Tuna Şahin 22201730 Assembly codes for Lab 2

part 1 :

```
ORG 0H
MOV A,#255
LOOPSTART:
MOV B,#10
DIV AB
PUSH B
JNZ LOOPSTART
ADDLOOP:
POP 7
ADD A,R7
MOV R1,SP
CJNE R1,#007H,ADDLOOP
MOV 30H,A
END
```

part 2: the code works iterably left-bottommost switch resets the code. You can use it while testing #,a,b,c,d keys all send the data

```
ORG 0H
MOV R0,#042H
ACALL SETUP
LJMP KEYBOARD_LOOP
CALC_LOOP:
        MOV R0,#042H
        MOV A,41H
        MOV B,#0AH
        MUL AB
        ADD A,40H
        MOV 43H,A
        MOV B,#07H
        DIV AB
        MOV 47H,B

        MOV A,42H
        MOV B,#10
        MUL AB
        MOV B,#7
        DIV AB

        MOV A,B

        MOV B,#10
        MUL AB
        MOV B,#7
        DIV AB

        MOV A,47H
        ADD A,B
        MOV B,#7
        DIV AB
        MOV 47H,B

        MOV A,#030H
        ADD A,47H

        MONTH_CALC:
                MOV DPTR,#NUMDAY
                MOV R4,42H ;hundereds place
                MOV R5,43H ;tens and ones places
                MOV R3,#12
                CJNE R4,#00H,NVM9
                CJNE R5,#00H,NVM9
                AJMP ERROR
```

part 2: the code works iterably left-bottommost switch resets the code. You can use it while testing #,a,b,c,d keys all send the data

```
                NVM9:
                CJNE R4,#003H,T5
                        CJNE R5,#67,T6
                        T6:
                        JNC ERROR

                        T5:
                        JNC ERROR

                NVM4:
                MLS:
                CJNE R5,#100,T2
                        SJMP NVM3
                        T2:
                        JNC NVM3
                        CJNE R4,#0,T3
                                SJMP NVM3
                        T3:
                        DEC R4                          ;if here it means r5 < 100 and r4 > 0
                        MOV A,R5
                        ADD A,#064H
                        MOV R5,A

                NVM3:
                ;IF HERE IT MEANS R4 IS EXHAUSTED OR R5>100
                CLR A
                MOVC A,@A+DPTR
                MOV 30H,A
                SUBB A,R5
                ;ACALL DELAY
                JNC FOUND_MONTH
                INC DPTR
                MOV A,R5
                SUBB A,30H
                MOV R5,A
                INC R5
                CLR A
                DEC R3
                SJMP MLS
FOUND_MONTH:
        MOV A,#12
        SUBB A,R3
        MOV R3,A
WRITE_DATA:
        MOV A,R3
```

part 2: the code works iterably left-bottommost switch resets the code. You can use it while testing #,a,b,c,d keys all send the data

```
MOV B,#3
MUL AB
MOV R2,A

MOV DPTR,#MONTHS
MOVC A,@A+DPTR
ACALL SEND_DATA
MOV A,R2
INC DPTR
MOVC A,@A+DPTR
ACALL SEND_DATA
MOV A,R2
INC DPTR
MOVC A,@A+DPTR
ACALL SEND_DATA

MOV A,#20H
ACALL SEND_DATA
;=========================
MOV A,R5
MOV B,#10
DIV AB
ADD A,#30H
ACALL SEND_DATA
MOV A,B
ADD A,#30H
ACALL SEND_DATA
;=========================
MOV A,#20H
ACALL SEND_DATA

MOV A,47H
MOV B,#3
MUL AB
MOV R2,A

MOV DPTR,#DAYS
MOVC A,@A+DPTR
ACALL SEND_DATA
MOV A,R2
INC DPTR
MOVC A,@A+DPTR
ACALL SEND_DATA
MOV A,R2
INC DPTR
```

part 2: the code works iterably left-bottommost switch resets the code. You can use it while testing #,a,b,c,d keys all send the data

```
        MOVC A,@A+DPTR
        ACALL SEND_DATA

        AJMP KEYBOARD_LOOP
ERROR:
        MOV DPTR,#ERROR_MESSAGE
        ERROR_LOOP:
        CLR A
        MOVC A,@A+DPTR
        INC DPTR
        JZ KEYBOARD_LOOP
        ACALL SEND_DATA
        SJMP ERROR_LOOP

WRITE_INPUT:
        MOV DPTR,#INPUT_MESSAGE
        WI_LOOP:
        CLR A
        MOVC A,@A+DPTR
        INC DPTR
        JZ NVM8
        ACALL SEND_DATA
        SJMP WI_LOOP
NVM8:
RET


RECORD_DATA:
        SUBB A,#30H
        MOV @R0,A
        DEC R0
        RET

KEYBOARD_LOOP:

        ACALL KEYBOARD          ; NOW A HAS THE KEY THAT IS PRESSED
        CJNE A,#02AH,NVM
                ACALL SETUP     ;CLR
                MOV R0,#042H
                SJMP KEYBOARD_LOOP
        NVM:
        CJNE A,#023H,NVM2
                MOV A,#0C0H     ;NEXT LINE
                ACALL SEND_COMMAND
                CJNE R0,#041H,NVM5
```

