Green University of Bangladesh

Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: (fall, Year:2024), B.Sc. in CSE (Day)

Lab Report #05

Course title: Microprocessor & Microcontroller Lab

Course Code: CSE 304 **Section:** 222_D13

Lab Experiment Name: Implement Array and String in Assembly Language Programming.

Student Details

Name	ID
Md. Moshiur Rahman	221902324

Submission Date : 30/11/2024
Course Teacher's Name : Md. Jahid Tanvir

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks: …………………………… **Signature:.........................**

Comments:.............................. **Date:..............................**

1. TITLE OF THE LAB REPORT EXPERIMENT

Implement Array and String in Assembly Language Programming

2. OBJECTIVES

- To understand the use of Array in Assembly Language Program.
- To understand the use of String in Assembly Language Program.

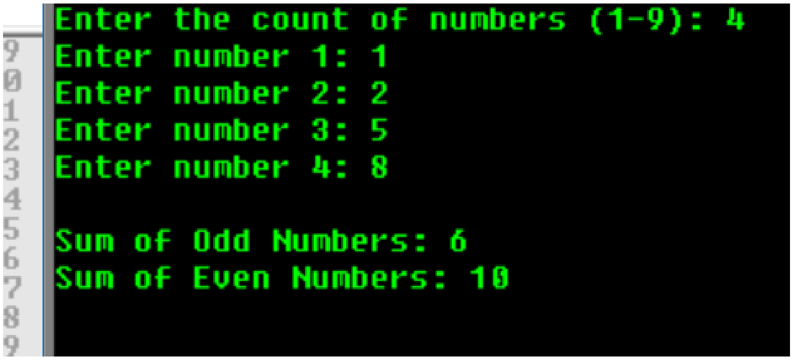
Write an assembly language code to take natural number series as input and as output, show:
a. The summation of odd numbers.
b. The summation of even numbers.
[NB: In this program you should use string for input and output message]

