Part 1 - Report

This assembly program *checks if the second number that we input into the machine is greater, lower or equal to the first number that we input.* This code is fully working.  
I decided to approach the solution like this: I take two inputs (The first one goes into register 0 (R0) and the second one goes into register 1 (R1)) using the *“INP Rd,2”* command. After that, I compare those using the command *“CMP”* which compares the value stored into R1 with the value stored into R0 *(CMP R1,R0)*. Then we use the branches as if statements to jump to the labels depending on the result of the comparison *(BGT, BLT, BEQ)*. Each label contains a *“MOV”* command that stores into R2 the sentence in ASCII characters that we are going to print out and then a branch to the *output label*. In the end, the *“out”* label contains the *“OUT”* commands that print the second value, the sentence and finally the first value. After these, we *HALT* the program.

Gianmarco Natuzzi,

BSc Games Development

Part 2 – Report

This assembly program *checks if the second number that we input into the machine is greater lower or equal to the first number that we input only if it’s between 1 and 13*. This code is fully working.  
I decided to approach the solution like this: I take an input that goes into register 0 with the *“INP R0,2”* command. Then I compare the value stored in R0 with 1 using the *“CMP”* command. If the value is lower than 1, the code *branches* to an *error label* that moves into R2 an error message written in ASCII, it *outputs* it and then *HALTS* the program. Then the number in R0 is *compared* with 13. If the value is greater than 13, the code *branches* to the same *error label* as before that moves into R2 the same error message written in ASCII, and then *outputs* it and *HALTS* the program.   
These two steps of comparison are repeated for the number that we get from the second INPUT *“INP R1,2”.*  
Now that we checked if the numbers are valid, we compare using the “*CMP”* command the values stored in R0 and R1. Then we use the branches as if statements to jump to the labels depending on the result of the comparison *(BGT, BLT, BEQ).* Each label contains a *“MOV”* command that stores into R2 the sentence in ASCII characters that we are going to print out and then a branch to the *output label*. In the end, the *“out”* label contains the *“OUT”* commands that print the second value, the sentence and finally the first value. After these, we *HALT* the program.

Gianmarco Natuzzi,

BSc Games Development