Assignment No: 02

Department of Computer Science

Iqra University Islamabad

Computer Organization and Assembly Language

Maqsood Ahmed

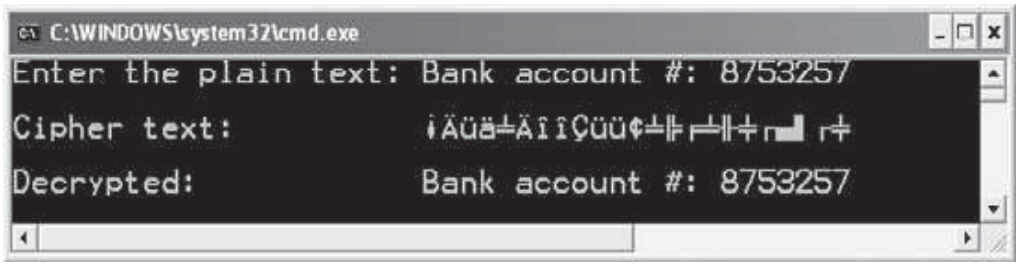
ID: 38186

Problem # 1: [CLO2]

Write a simple program that uses symmetric encryption, a process by which the same key is used for both encryption and decryption. The following steps occur in order at runtime.

1. **The user enters the plain text.**
2. **The program uses a single-character key to encrypt the plain text, producing the cipher text, which is displayed on the screen.**
3. **The program decrypts the cipher text, producing and displaying the original plain text.**

Here is sample output from the program:



Source Code:

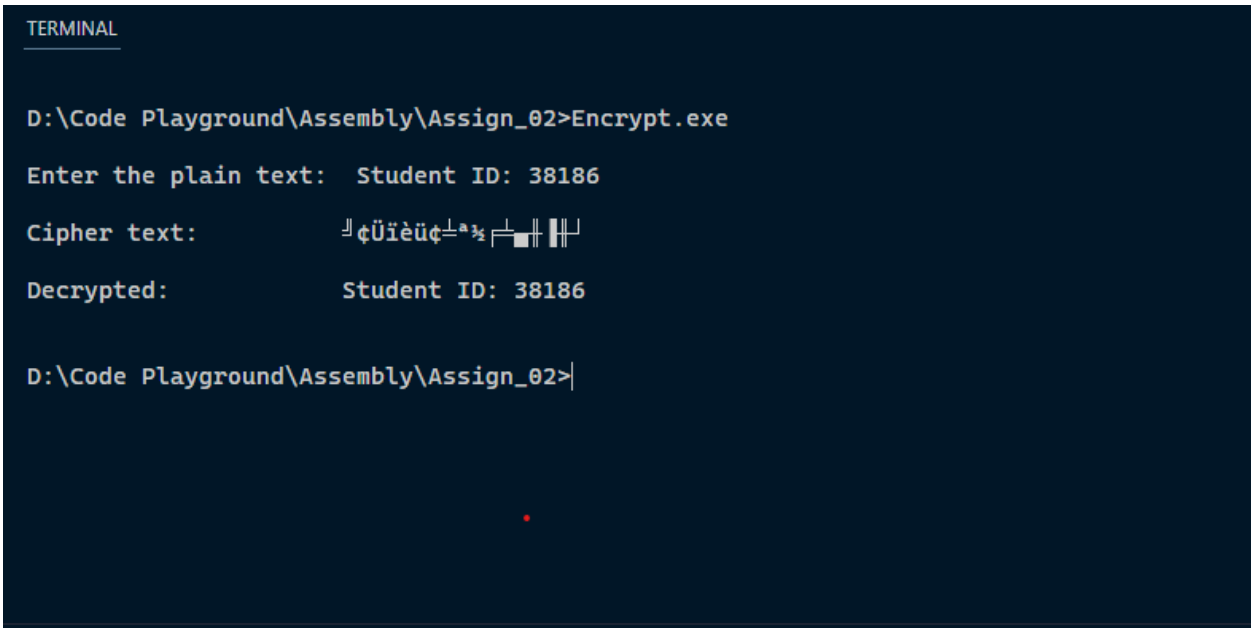
```
.686

.MODEL flat, stdcall

.STACK

INCLUDE Irvine32.inc


KEY = 239           ; any value between 1-255

BUFMAX = 128   ; maximum buffer size

.data

sPrompt BYTE "Enter the plain text:  ",0

sEncrypt BYTE "Cipher text:         ",0

sDecrypt BYTE "Decrypted:           ",0

buffer BYTE BUFMAX+1 DUP(0)
```

```
bufSize DWORD ?

.code
main PROC
   call Crlf              ; for new line
   call InputTheString    ; input the plain text
   call TranslateBuffer   ; encrpyt the buffer
   mov edx,OFFSET sEncrypt  ; display encrypted message
   call DisplayMessage
   call TranslateBuffer   ; decrpyt the buffer
   mov edx,OFFSET sDecrypt  ; display decrypted message
   call DisplayMessage
   exit
main ENDP

;-----------------------------------------------------------------------------
InputTheString PROC
   ; Prompts user for a plaintext string. Saves the string and its length
   ; Receives: nothing
   ; Returns:  nothing
;-----------------------------------------------------------------------------
   pushad                 ; save 32-bit registers
   mov edx, OFFSET sPrompt  ; display a prompt
   call WriteString
   mov ecx, BUFMAX        ; maximum character count
   mov edx, OFFSET buffer   ; point to the buffer
   call ReadString        ; input the string
   mov bufSize, eax       ; save the length
   call Crlf
```

```
    popad
    ret
InputTheString ENDP



;-----------------------------------------------------------------------
DisplayMessage PROC
    ; Displays the encrypted or decrypted message.
    ; Receives: EDX points to the message
    ; Returns:  nothing
;-----------------------------------------------------------------------
    pushad
    call WriteString
    mov edx, OFFSET buffer  ; display the buffer
    call WriteString
    call Crlf
    call Crlf
    popad
    ret
DisplayMessage ENDP



;-----------------------------------------------------------------------
TranslateBuffer PROC
    ; Translates the string by exclusive-ORing each byte with encryption
    ; key byte.
    ; Receives: nothing
    ; Returns:  nothing
;-----------------------------------------------------------------------
```

```
    pushad
    mov ecx, bufSize     ; loop counter
    mov esi, 0           ; index 0 in buffer
L1:
    xor buffer[esi], KEY  ; translate a byte
    inc esi              ; point to next byte
    loop L1
    popad
    ret
TranslateBuffer ENDP
END main
```

