Yingjie Lian
HW- 2
CS-3810
01.28.2018

Answer 1:

1. add $s1, $t2, $t3 # **$s1 is the sum of $t2 and $t3**
2. lw $s3, 8($gp) # **loading variable 8($gp) into $s3**
3. sw $s4, 12($s5) # **storing $s4 into variable 12($s5)**
4. addi $s1, $zero, 100 # **putting base address 100 into $s1**
5. bne $t1, $t2, else # **If $t1 is not equal to $t2, so else statement will be executed**

// bne is used if branches are not equal.

6. add $s1, $s1, $s1 # **$s1 is the sum of $s1 and $s1**

Answer 2: If we want to answer this question we should understand the MIPS instructions, here are my annotations for these instructions:

lw $s1, 0($gp) # load value in base address to $s1
addi $s1, $s1, 25 # increment it by 25
sw $s1, 0($gp) # store it back to same location
lw $s2, 12($gp) # load value in 1000 + 12y[2] in #s2
add $s2, $s2, $s1 # add it by value in $s1 because it is initialised to 0. $s2 = 25
sw $s2, 8($gp) # store it in y[1] 1000 + 8
sw $s2, 12($gp) # store it in y[2] 1000 + 12

So now, we can answer these two questions easily and clearly.

1. What are the memory addresses of variables x, y[0], and y[1]? **(15 points)**

Answer: Base address is 1000 and variable x, y[10] array are stored from there. Assuming variables are of type int, each variable occupies a memory of 4 Bytes.
Since 1 Byte = 8 Bits.
So x has address of 1000, y[0] has address of 1000 + 4 = 1004, y[1] has address of 1008, y[2] has address of 100C and so on.

2.What are the values of variables x, y[0], y[1], and y[2] at the end of the program? **(20 points).**

Answer: So at the end of the program x = 25, y[0] = 0, y[1] = 25, y[2] = 25.

Answer 3: Decimal: 146 = 1 * 10^2 + 4 * 10^1 + 6 * 10^0
Binary: **10010010** = 1 * 2^7 + 1 * 2^4 + 1 *2^1 = Decimal 146

Hexadecimal: **92** = 9 * 16 + 2 = Decimal 146

Answer 4: Binary: 1001100

Decimal: **76** because 1 * 2^6 + 1 * 2^3 + 1* 2^2 = 76

Hexadecimal: **4C** because we can calculate this by decimal number that we have already know,

Decimal: 76 / 16 = 4…...12, so it is 4C

Answer 5: Hexadecimal: 0x6d

Decimal: **109** because 6 * 16 + 13 = 109

Binary: **1101101** because we can calculate this by decimal number that we have already know,

Decimal: 109 / 2 = 54…...1
54 / 2 = 27…...0
27 / 2 = 13…...1
13 / 2 = 6 …...1
6 / 2 = 3…...0
3 / 2 = 1…...1
1 / 2 = 0…...1

Answer 6:

```
la $t0, ($gp) # load base address to $t0
addiu $t0, $t0, 4 # go to base address + 4 (i)
li $t1, 0 # initialise i to 0 and store in its location
sw $t1, ($t0)

loop: # enter for loop
addiu $t0, $t0, 4 # increment $t0 by 4 # initially it is a[0], keeps incrementing by i
mul $t2, $t1, 4 # i * 4
sw $t2, ($t0) # store in a[i]
addi $t1, $t1, 1 # increment i
bit $t1, 10, loop # loop till < 10
la $t0, ($gp) # load base address + 4
addiu $t0, $t0, 4
Sw $t1, ($t0) # store i value
```