CSE 331L / EEE 332L: Microprocessor Interfacing & Embedded System

Section: 7&9, Spring 2020

Lab - 02 (Input-Output Functions)

Program Structure

- **Code Segment:** holds the instructions of the program, instructions are organized as procedures.

  Procedure is a part of code that can be called from your program in order to make some specific task. Procedures make programs more structural and easier to understand. Generally procedure returns to the same point from where it was called.

  The syntax for procedure declaration:

```
name PROC
    ; here goes the code
    ; of the procedure ...
  RET
  name ENDP
```

  name - is the procedure name, the same name should be in the top and bottom, this is used to check the correct closing of procedures.

  Probably, you already know that RET instruction is used to return to the operating system. The same instruction is used to return from procedure (actually operating system sees your program as a special procedure).

  PROC and **ENDP** are compiler directives, so they are not assembled into any real machine code. Compiler just remembers the address of the procedure.
  CALL instruction is used to call a procedure.

```
ORG   100h
      MOV AL, 1
      CALL SUM
RET           ; return to the operating system.

SUM PROC
      MOV BL, 8
      ADD Al, BL
      RET     ; return to caller.
SUM ENDP
```

- **Data Segment:** contains all the variable definitions
- **Stack Segment:** contains a block of memory to store the stack, needs enough space to store.

Memory Models: SMALL, MEDIUM, COMPACT, LARGE, HUGE (Determines the size of code and data of the program)

Program Structure	**Example**

ORG 100H
.MODEL SMALL
.STACK 100H
 	
.DATA
;variables and constants
.CODE
MAIN PROC
;instructions of main procedure
RET
MAIN ENDP
; other procedures
END MAIN
	ORG 100H
	.MODEL SMALL
	.CODE
	
	MAIN PROC
	MOV AL, 1
	CALL SUM
	RET ;return to OS
	MAIN ENDP
	
	SUM PROC
	MOV BL, 8
	ADD Al, BL
	RET ; return to caller.
	SUM ENDP
	END MAIN

Functions

Function #	**Routine**
1	Single-key input
2	Single-key output
9	Character string output
4CH	DOS exit function

Function# 4CH

```
ORG 100H
.MODEL SMALL
.CODE

MAIN PROC
        ;BODY

        MOV AH, 4CH        ;terminate function number
        INT 21H            ;executes the function number 4ch
MAIN ENDP

;other procedures

END MAIN
```

Function# 1&2

Single-key Input	**Single-key output**

ORG 100H
.MODEL SMALL
 	
.CODE
MAIN PROC
 	
MOV AH, 1
INT 21H
 	
RET
MAIN ENDP
END MAIN
	ORG 100H
	.MODEL SMALL
	
	.CODE
	MAIN PROC
	
	MOV AH, 2
	MOV DL, AL
	INT 21H
	RET
	
	MAIN ENDP
	END MAIN

Single-key Input/Output

```
ORG 100H
.MODEL SMALL

.CODE
        MAIN PROC

        MOV AH, 1       ;input-key function
        INT 21H         ;ASCII code in AL

        MOV AH, 2       ;display character function
        MOV DL, AL      ;character from input stored in AL
        INT 21H         ;display character
        RET

        MAIN ENDP
END MAIN
```

Insert newline:

```
ORG 100H
.MODEL SMALL
.CODE
        MAIN PROC

        MOV AH, 1
        INT 21H
        MOV BL, AL

        MOV AH, 2
        MOV DL, 10      ;0AH: NEWLINE
        INT 21H
        MOV DL, 13      ;0DH: CARRIAGE RETURN,
                         ;BRINGS THE POINTER TO THE BEGINNING OF LINE
        INT 21H

        MOV AH, 2
        MOV DL, BL
        INT 21H
        RET

        MAIN ENDP
END MAIN
```

Multiple key Input

```
ORG 100H
.MODEL SMALL
.CODE
        MAIN PROC

        MOV AH, 1
        INT 21H
        MOV BL, AL

        INT 21H
        MOV BH, AL

        INT 21H
        MOV CL, AL

        MOV AH, 2
        MOV DL, 10
        INT 21H
        MOV DL, 13
        INT 21H

        MOV AH, 2
        MOV DL, BL
        INT 21H
        RET

        MAIN ENDP
END MAIN
```

TASK

1. Take 3 single-key input and display them using output function; each output should be in a separate line.
2. Show the output of Task 1 in reverse order all in one line using space between each two characters.
3. Take 2 numbers as input, add them and show the answer.
4. Take 2 numbers as input, subtract them and show the answer. (the answer should be positive)