, \$t3, 1	18:	1l	\$t3, \$t3, 1	14	0
60	0x0400003c	0x04100010	addi \$t3, \$t3, 1	19:	1l	\$t3, \$t3, 1	15	0
64	0x04000040	0x04100010	addi \$t3, \$t3, 1	20:	1l	\$t3, \$t3, 1	16	0
68	0x04000044	0x04100010	addi \$t3, \$t3, 1	21:	1l	\$t3, \$t3, 1	17	0
72	0x04000048	0x04100010	addi \$t3, \$t3, 1	22:	1l	\$t3, \$t3, 1	18	0
76	0x0400004c	0x04100010	addi \$t3, \$t3, 1	23:	1l	\$t3, \$t3, 1	19	0
80	0x04000050	0x04100010	addi \$t3, \$t3, 1	24:	1l	\$t3, \$t3, 1	20	0
84	0x04000054	0x04100010	addi \$t3, \$t3, 1	25:	1l	\$t3, \$t3, 1	21	0
88	0x04000058	0x04100010	addi \$t3, \$t3, 1	26:	1l	\$t3, \$t3, 1	22	0
92	0x0400005c	0x04100010	addi \$t3, \$t3, 1	27:	1l	\$t3, \$t3, 1	23	0
96	0x04000060	0x04100010	addi \$t3, \$t3, 1	28:	1l	\$t3, \$t3, 1	24	0
100	0x04000064	0x04100010	addi \$t3, \$t3, 1	29:	1l	\$t3, \$t3, 1	25	0
104	0x04000068	0x04100010	addi \$t3, \$t3, 1	30:	1l	\$t3, \$t3, 1	26	0
108	0x0400006c	0x04100010	addi \$t3, \$t3, 1	31:	1l	\$t3, \$t3, 1	27	0
112	0x04000070	0x04100010	addi \$t3, \$t3, 1	32:	1l	\$t3, \$t3, 1	28	0
116	0x04000074	0x04100010	addi \$t3, \$t3, 1	33:	1l	\$t3, \$t3, 1	29	0
120	0x04000078	0x04100010	addi \$t3, \$t3, 1	34:	1l	\$t3, \$t3, 1	30	0
124	0x0400007c	0x04100010	addi \$t3, \$t3, 1	35:	1l	\$t3, \$t3, 1	31	0
128	0x04000080	0x04100010	addi \$t3, \$t3, 1	36:	1l	\$t3, \$t3, 1	32	0
132	0x04000084	0x04100010	addi \$t3, \$t3, 1	37:	1l	\$t3, \$t3, 1	33	0
136	0x04000088	0x04100010	addi \$t3, \$t3, 1	38:	1l	\$t3, \$t3, 1	34	0
140	0x0400008c	0x04100010	addi \$t3, \$t3, 1	39:	1l	\$t3, \$t3, 1	35	0
144	0x04000090	0x04100010	addi \$t3, \$t3, 1	40:	1l	\$t3, \$t3, 1	36	0
148	0x04000094	0x04100010	addi \$t3, \$t3, 1	41:	1l	\$t3, \$t3, 1	37	0
152	0x04000098	0x04100010	addi \$t3, \$t3, 1	42:	1l	\$t3, \$t3, 1	38	0
156	0x0400009c	0x04100010	addi \$t3, \$t3, 1	43:	1l	\$t3, \$t3, 1	39	0
160	0x040000a0	0x04100010	addi \$t3, \$t3, 1	44:	1l	\$t3, \$t3, 1	40	0
164	0x040000a4	0x04100010	addi \$t3, \$t3, 1	45:	1l	\$t3, \$t3, 1	41	0
168	0x040000a8	0x04100010	addi \$t3, \$t3, 1	46:	1l	\$t3, \$t3, 1	42	0
172	0x040000ac	0x04100010	addi \$t3, \$t3, 1	47:	1l	\$t3, \$t3, 1	43	0
176	0x040000b0	0x04100010	addi \$t3, \$t3, 1	48:	1l	\$t3, \$t3, 1	44	0
180	0x040000b4	0x04100010	addi \$t3, \$t3, 1	49:	1l	\$t3, \$t3, 1	45	0
184	0x040000b8	0x04100010	addi \$t3, \$t3, 1	50:	1l	\$t3, \$t3, 1	46	0
188	0x040000bc	0x04100010	addi \$t3, \$t3, 1	51:	1l	\$t3, \$t3, 1	47	0
192	0x040000c0	0x04100010	addi \$t3, \$t3, 1	52:	1l	\$t3, \$t3, 1	48	0
196	0x040000c4	0x04100010	addi \$t3, \$t3, 1	53:	1l	\$t3, \$t3, 1	49	0
