

Bölüm 11: Kesme Fonksiyonları

Mikroişlemciler

BIOS ve DOS

Hizmet çağırma

INT 10h video

Öznitelik baytı

INT 16h klavye

INT 21h karakter G/Ç

Dizgi ve satır girişi

Tarih ve saat

INT 1Ah ve BIOS veri alanı

Mod 13h grafik

Program sonlandırma

Öğrenme Çıktıları

Hizmet çağırma

BIOS ve DOS hizmetlerini AH'deki fonksiyon numarası ve yazmaç parametreleriyle çağırarak.

Ekran çıktısı

INT 10h ve INT 21h ile karakter, renkli karakter ve dizgi yazdırmak; imleci konumlamak.

Klavye girişi

INT 16h ve INT 21h ile tuş okumak, tuş beklemeden yoklamak, satır okumak.

Saat ve sonlandırma

Tarih/saat bilgisini okuyup yazdırmak; programı RET, INT 20h ya da INT 21h/4Ch ile doğru sonlandırmak.

GİRİŞ

BIOS ve DOS Hizmetleri



- **BIOS** (Basic Input/Output System): anakarttaki ROM'da bulunan, donanımı doğrudan süren yordamlar. İşletim sistemi olmadan da çalışır.
- **DOS**: BIOS'un üstünde; dosya, dizgi, saat ve program yönetimi sunar.
- Her iki katmana da **yazılım kesmesi** (`INT n`) ile ulaşılır; çağıranın hizmet adresini bilmesi gerekmez.

Aynı iş farklı katmanlarda yapılabilir: ekrana karakter **INT 21h/02h** (DOS), **INT 10h/0Eh** (BIOS) ya da doğrudan **B800h** ekran belleğine yazarak. Üst katman daha taşınabilir, alt katman daha hızlı ve esnektir.

GİRİŞ

Hizmet Çağırma Kalıbı

1. **AH** ← fonksiyon (hizmet) numarası
2. Diğer parametreler: AL, BX, CX, DX, DS:DX, ES:BP ...
3. **INT n** (10h, 16h, 21h ...)
4. Sonuç yazmaçlarda: AL, AX, CX, DX, ZF/CF ...

```
mov ah, 02h    ; servis: karakter yaz
mov dl, 'A'    ; parametre
int 21h        ; DOS'u çağır
```

Kesme	Sağlayan	Görev
INT 10h	BIOS	video: mod, imleç, karakter, dizgi
INT 16h	BIOS	klavye
INT 1Ah	BIOS	saat sayacı, gerçek zaman saati
INT 13h	BIOS	disk
INT 20h	DOS	programı sonlandır (COM)
INT 21h	DOS	yüzlerce hizmet: G/Ç, dosya, saat, çıkış

Hizmetler kendi kullandıkları yazmaçları değiştirebilir (ör. INT 21h/02h ve 09h AL'yi). Sonra gerekecek değerleri **başka yazmaçta** ya da yığında saklayın.

GİRİŞ

Bir Hizmet Çağrısı Adım Adım: INT 21h

Kesme vektör tablosu

0040h	INT 10h → BIOS video
0058h	INT 16h → BIOS klavye
0080h	INT 20h → DOS
0084h	INT 21h → DOS (IP)
0086h	INT 21h → DOS (CS)

Yığın

boş

Yürütülen yer

0700:0105h

Çağrı öncesi: parametreler yazmaçlarda (AH = 09h, DX = msg). **CS:IP = 0700:0105h**,
komut **int 21h**.

Adım 1 / 5

Vektör tablosundaki adresler sistemden sisteme değişir; bu yüzden programlar hizmetleri adresle değil kesme numarasıyla çağırır. Ayrıntılar Bölüm 5: Kesmeler'de.

GİRİŞ

Bir Hizmet Çağrısı Adım Adım: INT 21h

Kesme vektör tablosu

0040h	INT 10h → BIOS video
0058h	INT 16h → BIOS klavye
0080h	INT 20h → DOS
0084h	INT 21h → DOS (IP)
0086h	INT 21h → DOS (CS)

Yığın

boş

Yürütülen yer

0700:0105h

$21h \times 4 = 0084h$

İşlemci vektör adresini hesaplar: $21h \times 4 = 84h \rightarrow 0000:0084h$.

Adım 2 / 5

Vektör tablosundaki adresler sistemden sisteme değişir; bu yüzden programlar hizmetleri adresle değil kesme numarasıyla çağırır. Ayrıntılar Bölüm 5: Kesmeler'de.

GİRİŞ

Bir Hizmet Çağrısı Adım Adım: INT 21h

Kesme vektör tablosu

0040h	INT 10h → BIOS video
0058h	INT 16h → BIOS klavye
0080h	INT 20h → DOS
0084h	INT 21h → DOS (IP)
0086h	INT 21h → DOS (CS)

Yığın

FFCCh	FLAGS
FFFAh	CS = 0700h
FFF8h	IP = 0107h

Yürütülen yer

0700:0105h

FLAGS, CS ve dönüş IP'si (**0107h**) yığına atılır; IF = TF = 0.

Adım 3 / 5

Vektör tablosundaki adresler sistemden sisteme değişir; bu yüzden programlar hizmetleri adresle değil kesme numarasıyla çağırır. Ayrıntılar Bölüm 5: Kesmeler'de.

GİRİŞ

Bir Hizmet Çağrısı Adım Adım: INT 21h

Kesme vektör tablosu

0040h	INT 10h → BIOS video
0058h	INT 16h → BIOS klavye
0080h	INT 20h → DOS
0084h	INT 21h → DOS (IP)
0086h	INT 21h → DOS (CS)

Yığın

FFFC h	FLAGS
FFFA h	CS = 0700h
FFF8 h	IP = 0107h

Yürütülen yer

DOS: 09h servisi

CS:IP ← [0000:0086h]:[0000:0084h]. DOS'un INT 21h yordamı AH'ye bakarak **09h** hizmetine dallanır.

Adım 4 / 5

Vektör tablosundaki adresler sistemden sisteme değişir; bu yüzden programlar hizmetleri adresle değil kesme numarasıyla çağırır. Ayrıntılar Bölüm 5: Kesmeler'de.

GİRİŞ

Bir Hizmet Çağrısı Adım Adım: INT 21h

Kesme vektör tablosu

0040h	INT 10h → BIOS video
0058h	INT 16h → BIOS klavye
0080h	INT 20h → DOS
0084h	INT 21h → DOS (IP)
0086h	INT 21h → DOS (CS)

Yığın

boş

Yürütülen yer

0700:0107h

Hizmet biter, **iret** ile IP, CS, FLAGS geri alınır. Sonuçlar yazmaçlarda (09h için AL = 24h).

Adım 5 / 5

Vektör tablosundaki adresler sistemden sisteme değişir; bu yüzden programlar hizmetleri adresle değil kesme numarasıyla çağırır. Ayrıntılar Bölüm 5: Kesmeler'de.