OUTPUT:



Problem # 2: [CLO2]

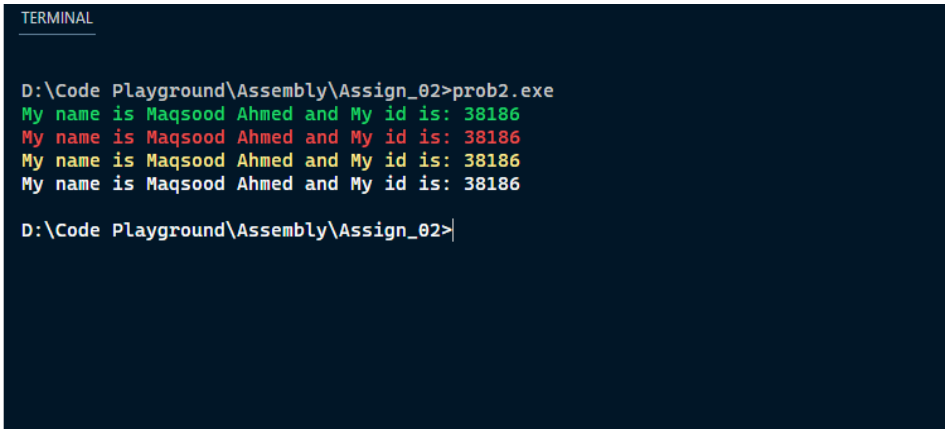
Write a program that displays the same string in four different colors, using a loop. Call the Set-TextColor procedure from the book's link library. Any colors may be chosen, but you may find it easiest to change the foreground color.

