ALIYAH KHAET REGACHO

CS243 F1 LABORATORY HANDS-ON EXERCISES

1. EXER12.ASM

```
; Filename: EXER12.ASM
; Programmer Name: ALIYAH KHAET REGACHO
; Date: SEPTEMBER 20, 2024
; Description: This assembly language program will display multiple string
variables on a single line.

.MODEL SMALL
.STACK 200H
.DATA

    String1 DB 'Line1', '$' ; DB = Define Byte String1 variable and assign
'Line 1' value.
    String2 DB 'Line2', '$'
    String3 DB 'Line3', '$'

.CODE

ProgramStart:

    mov ax, @data
    mov ds, ax
    mov ah, 9               ; DOS print string function
    mov dx, OFFSET String1  ; 1st string to print
    int 21h                 ; invoke DOS to print string

    mov dx, OFFSET String2  ; 2nd string to print
    int 21h                 ; invoke DOS to print string

    mov dx, OFFSET String3  ; 3rd string to print
    int 21h                 ; invoke DOS to print string

    mov ah, 4Ch             ; DOS terminate program function
    int 21h                 ; invoke DOS to end program

END ProgramStart
```

2. EXER13.ASM

```
; Filename: EXER13.ASM
; Programmer Name: ALIYAH KHAET REGACHO
; Date: SEPTEMBER 20, 2024
; Description: This assembly language program will display multiple string
variables on separate lines.

.MODEL small
.STACK 200h
.DATA

    String1 DB 'Line1', 0dh, 0ah, '$'
    String2 DB 'Line2', 0dh, 0ah, '$'
    String3 DB 'Line3', 0dh, 0ah, '$'

    ; carriage return (ASCII code 0Dh) positions the cursor to the left
side of the current line of characters
    ; line feed (ASCII code 0Ah) moves the cursor down one line on the
output device.

.CODE

ProgramStart:

    mov ax, @data
    mov ds, ax
    mov ah, 9               ; DOS print string function
    mov dx, OFFSET String1  ; 1st string to print
    int 21h                 ; invoke DOS to print string

    mov dx, OFFSET String2  ; 2nd string to print
    int 21h                 ; invoke DOS to print string

    mov dx, OFFSET String3  ; 3rd string to print
    int 21h                 ; invoke DOS to print string

    mov ah, 4Ch             ; DOS terminate program function
    int 21h                 ; invoke DOS to end program

END ProgramStart
```

3. EXER14.ASM

```
; Filename: EXER14.ASM
; Programmer Name: ALIYAH KHAET REGACHO
; Date: SEPTEMBER 20, 2024
; Description: This assembly language program will get character input and
; display back character input.

.MODEL small
.STACK 200h
.DATA

    inputChar db ?      ; variable to store the input character
    inputMsg  db 'Enter a character: $'
    outputMsg db 'You entered: $'
    newline db 0Dh, 0Ah, '$'

.CODE

Main:

    ; initialize the data segment
    mov ax, @data
    mov ds, ax

    ; display the input prompt message
    mov dx, offset inputMsg
    mov ah, 09h
    int 21h

    ; read a character from the keyboard
    mov ah, 01h         ; function to read a character
    int 21h             ; call DOS interrupt
    mov inputChar, al   ; store the character in inputChar

    ; display newline
    mov dx, offset newline
    mov ah, 09h
    int 21h

    ; display the output message
    mov dx, offset outputMsg
    mov ah, 09h
    int 21h
```

```
    ; display the character back
    mov dl, inputChar   ; load the character to DL
    mov ah, 02h         ; function to display a character
    int 21h

    ; exit program
    mov ax, 4C00h       ; function to exit program
    int 21h

END Main
```

4. EXER15.ASM

```
; Filename: EXER15.ASM
; Programmer Name: ALIYAH KHAET REGACHO
; Date: SEPTEMBER 20, 2024
; Description: This assembly language program will continuously get
; character input and display back input until Enter key is pressed.

.MODEL small
.STACK 200h
.DATA
.CODE

ProgramStart:

    EchoLoop:

        mov ah, 1       ; DOS keyboard input function
        int 21h         ; get the next key
        cmp al, 13      ; was the key the Enter key?
        jz EchoDone     ; yes, so we're done echoing
        mov dl, al      ; put the character into DL
        mov ah, 2       ; DOS display output function
        int 21h         ; display the character
        jmp EchoLoop    ; echo the next character

    EchoDone:

        mov ah, 4ch   ; DOS terminate program function
        int 21h       ; terminate the program

END ProgramStart
```

5. EXER16.ASM

```
; Filename: EXER16.ASM
; Programmer Name: ALIYAH KHAET REGACHO
; Date: SEPTEMBER 20, 2024
; Description: This assembly language program will get character input
; and determine if input is 'y' or 'Y' to display good morning message
; else display good afternoon message.

