Helena Fu
Period 4
Ms. Voldstad

2.23 & 2.26 through 2.30

2.23
    $t0 holds 0x00101000
    slt $t2, $0, $t0
    bne $t2, $0, ELSE
    j DONE
ELSE: addi $t2, $t2, 2
DONE:
	0 < 0x00101000 = -> $t2 = 0
	if $t2 != 0, else
	$t2 += 2
	$t2 = 3

2.26
LOOP: slt $t2, $0, $t1
    beq $t2, $0, DONE
    subi $t1, $t1, 1
    addi $s2, $s2, 2
    j LOOP
DONE:
2.26.1
$t1 is initialized to value 10
$s2 is zero.
2.26.2
Write equivalent C code
A   B   i   temp
$s1 $s2 $t1 $t2
2.26.3
$t1 = N
	0 < 10 $t2 = 1
	1 = 0 => DONE
	$t1 -= 1
	$s2 += 2
	
	When $t1 < 0, loops 10 times
	$s2 = 20
	i = 10;
	while (0 < i) {
	   i -= 1;
	   B += 2;
	}
	
	0 < N
	N -= 1;
	Number of MIPS instructions executed is n*5 + 2.

2.27
for (i = 0; i < a; i++)
   for (j = 0; j < b; j++)
      D[4 * j] = i + j;
 a   b   i   j
$s0 $s1 $t0 $t1
	LOOPF: slt $t2, $t0, $s0
	    beq $t2, $0, DONE
	    addi $t0, $t0, 1
	    j LOOPs
	LOOPS: slt $t3, $t1, $s1
	    beq $t3, $0, DONE
	    addi $t1, $t1, 1
	    mult $t1, 4
	    mfhi $t4
	    add $t4($s2), $t0, $t1
	    j LOOPF
	DONE:

	x addi $t0, $0, 0 # set $t0 to 0 i = 0 beq $0, $0, TEST1 # always branch to TEST1 LOOP1: addi $t1, $0, 0 # set $t1 to 0 j = 0 beq $0, $0, TEST2 # always branch to TEST2 LOOP2: add $t3, $t0, $t1 # put a + b into $t3 sll $t2, $t1, 4 # multiply j by 2^4 or 16 and put it into temp add $t2, $t2, $s2 # add temp to the address of the array and store in temp sw $t3, ($t2) # store a + b into the array at that address addi $t1, $t1, 1 # j++ TEST2: slt $t2, $t1, $s1 # if $t1 < $s1 j < b bne $t2, $0, LOOP2 # go to LOOP2 addi $t0, $t0, 1 # else add 1 to a for outer loop TEST1: slt $t2, $t0, $s0 # if $t0 < $s0 i < a bne $t2, $0, LOOP1 # go to LOOP1
2.28 How many MIPS instructions C code?	a = 10, b = 1, D[] = 0 14 158 instructions executed

2.29
addi $t1, $0, $0
LOOP: lw $s1, 0 ($s0)
add $s2, $s2, $s1
addi $s0, $s0, 4
addi $t1, $t1, 1
slti $t2, $t1, 100
bne $t2, $0, LOOP
	$t1 $s2 $s0
	i result MemArray
	int i = 0;
	int *s1 = *MemArray;
	result += s1;
	MemArray += 4;
	i += 1;
	Loop if !(i < 100)
	
	for (int i = 0; i < 100; i++)
	result += MemArray;
	
	addi $t1, $0, $0 # i = 0
	LOOP: lw $s1, 0($s0) # temp = first element of MemArray
	add $s2, $s2, $s1 # add first element to result
	addi $s0, $s0, 4 # increment the array to the next element
	addi $t1, $t1, 1 # i++
	slti $t2, $t1, 100 # i < 100? put 1 in $t2 for true

	bne $t2, $s0, LOOP # if 1 not equal to current address of array?
2.30	addi $t1, $s0, 400 LOOP: lw $s1, 0($s0) addi $t1, $s0, 400 # add 400 to address of array and store in $t1 //isn't this too far? 396 LOOP: lw $s1, 0($t1) # put the 100th element of the array in $s1 add $s2, $s2, $s1 # add element to the result addi $t1, $t1, -4 # subract 4 to move back 1 element bne $t1, $s0, LOOP # if array index isn't equal to the beginning of the array, loop