HW#5 (CSC390)

Due: 02/19/2018 (**Monday**) by **11:00PM**

(**Please turn in your code on the Blackboard**)

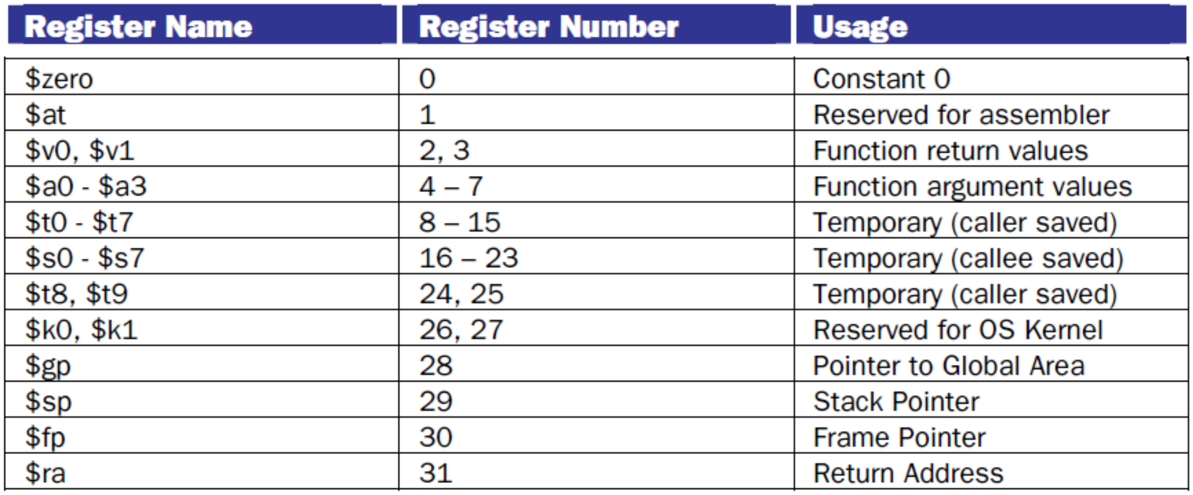
#Q1.

Convert the C function below to MIPS assembly language. Also write a MIPS assembly code to call the function with some initial value of n=5 and store the result in a suitable memory location, labeled as **F**. Consider $S0 will have the base address to store the result into F memory location. Also consider, the procedure uses $S0 to store the final result (i.e. 5+4+3+2+1) before it returns the results to the main program through $V0. Make sure that your assembly language code could be called from a standard C program (that is to say, make sure that you follow the MIPS calling conventions).

```
unsigned int sum(unsigned int n)
{
if (n == 0) return 0;
else return (n + sum(n-1));
}
```

Note down and explain the value of $PC, $ra, $Sp, and the content of stack memory location every time you call the procedure.

The stack grows downward (toward lower memory addresses). The following registers are used in the calling convention:



Register Name	Register Number	Usage
$zero	0	Constant 0
$at	1	Reserved for assembler
$v0, $v1	2, 3	Function return values
$a0 - $a3	4 – 7	Function argument values
$t0 - $t7	8 – 15	Temporary (caller saved)
$s0 - $s7	16 – 23	Temporary (callee saved)
$t8, $t9	24, 25	Temporary (caller saved)
$k0, $k1	26, 27	Reserved for OS Kernel
$gp	28	Pointer to Global Area
$sp	29	Stack Pointer
$fp	30	Frame Pointer
$ra	31	Return Address

Q2. [Q2.26 of your text book];

2.26 Consider the following MIPS loop:

```
LOOP: slt  $t2, $0,  $t1
      beq  $t2, $0,  DONE
      subi $t1, $t1, 1
      addi $s2, $s2, 2
      j    LOOP
DONE:
```

2.26.1 [5] <§2.7> Assume that the register `$t1` is initialized to the value 10. What is the value in register `$s2` assuming `$s2` is initially zero?

2.26.2 [5] <§2.7> For each of the loops above, write the equivalent C code routine. Assume that the registers `$s1`, `$s2`, `$t1`, and `$t2` are integers `A`, `B`, `i`, and temp, respectively.

2.26.3 [5] <§2.7> For the loops written in MIPS assembly above, assume that the register `$t1` is initialized to the value N. How many MIPS instructions are executed?