.MODEL small
.STACK 100h
.DATA

    TimePrompt DB 'Is it after 12 noon (Y/N)?$'
    GoodMorningMessage LABEL BYTE
        DB 13, 10, 'Good morning, world!', 13, 10, '$'
    GoodAfternoonMessage LABEL BYTE
        DB 13, 10, 'Good afternoon, world!', 13, 10, '$'

.CODE

ProgramStart:

    mov ax, @data
    mov ds, ax                  ; set DS to point to data segment
    mov dx, OFFSET TimePrompt   ; point to the time prompt
    mov ah, 9                   ; DOS print string function
    int 21h                     ; display time prompt

    mov ah, 1                   ; DOS get character function
    int 21h                     ; get single character response

    cmp al, 'y'                 ; typed lowercase 'y' for after noon?
    jz IsAfternoon              ; jz = jump if zero. Yes, it's afternoon
    cmp al, 'Y'                 ; typed uppercase 'Y' for afternoon?
    jnz IsMorning               ; no, it's before noon

IsAfternoon:

    mov dx, OFFSET GoodAfternoonMessage ; point to the afternoon greeting
    jmp DisplayGreeting                 ; point to the afternoon greeting

IsMorning:

    mov dx, OFFSET GoodMorningMessage   ; point to before noon greeting
```

```
    jmp DisplayGreeting                ; point to the morning greeting

DisplayGreeting:

    mov ah, 9       ; DOS print string function
    int 21h         ; display the greeting

    mov ah, 4Ch     ; DOS terminate program function
    int 21h         ; terminate program

END ProgramStart
```

6. EXER17.ASM

```
; Filename: EXER17.ASM
; Programmer Name: ALIYAH KHAET REGACHO
; Date: SEPTEMBER 20, 2024
; Description: This assembly language program will get string input and
; display back string.

.MODEL small
.STACK 100h
.DATA

    MAXIMUM_STRING_LENGTH EQU 1000
    StringInput DB  MAXIMUM_STRING_LENGTH DUP (?)
    InputPrompt DB 'Enter text: $'

.CODE

ProgramStart:

    mov ax, @data
    mov ds, ax      ; set DS to point to the data segment

    mov dx, OFFSET InputPrompt
    mov ah, 9
    int 21h

    mov ah, 3fh                     ; DOS read from handle function
    mov bx, 0                       ; standard input handle
    mov cx, MAXIMUM_STRING_LENGTH   ; read up to maximum number of
characters

    mov dx, OFFSET StringInput      ; store the string here
    int 21h                         ; get the string

    and ax, ax                      ; were any character read?
    jz Done                         ; no, so you're done
    mov cx, ax                      ; put string length in CX where
                                    ; you can use it as a counter

    push cx                         ; save the string length
    mov bx, OFFSET StringInput

    pop cx                          ; get back the string length
    mov ah, 40h                     ; DOS write from handle function
```

```
    mov bx, 1                       ; standard output handle
    mov dx, OFFSET StringInput      ; prepare to print the string
    int 21h                         ; print the string

Done:

    mov ah, 4ch                     ; DOS terminate program function
    int 21h                         ; terminate the program
END ProgramStart
```

7. EXER18.ASM

```
; Filename: EXER18.ASM
; Programmer Name: ALIYAH KHAET REGACHO
; Date: SEPTEMBER 20, 2024
; Description: This assembly language program will get string input and
; display the reverse of the string.

.MODEL small
.STACK 100h
.DATA

    MAXIMUM_STRING_LENGTH EQU 1000
    StringToReverse DB MAXIMUM_STRING_LENGTH DUP (?)
    ReverseString   DB MAXIMUM_STRING_LENGTH DUP (?)

.CODE

ProgramStart:

    mov ax, @data
    mov ds, ax      ; set DS to point to the data segment
    mov ah, 3fh     ; DOS read from handle function
    mov bx, 0       ; standard input handle
    mov cx, MAXIMUM_STRING_LENGTH   ; read up to maximum number of
characters

    mov dx, OFFSET StringToReverse ; store the string here
    int 21h                        ; get the string
    and ax, ax                     ; were any characters read?
    jz Done                        ; no, so you're done
    mov cx, ax                     ; put string length in CX, where
                                   ; you can use it as a counter
    push cx                        ; save the string length
    mov bx, OFFSET StringToReverse
    mov si, OFFSET ReverseString
    add si, cx
    dec si                         ; point to the end of the
                                   ; reverse string buffer

ReverseLoop:

    mov al, [bx]    ; get the next character
    mov [si], al    ; store the characters in reverse order
    inc bx          ; point to next character
```

```
    dec si          ; point to previous location
                    ; in reverse buffer

    loop ReverseLoop   ; move next character, if any
    pop cx          ; get back the string length
    mov ah, 40h     ; DOS write from handle function
    mov bx, 1       ; standard output handle
    mov dx, OFFSET ReverseString ; print this string
    int 21h                      ; print the reversed string

Done:

    mov ah, 4ch     ; DOS terminate program function
    int 21h         ; terminate the program
END ProgramStart
```

8. EXER19.ASM

```
; Filename: EXER19.ASM
; Programmer Name: ALIYAH KHAET REGACHO
; Date: SEPTEMBER 20, 2024
; Description: This assembly language program will get 3 character inputs.
; Then, it will display each character on its own line.

.MODEL small
.STACK 200h
.DATA

    Prompt1 DB 'Enter first character: $'
    Prompt2 DB 'Enter second character: $'
    Prompt3 DB 'Enter third character: $'

    Output1 DB 'The first character is $'
    Output2 DB 'The second character is $'
    Output3 DB 'The third character is $'

    newline DB 0Dh, 0Ah, '$'
    period  DB '.$'

    Char1 DB ?
