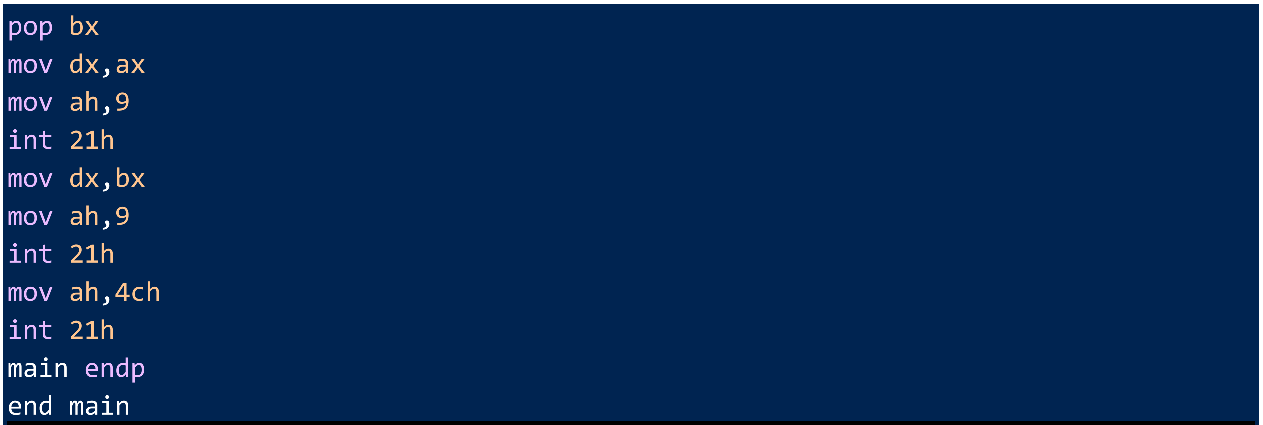
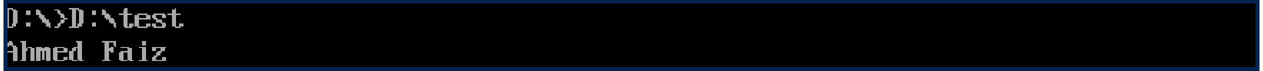
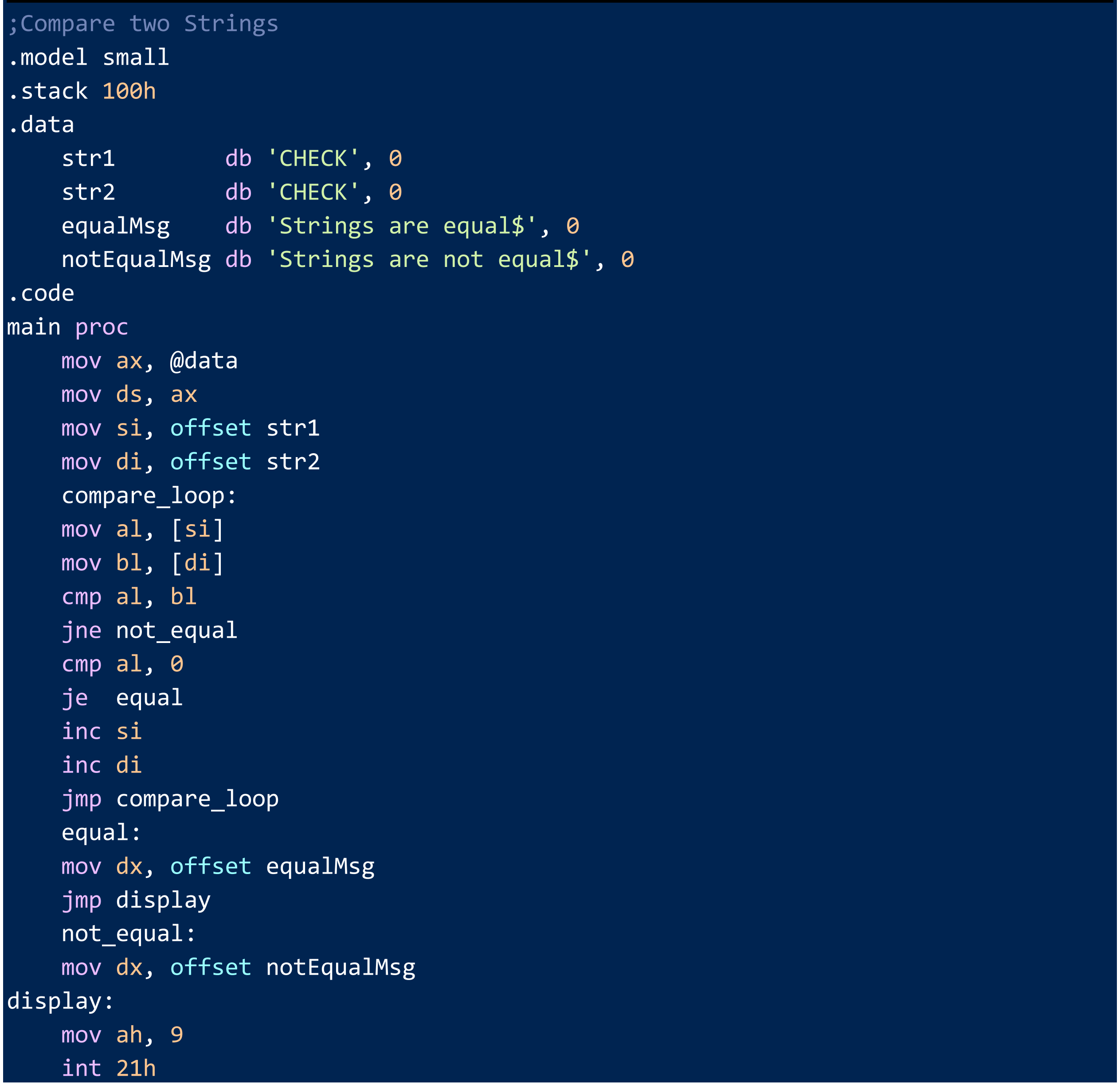
```
;Reverse a string
.model small
.stack 100h
.data
buffer db 6
.code
main proc
mov  ax, @data
mov  ds, ax
mov  dx, offset buffer
mov  ah, 0Ah
int  21h
mov  si, offset buffer+1
mov  cl, [buffer]
l1:
mov  al, [si]
push ax
inc  si
loop l1
mov  cl, [buffer]
l2:
pop  dx
mov  ah, 2
int  21h
loop l2
mov  ah, 4Ch
int  21h
main endp
end Main

;Swap Strings
.model small
.stack 100h
.data
varx db' Faiz$'
varz db 'Ahmed$'
.code
main proc
mov ax,@data
mov ds,ax
mov ax, offset varx
mov bx,offset varz
push ax
push bx
pop ax
```



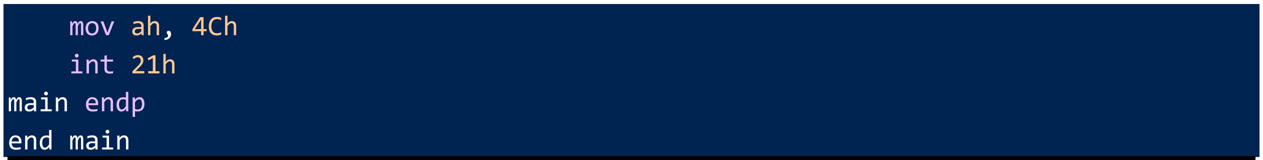
```
pop bx
mov dx,ax
mov ah,9
int 21h
mov dx,bx
mov ah,9
int 21h
mov ah,4ch
int 21h
main endp
