

```

CONFIGURE_LCD: ;THIS SUa
MOV a,#38H ;TWOSA
ACALL SEND_COMMAND
MOV a,#0FH ;DISPas
ACALL SEND_COMMAND
MOV a,#06H ;INCREas
ACALL SEND_COMMAND
MOV a,#01H ;CLEARas
ACALL SEND_COMMAND
MOV a,#80H ;FE
ACALL SEND_COMMAND
ret
; MOV P0, #0ffh ; Set P0 as 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
;jmp K2
;KB_OVER:
; ACALL DELAY
; MOV A, P0
; anl A, #00001111B
;cjne A, #00001111B, KB_OVER1
;jmp K2
;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

```

SEND_COMMAND:

```
MOV p1,a ;THE yh
```

```
clr p3.5 ;RS=gf
```

```
clr p3.6 ;R/hy
```

```
setb p3.7 ;Svf
```

```
ACALL DELAY
```

```
clr p3.7
```

```
ret
```

SEND_DATA:

```
MOV p1,a ;Ssd
```

```
setb p3.5 ;de
```

```
clr p3.6 ;qw
```

```
setb p3.7 ;aw
```

```
ACALL DELAY
```

clr p3.7

ret

DELAY:

push 0

push 1

MOV r1,#70

DELAY_OUTER_LOOP:

MOV r0,#124

djnz r0,\$

djnz r1,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

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
input_yazi: db 'input = ',0
ORG 0
clr a
ACALL CONFIGURE_LCD
MOV dptr,#input_yazi
clr a
MOVC a,@a+dptr; yazıyı kap gel
input_loop: ACALL SEND_DATA; input=... kısmı burası sürekli yollanacak sonra bir yerde
cursorukaydır
clr a
inc dptr
MOVC a,@a+dptr
MOV r6, a
cjne a,0,input_loop ; jump not equal null ch Clear A, incrementdptr, MOVnext memory location to
Auntil not equal 0.
MOV dptr,#0h; dptri resetle
MOV 60h,#3; 3 hane sayacak
k_lo:
ACALL KEYBOARD
;now, A has the key pressed
MOV 84h,a ; gelen sayıyı depola geçici
MOV r6, 60h
MOV a,r6
add a,#60h

```