    Char2 DB ?
    Char3 DB ?

.CODE

start:

    mov ax, @data
    mov ds, ax

    ; Prompt for 1st char
    mov ah, 09h
    lea dx, Prompt1
    int 21h

    ; Get 1st char
    mov ah, 01h
    int 21h
    mov Char1, al

    ; Output newline
```

```
    mov ah, 09h
    lea dx, newline
    int 21h

    ; Prompt for 2nd char
    mov ah, 09h
    lea dx, Prompt2
    int 21h

    ; Get 2nd char
    mov ah, 01h
    int 21h
    mov Char2, al

    ; Output newline
    mov ah, 09h
    lea dx, newline
    int 21h

    ; Prompt for 3rd char
    mov ah, 09h
    lea dx, Prompt3
    int 21h

    ; Get 3rd char
    mov ah, 01h
    int 21h
    mov Char3, al

    ; Output newline
    mov ah, 09h
    lea dx, newline
    int 21h

    ; Display 1st char
    mov ah, 09h
    lea dx, Output1
    int 21h

    mov ah, 02h
    mov dl, Char1
    int 21h

    mov ah, 09h
    lea dx, period
```

```
    int 21h

    ; Output newline
    mov ah, 09h
    lea dx, newline
    int 21h

    ; Display 2nd char
    mov ah, 09h
    lea dx, Output2
    int 21h

    mov ah, 02h
    mov dl, Char2
    int 21h

    mov ah, 09h
    lea dx, period
    int 21h

    ; Output newline
    mov ah, 09h
    lea dx, newline
    int 21h

    ; Display 3rd char
    mov ah, 09h
    lea dx, Output3
    int 21h

    mov ah, 02h
    mov dl, Char3
    int 21h

    mov ah, 09h
    lea dx, period
    int 21h

end start
```

9. EXER20.ASM

```
; Filename: EXER20.ASM
; Programmer Name: ALIYAH KHAET REGACHO
; Date: SEPTEMBER 20, 2024
; Description: This assembly language program will ask the user to input a
character.
; If it is an A/a, it will display the message "Yes, you have entered letter
A/a."
; If not, it will display the message "No, you have not entered letter A. You
entered character _."

.model small
.stack 100h
.data

    ; Input Prompt Display
    CharPrompt db 'Enter a character: $'
    YesMsg db 'Yes, you have entered letter $'
    NoMsg db 'No, you have not entered letter A. You entered character $'

    newline db 0Dh, 0Ah, '$'
    period db '.', '$'

    CharInput db ? ; To store the input

.code

start:

    mov ax, @data
    mov ds, ax

    ; Output the prompt
    mov ah, 09h
    lea dx, CharPrompt
    int 21h

    ; Store the character inputted
    mov ah, 01h
    int 21h
    mov CharInput, al

    ; Newline
    mov ah, 09h
```

```
    lea dx, newline
    int 21h

    ; Check if the character is 'A'
    cmp CharInput, 'A'
    je PrintYesMsg ; If yes, jump to PrintYesMsg
    ; If not, the PrintNoMsg will occur
    cmp CharInput, 'a'
    je PrintYesMsg

PrintNoMsg:

    ; Output the corresponding line
    mov ah, 09h
    lea dx, NoMsg
    int 21h

    ; Output the input
    mov ah, 02h
    mov dl, CharInput
    int 21h

    ; Period
    mov ah, 09h
    lea dx, period
    int 21h

    ; Newline
    mov ah, 09h
    lea dx, newline
    int 21h

    jmp ExitProgram

PrintYesMsg:

    ; Output the corresponding line
    mov ah, 09h
    lea dx, YesMsg
    int 21h

    ; Output the input
    mov ah, 02h
    mov dl, CharInput
    int 21h
```

```
    ; Period
    mov ah, 09h
    lea dx, period
    int 21h

    ; Newline
    mov ah, 09h
    lea dx, newline
    int 21h

ExitProgram:

    mov ah, 4Ch
    int 21h

end start
```

10. EXER21.ASM

```
; Filename: EXER21.ASM
; Programmer Name: ALIYAH KHAET REGACHO
; Date: SEPTEMBER 20, 2024
; Description: This assembly language program will ask for your first name,
middle name, and
; family name. Then display "Hello, FIRST NAME MIDDLE NAME FAMILY NAME!"

.MODEL SMALL
.STACK 100h
.DATA

    line1 db 'REGISTRATION FORM', 0Ah, '$'
    line2 db 'Enter First Name: $'
    line3 db 'Enter Middle Name: $'
    line4 db 'Enter Last Name: $'
    exclam db '!$'

    max_len EQU 1000

    first db max_len dup(?)
    middle db max_len dup (?)
    last db max_len dup (?)