GİRİŞ

Ekran ve Klavyede ASCII

Kod	Ad	Etki (0Eh / 02h)
07h	BEL	zil sesi
08h	BS	imleç bir geri
09h	TAB	sekme (DOS)
0Ah	LF	bir alt satır
0Dh	CR	satır başı (Enter tuşu)
1Bh	ESC	Esc tuşu

Karakter	ASCII	İlişki
boşluk	20h	
'0' ... '9'	30h-39h	rakam = kod - 30h
'A' ... 'Z'	41h-5Ah	bit 5 = 0
'a' ... 'z'	61h-7Ah	bit 5 = 1 (+20h)
'\$'	24h	09h hizmetinde dizgi sonu

yalnız CR (0Dh)

ABC

yalnız LF (0Ah)

ABC

CR + LF

ABC

Yeni satır için **iki** karakter gerekir: 0Dh, 0Ah (CR LF). Yalnız LF imleci aşağı indirir ama satır başına getirmez. Türkçe harfler (ç, ğ, ı, ö, ş, ü) 7 bitlik ASCII'de yoktur; kod sayfasına (CP857) göre gösterilir.

INT 10h: BIOS Video Hizmetleri

AH	Hizmet	Giriş	Çıkış / not
00h	video modu seç	AL = mod (03h: 80×25 renkli metin)	ekran temizlenir
01h	imleç şekli	CH = başlangıç, CL = bitiş satırı	CH = 20h → imleç gizli
02h	imleç konumla	BH = sayfa, DH = satır, DL = sütun	
03h	imleç konumu oku	BH = sayfa	DH, DL = konum; CH, CL = şekil
06h	pencereyi yukarı kaydır	AL = satır (0 = temizle), BH = öznitelik, CH,CL – DH,DL = köşeler	ekran/pencere silme
09h	karakter + öznitelik yaz	AL = karakter, BH = sayfa, BL = öznitelik, CX = sayı	imleç ilerlemez
0Ah	yalnız karakter yaz	AL, BH, CX	mevcut öznitelik korunur
0Eh	teletype çıkışı	AL = karakter, BH = sayfa	imleç ilerler; CR, LF, BS, BEL yorumlanır
13h	dizgi yaz	AL = mod, BH, BL, CX = uzunluk, DH,DL, ES:BP = dizgi	AT ve sonrası BIOS

AH = 00h: Video Modu ve Metin Ekranı

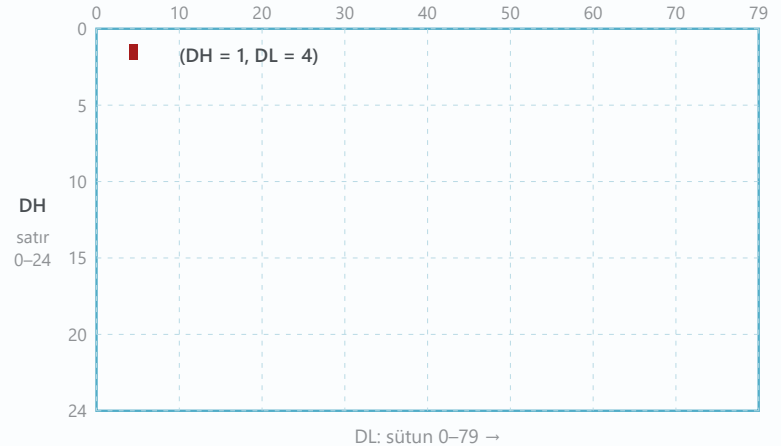
AL	Mod	Bellek
01h	40×25 metin, 16 renk	B800h
03h	80×25 metin, 16 renk	B800h
07h	80×25 tek renk	B000h
04h	320×200, 4 renk	B800h
13h	320×200, 256 renk	A000h

```

mov ah, 00h
mov al, 03h    ; ya da mov ax, 0003h
int 10h        ; temiz 80×25 ekran

```

emu8086 ekranı varsayılan olarak 80×25 metin modundadır; grafik modları sınırlı desteklenir.



İmleç Konumu: AH = 02h / 03h

- Satır **DH** (0–24), sütun **DL** (0–79); sol üst köşe (0, 0).
- BH** = görüntü sayfası. 80×25 modda 8 sayfa (0–7) vardır; normalde 0 kullanılır.
- İkisi tek komutla: `mov dx, 0C28h` → DH = 12, DL = 40 (ekranın ortası).
- AH = 03h konumu okur: dönüşte DH, DL; ayrıca CH, CL imleç şekli.
- INT 21h ile yazılan karakterler de bu imleç konumundan başlar.

(12, 40) → (12, 41)



; imleci ortaya al

`mov ah, 02h`

`mov bh, 0`

`mov dx, 0C28h` ; 12. satır, 40. sütun

`int 10h`

; imleci bir sağa kaydır

`mov ah, 03h`

`mov bh, 0`

`int 10h` ; DH, DL = konum

`inc dl`

`mov ah, 02h`

`int 10h`

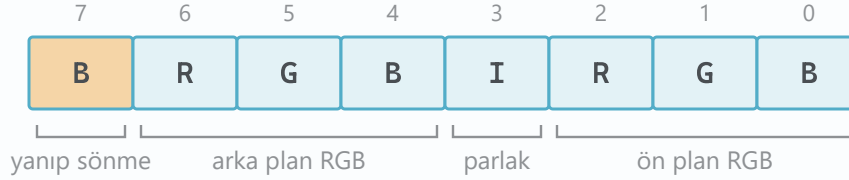
DH = 0Ch = 12 (satır), DL = 28h = 40 (sütun)

03h ile oku → inc dl → 02h ile yaz: (12, 41)

bağıl konum $(12 \times 80 + 40) \times 2 = 07D0h$

Ekran belleği karşılığı: (satır, sütun) hücresinin bağıl konumu = **(satır × 80 + sütun) × 2**. Örneğin (1, 4) → $84 \times 2 = \mathbf{00A8h}$; çift adreste karakter, tek adreste öznitelik.

Öznitelik (Attribute) Baytı



Örnek: 1Eh = 0001 1110b → mavi zemin, sarı (parlak kahverengi) yazı

0 siyah	1 mavi	2 yeşil	3 camgöbeği
4 kırmızı	5 eflatun	6 kahve	7 açık gri
8 koyu gri	9 açık mavi	A açık yeşil	B açık camg.
C açık kırmızı	D açık eflatun	E sarı	F beyaz

- Öznitelik = arka plan × 16 + ön plan. **07h** varsayılan (siyah üstüne açık gri), **1Fh** mavi üstüne beyaz, **4Eh** kırmızı üstüne sarı.
- Bit 7 = 1 ise karakter yanıp söner (ya da BIOS ayarına göre arka plan parlak olur): **8Fh**.

Öznitelik Gezgini

Ön plan

E sarı



Arka plan

4 kırmızı



☐ yanıp sönme Karakter

A



0 1 0 0 1 1 1 0

BL = 4Eh

```
mov ah, 09h · mov al, 'A' · mov bh, 0 · mov bl, 4Eh · mov cx, 3 · int 10h
```

Seçilen renklerden öznitelik baytı ve INT 10h/09h çağrısı üretilir.