200	0x040000c8	0x04100010	addi \$t3, \$t3, 1	54:	1l	\$t3, \$t3, 1	50	0
204	0x040000cc	0x04100010	addi \$t3, \$t3, 1	55:	1l	\$t3, \$t3, 1	51	0
208	0x040000d0	0x04100010	addi \$t3, \$t3, 1	56:	1l	\$t3, \$t3, 1	52	0
212	0x040000d4	0x04100010	addi \$t3, \$t3, 1	57:	1l	\$t3, \$t3, 1	53	0
216	0x040000d8	0x04100010	addi \$t3, \$t3, 1	58:	1l	\$t3, \$t3, 1	54	0
220	0x040000dc	0x04100010	addi \$t3, \$t3, 1	59:	1l	\$t3, \$t3, 1	55	0
224	0x040000e0	0x04100010	addi \$t3, \$t3, 1	60:	1l	\$t3, \$t3, 1	56	0
228	0x040000e4	0x04100010	addi \$t3, \$t3, 1	61:	1l	\$t3, \$t3, 1	57	0
232	0x040000e8	0x04100010	addi \$t3, \$t3, 1	62:	1l	\$t3, \$t3, 1	58	0
236	0x040000ec	0x04100010	addi \$t3, \$t3, 1	63:	1l	\$t3, \$t3, 1	59	0
240	0x040000f0	0x04100010	addi \$t3, \$t3, 1	64:	1l	\$t3, \$t3, 1	60	0
244	0x040000f4	0x04100010	addi \$t3, \$t3, 1	65:	1l	\$t3, \$t3, 1	61	0
248	0x040000f8	0x04100010	addi \$t3, \$t3, 1	66:	1l	\$t3, \$t3, 1	62	0
252	0x040000fc	0x04100010	addi \$t3, \$t3, 1	67:	1l	\$t3, \$t3, 1	63	0
256	0x04000100	0x04100010	addi \$t3, \$t3, 1	68:	1l	\$t3, \$t3, 1	64	0
260	0x04000104	0x04100010	addi \$t3, \$t3, 1	69:	1l	\$t3, \$t3, 1	65	0
264	0x04000108	0x04100010	addi \$t3, \$t3, 1	70:	1l	\$t3, \$t3, 1	66	0
268	0x0400010c	0x04100010	addi \$t3, \$t3, 1	71:	1l	\$t3, \$t3, 1	67	0
272	0x04000110	0x04100010	addi \$t3, \$t3, 1	72:	1l	\$t3, \$t3, 1	68	0
276	0x04000114	0x04100010	addi \$t3, \$t3, 1	73:	1l	\$t3, \$t3, 1	69	0
280	0x04000118	0x04100010	addi \$t3, \$t3, 1	74:	1l	\$t3, \$t3, 1	70	0
284	0x0400011c	0x04100010	addi \$t3, \$t3, 1	75:	1l	\$t3, \$t3, 1	71	0
288	0x04000120	0x04100010	addi \$t3, \$t3, 1	76:	1l	\$t3, \$t3, 1	72	0
292	0x04000124	0x04100010	addi \$t3, \$t3, 1	77:	1l	\$t3, \$t3, 1	73	0
296	0x04000128	0x04100010	addi \$t3, \$t3, 1	78:	1l	\$t3, \$t3, 1	74	0
300	0x0400012c	0x04100010	addi \$t3, \$t3, 1	79:	1l	\$t3, \$t3, 1	75	0
304	0x04000130	0x04100010	addi \$t3, \$t3, 1	80:	1l	\$t3, \$t3, 1	76	0
308	0x04000134	0x04100010	addi \$t3, \$t3, 1	81:	1l	\$t3, \$t3, 1	77	0
312	0x04000138	0x04100010	addi \$t3, \$t3, 1	82:	1l	\$t3, \$t3, 1	78	0
316	0x0400013c	0x04100010	addi \$t3, \$t3, 1	83:	1l	\$t3, \$t3, 1	79	0
320	0x04000140	0x04100010	addi \$t3, \$t3, 1	84:	1l	\$t3, \$t3, 1	80	0
324	0x04000144	0x04100010	addi \$t3, \$t3, 1	85:	1l	\$t3, \$t3, 1	81	0
328	0x04000148	0x04100010	addi \$t3, \$t3, 1	86:	1l	\$t3, \$t3, 1	82	0
332	0x0400014c	0x04100010	addi \$t3, \$t3, 1	87:	1l	\$t3, \$t3, 1	83	0
336	0x04000150	0x04100010	addi \$t3, \$t3, 1	88:	1l	\$t3, \$t3, 1	84	0