    msg db 'Hello, $'

.CODE

ProgramStart:

    mov ax, @data
    mov ds, ax

    lea dx, line1
    call printString

    lea dx, line2
    call printString

    lea dx, first
    mov cx, max_len
    call getString

    push ax
```

```
lea dx, line3
call printString

lea dx, middle
mov cx, max_len
call getString

pop dx
push ax
push dx

lea dx, line4
call printString

lea dx, last
mov cx, max_len
call getString

pop dx
pop cx
push ax
push cx
push dx

mov dx, OFFSET msg
call printString

mov dx, OFFSET first
pop cx
call printNumString

mov dx, OFFSET middle
pop cx
call printNumString

mov dx, OFFSET last
pop cx
call printNumString

mov dx, OFFSET exclam
pop cx
call printString

call endline
```

```
    int 27h

getString:

    push bx
    mov ah, 3Fh
    mov bx, 0
    mov cx, max_len
    int 21h

    pop bx
    ret

printNumString:

    push ax
    push bx
    mov ah, 40h
    add cx, -2
    mov bx, 1
    int 21h

    mov ah, 02h
    mov dl, ' '
    int 21h

    pop bx
    pop ax
    ret

printString:

    push ax
    mov ah, 09h
    int 21h
    pop ax
    ret

endline:

    push ax
    push bx
    mov ah, 02h
    mov dl, 0Ah
    int 21h
```

```
    pop bx
    pop ax
    ret

END ProgramStart
```

11. EXER22.ASM

```
; Filename: EXER22.ASM
; Programmer Name: Aliyah Khaet Regacho
; Date: September 20, 2024
; Description: Create a College Enrollment Form. Get user input.
; Display a summary of the inputs.
; Include the necessary documentation as comments in your code.

.model small
.stack 200h
.data
 prompt1 db 'Cebu Institute of Technology - University', 13,10
 db 'STUDENT ENROLLMENT FORM', 13, 10, 10
 db 'Please enter the following information:', 13, 10, '$'

 idPrompt db 'Student ID Number: $'
 firstNamePrompt db 13, 10,'First Name: $'
 middleNamePrompt db 13, 10,'Middle Name: $'
 lastNamePrompt db 13, 10,'Last Name: $'
 addressPrompt db 13,10,'Address: $'
 coursePrompt db 13,10,'Course: $'
 yearPrompt db 13,10,'Year: $'
 bdayMonthPrompt db 13,10,'Birthday Month: $'
 bdayDayPrompt db 13,10,'Birthday Day: $'
 bdayYearPrompt db 13,10,'Birthday Year: $'
 emailPrompt db 13,10,'Email Address: $'

 ; Variables to store inputs
 idInput db 20, 20 dup(?)
 firstNameInput db 20, 20 dup(?)
 middleNameInput db 20, 20 dup(?)
 lastNameInput db 20, 20 dup(?)
 addressInput db 100, 100 dup(?)
 courseInput db 20, 20 dup(?)
 yearInput db 10, 10 dup(?)
 bdayMonthInput db 15, 15 dup(?)
 bdayDayInput db 10, 10 dup(?)
 bdayYearInput db 10, 10 dup(?)
 emailInput db 50, 50 dup(?)

 prompt2 db 13,10,10,'SUMMARY',13,10
 db 'Please check if all information are correct.',13,10,10 , '$'

 idOutput db 'ID Number: $'
```

```
 fullNameOutput db 13,10,'Full Name: $'
 addressOutput db 13,10,'Address: $'
 courseYearOutput db 13,10,'Course & Year: $'
 bdayOutput db 13,10,'Birthday: $'
 emailOutput db 13,10,'Email Address: $'

 prompt3 db 13,10,10,'Thank you for enrolling at CIT-U.',13,10
 db 'Copyright 2024',13,10
 db 'Programmer: ALIYAH KHAET REGACHO' , '$'

.code

; For printing

print PROC
 mov ah, 09h
 int 21h
 ret
print ENDP

; For scanning the input

scan PROC
 mov ah, 0AH
 int 21h
 ret
scan ENDP

; For displaying

displaychar PROC
 mov dl, [si]
 mov ah, 02h
 int 21h
 inc si
 ret
displaychar ENDP

; Main
 main:
 mov ax, @data
 mov ds, ax

; Output prompt 1
```

```
 lea dx, prompt1
 call print

; Output ID Prompt and scan its input

 lea dx, idPrompt
 call print
 lea dx, idInput
 call scan

; Output First Name Prompt and scan its input

 lea dx, firstNamePrompt
 call print
 lea dx, firstNameInput
 call scan

; Output Middle Name Prompt and scan its input

 lea dx, middleNamePrompt
 call print
 lea dx, middleNameInput
 call scan

; Output Last Name Prompt and scan its input

 lea dx, lastNamePrompt
 call print
 lea dx, lastNameInput
 call scan

; Output Address Prompt and scan its input

 lea dx, addressPrompt
 call print
 lea dx, addressInput
 call scan

; Output Course Prompt and scan its input

 lea dx, coursePrompt
 call print
 lea dx, courseInput
 call scan
```

```
; Output Year Prompt and scan its input

 lea dx, yearPrompt
 call print
 lea dx, yearInput
 call scan

; Output Birthday Prompt and scan its input

 lea dx, bdayMonthPrompt
 call print
 lea dx, bdayMonthInput
 call scan

 lea dx, bdayDayPrompt
 call print
 lea dx, bdayDayInput
 call scan

 lea dx, bdayYearPrompt
 call print
 lea dx, bdayYearInput
 call scan

; Output Email Address Prompt and scan its input

 lea dx, emailPrompt
 call print
 lea dx, emailInput
 call scan

; Output "SUMMARY..."