end main
```



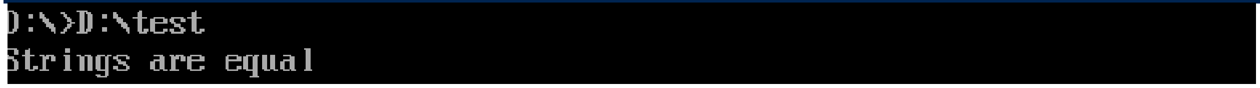
```
D:\>D:\test
Ahmed Faiz
```



```
;Compare two Strings
.model small
.stack 100h
.data
    str1        db 'CHECK', 0
    str2        db 'CHECK', 0
    equalMsg    db 'Strings are equal$', 0
    notEqualMsg db 'Strings are not equal$', 0
.code
main proc
    mov ax, @data
    mov ds, ax
    mov si, offset str1
    mov di, offset str2
    compare_loop:
    mov al, [si]
    mov bl, [di]
    cmp al, bl
    jne not_equal
    cmp al, 0
    je  equal
    inc si
    inc di
    jmp compare_loop
    equal:
    mov dx, offset equalMsg
    jmp display
    not_equal:
    mov dx, offset notEqualMsg
display:
    mov ah, 9
    int 21h
```



```
    mov ah, 4Ch
    int 21h
main endp
end main
```



```
D:\>D:\test
Strings are equal
```