EEE316 Microprocessors
Pre-Laboratory Report

Name	**:**	**Turhan Can Kargin**
ID Number	**:**	**150403005**
Lab. Number	**:**	**4**

Objectives of the Laboratory Assignment:

Objectives of this lab are memory operations by using a MPLAB simulator, constructing loops, and learning how to use tables, pointers and how to write in ASCII characters.

Code and Comments:

1.

```
org 0h
COUNTREG    EQU 0X00 ; Define processes
CNTVAL      EQU D'5'
COUNTREG1   EQU 0X10
L_BYTE      EQU 0X16
H_BYTE      EQU 0X15
CNTVAL1     EQU 0X0A
NUM         EQU 0X52
MYQ         EQU 0X50

        MOVLW CNTVAL ; for looping
        MOVWF COUNTREG

        LFSR 0,0X20 ; Pointer for 0x20
        LFSR 2,0X60 ; Pointer for 0x60


TURHAN  MOVFF POSTINC0,POSTINC2 ; Here we saved the numbers from loc 20-24 to 60-64
        DECF COUNTREG,F
        BNZ TURHAN

        MOVLW CNTVAL ; Loop
        MOVWF COUNTREG
        LFSR 1,0X30 ; Pointer for 0x30
        LFSR 2,0X70 ; Pointer for 0x70
```

```
TURHAN1     MOVFF POSTINC1,POSTINC2 ; Here we saved the numbers from loc 30-34 to 70-74
        DECF COUNTREG,F
        BNZ TURHAN1


        LFSR 2,0X60 ; Pointer for 0x60
        LFSR 1,0X70 ; Pointer for 0x70
        LFSR 0,0X20 ; Pointer for 0x20

        MOVFF POSTINC2,POSTINC0 ; Here we put the numbers from 0x60-0x64 and 0x70-0x74
        MOVFF POSTINC1,POSTINC0 ; to loc 0x20-0x29
        MOVFF POSTINC2,POSTINC0
        MOVFF POSTINC1,POSTINC0
        MOVFF POSTINC2,POSTINC0
        MOVFF POSTINC1,POSTINC0
        MOVFF POSTINC2,POSTINC0
        MOVFF POSTINC1,POSTINC0
        MOVFF POSTINC2,POSTINC0
        MOVFF POSTINC1,POSTINC0

        MOVLW CNTVAL1  ; Loop
        MOVWF COUNTREG1
        LFSR 0,0X20    ; Pointer for 0x20
        CLRF WREG      ; W=0
        CLRF H_BYTE    ; Clear High Byte
B5      ADDWF POSTINC0, W ; Here is to sum all numbers in loc 0x20 - 0x29
        BNC OVER
        INCF H_BYTE,F
OVER    DECF COUNTREG1,F
        BNZ B5
        MOVWF L_BYTE

        CLRF MYQ ; Here is for taking average of these numbers
        MOVFF L_BYTE, NUM
        MOVLW CNTVAL1
B9      INCF MYQ, F
        SUBWF NUM,F
        BC B9
        DECF MYQ,F
        END
```

2.

```
org 0h
COUNTREG    EQU 0X00 ; Define steps
CNTVAL      EQU D'3'

        MOVLW CNTVAL ; How many numbers to sum
        MOVWF COUNTREG

        CLRF 0X90 ; Clearing locations
        CLRF 0X91
        CLRF WREG
        CLRF 0X50
        CLRF 0X51
        CLRF 0X52

        LFSR 0,0X10 ; Pointer for 10h locs
        LFSR 1,0X30 ; Pointer for 30h locs
        LFSR 2,0X50 ; Pointer for 50h locs


TURHAN      MOVLW 0X00 ; W=0
        MOVWF 0X90 ; 0x90 loc = 0
        MOVWF 0X91 ; 0x91 loc = 0
        BCF STATUS,C ; Clear carry
        MOVF POSTINC0,W ; move 10h loc to wreg and increase
        ADDWF POSTINC1,W ; add wreg to loc 30h and increase
        MOVWF POSTINC2 ; move wreg to loc 50h and increase
        BNC OVER ; Check carry
        INCF 0X90,F ; If C = 1 increment loc 90h
OVER        MOVF POSTINC0,W ; move 11h loc to wreg and increase
        ADDWF POSTINC1,W ; add wreg to loc 31h and increase
        MOVWF INDF2 ; move wreg to loc 51h and stay the same
        BNC OVER2 ; Check carry
        INCF 0X91,F ; If C = 1 increment loc 0x91

OVER2       MOVF 0X90,W ; Here for adding carries to bytes
        ADDWF POSTINC2,F
        MOVF 0X91,W
        ADDWF POSTINC2,F
        DECF COUNTREG,F
        BNZ TURHAN
        END
```