Source Code:

```
.686
.MODEL flat, stdcall
.STACK
INCLUDE Irvine32.inc

.data
   string BYTE "My name is Maqsood Ahmed and My id is: 38186", 0
   colors DWORD 0Ah, 0Ch, 0Eh, 0Fh  ; Blue, Red, Yellow, Bright White

.code
main PROC
   mov ecx, 4                ; Loop counter for 4 colors
   lea esi, colors           ; Load address of colors array
L1:
   mov eax, [esi]            ; Load current color from array
   push eax                  ; Push color onto stack
   call SetTextColor         ; Set the text color
   mov edx, OFFSET string
   call WriteString          ; Display the string
   call Crlf                 ; Move to the next line
   add esi, 4                ; Move to the next color
   loop L1
```

```
    exit
main ENDP
END main
```

OUTPUT:



Problem # 3: [CLO2]

Write a sequence of statements using indexed addressing that copies an element in a doubleword array to the previous position in the same array.

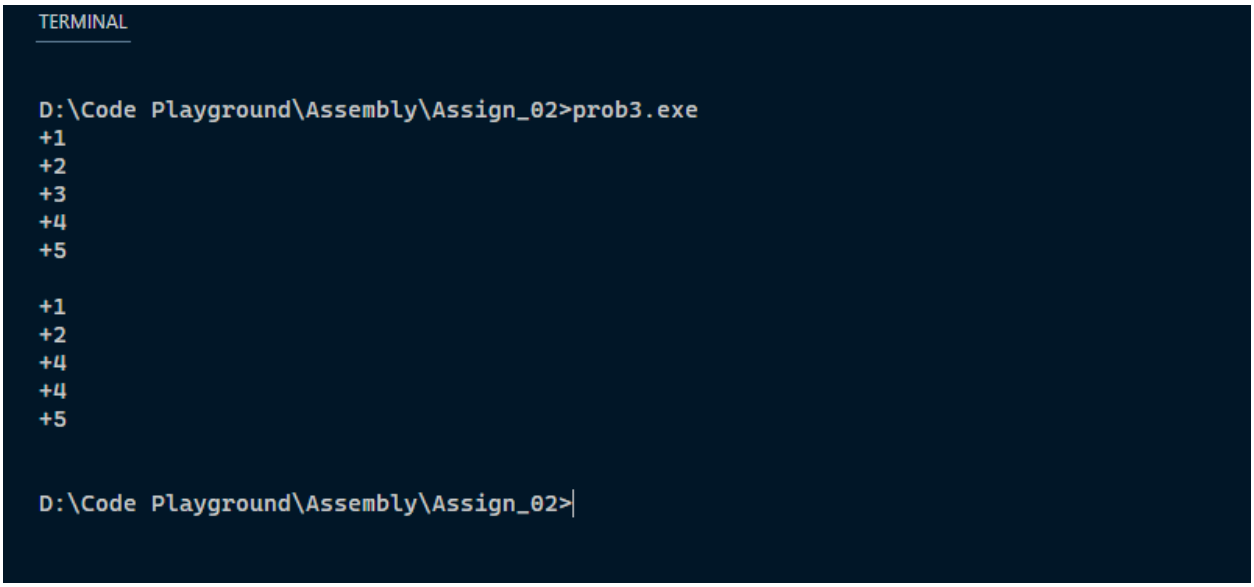
Source Code:

```
 .686
.MODEL flat, stdcall
.STACK
INCLUDE Irvine32.inc
.data
array DWORD 1, 2, 3, 4, 5
.code
main PROC
   ; Display the array before modification
   mov edx, OFFSET array
   call PrintArray

   ; Set up to copy the element at index 3 to index 2
   mov esi, OFFSET array     ; Load the address of the array into ESI
   mov ecx, 3                ; Index of the element we want to copy (zero-based)

   ; Move the value of array[ecx] into EAX
   mov eax, [esi + ecx * 4]

   ; Decrement ECX to point to the previous element
   dec ecx

   ; Copy the value from EAX to array[ecx]
```

```
    mov [esi + ecx * 4], eax

    ; Display the array after modification
    mov edx, OFFSET array
    call PrintArray

    exit
main ENDP

;-----------------------------------------------------------------------
PrintArray PROC
; Prints the contents of an array of DWORDs. EDX points to the array.
;-----------------------------------------------------------------------
    push esi
    push ecx
    mov esi, edx  ; ESI now points to the array
    mov ecx, 5    ; Number of elements in the array

L1:
    mov eax, [esi]
    call WriteInt
    call Crlf
    add esi, 4    ; Move to the next element
    loop L1

    ; Print a new line after the array
    call Crlf
```

```
    pop ecx
    pop esi
    ret
PrintArray ENDP
END main
```

OUTPUT:



The End