 lea dx, prompt2
 call print

; Print ID

 lea dx, idOutput
 call print
 lea si, idInput + 2
 mov cl, [idInput + 1]
 DisplayId:
 call displaychar
 loop DisplayId
```

```
; Print Full Name

 lea dx, fullNameOutput
 call print
 lea si, lastNameInput + 2
 mov cl, [lastNameInput+1]
 DisplayLastName:
 call displaychar
 loop DisplayLastName
 mov dl, ','
 mov ah, 02h
 int 21h
 mov dl, ' '
 mov ah, 02h
 int 21h


 lea si, firstNameInput + 2
 mov cl, [firstNameInput+1]
 DisplayFirstName:
 call displaychar
 loop DisplayFirstName
 mov dl, ' '
 mov ah, 02h
 int 21h


 lea si, middleNameInput + 2
 mov cl, [middleNameInput+1]
 DisplayMiddleName:
 call displaychar
 loop DisplayMiddleName

; Print Address

 lea dx, addressOutput
 call print
 lea si, addressInput + 2
 mov cl, [addressInput+1]
 DisplayAddress:
 call displaychar
 loop DisplayAddress

; Print Course and Year
```

```
 lea dx, courseYearOutput
 call print
 lea si, courseInput + 2
 mov cl, [courseInput +1]
 DisplayCourse:
 call displaychar
 loop DisplayCourse
 mov dl, ' '
 mov ah, 02h
 int 21h


 lea si, yearInput + 2
 mov cl, [yearInput + 1]
 DisplayYear:
 call displaychar
 loop DisplayYear

; Print Birthday

 lea dx, bdayOutput
 call print
 lea si, bdayMonthInput+2
 mov cl, [bdayMonthInput+1]
 DisplayBdayMonth:
 call displaychar
 loop DisplayBdayMonth
 mov dl, ' '
 mov ah, 02h
 int 21h

 lea si, bdayDayInput+2
 mov cl, [bdayDayInput+1]
 DisplayBdayDay:
 call displaychar
 loop DisplayBdayDay
 mov dl, ','
 mov ah, 02h
 int 21h
 mov dl, ' '
 mov ah, 02h
 int 21h

 lea si, bdayYearInput +2
```

```
 mov cl, [bdayYearInput+1]
 DisplayBdayYear:
 call displaychar
 loop DisplayBdayYear

; Print Email Address

 lea dx, emailOutput
 call print
 lea si, emailInput+2
 mov cl, [emailInput+1]
 DisplayEmail:
 call displaychar
 loop DisplayEmail

; Print Last Part

 lea dx, prompt3
 call print
 mov ax, 4C00h
 int 21h

end main
```

12. EXER23.ASM

```
;Filename: EXER23.ASM
;Programmer Name: ALIYAH KHAET REGACHO
;Date: September 20, 2024
;Description:  Create an Automated Teller Machine (ATM) main menu screen.
;              Ask the user to input a number corresponding to an item in
the main menu.
;              Print back the number input by user and the corresponding
;              transaction.

.model small
.stack 100h
.data
    prompt         db 'Cebu Institute of Technology - University', 13,10
                   db 'Automated Teller Machine',13,10,10
                   db 'Main Menu',13,10
                   db '1 Balance Inquiry',13,10
                   db '2 Withdrawal',13,10
                   db '3 Deposit',13,10
                   db '4 Transfer',13,10
                   db '5 Bills Payment',13,10
                   db '6 Change Pin',13,10
                   db '7 Exit',13,10
                   db 'Enter number of your choice: ', '$'

    choice         db ?

    ending         db 'Thank you for banking with us!',13,10
                   db 'Copyright 2024',13,10
                   db 'Programmer: ALIYAH KHAET REGACHO' , '$'

    balanceInquiry db 13,10,10,'You have chosen 1 Balance Inquiry.',
13,10,10, '$'
    withdrawal     db 13,10,10,'You have chosen 2 Withdrawal.', 13,10,10,
'$'
    deposit        db 13,10,10,'You have chosen 3 Deposit.', 13,10,10, '$'
    transfer       db 13,10,10,'You have chosen 4 Transfer.', 13,10,10,
'$'
    billsPayment   db 13,10,10,'You have chosen 5 Bills Payment.',
13,10,10, '$'
    changePin      db 13,10,10,'You have chosen 6 Change Pin.', 13,10,10,
'$'
    exitMsg        db 13,10,10,'You have chosen 7 Exit.', 13,10,10,'$'
.code
```

```
main:
    mov ax, @data
    mov ds, ax

    mov ah, 09h
    lea dx, prompt
    int 21h

; Ask for user input
    mov ah, 01h
    int 21h
    sub al, '0'
    mov choice, al

; Display the corresponding transaction
    cmp choice, 1
    je  displayBalanceInquiry
    cmp choice, 2
    je  displayWithdrawal
    cmp choice, 3
    je  displayDeposit
    cmp choice, 4
    je  displayTransfer
    cmp choice, 5
    je  displayBillsPayment
    cmp choice, 6
    je  displayChangePin
    cmp choice, 7
    je  displayExit

; If invalid choice
    mov dx, offset ending
    mov ah, 09h
    int 21h
    jmp exitProgram

displayBalanceInquiry:
    lea dx, balanceInquiry
    jmp printOutput

displayWithdrawal:
    lea dx, withdrawal
    jmp printOutput

displayDeposit:
```

```
        lea dx, deposit
        jmp printOutput

    displayTransfer:
        lea dx, transfer
        jmp printOutput

    displayBillsPayment:
        lea dx, billsPayment
        jmp printOutput

    displayChangePin:
        lea dx, changePin
        jmp printOutput

    displayExit:
        lea dx, exitMsg
        jmp printOutput

    printOutput:
        mov ah, 09h
        int 21h
        jmp exitProgram

    exitProgram:
        mov ah, 09h
        lea dx, ending
        int 21h

        mov ah, 4Ch
        int 21h

end main
```

13. EXER24.ASM

```
; Filename: EXER24.ASM
; Programmer Name: ALIYAH KHAET REGACHO
; Date: September 20, 2024
; Description: Edit your Laboratory Prelim Hands-on Exam. Before the form
displays,
;              ask the user to input all the needed data. Afterwards,
display the
;              form with all the data entered by the user.

