Lab 1

Assignment 1

Question: Which lines of code had to be changed? Why?
Answer:

Old	new
#14 addi $s0,$s0,1 #16 li $t0,0x5b	#14 addi $s0,$s0,3 #16 li $t0,0x5d

To skip two characters, you have to increment by 3 instead of 1. And to avoid missing the check to exit the loop, you have to change the stop value.

Assignment 2

Question 1: Your subroutine hexasc is called with an integer-value as an argument in register $a0, and returns a return-value in register $v0. If the argument is 17, what is the return-value? Why?

Answer 1:
If $a0=17 is used (...0001 0001), then the code using addi will remove all bits not in the last nibble, thus leaving us with a0=1 (...0000 0001). This value will result in the output “1”.

Question 2: If your solution contains a conditional-branch instruction: which input values cause the instruction to actually branch to another location? This is called a taken branch.

Answer 2:
The code will jump branch if the given value ($a0) is equal or less than 9.

Assignment 3

Question 1: Which registers are saved and restored by your subroutine? Why?

Answer 1: $ra, $a0,

Question 2: Which registers are used but not saved? Why are these not saved?

Answer 2: $t0–>$t5, $v0

Question 3: Assume the time is 16:53. Which lines of your code handle the '5'?

Answer 3: 109-110, 130-133, 149

Assignment 4

Question 1: If the argument value in register $a0 is zero, which instructions in your subroutine are executed? How many times each? Why?

Answer 1: No looping occurs because it’s set to 0. delay >> delay_while >> delay_end

Question 2: Repeat the previous question for a negative number: -1.

Answer 2: Same as answer 1.

Assignment 6

Question 1: What is the effect of the assembler directive .global? Why is the directive particularly important in this assignment? The teachers will help you with this if necessary.

Answer 1:
.Global makes the defined functions accessible outside of the file, i.e when we call the functions in the C code.