Leonard Chan

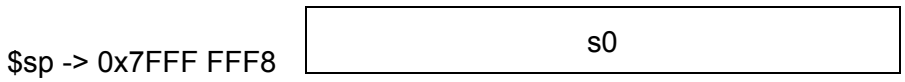
1.
```
addi    $t0, $zero, 0   # t0 is the counter (i) for the outer loop
loop1:
beq     $t0, $s0, end1  # leave loop1 when i == a
addi    $t1, $zero, 0   # t1 is the counter (j) for the inner loop
loop2:
beq     $t1, $s1, end2  # leave loop2 when j == b
ssl     $t2, $t1, 4     # save 4*j (16*j bytes) into some temp variable as the index
add     $t3, $t0, $t1   # save the sum into a temporary register (t3)
add     $t4, $s2, $t2   # save the address for D[4*j] into some register
sw      $t3, 0($t4)     # store the word/value in that register
addi    $t1, $t1, 1     # increment j by 1 (j++)
j       loop2           # go through another iteration of loop2
end2:
addi    $t0, $t0, 1     # increment i by 1 (i++)
j       loop1           # go through another iteration of loop1
end1:
```

    a. 12
    b. 122

2.
```
sw      $zero, 0($gp)           # initialize the global to zero
main:
addi    $s0, $zero, 10          # initialize and set s0 (x) to 10
addi    $s1, $zero, 20          # initialize and set s1 (y) to 20
add     $a0, $zero, $s0         # load x as the first arg
add     $a1, $zero, $s1         # load y as the second arg
jal     myFunc                  # call myFunc
add     $s2, $zero, $v0         # set s2 (z) as the return value
j       end
myFunc:
addi    $sp, $sp, -4            # make space for the global to be loaded on
sw      $s0, 0($sp)             # save whatever was in s0
lw      $s0, 0($gp)             # load the global
sub     $v0, $a0, $a1           # store x-y into the return register
add     $v0, $v0, $s0           # add the global onto the difference
lw      $s0, 0($sp)             # reload the old value of s0
addi    $sp, $sp, 4             # reset the stack pointer
jr      $ra                     # return back to main
end:
```

During function call

Stack: Space for s0 was made (4 bytes)



Heap: nothing since no memory is dynamically allocated

After function call

Stack: Space no longer needed

Heap: nothing since no memory is dynamically allocated