Karakter Yazma: AH = 09h mı, 0Eh mi?

Özellik	AH = 09h	AH = 0Eh (teletype)
Girişler	AL = karakter, BH = sayfa, BL = öznitelik, CX = tekrar	AL = karakter, BH = sayfa
Renk	var (BL)	metin modunda yok (hücresinin rengi korunur)
İmleç	ilerlemez	ilerler, satır sonunda alt satıra geçer
CR, LF, BS, BEL	sembol olarak basılır (␣, ▯ ...)	komut olarak yorumlanır
Ekran kaydırma	yok	son satırda ekran yukarı kayar
Tipik kullanım	renkli kutular, çerçeveler, aynı karakteri CX kez	düz metin akışı

AH = 09h, AL = '*', CX = 3



imleç başta kaldı

AH = 0Eh, AL = 'A'



imleç bir sağda

Renkli metin için yaygın kalıp: önce **09h** ile karakteri renkli yaz (CX = 1), sonra imleci **02h** ile bir sağa taşı; ya da tüm dizgi için **13h** kullan.

Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 ► mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

AH = 00h (mod seç), AL = 03h (80×25, 16 renk metin).
İki yazmaç tek komutla yüklenir.

Adım 1 / 17

DEĞİŞKENLER

AX = 0003h

BX = 0000h

CX = 0000h

DX = 0000h

ÇIKTI



Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 ► int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

`int 10h` : mod ayarlanır, **ekran temizlenir**, imleç (0,0)'a gider.

Adım 2 / 17

DEĞİŞKENLER

AX = 0003h

BX = 0000h

CX = 0000h

DX = 0000h

ÇIKTI



Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
▶ 4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

AH = 02h: imleç konumunu ayarla.

Adım 3 / 17

DEĞİŞKENLER

AX = 0203h

BX = 0000h

CX = 0000h

DX = 0000h

ÇIKTI



Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 ► mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

BH = görüntü sayfası (metin modunda 0–7; normalde 0).

Adım 4 / 17

DEĞİŞKENLER

AX = 0203h

BX = 0000h

CX = 0000h

DX = 0000h

ÇIKTI



Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 ► mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

DX = 0104h → **DH = 1** (satır), **DL = 4** (sütun). Sol üst köşe (0,0).

Adım 5 / 17

DEĞİŞKENLER

AX = 0203h

BX = 0000h

CX = 0000h

DX = 0104h

ÇIKTI



Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 ► int 10h
8 mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

İmleç satır 1, sütun 4'e taşınır. Ekranda henüz bir şey yok.

Adım 6 / 17

DEĞİŞKENLER

AX = 0203h

BX = 0000h

CX = 0000h

DX = 0104h

ÇIKTI



Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 ► mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

AH = 09h: imleç konumuna AL'yi BL özniteliğiyle CX kez yaz.

Adım 7 / 17

DEĞİŞKENLER

AX = 0903h

BX = 0000h

CX = 0000h

DX = 0104h

ÇIKTI



Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 mov ah, 09h ; karakter + öznitelik
▶ mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

Yazılacak karakter AL = '*' (2Ah).

Adım 8 / 17

DEĞİŞKENLER

AX = 092Ah

BX = 0000h

CX = 0000h

DX = 0104h

ÇIKTI



Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
▶ 10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

BL = 4Eh: arka plan 4 (kırmızı), ön plan E (sarı).

Adım 9 / 17

DEĞİŞKENLER

AX = 092Ah

BX = 004Eh

CX = 0000h

DX = 0104h

ÇIKTI



Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 ► mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

Tekrar sayısı CX = 3.

Adım 10 / 17

DEĞİŞKENLER

AX = 092Ah

BX = 004Eh

CX = 0003h

DX = 0104h

ÇIKTI



Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h      ; AH=00h, AL=03h
3 int 10h            ; ekran silinir
4 mov ah, 02h        ; imleç konumla
5 mov bh, 0          ; sayfa 0
6 mov dx, 0104h      ; DH=1, DL=4
7 int 10h
8 mov ah, 09h        ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh        ; kırmızı üstüne sarı
11 mov cx, 3          ; 3 kez
12 ► int 10h          ; imleç ilerlemez!
13 mov ah, 0Eh        ; teletype
14 mov al, 'A'
15 int 10h            ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

Ekrana *** yazılır, ama 09h **imleci ilerletmez**: imleç hâlâ (1,4).

Adım 11 / 17

DEĞİŞKENLER

AX = 092Ah

BX = 004Eh

CX = 0003h

DX = 0104h

ÇIKTI

Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
▶ 13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

AH = 0Eh: teletype (daktilo) çıkışı.

Adım 12 / 17

DEĞİŞKENLER

AX = 0E2Ah

BX = 004Eh

CX = 0003h

DX = 0104h

ÇIKTI

Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h      ; AH=00h, AL=03h
3 int 10h            ; ekran silinir
4 mov ah, 02h        ; imleç konumla
5 mov bh, 0          ; sayfa 0
6 mov dx, 0104h      ; DH=1, DL=4
7 int 10h
8 mov ah, 09h        ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh        ; kırmızı üstüne sarı
11 mov cx, 3          ; 3 kez
12 int 10h            ; imleç ilerlemez!
13 mov ah, 0Eh        ; teletype
14 ► mov al, 'A'
15 int 10h            ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

AL = 'A'.

Adım 13 / 17

DEĞİŞKENLER

AX = 0E41h

BX = 004Eh

CX = 0003h

DX = 0104h

ÇIKTI

Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h      ; AH=00h, AL=03h
3 int 10h            ; ekran silinir
4 mov ah, 02h        ; imleç konumla
5 mov bh, 0          ; sayfa 0
6 mov dx, 0104h      ; DH=1, DL=4
7 int 10h
8 mov ah, 09h        ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh        ; kırmızı üstüne sarı
11 mov cx, 3          ; 3 kez
12 int 10h           ; imleç ilerlemez!
13 mov ah, 0Eh        ; teletype
14 mov al, 'A'
15 ► int 10h          ; imleç ilerler
16 mov al, 'B'
17 int 10h
18 ret
```

'A', imlecin bulunduğu (1,4)'e yazılır ve ilk yıldızın **üzerine** gelir; imleç (1,5)'e ilerler.

Adım 14 / 17

DEĞİŞKENLER

AX = 0E41h

BX = 004Eh

CX = 0003h

DX = 0104h

ÇIKTI

A**

Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
▶ 16 mov al, 'B'
17 int 10h
18 ret
```

AL = 'B'. AH hâlâ 0Eh.