```
MOV r6,a; sonradan kullan
MOV r0,a
MOV a,84h
cjne a,'#'A',fed_continue ; a is ascii 65
ljmp oku ; a girildiğinde durdur
fed_continue:
MOV @r0,a
ACALL SEND_DATA
djnz 60h, k_lo
kb_takı: ACALL KEYBOARD
cjne a,'#'A',asd
ljmp oku
asd: ljmp kb_takı
oku:
MOV r0,60h
cjne r0,#03h,bas_ayarla ; data girilmedi data al k_lo
ljmp k_lo
MOV a,#0C0h; cursor ayarla
ACALL SEND_COMMAND
bas_ayarla: MOV a,#0C0h;cursor ayarla
ACALL SEND_COMMAND; cursor komutunu yolla babuş
clr c
cjne r0,#1h,gelis ; 2 rakam girildi ilk girilen onlar basamağı ikincisi birler yerlerini değiştir
MOV 61h,62h
MOV 62h,63h
MOV 63h,#30h; ascii 0
ljmp ascii_hexa
gelis: jnc gth1;eşit değilse carry f =0 zıpla
ljmp ascii_hexa
gth1:
MOV 61h,63h
```

MOV 62h,#30h;0

MOV 63h,#30h;0

ljmp ascii_hexa

; 63h 62h 61h

ascii_hexa:

MOV a,63h ; a'ya 63h adresindeki değeri yükle

clr c ; Taşıma bayrağını temizle

subb a,#30h ; ASCII değerinden '0' karakterinin ASCII değerini çıkararak heksadesimal değeri bul

MOV 63h,a ; Hesaplanan heksadesimal değeri 63h adresine koy

MOV a,62h ; a'ya 62h adresindeki değeri yükle

subb a,#30h ; ASCII değerinden '0' karakterinin ASCII değerini çıkararak heksadesimal değeri bul

MOV 62h,a ; Hesaplanan heksadesimal değeri 62h adresine koy

MOV a,61h ; a'ya 61h adresindeki değeri yükle

subb a,#30h ; ASCII değerinden '0' karakterinin ASCII değerini çıkararak heksadesimal değeri bul

MOV 61h,a ; Hesaplanan heksadesimal değeri 61h adresine koy

MOV r1,63h ; r1'e 63h adresindeki değeri yükle

MOV r2,62h ; r2'ye 62h adresindeki değeri yükle

MOV r3,61h ; r3'e 61h adresindeki değeri yükle

MOV A,r2 ; A'ya r2'nin içeriğini yükle

MOV dptr,#0h ; DPTR'ye 0h adresini yükle

MOV r0,#0 ; r0'ı sıfırla

MOV b, #0Ah ; onlar basamağını bulmak için 10 ile çarp

mul ab ; A ile B'yi çarp, sonucu A'ya kaydet

add a,r3 ; A'ya r3'ü ekle

MOV b, #64h ; 100 ile çarpmak için 64h'i B'ye yükle

MOV r4,a ; r4'e A'ya yüklenen değeri kopyala

MOV a,r1 ; A'ya r1'in içeriğini yükle

mul ab ; A ile B'yi çarp, sonucu A'ya kaydet

jnb ov,adim_1; taşma yoksa adim_1'e git

MOV r0,#01h ; r0'ı 1'e yükle

adim_1:

add a,r4 ; A'ya r4'ü ekle

jnb cy,adim_2; taşma yoksa adim_2'ye git

MOV r0,#01h ; r0'ı 1'e yükle

adim_2:

MOV r1,a ; r1'e A'ya yüklenen değeri kopyala

MOV r4,#0h ; r4'ü sıfırla

cjne r0,#01h, elde ; eğer r0 1'e eşit değilse elde'ye git

MOV r4,#04h ; r4'e 4'ü yükle

elde: ; yıl pazartesiyle başlıyor direkt yaz

MOV b, #07h ; b'ye 7h'yi yükle

MOV a,r1 ; A'ya r1'in içeriğini yükle

div ab ; A'yı B'ye böl, sonucu A'ya kaydet

MOV a,b ; A'yı B'ye yükle

add a,r4 ; A'ya r4'ü ekle

MOV b, #07h ; b'ye 7h'yi yükle

div ab ; A'yı B'ye böl, sonucu B'ye kaydet

MOV r4,b ; r4'e B'yi yükle

mov 90h, a ; 90h adresine A'yı yükle (Yılın başladığı günü tutar, 0'dan Pazar, 1'den Pazartesi)

mov 90h, a

GUN1:

CJNE R4,#1H,GUN7

MOV R7, #'p'

MOV R6, #'z'

MOV R5, #'t'

IJMP ay_bul

mov 90h, a

GUN2:

MOV R7, #'s'

MOV R6, #'a'


```
MOV R5, #'l'
ljmp ay_bul
mov 90h, a
GUN3:
CJNE R4,#3H,GUN5
MOV R7, #'c'
MOV R6, #'r'
MOV R5, #'s'
IJMP ay_bul
mov 90h, a
GUN4:
MOV R7, #'p'
MOV R6, #'r'
MOV R5, #'s'
IJMP ay_bul
mov 90h, a
GUN5: JC GUN1
CJNE R4,#5H,GUN6
MOV R7, #'c'
MOV R6, #'u'
MOV R5, #'m'
mov 90h, a
IJMP ay_bul

