Lab 6
Christopher Ahrens

TPS #1

1. Text Segment: 0x00400000 Data Segment: 0x10010000
2. The source code may correspond to more than one line of basic code. It seems like the basic code uses fewer instructions and consists mostly of addition operations. The basic language seems to more or less correspond to the source instructions, but there are occasionally some differences, or more lines required to perform a command.
3. Between two address locations in the data segment, there is a difference of 0x20, or 8 bytes.
4. There are 8 columns in each address location.
5. Only the first address seems to have values for the columns in its address location. It seems that if one column is zero, the entire row is zero.
6. In .data: m: .word 20
7. This is stored at 0x10010004
8. str1: .asciiz “I love CSE31!”
9. 0x10010008, 0x1001000c, 0x10010010
10. It stores the bytes in the correct order, however the order of the characters within those bytes is reversed, which matches what we’ve learned about how characters are stored within bytes.
11. First, use ‘li $v0, 4’ to load the code to print a string. Then use ‘la $a0, str1’ to load the print register with str1. Then, call ‘syscall’ to print.
12. $t3 will store the address of n. In order to store the word itself, we must save the value into a register using lw.

TPS #2

1. Each inequality can be performed by having these three commands, and creating more opcodes is not needed when one can simply keep this in account in code.

   TPS #2 code in compare.s