Adım 15 / 17

DEĞİŞKENLER

AX = 0E42h

BX = 004Eh

CX = 0003h

DX = 0104h

ÇIKTI

A**

Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
▶ int 10h
18 ret
```

'B' (1,5)'e yazılır: ekranda **AB***. 0Eh metin modunda öznitelik almaz; hücrelerdeki 4Eh rengi korunur.

Adım 16 / 17

DEĞİŞKENLER

AX = 0E42h

BX = 004Eh

CX = 0003h

DX = 0104h

ÇIKTI

AB*

Kod İzleme: Mod, İmleç, 09h ve 0Eh

```
1 org 100h
2 mov ax, 0003h ; AH=00h, AL=03h
3 int 10h ; ekran silinir
4 mov ah, 02h ; imleç konumla
5 mov bh, 0 ; sayfa 0
6 mov dx, 0104h ; DH=1, DL=4
7 int 10h
8 mov ah, 09h ; karakter + öznitelik
9 mov al, '*'
10 mov bl, 4Eh ; kırmızı üstüne sarı
11 mov cx, 3 ; 3 kez
12 int 10h ; imleç ilerlemez!
13 mov ah, 0Eh ; teletype
14 mov al, 'A'
15 int 10h ; imleç ilerler
16 mov al, 'B'
17 int 10h
▶ ret
```

09h imleci ilerletmez; yazdıktan sonra imleç AH = 02h ile ayrıca ilerletilir.

Adım 17 / 17

DEĞİŞKENLER

AX = 0E42h

BX = 004Eh

CX = 0003h

DX = 0104h

ÇIKTI

AB*

Kod İzleme: AH = 13h ile Renkli Dizgi

```
1 org 100h
2 ► mov ax, 1301h ; AH=13h, AL=01h
3 mov bx, 001Eh ; BH=0, BL=1Eh
4 mov cx, 7 ; dizgi uzunluğu
5 mov dx, 0205h ; satır 2, sütun 5
6 lea bp, mesaj ; ES:BP → dizgi
7 int 10h
8 ret
9 mesaj db 'Merhaba'
```

AH = 13h (dizgi yaz). AL = 01h: tüm karakterler BL özniteliğinde, yazdıktan sonra **imleç ilerler**.

Adım 1 / 7

DEĞİŞKENLER

AX = 1301h

BX = 0000h

CX = 0000h

DX = 0000h

BP = 0000h

ÇIKTI



Kod İzleme: AH = 13h ile Renkli Dizgi

```
1 org 100h
2 mov ax, 1301h ; AH=13h, AL=01h
3 ► mov bx, 001Eh ; BH=0, BL=1Eh
4 mov cx, 7 ; dizgi uzunluğu
5 mov dx, 0205h ; satır 2, sütun 5
6 lea bp, mesaj ; ES:BP → dizgi
7 int 10h
8 ret
9 mesaj db 'Merhaba'
```

BH = 0 (sayfa), BL = 1Eh: mavi (1) zemin, sarı (E) yazı.

Adım 2 / 7

DEĞİŞKENLER

AX = 1301h

BX = 001Eh

CX = 0000h

DX = 0000h

BP = 0000h

ÇIKTI



Kod İzleme: AH = 13h ile Renkli Dizgi

```
1 org 100h
2 mov ax, 1301h ; AH=13h, AL=01h
3 mov bx, 001Eh ; BH=0, BL=1Eh
▶ 4 mov cx, 7 ; dizgi uzunluğu
5 mov dx, 0205h ; satır 2, sütun 5
6 lea bp, mesaj ; ES:BP → dizgi
7 int 10h
8 ret
9 mesaj db 'Merhaba'
```

CX = karakter sayısı. '\$' gibi bir bitiş işareti **yoktur**.

Adım 3 / 7

DEĞİŞKENLER

AX = 1301h

BX = 001Eh

CX = 0007h

DX = 0000h

BP = 0000h

ÇIKTI



Kod İzleme: AH = 13h ile Renkli Dizgi

```
1 org 100h
2 mov ax, 1301h ; AH=13h, AL=01h
3 mov bx, 001Eh ; BH=0, BL=1Eh
4 mov cx, 7 ; dizgi uzunluğu
5 ► mov dx, 0205h ; satır 2, sütun 5
6 lea bp, mesaj ; ES:BP → dizgi
7 int 10h
8 ret
9 mesaj db 'Merhaba'
```

DH = 2, DL = 5: yazmanın başlayacağı konum.

Adım 4 / 7

DEĞİŞKENLER

AX = 1301h

BX = 001Eh

CX = 0007h

DX = 0205h

BP = 0000h

ÇIKTI



Kod İzleme: AH = 13h ile Renkli Dizgi

```
1 org 100h
2 mov ax, 1301h ; AH=13h, AL=01h
3 mov bx, 001Eh ; BH=0, BL=1Eh
4 mov cx, 7 ; dizgi uzunluğu
5 mov dx, 0205h ; satır 2, sütun 5
▶ lea bp, mesaj ; ES:BP → dizgi
7 int 10h
8 ret
9 mesaj db 'Merhaba'
```

Dizginin adresi **ES:BP**'de olmalıdır (DS:DX değil!). COM programında ES = DS: BP = 0113h.

Adım 5 / 7

DEĞİŞKENLER

AX = 1301h

BX = 001Eh

CX = 0007h

DX = 0205h

BP = 0113h

ÇIKTI



Kod İzleme: AH = 13h ile Renkli Dizgi

```
1 org 100h
2 mov ax, 1301h ; AH=13h, AL=01h
3 mov bx, 001Eh ; BH=0, BL=1Eh
4 mov cx, 7 ; dizgi uzunluğu
5 mov dx, 0205h ; satır 2, sütun 5
6 lea bp, mesaj ; ES:BP → dizgi
7 ► int 10h
8 ret
9 mesaj db 'Merhaba'
```

Satır 2, sütun 5'ten itibaren **Merhaba** yazılır; imleç (2,12)'ye gider.

Adım 6 / 7

DEĞİŞKENLER

AX = 1301h

BX = 001Eh

CX = 0007h

DX = 0205h

BP = 0113h

ÇIKTI

Merhaba

Kod İzleme: AH = 13h ile Renkli Dizgi

```
1 org 100h
2 mov ax, 1301h    ; AH=13h, AL=01h
3 mov bx, 001Eh    ; BH=0, BL=1Eh
4 mov cx, 7        ; dizgi uzunluğu
5 mov dx, 0205h    ; satır 2, sütun 5
6 lea bp, mesaj    ; ES:BP → dizgi
7 int 10h
8 ► ret
9 mesaj db 'Merhaba'
```

AL = 00h olsaydı dizgi yine yazılır ama imleç eski yerinde kalırdı. AL = 02h/03h'te dizgi karakter, öznitelik çiftlerinden oluşur.

Adım 7 / 7

DEĞİŞKENLER

AX = **1301h**

BX = **001Eh**

CX = **0007h**

DX = **0205h**

BP = **0113h**

ÇIKTI

Merhaba

AH = 06h: Pencere Kaydırma ve Ekran Temizleme

Yazmaç	Anlamı
AL	kaydırılacak satır sayısı; 0 = pencereyi temizle
BH	boşalan satırların özniteliği
CH, CL	sol üst köşe (satır, sütun)
DH, DL	sağ alt köşe (satır, sütun)

- AH = 07h aynı işi aşağı doğru yapar.
- İmleç yerinde kalır; temizledikten sonra 02h ile (0,0)'a alın.

; tüm ekranı mavi zemine temizle