.MODEL small
.STACK 100h
.DATA
menu    db
'
', 13, 10
        db '                    Cebu Institute of Technology -
University               ', 13, 10
        db '                         VEHICLE STICKER APPLICATION
FORM                  ', 13, 10
        db '                           Please fill out the form
below.                     ', 13, 10, 10
        db '   Personnel Type:                      V   Vehicle Sticker
Type:           V ', 13, 10, 10
        db '   Name of Applicant/Driver:                ID
Number:                        ', 13, 10, 10
        db '   Mobile
Number:                             Address:                          ', 13,
10, 10
        db '   Vehicle Make(s)/Brand:                   Plate
Number:                     ', 13, 10, 10
        db '   Vehicle Color:                       V   Vehicle
Type:                    V ', 13, 10, 10, 10
        db
'                                    SUBMIT                         ', 13,
10, 10
        db '                            Copyright 2024 ALIYAH KHAET
REGACHO      ', 13, 10, 10, 10
        db '                                       Thank You!            ', 13,
10, '$'

personnelType db 9, 0, 9 dup(0)
applicantName db 14, 0, 14 dup(0)
mobileNumber db 12, 0, 12 dup(0)
```

```
carMake db 12, 0, 12 dup(0)
carColor db 9, 0, 9 dup(0)
stickerType db 9, 0, 9 dup(0)
idNum db 12, 0, 12 dup(0)
address db 12, 0, 12 dup(0)
plateNumber db 12, 0, 12 dup(0)
carType db 9, 0, 9 dup(0)

row0 db 'Cebu Institute of Technology - University', 0dh, 0ah, '$'
row1 db 'VEHICLE STICKER APPLICATION FORM', 0dh, 0ah, '$'
row2 db 'Please enter the needed information:', 0dh, 0ah, '$'
row3 db 'Personnel Type: $'
row4 db 'Name of Applicant/Driver: $'
row5 db 'Mobile Number: $'
row6 db 'Vehicle Make(s)/Brand: $'
row7 db 'Vehicle Color: $'
row8 db 'Vehicle Sticker Type: $'
row9 db 'ID Number: $'
rowA db 'Address: $'
rowB db 'Plate Number: $'
rowC db 'Vehicle Type: $'

nxt DB 0dh, 0ah, '$'

.CODE

PrintString:
    MOV ah, 09h
    INT 21h
    RET

NextL:
    lea dx, nxt
    MOV ah, 09h
    INT 21h
    RET


inputForm proc

    lea dx, row0
    call PrintString

    lea dx, row1
    call PrintString
```

```
lea dx, row2
call PrintString
call NextL

lea dx, row3
call PrintString
lea dx, personnelType
mov ah, 0ah
int 21h
CALL NextL

lea dx, row4
call PrintString
lea dx, applicantName
mov ah, 0ah
int 21h
CALL NextL

lea dx, row5
call PrintString
lea dx, mobileNumber
mov ah, 0ah
int 21h
CALL NextL

lea dx, row6
call PrintString
lea dx, carMake
mov ah, 0ah
int 21h
CALL NextL

lea dx, row7
call PrintString
lea dx, carColor
mov ah, 0ah
int 21h
CALL NextL

lea dx, row8
call PrintString
lea dx, stickerType
mov ah, 0ah
int 21h
```

```
    CALL NextL

    lea dx, row9
    call PrintString
    lea dx, idNum
    mov ah, 0ah
    int 21h
    CALL NextL

    lea dx, rowA
    call PrintString
    lea dx, address
    mov ah, 0ah
    int 21h
    CALL NextL

    lea dx, rowB
    call PrintString
    lea dx, plateNumber
    mov ah, 0ah
    int 21h
    CALL NextL

    lea dx, rowC
    call PrintString
    lea dx, carType
    mov ah, 0ah
    int 21h
    CALL NextL

    RET

inputForm endp


main proc

    mov ax, @data ; db setup
    mov ds, ax

    call inputForm

    mov ah, 00h
    mov al, 03h   ; display setup
    int 10h
```

```
    call printForm

    ; personnelType
    mov ah, 02h          ; Function to set cursor position
    mov bh, 00h          ; Page number (0 for standard screen)
    mov dh, 5            ; Row (0-based)
    mov dl, 029          ; Column (0-based)
    int 10h              ; Call BIOS interrupt

; Print Personnel
    lea si, personnelType + 2
    mov cl, [personnelType+1]
PrintPersonnel:
    mov dl, [si]
    cmp dl, 0dh
    je DonePersonnel
    mov ah, 02h
    int 21h
    inc si
    loop PrintPersonnel
DonePersonnel:
    CALL NextL


    ; applicantName
    mov ah, 02h
