**%macro print 2**

mov rdx, %2 ; rdx = length of string

mov rsi, %1 ; rsi = address of string

mov rdi, 1 ; rdi = stdout

call write ; write string

%endmacro

**segment .data**

MAX\_CHARS: equ 9 ; max chars allowed for string

NEWLINE: equ 10 ; newline char

NULL: equ 0 ; null entry

prompt db "Enter the string",NEWLINE,NULL ; prompt for input

PROMPT\_LEN: equ $ - prompt - 1 ; length of prompt

errorStr db "String was truncated",NEWLINE,NULL ; error string

ERR\_LEN: equ $ - errorStr - 1 ; length of error

**segment .bss**

input resb 9

**segment .text**

extern write, getchar, putchar

global main

**printError**:

print errorStr, ERR\_LEN ; print the error string

ret

**getString**:

xor rbx, rbx ; set rbx = 0

mov r15, [rsp + 8] ; set r15 to input string

**getStringLoop**:

call getchar ; rax = getchar()

cmp rax, NEWLINE ; check for end of terminal input

je doneGetString

cmp rax, -1 ; check for end of file

je doneGetString

cmp rbx, MAX\_CHARS ; check if max length reached

je overSize

mov [r15 + rbx], al ; string[rbx] = getchar()

inc rbx ; rbx++

jmp getStringLoop ; goto input

**overSize**:

call printError

**doneGetString**:

ret

**main**:

print prompt, PROMPT\_LEN ; print prompt

push rbp ; save LV frame

mov rbp, rsp ; new LV frame to top of stack

push input ; pointer to input as argument

call getString ; call getString function

mov rsp, rbp ; restore the SP

pop rbp ; restore LV frame

print input, rbx ; print input string

**done**:

mov dil, NEWLINE ; move console to next line

call putchar

xor rax, rax ; set return status to 0

ret