Assembler Directives
1.) Examine the following C program variable declarations and translate them to the appropriate directives.

Submission Instructions: Enter the number of your desired selection.

a. short int x = 6;
   1. x: .word 6
   2. x: .half 6
   3. x: .space 6
   4. x: .align 6

b. unsigned char msg[8] = {1,4,9,7,3,6,8,2};
   1. msg: .unsigned 1,4,9,7,3,6,8,2
   2. msg: .byte 0x14973682
   3. msg: .byte 1,4,9,7,3,6,8,2
   4. msg: .half 1,4,9,7,3,6,8,2

c. int data[100];
   1. data: .word 100
   2. data: .space 100
   3. data: .space 200
   4. data: .space 0x190

d. char str[] = “hello
   1. str: .ascii “hello
   2. str: .asciiz “hello
   3. str: .byte “hello
   4. str: .asciiz “hello"

HLL to Assembly Translation
2.) Translate the following C code statements to an equivalent assembly language implementation.

```
short data[20];
...
for(int i=0;i < 20; i++){
   data[i] = data[i] + 5;
}

// let i be stored in $t0
```

```
data .space 40 or 0x28___
.text
...
la $s0,data
add $t0,$zero,$zero
li $t4,20 or 0x14___
L0: slt $t1,$t0,$t4
L1: beq $t1,$zero,L9___
L2: sll $t2,$t0,1___
L3: add $s1,$s0,$t2
L4: lh $t3,0($s1)___
```
Subroutines and Assembly

3.) In addition to passing arguments in registers or by using the stack, another possible method (that no one would ever use) is to provide space for the arguments in the code itself (though you should note that this code will NOT assemble using MARS because it does not allow data directives in the text section). However, please study the code below to understand how this method works and answer the given questions. Show any work on an extra page and submit that with your homework. **SHOW ALL VALUES in HEX.**

Assume that the `.data` section starts at 0x10010000.

- **a.)** Before execution of “jal AVG”, what are the contents of $t0, $t1, and $t2.
- **b.)** Before execution of “sra $v0,$v0,1”, what are the contents of $v0, $t3, & $ra?
- **c.)** After program execution, what are the contents of $ra and the word at address RES.

```assembly
x .space 4
y .space 4
...
la $s0, x
la $s1, y
addi $t0, $zero, 10
addi $t1, $zero, 5
lw $s2, 0($s0)
lw $s3, 0($s1)
lw $t2, 0($s2)
lw $t3, 0($s3)
blt $t2, $t0, _L3___
bge $t3, $t1, _L6___

_code A instructions
L1: C...
L2: b _L8___
L3: blt_ $t2, $t0, _L5___
L4: ble_ $t3, $t1, _L7___

_code B instructions
L5: C...
L6: b _L8___
L7: C...
L8: ...
```
a.) $t0 = 0x10010000$  
   $t1 = 0xffffffffbead$  
   $t2 = 0xffffffffbead$

b.) $v0 = 0xfffffffff8$  
    $t3 = 0xfffffffffa$  
    $ra = 0x0040001c$

c.) $ra = 0x00400024$  
    M[RES] = 0xfffffffffc