

a.) The way to develop this code was through the resources of stack overflow related to assembly language and the slideshows given during lab. The majority was through online research into function calling and declaration. The decision to use XOR operation for comparison of inserted strings was a conclusion made because of the desired outcome. A comparison of alike and unalike using XOR operation required minimal extra operations.

b.)

Unset

```
section .data
prompt1 db "Enter first string: ", 0
prompt2 db "Enter second string: ", 0
result_msg db "Hamming Distance: ", 0
newline db 10

section .bss
str1 resb 256      ; Buffer for the first input string
str2 resb 256      ; Buffer for the second input string
hamming_num resb 3 ; Buffer to store the Hamming distance as ASCII (up to 2
digits + null terminator)

section .text
global _start

_start:
    ;; Prompt and read the first string
    mov rax, 1
    mov rdi, 1
    mov rsi, prompt1
    mov rdx, 19
    syscall

    mov rax, 0
    mov rdi, 0
    mov rsi, str1
    mov rdx, 256
    syscall

    ;; Remove newline character if present
    dec rax
    mov byte [rsi + rax], 0

    ;; Prompt and read the second string
    mov rax, 1
    mov rdi, 1
    mov rsi, prompt2
    mov rdx, 20
```

```
    syscall

    mov rax, 0
    mov rdi, 0
    mov rsi, str2
    mov rdx, 256
    syscall

    ;; Remove newline character if present
    dec rax
    mov byte [rsi + rax], 0

    ;; Initialize pointers
    mov rsi, str1
    mov rdi, str2
    xor rcx, rcx      ; Hamming distance counter

compare_loop:
    mov al, [rsi]      ; Load character from str1
    mov bl, [rdi]      ; Load character from str2
    test al, al        ; Check if end of string
    jz done_compare
    test bl, bl        ; Also check if str2 ends
    jz done_compare

    cmp al, bl
    je skip_increment
    inc rcx            ; Increment Hamming distance if characters differ

skip_increment:
    inc rsi
    inc rdi
    jmp compare_loop

done_compare:
    ;; Convert the Hamming distance to ASCII
    mov rax, rcx
    add rax, '0'
    mov [hamming_num], al
    mov byte [hamming_num+1], 0

    ;; Print "Hamming Distance: "
    mov rax, 1
    mov rdi, 1
```

```
mov rsi, result_msg
mov rdx, 18
syscall

;; Print the Hamming distance
mov rax, 1
mov rdi, 1
mov rsi, hamming_num
mov rdx, 1
syscall

;; Print newline
mov rax, 1
mov rdi, 1
mov rsi, newline
mov rdx, 1
syscall

;; Exit
mov rax, 60
xor rdi, rdi
syscall
```

Nate Thies (ZA97852)

c.)

```
Success. Logging you in...

UMBC Division of Information Technology          http://doit.umbc.edu/
-----
If you have any questions or problems regarding these systems, please call the
DoIT Technology Support Center at 410-455-3838, or submit your request on the
web by visiting http://my.umbc.edu/help/request

Remember that the Division of Information Technology will never ask for your
password. Do NOT give out this information under any circumstances.
-----

Last login: Thu Mar  6 22:03:05 2025 from [REDACTED]
nthies1@linux4 ~]$ cd CMPE310
nthies1@linux4 ~/CMPE310$ ls
HammingDistance  LabHw  SimpleAdd
nthies1@linux4 ~/CMPE310$ cd HammingDistance
nthies1@linux4 HammingDistance]$ ls
ham  ham.asm  ham.asm~  ham.o
nthies1@linux4 HammingDistance]$ ./ham
Enter first string:foo
Enter second string:bar
Hamming Distance: 3
nthies1@linux4 HammingDistance]$ ./ham
Enter first string:little doggie
Enter second string:ham sandie
Hamming Distance: 9
nthies1@linux4 HammingDistance]$
```