The University of Jordan
King Abdullah II School for Information Technology
Computer Science Department
CS 238 Computer Organizations – Homework#1

Instructor: Dr. Jamal Alsakran Due Date: 31/10/2016

- Answer only 4 exercises out of the 6 exercises below
- Due Date: 31/10/2016 in class

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;`

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

b. 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?

Exercise 2: 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, $s0, LOOP`

a. What is the total number of MIPS instructions executed?

b. 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.

c. Rewrite the loop to reduce the number of MIPS instructions executed.

Exercise 3: For the following problems, the table holds C code functions. Assume that the first function listed in the table is called first. You will be asked to translate these C code routines into MIPS assembly.

a.

```
int fib(int n){
    if (n==0)
       return 0;
    else if (n == 1)
       return 1;
    else
       fib(n-1) + fib(n-2);
```

b.

```
int positive(int a, int b) {
    if (addit(a, b) > 0)
       return 1;
    else
       return 0;
    }
int addit(int a, int b) {
    return a+b;
    }
```

a. Implement the C code in the table in MIPS assembly. What is the total number of MIPS instructions needed to execute the function?

b. Functions can often be implemented by compilers "in-line." An in-line function is when the body of the function is copied into the program space, allowing the overhead of the function call to be eliminated. Implement an "in-line" version of the the C code in the table in MIPS assembly. What is the reduction in the total number of MIPS assembly instructions needed to complete the function? Assume that the C variable n is initialized to 5.

c. For each function call, show the contents of the stack after the function call is made. Assume the stack pointer is originally at address 0x7ffffffc.

Exercise 4: The following three problems in this Exercise refer to a function f that calls another function func. The code for C function func is already compiled in another module. The function declaration for func is "int func(int a, int b);". The code for function f is as follows:

a.

```
int f(int a, int b, int c, int d){
  return func(func(a,b),c+d);
}
```

b.

```
int f(int a, int b, int c, int d){
  if(a+b>c+d)
    return func(a+b,c+d);
  return func(c+d,a+b);
}
```

a. Translate function f into MIPS assembly language. If you need to use registers $t0 through $t7, use the lower-numbered registers first.

b. Right before your function f from part (a) returns, what do we know about contents of registers $t5, $s3, $ra, and $sp? Keep in mind that we know what the entire function f looks like, but for function func we only know its declaration.

Exercise 5: 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 that function inputs are passed using registers $a0–$a3 and returned in register $r0. 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;
}
```

a. Write MIPS assembly code for the code in the table above.

b. Show the contents of the stack and the static data segments after each function call.

c. If the leaf function could use temporary registers ($t0, $t1, etc.), write the MIPS code for the code in the table above.

Exercise 6: The following three problems in this Exercise refer to this function, written in MIPS assembly

a.

```
f: add   $v0,$a1,$a0
   bnez  $a2,L
   sub   $v0,$a0,$a1
L: jr    $v0
```

b.

```
f: add   $a2,$a3,$a2
   slt   $a2,$a2,$a0
   move  $v0,$a1
   beqz  $a2, L
   jr    $ra
L: move  $a0,$a1
   jal   g       ; Tail call
```

a. This code contains a mistake that violates the MIPS calling convention. What is this mistake and how should it be fixed?

b. What is the C equivalent of this code? Assume that the function's arguments are named a, b, c, etc. in the C version of the function.

c. At the point where this function is called register $a0, $a1, $a2, and $a3 have values 1, 100, 1000, and 30, respectively. What is the value returned by this function? If another function g is called from f, assume that the value returned from g is always 500.