CSCI 240 Computer Organization and Assembly Language Programming
Lab - The Guessing Game

Input

The LC-3 assembly language facilitates the reading of ASCII valued characters from the console using the `GETC` pseudo-op which is equivalent to the `TRAP x20` instruction. This instruction reads a *single* ASCII valued character from the console and stores the value in register `R0` with variable definition `char R0`. The `GETC` pseudo-op is equivalent to the C instruction `scanf("%c", &R0);` or the C++ instruction `std::cout << R0;`

Reading a String

The following LC-3 assembly language program reads a string from the console, character-by-character, until a `\n` is encountered in which case it will terminate and echo the string.

```
            .ORIG x3000
            LD R1, PTR  ; Initialize R1 to the base pointer of the string

            ; Read character from the console, if not equal to '\n' then save the
            ; character to memory; otherwise, terminate
ReadChar    GETC

            ; Test if character is '\n' (x000A)
            ADD R2, R0, #-10
            BRz Terminate

            ; If the read character is not '\n'
            STR R0, R1, #0       ; Save read character to memory
            ADD R1, R1, #1  ; Increment the base pointer
            BRnzp ReadChar

            ; Append the null character '\0' (0x0000) to the string and output
            ; the string to the console
Terminate   AND R0, R0, #0 ; Zero out register
            LDR R1, R0, #0 ; Write '\0' (0x0000) to memory
            LD R0, PTR     ; Load string base pointer to R0
            PUTS           ; Print string found at R0
            HALT
; PTR is the base pointer to the stored string
NEWLINE     .FILL x00A0
PTR         .FILL x4000
            .END
```

Output

The LC-3 assembly language facilitates the printing of *C-strings* using the `PUTS` pseudo-op which is equivalent to the `TRAP x22` instruction. This instruction prints ASCII valued characters beginning at the memory address stored in register `R0` until a null character is found. For example,

```
            .ORIG x3000
             LD R0, SomeString

            .ORIG x3010
SomeString .STRINGZ "It's a TRAP!"
```

will output `It's a TRAP!` to the console. This is equivalent to the following C program snippet:

```c
const char* str= "It's a TRAP!";
char* R0 = str;
printf("%s", R0);
```

Guessing Game

Write an assembly language program which will simulate a simple guessing game. For example, suppose the program has stored the value 6. The program should continually ask the user to guess a number between 0 and 9 (see sample program input/output below).

```
Guess a number between 0 and 9 (inclusive): 8
Too big.

Guess a number between 0 and 9 (inclusive): 4
Too small.

Guess a number between 0 and 9 (inclusive): 6
Correct! You took 3 guesses
```

Submission

Submit the following files to Canvas:

- LC-3 assembly language source file (`GussingGame.asm`)
- Console output similar to the sample input/ouput described in the lab (`output.txt`).