Lebanese American University

School of Arts and Sciences

Department of Computer Science and Mathematics

CSC 320 – Computer Organization

Problem Set 4: Language of the Computer

Exercise 1

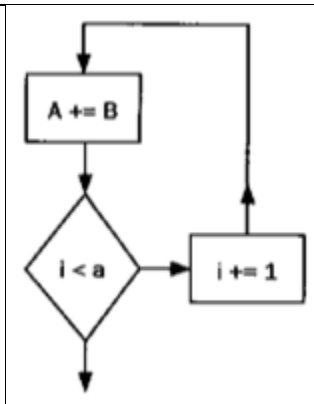
For these problems, the table holds some C code. You will be asked to evaluate these C code statements in MIPS assembly code.

a.	for (i = 0; i < a; i++) a+= b;

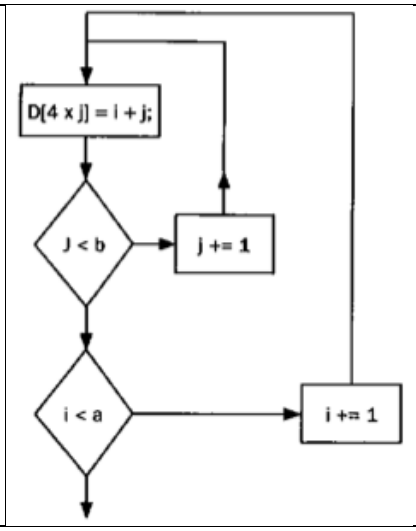
b.	for (i = 0; i < a; i++){
	for(j = 0; j < b; j++;
	D[4 * j] = i + j;
	}

1.1 For the table above, draw a control-flow graph of the C code.

Solution:

a.	

b.



1.2 For the table above, translate the C code to MIPS assembly code. Use a minimum number of instructions. Assume that the value a, b i, j are in registers $s0, $s1, $t0, $t1, respectively. Also, assume that register $s2 holds the base address of the array D.

Solution:

a.

```
        addi  $t0,  $0,   0
        beq   $0,   $0,   TEST
LOOP:   add   $s0,  $s0,  $sl
        addi  $t0,  $t0,  l
TEST:   slt   $t2,  $t0,  $s0
        bne   $t2,  $0,   LOOP
```

b.

```
        addi  $t0,  $0,   0
        beq   $0,   $0,   TEST1
LOOP1:  addi  $t1,  $0,   0
        beq   $0,   $0,   TEST2
LOOP2:  add   $t3,  $t0,  $t1
        sll   $t2,  $t1,  4
        add   $t2,  $t2,  $s2
        sw    $t3,  ($t2)
        addi  $t1,  $t1,  1
TEST2:  slt   $t2,  $t1,  $s1
        bne   $t2,  $0,   LOOP2
        addi  $t0,  $t0,  1
TEST1:  slt   $t2,  $t0,  $s0
        bne   $t2,  $0,   LOOP1
```

1.3 How many MIPS instructions does it take to implement the C code? If the variables a and b are initialized to 10 and 1 and all elements of D are initially 0, what is the total number of MIPS instructions that is executed to complete the loop?

For these problems, the table holds MIPS assembly code fragments. You will be asked to evaluate each of the code fragments, familiarizing you with the different MIPS branch instructions.

a.		addi	$t1,	$0,	50
	Loop:	lw	$s1,	0($s0)	
		add	$s2,	$s2,	$s1
		lw	$s1,	4($s0)	
		add	$s2,	$s2,	$s1
		addi	$s0,	$s0,	8
		subi	$t1,	$t1,	1
		bne	$t1,	$0,	Loop
b.		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

Solution:

a.	6 instructions to implement and infinite instructions executed
b.	14 instructions to implement and 158 instructions executed

1.4 What is the total number of MIPS instructions executed?

Solution:

a.	351
b.	601

1.5 Translate the loops above into C. Assume that the C-level integer i is held in register $t1, $s2 holds the C-level integer called *result*, and $s0 holds the base address of the integer *MemArray*.

Solution:

a.	j = 0;
	for (i=50; i>0; i--){
	  result += MemArray[j];
	  result += MemArray[j+1];
	  j += 2;

b.	j = 0;
	for (i=0; i<100; i++){
	  result += MemArray[j];
	  j += 1;

1.6 Rewrite the loop in MIPS assembly to reduce the number of MIPS instructions executed.

Solution:

a.	addi $t1, $s0, 400
	LOOP: lw $s1, 0($s0)
	add $s2, $s2, $s1
	lw $s1, 4($s0)
	add $s2, $s2, $s1
	addi $s0, $s0, 8
	bne $s0, $t1, LOOP

b.	addi $t1, $s0, 400
	LOOP: lw $s1, 0($t1)
	add $s2, $s2, $s1
	addi $t1, $t1, -4
	bne $t1, $s0, LOOP

Exercise 2

Assume that the stack and the static data segments are empty and that the stack and global pointers start at address 0x7fff fffc and 0x1000 8000, respectively. Assume the calling conventions as specified in Figure 2.11 and that function inputs are passed using registers $a0 -$a3 and returned in register $v0. Assume that leaf functions may only use saved registers.

a.

```
int my_global = 100
main()
{
        int x = 10;
        int y = 20;
        int z;
        z = my_function(x, y);
}
int my_function (int x, int y)
{
        return x – y +  my_global;
}
```

b.

```
int my_global = 100;
main()
{
        int z;
        my_global += 1;
        z = leaf_function(my_global)
}
int leaf_function(int x)
{
        return x + 1;
}
```

2.1 Write MIPS assembly code for the code in the table below.

Solution:

a.

```
MAIN:   addi    $sp,    $sp,    -4
        sw      $ra,    ($sp)

        addi    $a0,    0,      10
        addi    $a1,    $0,     20
        jal     FUNC
        add     $t2,    $v0,    $0

        lw      $ra,    ($sp)
        addi    $sp,    $sp,    4
        jr      $ra
                        #
FUNC:   lw      $t1,    0($s0),         #assume $s0 has global variable base
        sub     $t0,    $a0,    $a1
        addi    $v0,    $t0,    $t1
        jr      $ra
```

b.

```
MAIN:   addi    $sp,    $sp,    -4
        sw      $ra,    ($sp)

        lw      $t1,    0($s0)      #assume $s0 has global variable base
        addi    $a0,    $t1,    1
        jal     LEAF
        add     $t2,    $v0,    $0

        lw      $ra,    ($sp)
        addi    $sp,    $sp,    4
        jr      $ra

LEAF:   addi    $v0,    $a0,    1
        jr      $ra
```

Exercise 3

This exercise explores ASCII and Unicode conversion. The following table shows strings of characters.

a.	Hello world
b.	0123456789

3.1 Translate the strings into decimal ASCII byte values.

Solution:

a.	72 101 108 108 111 8 119 111 114 108 100
b.	48 49 50 51 52 53 54 55 56 57

3.2 The following table shows hexadecimal ASCII character values. Translate the hexadecimal ASCII values to text.

a.	41 44 44
b.	4D 49 50 53

Solution:

a.	ADD
b.	MIPS