3. IMPLEMENTATION

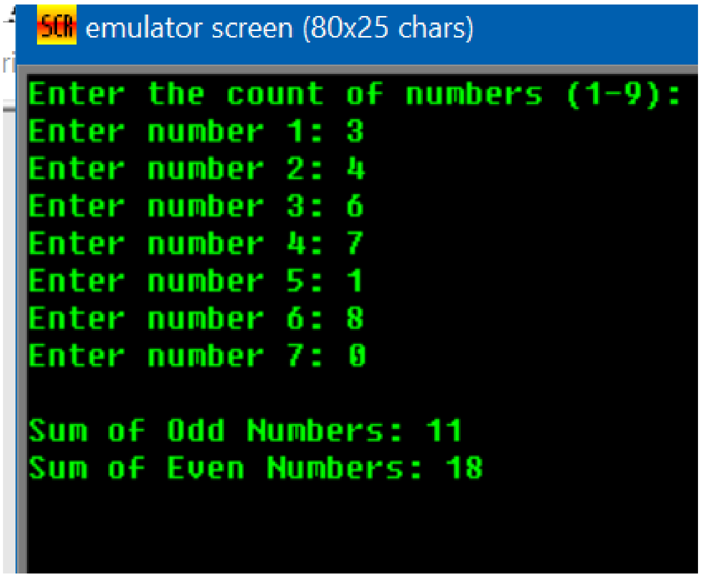
Src code:

```
DATA SEGMENT
    ; Input and output messages
    ENTER_COUNT_MSG         DB  'Enter the count of numbers (1-9): $'
    ENTER_NUM_MSG   DB  'Enter number $'
    COLON_MSG       DB  ': $'
    ODD_SUM_MSG     DB  'Sum of Odd Numbers: $'
    EVEN_SUM_MSG    DB  'Sum of Even Numbers: $'
    NEWLINE         DB  0DH, 0AH, '$'

    ; Variables for calculations
    COUNT       DB  ?   ; Number of input numbers
    CURRENT_NUM     DB  ?   ; Current number being processed
    ODD_SUM         DW  0           ; Summation of odd numbers
    EVEN_SUM        DW  0           ; Summation of even numbers
    LOOP_COUNTER    DB  ?   ; Loop counter for input
DATA ENDS

CODE SEGMENT
    ASSUME CS:CODE, DS:DATA

; Macro to print a string
PRINT_STRING MACRO MSG
    LEA DX, MSG
    MOV AH, 09H
    INT 21H
ENDM

START:
    ; Initialize data segment
    MOV AX, DATA
    MOV DS, AX

    ; Prompt for count of numbers
    PRINT_STRING ENTER_COUNT_MSG

    ; Read count of numbers (1-9)
    MOV AH, 01H
    INT 21H
    SUB AL, 30H
    MOV COUNT, AL
    MOV LOOP_COUNTER, AL

    ; Print newline
    PRINT_STRING NEWLINE

    ; Input loop for numbers
INPUT_LOOP:
    ; Prompt for current number input
    PRINT_STRING ENTER_NUM_MSG

    ; Print current loop iteration
    MOV AL, LOOP_COUNTER
    SUB AL, COUNT
    ADD AL, 31H  ; Convert to ASCII
    MOV DL, AL
    MOV AH, 02H
    INT 21H

    PRINT_STRING COLON_MSG

    ; Read number
    MOV AH, 01H
    INT 21H
    SUB AL, 30H
    MOV CURRENT_NUM, AL

    ; Check if number is odd or even
    TEST CURRENT_NUM, 01H
    JZ EVEN_NUMBER

    ; Odd number - add to ODD_SUM
    MOV AX, ODD_SUM
    ADD AL, CURRENT_NUM
    ADC AH, 0
    MOV ODD_SUM, AX
    JMP CONTINUE_LOOP

EVEN_NUMBER:
    ; Even number - add to EVEN_SUM
    MOV AX, EVEN_SUM
    ADD AL, CURRENT_NUM
    ADC AH, 0
    MOV EVEN_SUM, AX

CONTINUE_LOOP:
    ; Print newline
    PRINT_STRING NEWLINE

    ; Decrement loop counter
    DEC COUNT
    JNZ INPUT_LOOP

    ; Print results
    PRINT_STRING NEWLINE
    PRINT_STRING ODD_SUM_MSG

    ; Print odd sum
    MOV AX, ODD_SUM
    CALL PRINT_NUMBER
    PRINT_STRING NEWLINE
    PRINT_STRING EVEN_SUM_MSG

    ; Print even sum
    MOV AX, EVEN_SUM
    CALL PRINT_NUMBER

    ; Exit program
    MOV AH, 4CH
    INT 21H

; Procedure to print a 16-bit number
PRINT_NUMBER PROC
    ; Save registers
    PUSH BX
    PUSH CX
    PUSH DX

    ; Check if number is zero
    CMP AX, 0
    JNE NON_ZERO

    ; If zero, print '0'
    MOV DL, '0'
    MOV AH, 02H
    INT 21H
    JMP PRINT_NUMBER_END

NON_ZERO:
    ; Initialize digit counter
    MOV CX, 0
    MOV BX, 10

CONVERT_LOOP:
    ; Divide by 10
    MOV DX, 0
    DIV BX

    ; Push remainder (digit) to stack
    PUSH DX
    INC CX

    ; Check if quotient is zero
    CMP AX, 0
    JNE CONVERT_LOOP

PRINT_LOOP:
    ; Pop and print digits
    POP DX
    ADD DL, '0'
    MOV AH, 02H
    INT 21H
    LOOP PRINT_LOOP

PRINT_NUMBER_END:
    ; Restore registers
    POP DX
    POP CX
    POP BX
    RET
PRINT_NUMBER ENDP

CODE ENDS
END START
```

Test(Output)



```
Enter the count of numbers (1-9): 4
Enter number 1: 1
Enter number 2: 2
Enter number 3: 5
Enter number 4: 8

Sum of Odd Numbers: 6
Sum of Even Numbers: 10
```



```
Enter the count of numbers (1-9):
Enter number 1: 3
Enter number 2: 4
Enter number 3: 6
Enter number 4: 7
Enter number 5: 1
Enter number 6: 8
Enter number 7: 0

Sum of Odd Numbers: 11
Sum of Even Numbers: 18
```

4. ANALYSIS AND DISCUSSION [3 marks]

The program effectively calculates the sum of odd and even numbers entered by the user. It utilizes registers to accumulate the sums, optimizing memory access. The input validation and output formatting are well-implemented, providing a polished user experience.

5. SUMMARY:

The program makes good use of macros to simplify the string printing logic. It also has a custom PRINT_NUMBER procedure to handle the conversion and display of the 16-bit sum values.