3.

```
        org 0h
        COUNTER EQU 0X00
        MOVLW 0X06 ; enter here how many character you have
        MOVWF COUNTER ; for looping
        LFSR 0,0X10 ; Pointer for loc 0x10
        LFSR 1,0X25 ; Pointer for loc 0x25

L1      MOVFF POSTINC0,POSTDEC1 ; Move loc 10h (and increase) to loc 25 (and decrease)
        DECF COUNTER,F
        BNZ L1

        MOVLW 0X06
        MOVWF COUNTER
        LFSR 0,0X10 ; Pointer for loc 0x10
        LFSR 1,0X20 ; Pointer for loc 0x20

L2      MOVFF POSTINC1,POSTINC0 ; Move loc 20h to loc 10 and increase both of them
        DECF COUNTER,F
        BNZ L2
        END
```

BONUS.

```
org 0000H ; burn into ROM starting at 0

     ;Turhan Can
     MOVLW 0X00 ; Wreg = 0 Look-up table low-byte addr
     MOVWF TBLPTRL ; Look-up table low-byte addr
     MOVLW 0X04 ; Wreg = 5 look-up table high-byte addr
     MOVWF TBLPTRH ; Look-up table high-byte addr
     TBLRD* ; TABLAT = 'T' char pointed to TABPTR
     MOVFF TABLAT,0x00 ; Send it to loc 0x00
     INCF TBLPTRL,F ; TBLPTRL = 01 pointing to next (401)
     TBLRD* ; TABLAT = 'U' char pointed to TABPTR
     MOVFF TABLAT,0x01 ; Send it to loc 0x01
     INCF TBLPTRL,F ; TBLPTRL = 02 pointing to next (402)
     TBLRD* ; TABLAT = 'R' char pointed to TABPTR
     MOVFF TABLAT,0x02 ; Send it to loc 0x02
     INCF TBLPTRL,F ; TBLPTRL = 03 pointing to next (403)
     TBLRD* ; TABLAT = 'H' char pointed to TABPTR
     MOVFF TABLAT,0x03 ; Send it to loc 0x03
     INCF TBLPTRL,F ; TBLPTRL = 04 pointing to next (404)
     TBLRD* ; TABLAT = 'A' char pointed to TABPTR
     MOVFF TABLAT,0x04 ; Send it to loc 0x04
     INCF TBLPTRL,F ; TBLPTRL = 05 pointing to next (405)
     TBLRD* ; TABLAT = 'N' char pointed to TABPTR
     MOVFF TABLAT,0x05 ; Send it to loc 0x05
     INCF TBLPTRL,F ; TBLPTRL = 06 pointing to next (406)
     TBLRD* ; TABLAT = ' ' char pointed to TABPTR
```

```
MOVFF TABLAT,0x06 ; Send it to loc 0x06
INCF TBLPTRL,F ; TBLPTRL = 07 pointing to next (407)
TBLRD* ; TABLAT = ' ' char pointed to TABPTR
MOVFF TABLAT,0x07 ; Send it to loc 0x07
INCF TBLPTRL,F ; TBLPTRL = 08 pointing to next (408)
TBLRD* ; TABLAT = 'C' char pointed to TABPTR
MOVFF TABLAT,0x08 ; Send it to loc 0x08
INCF TBLPTRL,F ; TBLPTRL = 09 pointing to next (409)
TBLRD* ; TABLAT = 'A' char pointed to TABPTR
MOVFF TABLAT,0x09 ; Send it to loc 0x09
INCF TBLPTRL,F ; TBLPTRL = 10 pointing to next (410)
TBLRD* ; TABLAT = 'N' char pointed to TABPTR
MOVFF TABLAT,0x0A ; Send it to loc 0x0A

;Kargin
MOVLW 0X10
MOVWF TBLPTRL
MOVLW 0X04
MOVWF TBLPTRH
TBLRD*
MOVFF TABLAT,0x10
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x11
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x12
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x13
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x14
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x15

;Student
MOVLW 0X20
MOVWF TBLPTRL
MOVLW 0X04
MOVWF TBLPTRH
TBLRD*
MOVFF TABLAT,0x20
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x21
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x22
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x23
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x24
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x25
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x26
```

```
;E.E.E
MOVLW 0X30
MOVWF TBLPTRL
MOVLW 0X04
MOVWF TBLPTRH
TBLRD*
MOVFF TABLAT,0x30
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x31
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x32
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x33
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x34

;I.K.C.U
MOVLW 0X40
MOVWF TBLPTRL
MOVLW 0X04
MOVWF TBLPTRH
TBLRD*
MOVFF TABLAT,0x40
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x41
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x42
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x43
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x44
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x45
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x46

;Logo
MOVLW 0X50
MOVWF TBLPTRL
MOVLW 0X04
MOVWF TBLPTRH
TBLRD*
MOVFF TABLAT,0x50
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x51
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x52
INCF TBLPTRL,F
TBLRD*
```

```
MOVFF TABLAT,0x53
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x54
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x55
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x56
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x57
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x58
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x59
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x5A
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x5B
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x5C
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x5D
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x5E
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x5F
     MOVLW 0X60
     MOVWF TBLPTRL
     MOVLW 0X04
     MOVWF TBLPTRH
     TBLRD*
     MOVFF TABLAT,0x60
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x61
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x62
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x63
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x64
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x65
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x66
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x67
```

```
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x68
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x69
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x6A
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x6B
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x6C
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x6D
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x6E
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x6F
     MOVLW 0X70
MOVWF TBLPTRL
MOVLW 0X04
MOVWF TBLPTRH
TBLRD*
MOVFF TABLAT,0x70
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x71
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x72
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x73
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x74
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x75
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x76
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x77
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x78
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x79
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x7A
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x7B
```

```
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x7C
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x7D
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x7E
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x7F

MOVLW 0X80
MOVWF TBLPTRL
MOVLW 0X04
MOVWF TBLPTRH
TBLRD*
MOVFF TABLAT,0x80
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x81
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x82
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x83
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x84
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x85
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x86
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x87
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x88
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x89
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x8A
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x8B
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x8C
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x8D
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x8E
INCF TBLPTRL,F
TBLRD*
MOVFF TABLAT,0x8F
```

```
     MOVLW 0X90
     MOVWF TBLPTRL
     MOVLW 0X04
     MOVWF TBLPTRH
     TBLRD*
     MOVFF TABLAT,0x90
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x91
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x92
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x93
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x94
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x95
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x96
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x97
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x98
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x99
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x9A
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x9B
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x9C
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x9D
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x9E
     INCF TBLPTRL,F
     TBLRD*
     MOVFF TABLAT,0x9F
HERE     GOTO HERE
```