GUN6: JC GUN4
MOV R7, #'c'
MOV R6, #'m'
MOV R5, #'t'
IJMP ay_bul
mov 90h, a
GUN7: JNC GUN2
```

MOV R7, #'p'

MOV R6, #'a'

MOV R5, #'z'

mov 90h, a

IJMP ay_bul

ay_bul:

cjne r0,#0,aktarma

ljmp ter

ter:

cjne r1,#0ffh,NE

MOV 23h,#1h

ljmp son

ljmp arcek

mov 90h, a

aktarma: ljmp cek1

mov 90h, a

arcek:

NE:jnc ata

MOV r2,#8h

MOV dptr,#ay_ilk_gün

ay_bul_loop1:

clr a

MOVC a,@a+dptr

MOV 50h,a

MOV a,r1

cjne a,50h,ay_karsilas_kucuk

ay_karsilas_kucuk: jnc ay_bulundu

MOV dptr,#ay_ilk_gün

MOV a,#8h

clr c

subb a,r2

MOV b,#3h

mul ab

mov 90h, a

add a,#0bh

MOV 52h,a

MOVC a,@a+dptr

MOV 20h,a

MOV a,52h

mov 90h, a

add a,#1

MOV 52h,a

mov 90h, a

MOVC a,@a+dptr

MOV 21h,a

MOV a,52h

add a,#1

MOVC a,@a+dptr

MOV 22h,a

cjne r2,#8,son1

MOV 23h,r1

son1:

cjne r2,#07h,son2

```
MOV a,r1
subb a,#1fh
MOV 23h,a
mov 90h, a
son2:
cjne r2,#06h,son3
MOV a,r1
subb a,#3ch
MOV 23h,a
mov 90h, a
son3:
cjne r2,#05h,son4
MOV a,r1
subb a,#5bh
MOV 23h,a

son4:
cjne r2,#04h,son5
MOV a,r1
subb a,#79h
MOV 23h,a
mov 90h, a
son5:
cjne r2,#03h,son6
MOV a,r1
subb a,#98h
MOV 23h,a
mov 90h, a
son6:
cjne r2,#02h,son7
MOV a,r1
```

subb a,#0b6h

MOV 23h,a

son7:

cjne r2,#01h,son

MOV a,r1

subb a,#0d5h

MOV 23h,a

ljmp son

ata:

MOV A,r1

ljmp son

ay_bulundu:

MOV 80h, #20h

inc dptr

djnz r2,ay_bul_loop1_aktarma

mov 90h, a

cek1:

cjne r1, #13h,sol

ay_bul_loop1_aktarma:

ljmp ay_bul_loop1

sol: jnc git

MOV a,r1

add a,#0ch

MOV 23h,a

ljmp son

git:

cjne r1, #22h,sol1 ;;;;;;;;;;

sol1: jnc git1

```
MOV a,r1
clr c
mov 90h, a
subb a,#12h
MOV 23h,a
ljmp son
```

```
git1:
cjne r1, #50h,sol2
mov 90h, a
sol2: jnc git2
MOV a,r1
clr c
subb a,#31h
MOV 23h,a
ljmp son
```

```
git2:
MOV a,r1
clr c
subb a,#4fh
MOV 23h,a
son: ; ay gün no , gün ad
MOV a ,20h ; ay adı harf1 fetch
ACALL SEND_DATA; ay adı harf1 yolla
MOV a ,21h; ay adı harf2 fetch
ACALL SEND_DATA; ay adı harf2 yolla
MOV a ,22h; ay adı harf3 fetch
ACALL SEND_DATA; ay adı harf3 yolla
MOV a ,#' '
ACALL SEND_DATA
MOV a,23h
```

```
MOV b,#0Ah
div ab
add a,#30h
ACALL SEND_DATA
MOV a,b
add a,#30h
ACALL SEND_DATA
MOV a,#' '
ACALL SEND_DATA
MOV a,r7
ACALL SEND_DATA
MOV a,r6
ACALL SEND_DATA
MOV a,r5
ACALL SEND_DATA
bitti_Sonunda: sjmp bitti_sonunda
org 800
;ASCII look-up table
KCODE0: DB '1', '2', '3', 'A'
KCODE1: DB '4', '5', '6', 'B'
KCODE2: DB '7', '8', '9', 'C'
KCODE3: DB '*', '0', '#', 'D'
org 1200

ay_ilk_gün: db
20h,3Dh,5Ch,7Ah,99h,0B7h,0D6h,0F5h,13h,32h,50h,'OCASUBMARNISMAYHAZTEMEYLEKIKASARA'
END
```