```
mov ax, 0600h ; AH=06h, AL=0
```

```
mov bh, 1Fh ; mavi/beyaz
```

```
mov cx, 0000h ; (0,0)
```

```
mov dx, 184Fh ; (24,79)
```

```
int 10h
```

; ortada 10×40'luk kırmızı kutu

```
mov ax, 0600h
```

```
mov bh, 4Fh
```

```
mov cx, 0714h ; (7,20)
```

```
mov dx, 103Bh ; (16,59)
```

```
int 10h
```

184Fh = satır 24 (18h), sütun 79 (4Fh). Mod 03h'ü yeniden seçmek de ekranı temizler, ama renkleri varsayılan (07h) döndürür.

Ekranı ve Bir Pencereyi Temizlemek

```
1 org 100h
```

```
▶ mov ax, 0600h ; AH=06h, AL=0
```

```
3 mov bh, 1Fh ; mavi/beyaz
```

```
4 mov cx, 0000h ; (0,0)
```

```
5 mov dx, 184Fh ; (24,79)
```

```
6 int 10h
```

```
7 mov ax, 0600h
```

```
8 mov bh, 4Fh ; kırmızı/beyaz
```

```
9 mov cx, 0714h ; (7,20)
```

```
10 mov dx, 103Bh ; (16,59)
```

```
11 int 10h
```

```
12 ret
```

AX

0600h

BH

00h

CX

0000h

DX

0000h

CH, CL = (0, 0)

DH, DL = (0, 0)

AH = 06h (pencereyi yukarı kaydır), AL = 0: kaydırma yerine **temizle**.

Adım 1 / 10

DEĞİŞKENLER

AX = 0600h

BH = 00h

CX = 0000h

DX = 0000h

Yazmaç değerleri yorumlayıcıda çalıştırılarak alındı; yorumlayıcı 06h'nin ekran etkisini benzetmez, ekran BIOS tanımına göre çizildi.

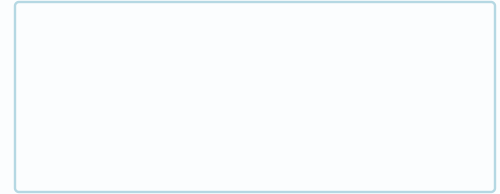
Ekranı ve Bir Pencereyi Temizlemek

```
1 org 100h
2 mov ax, 0600h ; AH=06h, AL=0
3 ► mov bh, 1Fh ; mavi/beyaz
4 mov cx, 0000h ; (0,0)
5 mov dx, 184Fh ; (24,79)
6 int 10h
7 mov ax, 0600h
8 mov bh, 4Fh ; kırmızı/beyaz
9 mov cx, 0714h ; (7,20)
10 mov dx, 103Bh ; (16,59)
11 int 10h
12 ret
```

AX	BH	CX	DX
0600h	1Fh	0000h	0000h

CH,CL = (0, 0)

DH,DL = (0, 0)



(0,0)

(24,79)

BH = 1Fh: boşalan hücrelerin özniteliği (zemin 1 mavi, yazı F beyaz).

Adım 2 / 10

DEĞİŞKENLER

AX = 0600h

BH = 1Fh

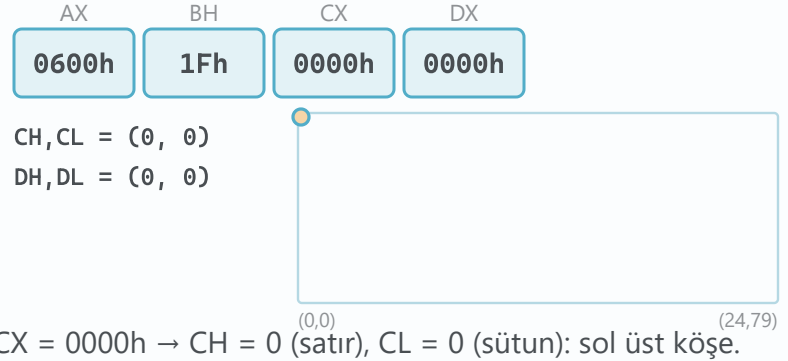
CX = 0000h

DX = 0000h

Yazmaç değerleri yorumlayıcıda çalıştırılarak alındı; yorumlayıcı 06h'nin ekran etkisini benzetmez, ekran BIOS tanımına göre çizildi.

Ekranı ve Bir Pencereyi Temizlemek

```
1 org 100h
2 mov ax, 0600h ; AH=06h, AL=0
3 mov bh, 1Fh ; mavi/beyaz
4 ► mov cx, 0000h ; (0,0)
5 mov dx, 184Fh ; (24,79)
6 int 10h
7 mov ax, 0600h
8 mov bh, 4Fh ; kırmızı/beyaz
9 mov cx, 0714h ; (7,20)
10 mov dx, 103Bh ; (16,59)
11 int 10h
12 ret
```



Adım 3 / 10

DEĞİŞKENLER

AX = 0600h

BH = 1Fh

CX = 0000h

DX = 0000h

Yazmaç değerleri yorumlayıcıda çalıştırılarak alındı; yorumlayıcı 06h'nin ekran etkisini benzetmez, ekran BIOS tanımına göre çizildi.

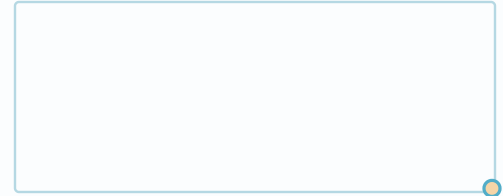
Ekranı ve Bir Pencereyi Temizlemek

```
1 org 100h
2 mov ax, 0600h ; AH=06h, AL=0
3 mov bh, 1Fh ; mavi/beyaz
4 mov cx, 0000h ; (0,0)
5 ► mov dx, 184Fh ; (24,79)
6 int 10h
7 mov ax, 0600h
8 mov bh, 4Fh ; kırmızı/beyaz
9 mov cx, 0714h ; (7,20)
10 mov dx, 103Bh ; (16,59)
11 int 10h
12 ret
```

AX	BH	CX	DX
0600h	1Fh	0000h	184Fh

CH,CL = (0, 0)

DH,DL = (24, 79)



DX = 184Fh → DH = 18h = **24**, DL = 4Fh = **79**: sağ alt köşe.

Adım 4 / 10

DEĞİŞKENLER

AX = 0600h

BH = 1Fh

CX = 0000h

DX = 184Fh

Yazmaç değerleri yorumlayıcıda çalıştırılarak alındı; yorumlayıcı 06h'nin ekran etkisini benzetmez, ekran BIOS tanımına göre çizildi.

Ekranı ve Bir Pencereyi Temizlemek

```
1 org 100h
2 mov ax, 0600h    ; AH=06h, AL=0
3 mov bh, 1Fh      ; mavi/beyaz
4 mov cx, 0000h    ; (0,0)
5 mov dx, 184Fh    ; (24,79)
```

► int 10h