340	0x04000154	0x04100010	addi \$t3, \$t3, 1	89:	1l	\$t3, \$t3, 1	85	0
344	0x04000158	0x04100010	addi \$t3, \$t3, 1	90:	1l	\$t3, \$t3, 1	86	0
348	0x0400015c	0x04100010	addi \$t3, \$t3, 1	91:	1l	\$t3, \$t3, 1	87	0
352	0x04000160	0x04100010	addi \$t3, \$t3, 1	92:	1l	\$t3, \$t3, 1	88	0
356	0x04000164	0x04100010	addi \$t3, \$t3, 1	93:	1l	\$t3, \$t3, 1	89	0
360	0x04000168	0x04100010	addi \$t3, \$t3, 1	94:	1l	\$t3, \$t3, 1	90	0
364	0x0400016c	0x04100010	addi \$t3, \$t3, 1	95:	1l	\$t3, \$t3, 1	91	0
368	0x04000170	0x04100010	addi \$t3, \$t3, 1	96:	1l	\$t3, \$t3, 1	92	0
372	0x04000174	0x04100010	addi \$t3, \$t3, 1	97:	1l	\$t3, \$t3, 1	93	0
376	0x04000178	0x04100010	addi \$t3, \$t3, 1	98:	1l	\$t3, \$t3, 1	94	0
380	0x0400017c	0x04100010	addi \$t3, \$t3, 1	99:	1l	\$t3, \$t3, 1	95	0
384	0x04000180	0x04100010	addi \$t3, \$t3, 1	100:	1l	\$t3, \$t3, 1	96	0
388	0x04000184	0x04100010	addi \$t3, \$t3, 1	101:	1l	\$t3, \$t3, 1	97	0
392	0x04000188	0x04100010	addi \$t3, \$t3, 1	102:	1l	\$t3, \$t3, 1	98	0
396	0x0400018c	0x04100010	addi \$t3, \$t3, 1	103:	1l	\$t3, \$t3, 1	99	0
400	0x04000190	0x04100010	addi \$t3, \$t3, 1	104:	1l	\$t3, \$t3, 1	100	0
404	0x04000194	0x04100010	addi \$t3, \$t3, 1	105:	1l	\$t3, \$t3, 1	101	0
408	0x04000198	0x04100010	addi \$t3, \$t3, 1	106:	1l	\$t3, \$t3, 1	102	0
412	0x0400019c	0x04100010	addi \$t3, \$t3, 1	107:	1l	\$t3, \$t3, 1	103	0
416	0x040001a0	0x04100010	addi \$t3, \$t3, 1	108:	1l	\$t3, \$t3, 1	104	0
420	0x040001a4	0x04100010	addi \$t3, \$t3, 1	109:	1l	\$t3, \$t3, 1	105	0
424	0x040001a8	0x04100010	addi \$t3, \$t3, 1	110:	1l	\$t3, \$t3, 1	106	0
428	0x040001ac	0x04100010	addi \$t3, \$t3, 1	111:	1l	\$t3, \$t3, 1	107	0
432	0x040001b0	0x04100010	addi \$t3, \$t3, 1	112:	1l	\$t3, \$t3, 1	108	0
436	0x040001b4	0x04100010	addi \$t3, \$t3, 1	113:	1l	\$t3, \$t3, 1	109	0
440	0x040001b8	0x04100010	addi \$t3, \$t3, 1	114:	1l	\$t3, \$t3, 1	110	0
444	0x040001bc	0x04100010	addi \$t3, \$t3, 1	115:	1l	\$t3, \$t3, 1	111	0
448	0x040001c0	0x04100010	addi \$t3, \$t3, 1	116:	1l	\$t3, \$t3, 1	112	0
452	0x040001c4	0x04100010	addi \$t3, \$t3, 1	117:	1l	\$t3, \$t3, 1	113	0
456	0x040001c8	0x04100010	addi \$t3, \$t3, 1	118:	1l	\$t3, \$t3, 1	114	0
460	0x040001cc	0x04100010	addi \$t3, \$t3, 1	119:	1l	\$t3, \$t3, 1	115	0
464	0x040001d0	0x04100010	addi \$t3, \$t3, 1	120:	1l	\$t3, \$t3, 1	116	0
468	0x040001d4	0x04100010	addi \$t3, \$t3, 1	121:	1l	\$t3, \$t3, 1	117	0
472	0x040001d8	0x04100010	addi \$t3, \$t3, 1	122:	1l	\$t3, \$t3, 1	118	0
476	0x040001dc	0x04100010	addi \$t3, \$t3, 1	123:	1l	\$t3, \$t3,		