part 2: the code works iterably left-bottommost switch resets the code. You can use it while testing #,a,b,c,d keys all send the data

```
                MOV 40H,42H
                MOV 41H,#0
                MOV 42H,#0
                MOV R0,#042H
                AJMP CALC_LOOP
        NVM5:
        CJNE R0,#040H,NVM6
                MOV 40H,41H
                MOV 41H,42H
                MOV 42H,#0
                MOV R0,#042H
                AJMP CALC_LOOP
        NVM6:
        CJNE R0,#03FH,NVM7
                AJMP CALC_LOOP
        NVM7:
                AJMP ERROR


                MOV R0,#042H
                AJMP CALC_LOOP
        NVM2:
        ACALL SEND_DATA         ; SEND DATA TO LCD SCREEN
        ACALL RECORD_DATA
        SJMP KEYBOARD_LOOP  ; DOING ALL OVER AGAIN

SETUP:
        MOV A,#38H ;TWO LINES, 5X7 MATRIX
        ACALL SEND_COMMAND
        MOV A,#0FH;DISPLAY ON, CURSOR BLINKING
        ACALL SEND_COMMAND
        MOV A,#06H;INCREMENT CURSOR (SHIFT CURSOR TO RIGHT)
        ACALL SEND_COMMAND
        MOV A,#01H ;CLEAR DISPLAY SCREEN
        ACALL SEND_COMMAND
        MOV A,#80H;FORCE CURSOR TO BEGINNING OF THE FIRST LINE
        ACALL SEND_COMMAND
        ACALL WRITE_INPUT
        RET

SEND_COMMAND:
        MOV P1,A
        CLR P3.5            ;RS=0 BEFORE SENDING COMMAND
        CLR P3.6            ;R/W=0 TO WRITE
        SETB P3.7     ;SEND A HIGH TO LOW SIGNAL TO ENABLE PIN
```

part 2: the code works iterably left-bottommost switch resets the code. You can use it while testing #,a,b,c,d keys all send the data

```
        ACALL DELAY
        CLR P3.7
        RET
SEND_DATA:
        MOV P1,A
        SETB P3.5           ;RS=0 BEFORE SENDING COMMAND
        CLR P3.6            ;R/W=0 TO WRITE
        SETB P3.7     ;SEND A HIGH TO LOW SIGNAL TO ENABLE PIN
        ACALL DELAY
        CLR P3.7
        RET

DELAY:
        PUSH 0
        PUSH 1
        MOV R7,#50
DELAY_OUTER_LOOP:
        MOV R6,#255
        DJNZ R6,$
        DJNZ R7,DELAY_OUTER_LOOP
        POP 1
        POP 0
        RET



KEYBOARD: ; takes the key pressed from the keyboard and puts it to A
        mov   P0, #0ffh      ;makes P0 input
K1:
        mov   P2, #0 ;ground all rows
        mov   A, P0
        anl     A, #00001111B
        cjne    A, #00001111B, K1
K2:
        acall   DELAY
        mov   A, P0
        anl     A, #00001111B
        cjne    A, #00001111B, KB_OVER
        sjmp   K2
KB_OVER:
        acall DELAY
        mov   A, P0
        anl     A, #00001111B
        cjne    A, #00001111B, KB_OVER1
        sjmp   K2
```

part 2: the code works iterably left-bottommost switch resets the code. You can use it while testing #,a,b,c,d keys all send the data

```
KB_OVER1:
        mov   P2, #11111110B
        mov   A, P0
        anl   A, #00001111B
        cjne  A, #00001111B, ROW_0
        mov   P2, #11111101B
        mov   A, P0
        anl   A, #00001111B
        cjne  A, #00001111B, ROW_1
        mov   P2, #11111011B
        mov   A, P0
        anl   A, #00001111B
        cjne  A, #00001111B, ROW_2
        mov   P2, #11110111B
        mov   A, P0
        anl   A, #00001111B
        cjne  A, #00001111B, ROW_3
        ljmp  K2

ROW_0:
        mov   DPTR, #KCODE0
        sjmp  KB_FIND
ROW_1:
        mov   DPTR, #KCODE1
        sjmp  KB_FIND
ROW_2:
        mov   DPTR, #KCODE2
        sjmp  KB_FIND
ROW_3:
        mov   DPTR, #KCODE3
KB_FIND:
        rrc   A
        jnc   KB_MATCH
        inc   DPTR
        sjmp  KB_FIND
KB_MATCH:
        clr   A
        movc  A, @A+DPTR; get ASCII code from the table
        ret

;ASCII look-up table
KCODE0:     DB    '1', '2', '3', '#'
KCODE1:     DB    '4', '5', '6', '#'
KCODE2:     DB    '7', '8', '9', '#'
KCODE3:     DB    '*', '0', '#', '#'
```

part 2: the code works iterably left-bottommost switch resets the code. You can use it while testing #,a,b,c,d keys all send the data

```
DAYS: DB 'SUN','MON','TUE','WED','THU','FRI','SAT'
MONTHS: DB 'JAN','FEB','MAR','APR','MAY','JUN','JUL','AUG','SEP','OCT','NOV','DEC'
NUMDAY: DB 31,29,31,30,31,30,31,31,30,31,30,31,31,29,31,30
ERROR_MESSAGE: DB 'ERROR',0
INPUT_MESSAGE: DB 'PLEASE INPUT NUMBER: ',0
ENDD:
SJMP $
END
```