```
7 mov ax, 0600h
8 mov bh, 4Fh      ; kırmızı/beyaz
9 mov cx, 0714h    ; (7,20)
10 mov dx, 103Bh   ; (16,59)
11 int 10h
12 ret
```

AX

0600h

BH

1Fh

CX

0000h

DX

184Fh

CH,CL = (0, 0)

DH,DL = (24, 79)



INT 10h: 80 × 25 ekranın tamamı mavi zemine silinir. İmleç yerinde kalır.

Adım 5 / 10

DEĞİŞKENLER

AX = 0600h

BH = 1Fh

CX = 0000h

DX = 184Fh

Yazmaç değerleri yorumlayıcıda çalıştırılarak alındı; yorumlayıcı 06h'nin ekran etkisini benzetmez, ekran BIOS tanımına göre çizildi.

Ekranı ve Bir Pencereyi Temizlemek

```
1 org 100h
2 mov ax, 0600h    ; AH=06h, AL=0
3 mov bh, 1Fh      ; mavi/beyaz
4 mov cx, 0000h    ; (0,0)
5 mov dx, 184Fh    ; (24,79)
6 int 10h
7 ► mov ax, 0600h
8 ► mov bh, 4Fh     ; kırmızı/beyaz
9 mov cx, 0714h    ; (7,20)
10 mov dx, 103Bh   ; (16,59)
11 int 10h
12 ret
```

AX	BH	CX	DX
0600h	4Fh	0000h	184Fh

CH,CL = (0, 0)

DH,DL = (24, 79)



İkinci çağrı: AX = 0600h, BH = 4Fh (kırmızı zemin, beyaz yazı).

Adım 6 / 10

DEĞİŞKENLER

AX = 0600h

BH = 4Fh

CX = 0000h

DX = 184Fh

Yazmaç değerleri yorumlayıcıda çalıştırılarak alındı; yorumlayıcı 06h'nin ekran etkisini benzetmez, ekran BIOS tanımına göre çizildi.

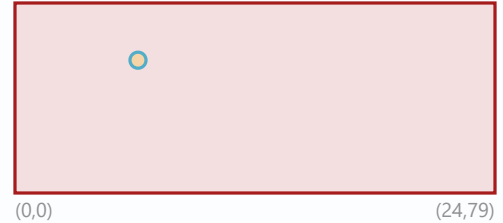
Ekranı ve Bir Pencereyi Temizlemek

```
1 org 100h
2 mov ax, 0600h    ; AH=06h, AL=0
3 mov bh, 1Fh      ; mavi/beyaz
4 mov cx, 0000h    ; (0,0)
5 mov dx, 184Fh    ; (24,79)
6 int 10h
7 mov ax, 0600h
8 mov bh, 4Fh      ; kırmızı/beyaz
9 ► mov cx, 0714h  ; (7,20)
10 mov dx, 103Bh   ; (16,59)
11 int 10h
12 ret
```

AX	BH	CX	DX
0600h	4Fh	0714h	184Fh

CH,CL = (7, 20)

DH,DL = (24, 79)



CX = 0714h → (7, 20).

Adım 7 / 10

DEĞİŞKENLER

AX = 0600h

BH = 4Fh

CX = 0714h

DX = 184Fh

Yazmaç değerleri yorumlayıcıda çalıştırılarak alındı; yorumlayıcı 06h'nin ekran etkisini benzetmez, ekran BIOS tanımına göre çizildi.

Ekranı ve Bir Pencereyi Temizlemek

```
1 org 100h
2 mov ax, 0600h ; AH=06h, AL=0
3 mov bh, 1Fh ; mavi/beyaz
4 mov cx, 0000h ; (0,0)
5 mov dx, 184Fh ; (24,79)
6 int 10h
7 mov ax, 0600h
8 mov bh, 4Fh ; kırmızı/beyaz
9 mov cx, 0714h ; (7,20)
▶ mov dx, 103Bh ; (16,59)
11 int 10h
12 ret
```

AX	BH	CX	DX
0600h	4Fh	0714h	103Bh

CH,CL = (7, 20)

DH,DL = (16, 59)



DX = 103Bh → (16, 59): 10 satır × 40 sütunluk kutu.

Adım 8 / 10

DEĞİŞKENLER

AX = 0600h

BH = 4Fh

CX = 0714h

DX = 103Bh

Yazmaç değerleri yorumlayıcıda çalıştırılarak alındı; yorumlayıcı 06h'nin ekran etkisini benzetmez, ekran BIOS tanımına göre çizildi.

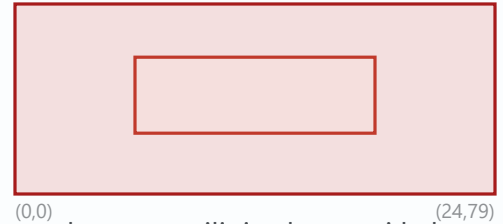
Ekranı ve Bir Pencereyi Temizlemek

```
1 org 100h
2 mov ax, 0600h    ; AH=06h, AL=0
3 mov bh, 1Fh      ; mavi/beyaz
4 mov cx, 0000h    ; (0,0)
5 mov dx, 184Fh    ; (24,79)
6 int 10h
7 mov ax, 0600h
8 mov bh, 4Fh      ; kırmızı/beyaz
9 mov cx, 0714h    ; (7,20)
10 mov dx, 103Bh   ; (16,59)
▶ int 10h
12 ret
```

AX	BH	CX	DX
0600h	4Fh	0714h	103Bh

CH,CL = (7, 20)

DH,DL = (16, 59)



INT 10h: yalnız bu pencere kırmızıya silinir; dışı mavi kalır.

Adım 9 / 10

DEĞİŞKENLER

AX = 0600h

BH = 4Fh

CX = 0714h

DX = 103Bh

Yazmaç değerleri yorumlayıcıda çalıştırılarak alındı; yorumlayıcı 06h'nin ekran etkisini benzetmez, ekran BIOS tanımına göre çizildi.

Ekranı ve Bir Pencereyi Temizlemek

```
1 org 100h
2 mov ax, 0600h ; AH=06h, AL=0
3 mov bh, 1Fh ; mavi/beyaz
4 mov cx, 0000h ; (0,0)
5 mov dx, 184Fh ; (24,79)
6 int 10h
7 mov ax, 0600h
8 mov bh, 4Fh ; kırmızı/beyaz
9 mov cx, 0714h ; (7,20)
10 mov dx, 103Bh ; (16,59)
11 int 10h
▶ ret
```

AX	BH	CX	DX
0600h	4Fh	0714h	103Bh

CH,CL = (7, 20)

DH,DL = (16, 59)



RET: program biter.

Adım 10 / 10

DEĞİŞKENLER

AX = 0600h

BH = 4Fh

CX = 0714h

DX = 103Bh

Yazmaç değerleri yorumlayıcıda çalıştırılarak alındı; yorumlayıcı 06h'nin ekran etkisini benzetmez, ekran BIOS tanımına göre çizildi.

BIOS'suz Çıktı: B800h Ekran Belleği

- Mod 03h'te ekran, **B800:0000h**'dan başlayan 4000 baytlık bellekte tutulur ($80 \times 25 \times 2$).
- Her hücre 2 bayt: **karakter** (çift adres) + **öznitelik** (tek adres).
- Belleğe yazılan değer ekran kartı tarafından anında gösterilir: kesme çağrısından çok daha hızlı.
- Sözcük olarak yazarken little-endian: **4E41h** → 41h ('A') düşük adrese, 4Eh özniteliğe.



; (1,4) hücrelerine sarı/kırmızı 'A'