Assignment 4

a. $i < j$

1	start:		
2		addi \$s1, \$zero, 3	# i = 3
3		addi \$s2, \$zero, 4	# j = 4
4		slt \$t0, \$s1, \$s2	# i < j
5		bne \$t0, \$zero, else	# branch to else if j < i
6		addi \$t1, \$t1, 1	# then part: x = x + 1
7		addi \$t3, \$zero, 1	# z = 1
8		j endif	# skip "else" part
9	else:	addi \$t2, \$t2, -1	# begin else part: y = y + 1
10		add \$t3, \$t3, \$t3	# z = 2 * z
11	endif:		

b. $i \geq j$

1	start:		
2		addi \$s1, \$zero, 3	# i = 3
3		addi \$s2, \$zero, 2	# j = 2
4		slt \$t0, \$s1, \$s2	# i > j
5		beq \$t0, \$zero, else	# branch to else if i >= j
6		addi \$t1, \$t1, 1	# then part: x = x + 1
7		addi \$t3, \$zero, 1	# z = 1
8		j endif	# skip "else" part
9	else:	addi \$t2, \$t2, -1	# begin else part: y = y + 1
10		add \$t3, \$t3, \$t3	# z = 2 * z
11	endif:		
12			

c. $i + j \leq 0$

1	start:		
2		addi \$s1, \$zero, -3	# i = -3
3		addi \$s2, \$zero, 2	# j = 2
4		add \$s3, \$s1, \$s2	# s3 = i + j
5		slt \$t0, \$zero, \$s3	# i + j <= 0
6		bne \$t0, \$zero, else	# branch to else if i + j > 0
7		addi \$t1, \$t1, 1	# then part: x = x + 1
8		addi \$t3, \$zero, 1	# z = 1
9		j endif	# skip "else" part
10	else:	addi \$t2, \$t2, -1	# begin else part: y = y + 1
11		add \$t3, \$t3, \$t3	# z = 2 * z
12	endif:		
13			

⇒ Kết quả:

Registers	Coproc 1	Coproc 0
Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	0
\$t1	9	1
\$t2	10	0
\$t3	11	1
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	0
\$s1	17	-3
\$s2	18	2
\$s3	19	-1
\$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	0
pc		4194344
hi		0
lo		0

d. $i + j > m + n$

```

1 start:
2     addi    $s1, $zero, 3      # i = 3
3     addi    $s2, $zero, 2      # j = 2
4     add     $s3, $s1, $s2      # s3 = i + j
5     addi    $s4, $zero, 1      # i = 1
6     addi    $s5, $zero, 2      # j = 2
7     add     $s6, $s4, $s5      # s6 = m + n
8     slt     $t0, $zero, $s3
9     bne     $t0, $zero, else
10    addi    $t1, $t1, 1         # then part: x = x + 1
11    addi    $t3, $zero, 1      # z = 1
12    j       endif             # skip "else" part
13 else:    addi    $t2, $t2, -1  # begin else part: y = y + 1
14          add     $t3, $t3, $t3  # z = 2 * z
15 endif:

```

⇒ Kết quả:

Registers	Coproc 1	Coproc 0
Name	Number	Value
\$zero	0	0
\$at	1	0
\$v0	2	0
\$v1	3	0
\$a0	4	0
\$a1	5	0
\$a2	6	0
\$a3	7	0
\$t0	8	1
\$t1	9	0
\$t2	10	-1
\$t3	11	0
\$t4	12	0
\$t5	13	0
\$t6	14	0
\$t7	15	0
\$s0	16	0
\$s1	17	3
\$s2	18	2
\$s3	19	5
\$s4	20	1
\$s5	21	2
\$s6	22	3
\$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	0
pc		4194356
hi		0
lo		0