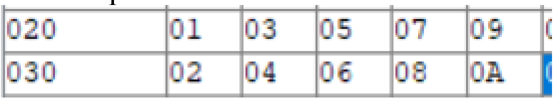
```
;Data is burned into code space starting at 400H
    ORG 400H
MYDATA  DB "TURHAN  CAN"
    ORG 410H
MYDATA1 DB "KARGIN"
    ORG 420H
MYDATA2 DB "STUDENT"
    ORG 430H
MYDATA3 DB "E.E.E"
    ORG 440H
MYDATA4 DB "I.K.C.U"
    ORG 450H
MYDATA5 DB ".-------. -------"
        ORG 460H
MYDATA6 DB "|.    .||       "
    ORG 470H
MYDATA7 DB "   ||   |       "
    ORG 480H
MYDATA8 DB "   ||   |       "
    ORG 490H
MYDATA9 DB "   ||    -------"
    END
```

EXPLANATIONS:

QUESTION-1 :

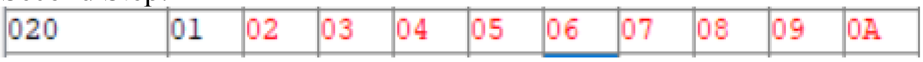
The aim of this question is to write to different lists which first one will be stored from address 0x20 to 0x2n (n will be entered by user to location 0x10) and second list will be stored from location 0x30 to 0x3n. Then, after execution the numbers will display 0x20 to 0x29 if there are 10 numbers. At the end, the average of these numbers will be shown at the location 0x50.

