EE 2310 Homework #5 Solutions – Writing Simple Programs

Note: All CLO's in this problem set tie to ABET program-level criterion a.

Name _____ CE _____ EE _____

Write each of the programs in this homework and run them on your SPIM emulator. DO NOT just write something in the area to the right. COMPOSE THE PROGRAMS ACCORDING TO THE DIRECTIONS AND MAKE SURE EACH RUNS PROPERLY AND GIVES THE CORRECT RESULT! Then turn in the programs when the homework is due.

1. **(CLO 5—Assy Lang.) Write a program to do the following: Load $t0 with 12 and call it x. Load 21 in $t1 and denote it as y. Finally, load 32 in $t2 and refer to it as z. Now, compute $3x^2+10y+5z$. Output the result using syscall 1 (remember the result has to be in $a0). Write down the output number on your answer sheet.**

```
        .text
main:
```

2. **(CLO 5—Assy Lang.) Write a program to do the following: Add the words t and u together and store the result in x. OR v and w, storing the result in y. Then multiply x and y and store the result in z. Add x through z together and output that result to the console using syscall 1.**

Remember: You can ONLY perform mathematical operations on data that is in <u>registers</u>.

```
        .data
t:      .word 100
u:      .word 54
v:      .word 37
w:      .word 23
x:      .word 0
y:      .word 0
z:      .word 0
```

3. **(CLO 5—Assy Lang.) The data declarations for the program below are done. Remembering that data can only be manipulated in registers, add data1 to data2. Then add that sum to data3. Multiply the resulting sum by data4, then subtract data 5 from that total. Finally, take the 2's complement of the result, store this final result in ans, and output it to the console. Before you output the answer, output the leader declared below: "Answer = " . Then output the answer you have calculated. Remember that outputting a phrase uses syscall 4.**

```
        .text
main:

        .data
str: .asciiz "Answer = "
data1: .word 95
data2: .word 270
data3: .word 88
data4: .word 23
data5: .word 456
ans:    .word 0
```

4. **(CLO 5—Assy Lang.) We are just beginning to study branch instructions, but they are actually quite easy to understand. In a branch instruction, a test is made for a certain condition, normally comparing two registers or perhaps determining the value of a single register. An example is bgez $t1,next. This branch instruction instructs the computer to evaluate the contents of the register $t1. If the contents are greater than or equal to 0 (≥ 0), then the computer immediately goes to the instruction labeled "next," wherever it is in the program, and executes it, continuing to execute instructions from that point. If the contents of $t1 are less than zero, the computer simply executes the instruction that follows the branch instruction.**

   Let's use that simple branch instruction to do the following: Load the two words declared in the data statement shown to the right into registers, and then determine if either is ≥ 0. If either number is ≥ 0, print it out using syscall 1 (you do not have to include a leader of any sort). If a number is negative, do not print it out. End the program with a syscall 10. Note: you can declare words as either decimal or hexadecimal. SPIM understands both number systems.

   If only one number is printed out, which number is it?

```
        .data
data1: .word 0x0001ead7
data2: .word 0x9800fffe
```

5. **(CLO 5—Assy Lang.) In the data statement below, only word aa is defined, while bb, cc, dd, ee, and ff are simply placeholders for results, and are initially declared with a value of 0. Data word aa is defined as shown. Divide aa by 9, and store in bb. Subtract bb from aa and store in cc. OR aa and cc and store the results in dd. AND aa and dd and store in ee. XOR dd and ee and store that final result in ff. Output only the value of ff, preceded by a leader that says, "Final answer = ". Stop the program as usual.**

```
        .data
aa:     .word 35987
bb:     .word 0
cc:     .word 0
dd:     .word 0
ee:     .word 0
ff:     .word 0
ldr:    .asciiz "Final answer = "
```