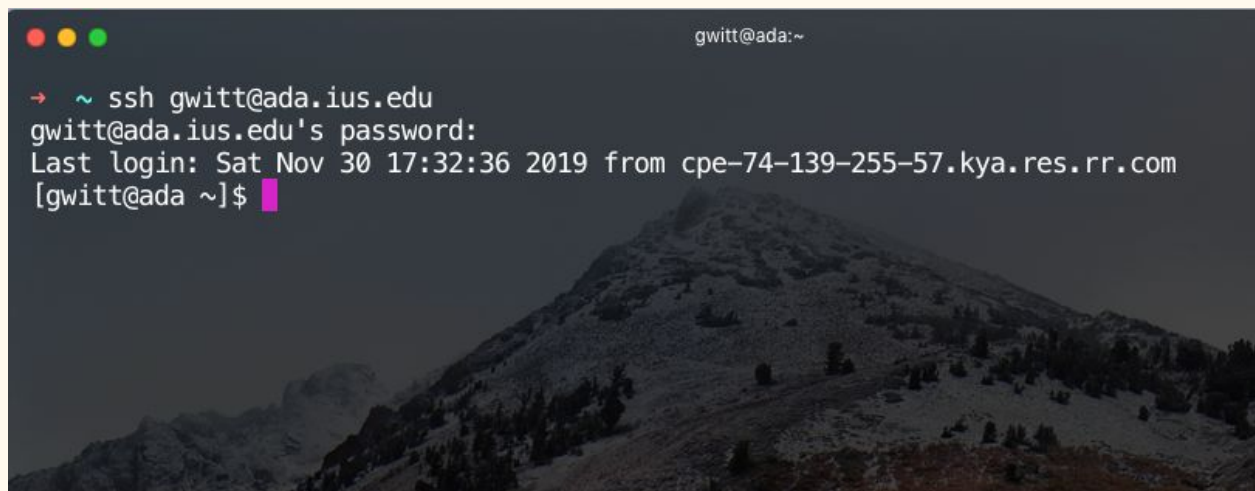


Greg Witt
Computer Structures

P-07

Login to ADA Via SSH

A terminal window with a dark background and a mountain landscape wallpaper. The window title bar shows three colored circles (red, yellow, green) on the left and the text 'gwitt@ada:~' on the right. The terminal text shows a user running 'ssh gwitt@ada.ius.edu', entering a password, and receiving a confirmation message about the last login. The prompt '[gwitt@ada ~]\$' is followed by a pink cursor.

```
→ ~ ssh gwitt@ada.ius.edu
gwitt@ada.ius.edu's password:
Last login: Sat Nov 30 17:32:36 2019 from cpe-74-139-255-57.kya.res.rr.com
[gwitt@ada ~]$
```

Code Execution with VIM

```

;External Function Calls:
extern printf, scanf

SECTION .data
fmt: db "Enter an argument: ", 0 ;welcome
fmt1: db "The argument is: %s", 10, 0 ;format to display the message
fmt2: db "%s", 0 ;variable for argument
fmt4: db " is your argument revers", 10, 0 ;label for reversing

SECTION .bss
input resw 8 ;takes input

global main
SECTION .text
main:

; Display Message for the user:

mov rdi, fmt ;sets up user expectations
mov rsi, input ;move value into rsi for displaying
mov rax, 0 ;zero out rax for printf
call printf ;calls printf to display the value from rsi

mov rdi, fmt2 ;declare variables
mov rsi, input ;moves input into rsi
mov al, 0 ;zeros out al register for scanf
call scanf ;calls to scanf

mov rdi, fmt1 ;display the argument for the user
mov rsi, input ;moves input into printf register for display again
mov rax, 0 ;zero out rax
call printf ;call to printf to display input

;Begins Programming for Reversal
mov rcx, rax ;moves 0 into rcx
mov rdi, input ;rsi and rdi will hold the input from the user for program
mov rsi, input
add rdi, rax ;adds the space needed for rdi from rax
dec rdi
shr rax, 1 ;divide the length of rax

loop:

