BSAI - LAB TASK

1. Write a program to calculate the area of various geometric shapes (circle, triangle, and rectangle) based on user input.

Code:

```
.model small
.stack 100h
.data
   choice_msg db 'Choose a shape:',13,10
            db '1. Circle',13,10
            db '2. Triangle',13,10
            db '3. Rectangle',13,10
            db 'Enter your choice (1/2/3): $'
   radius_msg db 'Enter radius of circle (in cm): $'
   base_msg db 'Enter base of triangle (in cm): $'
   height_msg db 'Enter height of triangle (in cm): $'
   length_msg db 'Enter length of rectangle (in cm): $'
   width_msg db 'Enter width of rectangle (in cm): $'
   area_msg db 'Area = $'
.code
   main proc
      mov ax, @data
      mov ds, ax
      mov ah, 09h
      lea dx, choice_msg
```

```
    int 21h
input_choice:
    mov ah, 01h
    int 21h
    sub al, 30h   ; Convert ASCII to integer
    cmp al, '1'
    je calculate_circle
    cmp al, '2'
    je calculate_triangle
    cmp al, '3'
    je calculate_rectangle
    jmp input_choice
calculate_circle:
    mov ah, 09h
    lea dx, radius_msg
    int 21h
    call get_input
    mov bx, ax
    mov ax, bx
    mul bx
    mov cx, 314     ; Value of pi (3.14) multiplied by 100 for precision
    mul cx
    mov bx, 100
    div bx         ; Divide by 100 to get the final result
    call display_area
    jmp exit_program
calculate_triangle:
    mov ah, 09h
    lea dx, base_msg
    int 21h
    call get_input
    mov bx, ax
    mov ah, 09h
    lea dx, height_msg
    int 21h
    call get_input
    add ax, bx
    mul bx
    mov ax, ax
    mov bx, 2
    div bx
    call display_area
    jmp exit_program
calculate_rectangle:
    mov ah, 09h
    lea dx, length_msg
    int 21h
    call get_input
    mov bx, ax
    mov ah, 09h
    lea dx, width_msg
    int 21h
    call get_input
    mul bx
    call display_area
    jmp exit_program
get_input:
    mov ah, 01h
    int 21h
    sub al, 30h   ; Convert ASCII to integer
    mov bl, al
    mov ax, 0
input_loop:
    mov ah, 01h
    int 21h
    cmp al, 13    ; Check for carriage return
    je input_done
    sub al, 30h   ; Convert ASCII to integer
    mov cl, al
    mov al, bl
    mul bx
    mov bl, 10
    add ax, cx
    jmp input_loop
input_done:
        ret
    display_area:
        mov ah, 09h
        lea dx, area_msg
        int 21h
        mov si, 10
    convert_and_display:
        xor dx, dx
        div si
        add dl, 30h
        push dx
        cmp ax, 0
        jz display_loop
        jmp convert_and_display
    display_loop:
        pop dx
        mov ah, 02h
        int 21h
        loop display_loop
        jmp exit_program
    exit_program:
        mov ax, 4C00h
        int 21h
    main endp
end main
```

2. Design an assembly language program to convert temperature from Celsius to Fahrenheit or vice versa.

Code:

```
.model small
.stack 100h
.data
   choice_msg db 'Choose conversion:',13,10
              db '1. Celsius to Fahrenheit',13,10
              db '2. Fahrenheit to Celsius',13,10
              db 'Enter your choice (1/2): $'
   temp_msg db 'Enter temperature: $'
   result_msg db 'Result = $'
.code
   main proc
      mov ax, @data
      mov ds, ax
      mov ah, 09h
      lea dx, choice_msg
      int 21h
   input_choice:
      mov ah, 01h
      int 21h
      sub al, 30h   ; Convert ASCII to integer
      cmp al, '1'
```

```
    je celsius_to_fahrenheit
    cmp al, '2'
    je fahrenheit_to_celsius
    jmp input_choice
celsius_to_fahrenheit:
    mov ah, 09h
    lea dx, temp_msg
    int 21h
    call get_input
    mov bx, ax
    mov ax, bx
    imul bx, 9
    mov cx, 5
    idiv cx
    add ax, 32
    call display_result
    jmp exit_program
fahrenheit_to_celsius:
    mov ah, 09h
    lea dx, temp_msg
    int 21h
    call get_input
    mov bx, ax
    sub bx, 32
    mov ax, bx
    imul bx, 5
    mov cx, 9
    idiv cx
    call display_result
    jmp exit_program
get_input:
    mov ah, 01h
    int 21h
    sub al, 30h   ; Convert ASCII to integer
    mov bl, al
    mov ax, 0
input_loop:
    mov ah, 01h
    int 21h
    cmp al, 13    ; Check for carriage return
    je input_done
    sub al, 30h   ; Convert ASCII to integer
    mov cl, al
    mov al, bl
    mul bx
    mov bl, 10
    add ax, cx
    jmp input_loop
input_done:
    ret
    display_result:

        mov ah, 09h

        lea dx, result_msg

        int 21h

        mov si, 10

    convert_and_display:

        xor dx, dx

        div si

        add dl, 30h

        push dx

        cmp ax, 0

        jz display_loop

        jmp convert_and_display

    display_loop:

        pop dx

        mov ah, 02h

        int 21h

        loop display_loop

        jmp exit_program

    exit_program:

        mov ax, 4C00h

        int 21h

    main endp

end main
```

3. Implement an assembly language program to check if a given number is prim or not.

Code:

```
.model small
.stack 100h
.data
   num_msg db 'Enter a number: $'
   prime_msg db 'The number is PRIME.',13,10,'$'
           db 'The number is NOT PRIME.',13,10,'$'
.code
   main proc
      mov ax, @data
      mov ds, ax
      mov ah, 09h
      lea dx, num_msg
      int 21h
      call get_input
      mov bx, ax
      call is_prime
      cmp bx, 0
      je not_prime
      mov ah, 09h
      lea dx, prime_msg
      int 21h
```

```
    jmp exit_program
not_prime:
    mov ah, 09h
    lea dx, prime_msg
    int 21h
exit_program:
    mov ax, 4C00h
    int 21h
main endp
get_input:
    mov ah, 01h
    int 21h
    sub al, 30h   ; Convert ASCII to integer
    mov bl, al
    mov ax, 0
input_loop:
    mov ah, 01h
    int 21h
    cmp al, 13    ; Check for carriage return
    je input_done
    sub al, 30h   ; Convert ASCII to integer
    mov cl, al
    mov al, bl
    mul bx
    mov bl, 10
```

```
        add ax, cx
        jmp input_loop
    input_done:
        ret
    is_prime:
        mov cx, 2
    check_prime_loop:
        mov dx, 0
        div cx
        cmp dx, 0
        je not_prime_exit
        inc cx
        cmp cx, ax
        jb check_prime_loop
        mov bx, 1
        ret
    not_prime_exit:
        xor bx, bx
        ret
end main
```