```
mov ax, 0B800h
```

```
mov es, ax
```

```
mov di, (1*80+4)*2 ; 00A8h
```

```
mov ax, 4E41h
```

```
mov es:[di], ax
```

; tüm ekranı '.' ile doldur

```
xor di, di
```

```
mov ax, 072Eh
```

```
mov cx, 2000
```

```
cld
```

```
rep stosw
```

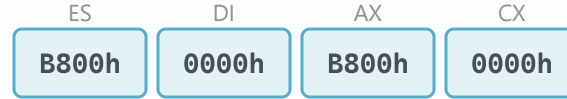
```
mov es:[di], ax (AX = 4E41h)
```

düşük bayt 41h → çift adres (karakter)

Doğrudan yazma imleci **hareket ettirmez** ve BIOS'un tuttuğu imleç bilgisini güncellemez. emu8086 bu belleği de destekler.

Ekran Belleğine Doğrudan Yazmak

```
1 org 100h
2 ▶ mov ax, 0B800h
3 ▶ mov es, ax
4 mov di, (1*80+4)*2 ; 00A8h
5 mov ax, 4E41h
6 mov es:[di], ax
7 xor di, di
8 mov ax, 072Eh
9 mov cx, 3 ; ilk 3 hücre
10 cld
11 rep stosw
12 ret
```



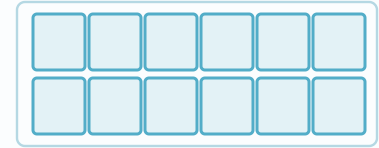
B800:0000...



B800:00A8



karakter · öznitelik



ekran: satır 0-1, sütun 0-5

ES = **B800h**: mod 03h metin ekranının kesimi. Kesim yazmacına sabit doğrudan yazılamadığı için AX üzerinden.

Adım 1 / 9

DEĞİŞKENLER

ES = B800h

DI = 0000h

AX = B800h

CX = 0000h

Ekran Belleğine Doğrudan Yazmak

```
1 org 100h
2 mov ax, 0B800h
3 mov es, ax
4 ► mov di, (1*80+4)*2 ; 00A8h
5 mov ax, 4E41h
6 mov es:[di], ax
7 xor di, di
8 mov ax, 072Eh
9 mov cx, 3 ; ilk 3 hücre
10 cld
11 rep stosw
12 ret
```



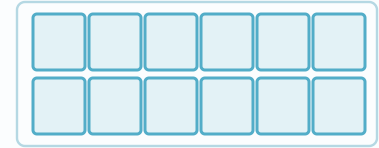
B800:0000...



B800:00A8



karakter · öznitelik



ekran: satır 0-1, sütun 0-5

(1, 4) hücresi: $(1 \times 80 + 4) \times 2 = 168 = 00A8h$. Çarpım derleme zamanında hesaplanır.

Adım 2 / 9

DEĞİŞKENLER

ES = B800h

DI = 00A8h

AX = B800h

CX = 0000h

Ekran Belleğine Doğrudan Yazmak

```
1 org 100h
2 mov ax, 0B800h
3 mov es, ax
4 mov di, (1*80+4)*2 ; 00A8h
5 ► mov ax, 4E41h
6 mov es:[di], ax
7 xor di, di
8 mov ax, 072Eh
9 mov cx, 3 ; ilk 3 hücre
10 cld
11 rep stosw
12 ret
```



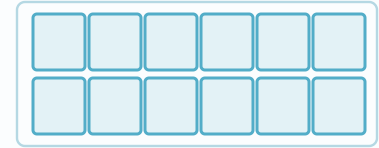
B800:0000...



B800:00A8



karakter · öznitelik



ekran: satır 0–1, sütun 0–5

AX = 4E41h: AL = 41h ('A', karakter), AH = 4Eh (öznitelik: kırmızı zemin, sarı yazı).

Adım 3 / 9

DEĞİŞKENLER

ES = B800h

DI = 00A8h

AX = 4E41h

CX = 0000h

Ekran Belleğine Doğrudan Yazmak

```
1 org 100h
2 mov ax, 0B800h
3 mov es, ax
4 mov di, (1*80+4)*2 ; 00A8h
5 mov ax, 4E41h
▶ mov es:[di], ax
7 xor di, di
8 mov ax, 072Eh
9 mov cx, 3 ; ilk 3 hücre
10 cld
11 rep stosw
12 ret
```



MOV ES:[DI], AX: little-endian → B80A8h ← **41h**, B80A9h ← **4Eh**. Karakter anında ekranda; imleç yerinde kalır.

Adım 4 / 9

DEĞİŞKENLER

ES = B800h

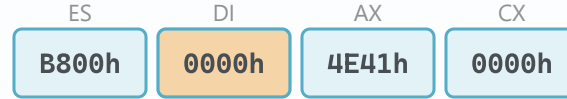
DI = 00A8h

AX = 4E41h

CX = 0000h

Ekran Belleğine Doğrudan Yazmak

```
1 org 100h
2 mov ax, 0B800h
3 mov es, ax
4 mov di, (1*80+4)*2 ; 00A8h
5 mov ax, 4E41h
6 mov es:[di], ax
7 xor di, di
8 mov ax, 072Eh
9 mov cx, 3 ; ilk 3 hücre
10 cld
11 rep stosw
12 ret
```



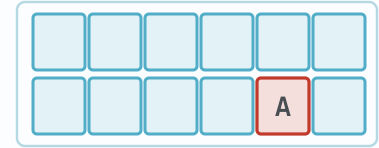
B800:0000...



B800:00A8



karakter · öznitelik



ekran: satır 0-1, sütun 0-5

DI = 0: sol üst köşe.

Adım 5 / 9

DEĞİŞKENLER

ES = B800h

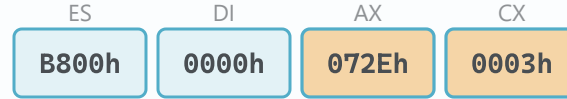
DI = 0000h

AX = 4E41h

CX = 0000h

Ekran Belleğine Doğrudan Yazmak