    mov bh, 00h
    mov dh, 7
    mov dl, 029
    int 10h

; Print Name
    lea si, applicantName + 2
    mov cl, [applicantName+1]
PrintName:
    mov dl, [si]
    cmp dl, 0dh
    je DoneName
    mov ah, 02h
    int 21h
    inc si
    loop PrintName
DoneName:
    CALL NextL
```

```
    ; mobileNumber
    mov ah, 02h          ; Function to set cursor position
    mov bh, 00h          ; Page number (0 for standard screen)
    mov dh, 9            ; Row (0-based)
    mov dl, 029          ; Column (0-based)
    int 10h              ; Call BIOS interrupt

; Print Mobile Number
    lea si, mobileNumber + 2
    mov cl, [mobileNumber+1]
PrintNumber:
    mov dl, [si]
    cmp dl, 0dh
    je DoneNumber
    mov ah, 02h
    int 21h
    inc si
    loop PrintNumber
DoneNumber:
    CALL NextL

    ; carMake
    mov ah, 02h          ; Function to set cursor position
    mov bh, 00h          ; Page number (0 for standard screen)
    mov dh, 11           ; Row (0-based)
    mov dl, 029          ; Column (0-based)
    int 10h              ; Call BIOS interrupt

    ; Print car make
    lea si, carMake + 2
    mov cl, [carMake+1]
PrintMake:
    mov dl, [si]
    cmp dl, 0dh
    je DoneMake
    mov ah, 02h
    int 21h
    inc si
    loop PrintMake
DoneMake:
    CALL NextL

    ; carColor
    mov ah, 02h          ; Function to set cursor position
    mov bh, 00h          ; Page number (0 for standard screen)
```

```
    mov dh, 13           ; Row (0-based)
    mov dl, 029          ; Column (0-based)
    int 10h              ; Call BIOS interrupt

    ; Print car color
    lea si, carColor + 2
    mov cl, [carColor+1]
PrintColor:
    mov dl, [si]
    cmp dl, 0dh
    je DoneColor
    mov ah, 02h
    int 21h
    inc si
    loop PrintColor
DoneColor:
    CALL NextL

    ; stickerType
    mov ah, 02h          ; Function to set cursor position
    mov bh, 00h          ; Page number (0 for standard screen)
    mov dh, 5            ; Row (0-based)
    mov dl, 066          ; Column (0-based)
    int 10h              ; Call BIOS interrupt

    ; Print sticker type
    lea si, stickerType + 2
    mov cl, [stickerType+1]
PrintSticker:
    mov dl, [si]
    cmp dl, 0dh
    je DoneSticker
    mov ah, 02h
    int 21h
    inc si
    loop PrintSticker
DoneSticker:
    CALL NextL

    ; idNum
    mov ah, 02h          ; Function to set cursor position
    mov bh, 00h          ; Page number (0 for standard screen)
    mov dh, 7            ; Row (0-based)
    mov dl, 066          ; Column (0-based)
    int 10h              ; Call BIOS interrupt
```

```
    ; Print ID Number
    lea si, idNum + 2
    mov cl, [idNum+1]
PrintID:
    mov dl, [si]
    cmp dl, 0dh
    je DoneID
    mov ah, 02h
    int 21h
    inc si
    loop PrintID
DoneID:
    CALL NextL

    ; address
    mov ah, 02h          ; Function to set cursor position
    mov bh, 00h          ; Page number (0 for standard screen)
    mov dh, 9            ; Row (0-based)
    mov dl, 066          ; Column (0-based)
    int 10h              ; Call BIOS interrupt

; Print Address
    lea si, address + 2
    mov cl, [address+1]
PrintAddress:
    mov dl, [si]
    cmp dl, 0dh
    je DoneAddress
    mov ah, 02h
    int 21h
    inc si
    loop PrintAddress
DoneAddress:
    CALL NextL

    ; plateNumber
    mov ah, 02h          ; Function to set cursor position
    mov bh, 00h          ; Page number (0 for standard screen)
    mov dh, 11           ; Row (0-based)
    mov dl, 066          ; Column (0-based)
    int 10h              ; Call BIOS interrupt

    ; Print Plate Number
    lea si, plateNumber + 2
```

```
    mov cl, [plateNumber+1]
PrintPlate:
    mov dl, [si]
    cmp dl, 0dh
    je DonePlate
    mov ah, 02h
    int 21h
    inc si
    loop PrintPlate
DonePlate:
    CALL NextL

    ; carType
    mov ah, 02h          ; Function to set cursor position
    mov bh, 00h          ; Page number (0 for standard screen)