Assignment 5

a. $i < n$

```
1  .text
2
3  loop:  add    $s1, $s1, $s4    #i=i+step
4         add    $t1, $s1, $s1    #t1=2*s1
5         add    $t1, $t1, $t1    #t1=4*s1
6         add    $t1, $t1, $s2    #t1 store the address of A[i]
7         lw     $t0, 0($t1)      #load value of A[i] in $t0
8         add    $s5, $s5, $t0    #sum=sum+A[i]
9         slt    $t8, $s1, $s3    # if i < n
10        bne    $t8, $zero, loop    #if i != n, goto loop
```

+ Step 1: khởi tạo $i = 0$

+ Step 2: lưu địa chỉ của mảng

+ Step 3: lưu giá trị của n

+ Step 4: lưu bước nhảy của i

+ Step 5: khởi tạo $\text{sum} = A[0]$

+ Step 6: tăng i

+ step 7, 8: $t1 = 4s1$

+ Step 9: $t1$ lưu địa chỉ của $s2$

+ Step 10: load dữ liệu của $A[i]$ vào $t0$

+ Step 11: $s5 = s5 + t0$

+ Step 12: So sánh nếu $i < n$ thì $t8 = 1$, ngược lại bằng 0

+ Step 13: Nếu $t8 \neq 0$ thì quay lại vòng lặp

b. $i \leq n$


```

1  .text
2      li      $s1, 0          # gán i = 0
3
4  loop:  add    $s1, $s1, $s4   #i=i+step
5          add    $t1, $s1, $s1  #t1=2*s1
6          add    $t1, $t1, $t1  #t1=4*s1
7          add    $t1, $t1, $s2  #t1 store the address of A[i]
8          lw     $t0, 0($t1)    #load value of A[i] in $t0
9          add    $s5, $s5, $t0  #sum=sum+A[i]
10         slt    $t8, $s3, $s1  # if i <= n
11         beq    $t8, $zero, loop    #if i != n, goto loop

```

- + Step 1: khởi tạo $i = 0$
- + Step 2: lưu địa chỉ của mảng
- + Step 3: lưu giá trị của n
- + Step 4: lưu bước nhảy của i
- + Step 5: khởi tạo $\text{sum} = A[0]$
- + Step 6: tăng i
- + step 7, 8: $t1 = 4s1$
- + Step 9: $t1$ lưu địa chỉ của $s2$
- + Step 10: load dữ liệu của $A[i]$ vào $t0$
- + Step 11: $s5 = s5 + t0$
- + Step 12: So sánh nếu $i \leq n$ thì $t8 = 1$, ngược lại bằng 0
- + Step 13: Nếu $t8 \neq 0$ thì quay lại vòng lặp

c. $\text{sum} \geq 0$

```

1  .data
2  A: .space 16      # Khai báo mảng A có 4 phần tử
3
4  .text
5      li      $s1, -1          # gán i = -1
6      li      $s5, 0           # sum = 0
7
8  loop: add     $s1, $s1, $s4    # i=i+step
9      add     $t1, $s1, $s1     # t1=2*s1
10     add     $t1, $t1, $t1     # t1=4*s1
11     add     $t1, $t1, $s2     # t1 store the address of A[i]
12     lw      $t0, 0($t1)       # load value of A[i] in $t0
13     add     $s5, $s5, $t0     # sum=sum+A[i]
14     sge     $t8, $s5, $zero   # if 0 <= sum
15     bne     $t8, $zero, loop  # if i != n, goto loop

```

- + Step 1: khởi tạo $i = -1$
- + Step 2: lưu địa chỉ của mảng
- + Step 3: lưu giá trị của n
- + Step 4: lưu bước nhảy của i
- + Step 5: khởi tạo $\text{sum} = A[0]$
- + Step 6: tăng i
- + step 7, 14: $t1 = 4s1$
- + Step 8: $t1$ lưu địa chỉ của $s2$
- + Step 9: load dữ liệu của $A[i]$ vào $t0$
- + Step 10: $s5 = s5 + t0$
- + Step 11: So sánh nếu $\text{sum} \geq 0$ thì $t8 = 1$, ngược lại bằng 0
- + Step 12: Nếu $t8 \neq 0$ thì quay lại vòng lặp

