CS231 Computer Systems and Organization
Department of Computer Science
SEDS

Lab 4

Submit your work to moodle before the deadline

1. Implement a procedure **reverse** in MIPS assembly language that, given a string ***S*** and its ***length***, reverses ***S***.

For example, if ***S*** =“Hello” and ***length*** = 5, then after calling your procedure ***S*** becomes “olleH”, and this reversed ***S*** should be printed out. (NOTE: **S** = “H ello” and ***length*** = 6, ***S*** becomes “olle H”, assuming each space will be calculated as an each length; also special characters will not be considered).

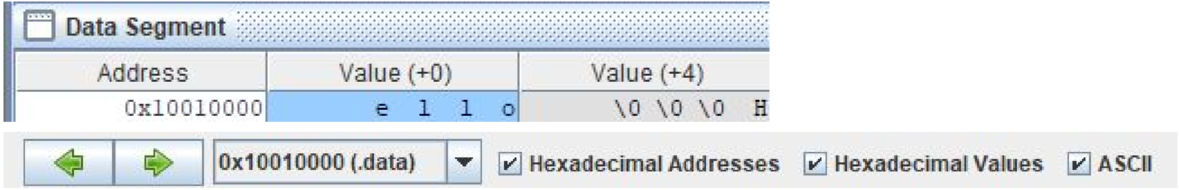
In the program, we assume the variables (e.g., ***S*** and ***length***) should be declared and initialized manually in the **.data** section. (Need to be tested by changing the ***S*** and ***length*** manually.)

The signature of this procedure in a high level language would look like this:

void reverse(char String[], int length);

Output: for ***S*** =“**Hello**”

With the printed **olleH**
The string ***S*** MUST have **olleH** (,with ASCII representation; the address might be different)



Data Segment

Address	Value (+0)	Value (+4)
0x10010000	e l l o	\0 \0 \0 H

For the optional problem, you need to refer more SYSCALL system services, in addition to the below examples: https://courses.missouristate.edu/KenVollmar/mars/Help/SyscallHelp.html

NOTES: How to print Integers and Strings/space/newline using ‘syscall’

```
        .data
x:              .word    5
msg1:           .asciiz  "x="
nl:             .asciiz  "\n"
space:          .asciiz  " "

        .text
main:
        # Register assignments
        # $s0 = x

        # Initialize registers
        lw      $s0, x          # Reg $s0 = x

        # Print msg1
        li      $v0, 4          # print_string syscall code = 4
        la      $a0, msg1
        syscall
```

```
# Print result (x)
li      $v0,1           # print_int syscall code = 1
move    $a0, $s0        # Load integer to print in $a0
syscall

# Print newline
li      $v0,4           # print_string syscall code = 4
la      $a0, nl
syscall

# Exit
li      $v0,10          # exit
syscall
```