First Step:



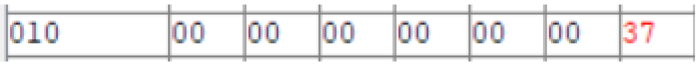
```
020  01  03  05  07  09  (
030  02  04  06  08  0A
```

Second Step:



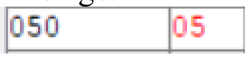
```
020  01  02  03  04  05  06  07  08  09  0A
```

Summation:



```
010  00  00  00  00  00  00  37
```

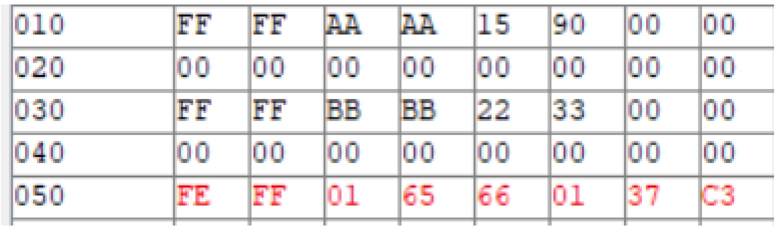
Average:



```
050  05
```

QUESTION-2 :

The aim of this question is to sum two list of unsigned numbers (starting at 0x10 and second is starting at 0x30) and put the result into the list starting at loc 0x50. The length and numbers will be defined by the user.



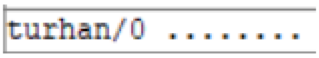
010	FF	FF	AA	AA	15	90	00	00
020	00	00	00	00	00	00	00	00
030	FF	FF	BB	BB	22	33	00	00
040	00	00	00	00	00	00	00	00
050	FE	FF	01	65	66	01	37	C3

(In the code section, I wrote the code as the low byte is at 0x10 adress and high byte is at 0x11 adress and same for second list. And the high byte of the result is at 0x52 location.)

QUESTION-3 :

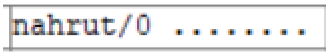
The aim of this question is defining a string and after execution given string will be written in inverse order. The list stats at 0x10.

Before Execution:



```
turhan/0 ........
```

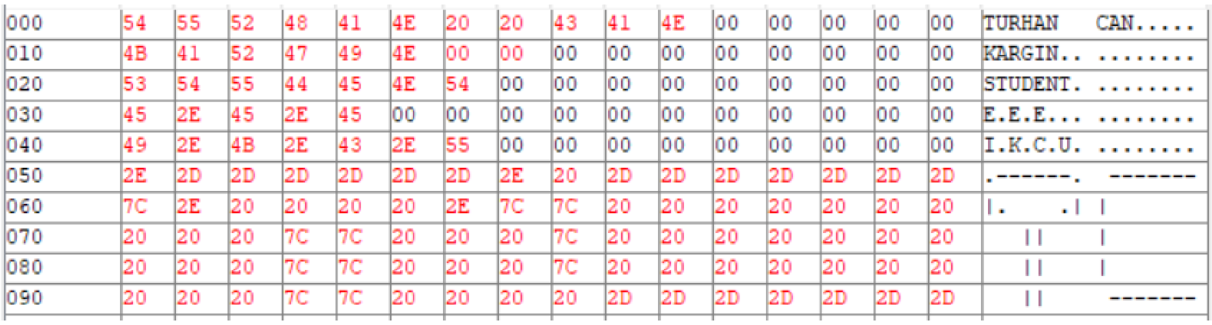
After Execution:



```
nahrut/0 ........
```

QUESTION-BONUS:

The aim of this question is to write my name, surmane, positoin, department, university, initials and logo in ASCII part.



000	54	55	52	48	41	4E	20	20	43	41	4E	00	00	00	00	00	TURHAN CAN.....
010	4B	41	52	47	49	4E	00	00	00	00	00	00	00	00	00	00	KARGIN.. ....
020	53	54	55	44	45	4E	54	00	00	00	00	00	00	00	00	00	STUDENT. ....
030	45	2E	45	2E	45	00	00	00	00	00	00	00	00	00	00	00	E.E.E... ....
040	49	2E	4B	2E	43	2E	55	00	00	00	00	00	00	00	00	00	I.K.C.U. ....
050	2E	2D	2D	2D	2D	2D	2D	2E	20	2D	2D	2D	2D	2D	2D	2D	.------. -------
060	7C	2E	20	20	20	20	2E	7C	7C	20	20	20	20	20	20	20	|. .| |
070	20	20	20	7C	7C	20	20	20	7C	20	20	20	20	20	20	20	|| |
080	20	20	20	7C	7C	20	20	20	7C	20	20	20	20	20	20	20	|| |
090	20	20	20	7C	7C	20	20	20	20	2D	2D	2D	2D	2D	2D	2D	|| -------

Note:

This document will be prepared before the lab session. Unless you bring this document in the desired format or prepared, you will not be let to the session.