d. $A[i] == 0$

```

4  .text
5      li      $s1, -1          # gán i = -1
6      li      $s5, 0           # sum = 0
7      li      $s4, 1           # step = 1
8
9  loop: add     $s1, $s1, $s4    # i=i+step
10     add     $t1, $s1, $s1     # t1=2*s1
11     add     $t1, $t1, $t1     # t1=4*s1
12     add     $t1, $t1, $s2     # t1 store the address of A[i]
13     lw      $t0, 0($t1)       # load value of A[i] in $t0
14     add     $s5, $s5, $t0     # sum=sum+A[i]
15     seq     $t8, $t0, $zero   # if A[i] == 0
16     bne     $t8, $zero, loop  # if i != n, goto loop

```

- + Step 1: khởi tạo $i = -1$
- + Step 2: lưu địa chỉ của mảng
- + Step 3: lưu giá trị của n
- + Step 4: lưu bước nhảy của i
- + Step 5: khởi tạo $\text{sum} = 0$
- + Step 6: tăng i
- + step 7: $t1 = 4s1$
- + Step 8: $t1$ lưu địa chỉ của $\$s2$
- + Step 9: load dữ liệu của $A[i]$ vào $t0$
- + Step 10: $s5 = s5 + t0$
- + Step 11: Nếu $t0 == 0$ thì quay lại vòng lặp

Assignment 6

```
.data
A: .word -5, -4, 4528, 15      #Khai bao mang A co 4 phan tu
```

```
.text
    li    $s1, -1             #gan i = -1
    la    $s2, A               # $2 = &A
    li    $s3, 4               # n = 4
    li    $s4, 1               # step = 1
    li    $s5, 0               # max = 0
```

```
loop:
    add    $s1, $s1, $s4       #i=i+step
    add    $t1, $s1, $s1       #t1=2*s1
    add    $t1, $t1, $t1       #t1=4*s1
    add    $t1, $t1, $s2       #t1 store the address of A[i]
```

```
    add    $s1, $s1, $s4       #i=i+step
    add    $t1, $s1, $s1       #t1=2*s1
    add    $t1, $t1, $t1       #t1=4*s1
    add    $t1, $t1, $s2       #t1 store the address of A[i]
    lw     $t0, 0($t1)         #load value of A[i] in $t0
    abs    $t8, $t0
    sgt     $t7, $t8, $s5       # if max > A[i]
    bne     $t7, $zero, else    #lap lai
    j       endif
```

```
else : add    $s5, $zero, $t8
```

```
endif : bne    $s1, $s3, loop
```

- + Step 1: khởi tạo $i = -1$
- + Step 2: lưu địa chỉ của mảng vào $\$s2$
- + Step 3: lưu giá trị của $n = 4$
- + Step 4: lưu bước nhảy của i
- + Step 5: khởi tạo $\text{max} = 0$
- + Step 6: tăng i
- + step 7: $t1 = 4s1$
- + Step 6: $t1$ lưu địa chỉ của $\$s2$
- + Step 9: load dữ liệu của $A[i]$ vào $t0$
- + Step 10: Lưu giá trị tuyệt đối của $t0$ vào $t8$
- + Step 11: Nếu $t7 > s5$ thì $t8 = 1$
- + Step 12: Nếu $t7 \neq 0$ thì rẽ nhánh đến else
- + Step 13: Khi rẽ đến else thì gán $t8$ cho $t5$
- + step 14: đóng if
- + Step 15: Khi $i \neq n$ thì thực hiện tiếp tục vòng lặp

⇒ **Kết quả:** $\$s5 = 4528$

Registers		Coproc 1	Coproc 0
Name	Number	Value	
\$zero	0	0	
\$at	1	0	
\$v0	2	0	
\$v1	3	0	
\$a0	4	0	
\$a1	5	0	
\$a2	6	0	
\$a3	7	0	
\$t0	8	0	
\$t1	9	268501008	
\$t2	10	0	
\$t3	11	0	
\$t4	12	0	
\$t5	13	0	
\$t6	14	0	
\$t7	15	0	
\$s0	16	0	
\$s1	17	4	
\$s2	18	268500992	
\$s3	19	4	
\$s4	20	1	
\$s5	21	4528	
\$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	0	
pc		4194380	
hi		0	
lo		0	