```

3,1 Top

```

;External Function call printf          ;calls printf to display the value from rsi

;Extern printf, mov rdi, fmt2          ;declare variables
;mov rsi, input          ;moves input into rsi
SECTION .data
;mov al, 0          ;zeros out al register for scanf
;call scanf argument: " " ;calls to scanf
;fmt1: db "The argument is: %s", 10, 0 ;format to display 1
;fmt2: db "able for: %s", 10, 0 ;format to display 2
;fmt4: db "display again", 10, 0 ;format to display 4
;mov rdi, input ;moves input into printf register for display again
;mov rax, 0 ;zero out rax
;call printf ;call to printf to display input

SECTION .bss
;Begins Programming for Reversal
;input: mov rcx, rax ;moves 0 into rcx
;mov rdi, input ;rsi and rdi will hold the input from the user for program
;global: mov rsi, input
SECTION .text
;add rdi, rax ;adds the space needed for rdi from rax
;dec rdi
;shr rax, 1 ;divide the length of rax

;loop:
;mov bl, [rsi] ;swaps the components of rsi, rdi using 8bit registers
;mov bh, [rdi]
;mov [rsi], bh
;mov [rdi], bl
;inc rsi ;increment rsi
;dec rdi ;decrement rdi
;dec rax ;decreases the counter variable
;jnz loop ;jnz will determine when the loop is zero

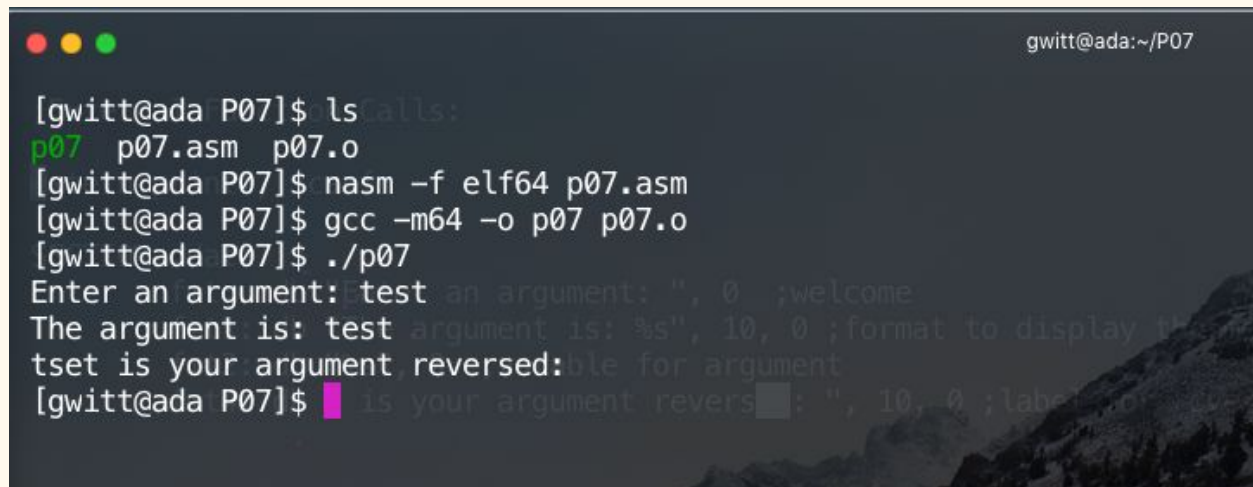
;Display Reversed Characters
;mov rdx, rcx ;moves rcx into rdx for displaying again
;mov rdi, 1
;mov rax, 1
;syscall

;mov rdi, fmt4 ;displays the fourth parameter
;mov rax, 0
;call printf ;calls the printf function for displaying format

;mov rax, 60 ; exit process
;xor rdi, rdi
;syscall ;ends program

```

Compile Code via Command Line



```
gwitt@ada:~/P07
[gwitt@ada P07]$ ls
p07  p07.asm  p07.o
[gwitt@ada P07]$ nasm -f elf64 p07.asm
[gwitt@ada P07]$ gcc -m64 -o p07 p07.o
[gwitt@ada P07]$ ./p07
Enter an argument: test
The argument is: test
tset is your argument, reversed:
[gwitt@ada P07]$
```