    mov dh, 13           ; Row (0-based)
    mov dl, 066          ; Column (0-based)
    int 10h              ; Call BIOS interrupt

    ; Print Car type
    lea si, carType + 2
    mov cl, [carType+1]
PrintType:
    mov dl, [si]
    cmp dl, 0dh
    je DoneType
    mov ah, 02h
    int 21h
    inc si
    loop PrintType
DoneType:

    mov ah, 02h          ; Function to set cursor position
    mov bh, 00h          ; Page number (0 for standard screen)
    mov dh, 21           ; Row (0-based)
    mov dl, 80           ; Column (0-based)
    int 10h              ; Call BIOS interrupt

    mov ax, 4C00h ; return 0
    int 21h

main endp

printForm proc
```

```
    xor al, al

    ; Grey BG
    mov ah, 06h
    mov ch, 1 ; row start
    mov cl, 2 ; col start
    mov dh, 19  ; row end
    mov dl, 78  ; col end
    mov bh, 70h ; grey bg with black text
    int 10h

    ; header red bg
    mov ah, 06h
    mov ch, 1
    mov cl, 3
    mov dh, 3
    mov dl, 77
    mov bh, 4fh ; red bg with white text
    int 10h
    ; header yellow blink line
    mov ah, 06h
    mov ch, 3
    mov cl, 3
    mov dh, 3
    mov dl, 77
    mov bh, 0ceh ; red bg with yellow blinking text
    int 10h


    ; black bg left
    mov ah, 06h
    mov ch, 5
    mov cl, 29
    mov dh, 5
    mov dl, 41
    mov bh, 0fh ; black bg with white text
    int 10h
    ; black bg right
    mov ah, 06h
    mov ch, 5
    mov cl, 66
    mov dh, 5
    mov dl, 77
    mov bh, 0fh ; black bg with white text
    int 10h
```

```
 ; black bg left
 mov ah, 06h
 mov ch, 7
 mov cl, 29
 mov dh, 7
 mov dl, 41
 mov bh, 0fh ; black bg with white text
 int 10h
 ; black bg right
 mov ah, 06h
 mov ch, 7
 mov cl, 66
 mov dh, 7
 mov dl, 77
 mov bh, 0fh ; black bg with white text
 int 10h

; black bg left
 mov ah, 06h
 mov ch, 9
 mov cl, 29
 mov dh, 9
 mov dl, 41
 mov bh, 0fh ; black bg with white text
 int 10h
 ; black bg right
 mov ah, 06h
 mov ch, 9
 mov cl, 66
 mov dh, 9
 mov dl, 77
 mov bh, 0fh ; black bg with white text
 int 10h

 ; black bg left
 mov ah, 06h
 mov ch, 9
 mov cl, 29
 mov dh, 9
 mov dl, 41
 mov bh, 0fh ; black bg with white text
 int 10h
 ; black bg right
 mov ah, 06h
```

```
    mov ch, 9
    mov cl, 66
    mov dh, 9
    mov dl, 77
    mov bh, 0fh ; black bg with white text
    int 10h

    ; black bg left
    mov ah, 06h
    mov ch, 11
    mov cl, 29
    mov dh, 11
    mov dl, 41
    mov bh, 0fh ; black bg with white text
    int 10h
    ; black bg right
    mov ah, 06h
    mov ch, 11
    mov cl, 66
    mov dh, 11
    mov dl, 77
    mov bh, 0fh ; black bg with white text
    int 10h

    ; black bg left
    mov ah, 06h
    mov ch, 13
    mov cl, 29
    mov dh, 13
    mov dl, 41
    mov bh, 0fh ; black bg with white text
    int 10h
    ; black bg right
    mov ah, 06h
    mov ch, 13
    mov cl, 66
    mov dh, 13
    mov dl, 77
    mov bh, 0fh ; black bg with white text
    int 10h

    ; Upper Left V
    mov ah, 06h
    mov ch, 5
    mov cl, 39
```

```
    mov dh, 5
    mov dl, 41
    mov bh, 4fh ; red bg with white text
    int 10h
    ; Upper Right V
    mov ah, 06h
    mov ch, 5
    mov cl, 75
    mov dh, 5
    mov dl, 77
    mov bh, 4fh ; red bg with white text
    int 10h
    ; Lower Right V
    mov ah, 06h
    mov ch, 13
    mov cl, 39
    mov dh, 13
    mov dl, 41
    mov bh, 4fh ; red bg with white text
    int 10h
    ; Lower Left V
    mov ah, 06h
    mov ch, 13
    mov cl, 75
    mov dh, 13
    mov dl, 77
    mov bh, 4fh ; red bg with white text
    int 10h


    ; Red BG Submit Button
    mov ah, 06h
    mov ch, 16
    mov cl, 35
    mov dh, 16
    mov dl, 42
    mov bh, 4eh ; red bg with yellow text
    int 10h

    ; Blinking Yellow thankyou
    mov ah, 06h
    mov ch, 21
    mov cl, 0
    mov dh, 21
    mov dl, 78
```

```
    mov bh, 8eh
    int 10h

    ; print
    mov ah, 09h
    mov dx, offset menu
    int 21h

    ret

printForm endp


end main
```