```
1 org 100h
2 mov ax, 0B800h
3 mov es, ax
4 mov di, (1*80+4)*2 ; 00A8h
5 mov ax, 4E41h
6 mov es:[di], ax
7 xor di, di
▶ mov ax, 072Eh
▶ mov cx, 3 ; ilk 3 hücre
10 cld
11 rep stosw
12 ret
```



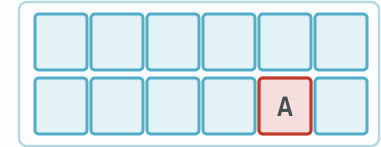
B800:0000...



B800:00A8



karakter · öznitelik



ekran: satır 0-1, sütun 0-5

AX = 072Eh ('.' + 07h açık gri), CX = 3.

Adım 6 / 9

DEĞİŞKENLER

ES = B800h

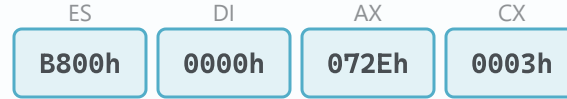
DI = 0000h

AX = 072Eh

CX = 0003h

Ekran Belleğine Doğrudan Yazmak

```
1 org 100h
2 mov ax, 0B800h
3 mov es, ax
4 mov di, (1*80+4)*2 ; 00A8h
5 mov ax, 4E41h
6 mov es:[di], ax
7 xor di, di
8 mov ax, 072Eh
9 mov cx, 3 ; ilk 3 hücre
▶ cld
11 rep stosw
12 ret
```



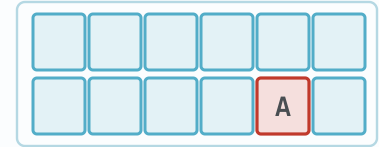
B800:0000...



B800:00A8



karakter · öznitelik



ekran: satır 0-1, sütun 0-5

CLD: DF = 0, DI artarak ilerler.

Adım 7 / 9

DEĞİŞKENLER

ES = B800h

DI = 0000h

AX = 072Eh

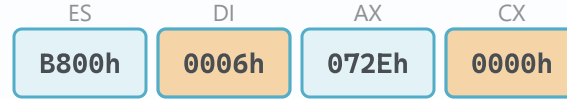
CX = 0003h

Ekran Belleğine Doğrudan Yazmak

```

1 org 100h
2 mov ax, 0B800h
3 mov es, ax
4 mov di, (1*80+4)*2 ; 00A8h
5 mov ax, 4E41h
6 mov es:[di], ax
7 xor di, di
8 mov ax, 072Eh
9 mov cx, 3 ; ilk 3 hücre
10 cld
11 ► rep stosw
12 ret

```



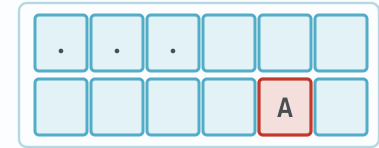
B800:0000...



B800:00A8



karakter · öznitelik



ekran: satır 0-1, sütun 0-5

REP STOSW: ES:[DI] ← AX, DI += 2, CX kez → ilk 3 hücre '.' oldu; DI = 0006h, CX = 0. (Tüm ekran için CX = 2000.)

Adım 8 / 9

DEĞİŞKENLER

ES = B800h

DI = 0006h

AX = 072Eh

CX = 0000h

Ekran Belleğine Doğrudan Yazmak

```
1 org 100h
2 mov ax, 0B800h
3 mov es, ax
4 mov di, (1*80+4)*2 ; 00A8h
5 mov ax, 4E41h
6 mov es:[di], ax
7 xor di, di
8 mov ax, 072Eh
9 mov cx, 3 ; ilk 3 hücre
10 cld
11 rep stosw
▶ ret
```



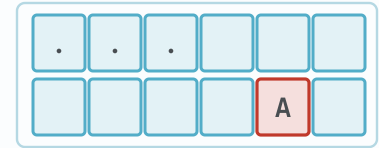
B800:0000...



B800:00A8



karakter · öznitelik



ekran: satır 0-1, sütun 0-5

RET: program biter.

Adım 9 / 9

DEĞİŞKENLER

ES = B800h

DI = 0006h

AX = 072Eh

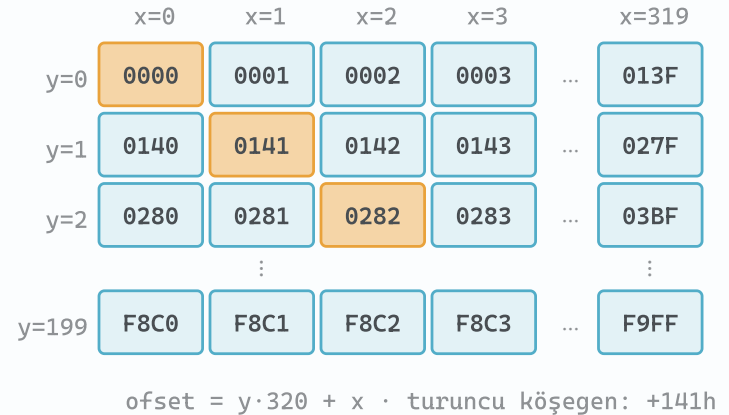
CX = 0000h

Mod 13h: 320×200, 256 Renk

- `mov ax, 0013h` / `int 10h` : 320 × 200 grafik modu. Görüntü belleği **A000:0000h – A000:F9FFh** (64 000 bayt).
- Her piksel **1 bayt**: renk numarası (palet girişi). İlk 16 renk metin renkleriyle aynıdır: 0Eh sarı, 0Fh beyaz.
- Satır başına 320 = **140h** bayt: **bağıl konum = $y \cdot 320 + x$** .
- Örnek: (100, 50) → $50 \cdot 320 + 100 = 16\,100 = 3EE4h$.

INT 10h / AH = 0Ch: piksel yaz	
AL	renk
BH	sayfa (mod 13h'te 0)
CX / DX	x (sütun 0–319) / y (satır 0–199)

AH = 0Dh aynı CX, DX ile pikselin rengini AL'ye okur.



Köşegen Çizgi: BIOS ile ve A000h ile

INT 10h / 0Ch: koordinatla

```
mov ax, 0013h
int 10h          ; 320x200, 256 renk
xor cx, cx       ; x = 0
xor dx, dx       ; y = 0
mov bh, 0        ; sayfa
nokta: mov ax, 0C0Eh ; AH=0Ch, AL=sarı
int 10h
inc cx           ; bir sağ
inc dx           ; bir alt
cmp dx, 200
jb nokta         ; 200 piksel
```

Neden 141h? +140h (320) bir alt satırın aynı sütununa, +1 bir sağ sütuna gider: 141h = 321. Son piksel (199, 199) = $199 \cdot 321 = \text{F987h}$. Sağ üstten inen ters köşegen için 13Fh (320 - 1) eklenir.

emu8086'nın grafik desteği sınırlıdır: 0Ch yavaş çizer, A000h'a doğrudan yazma her sürümde görünmeyebilir; DOSBox'ta tam görünür.

A000h belleği: bağıl konumla

```
mov ax, 0013h
int 10h
mov ax, 0A000h
mov es, ax
xor di, di       ; (0,0) → offset 0
mov al, 0Eh      ; sarı
mov cx, 200
ciz: mov es:[di], al
add di, 141h     ; 320 + 1
loop ciz
mov ah, 0
int 16h          ; tuş bekle
mov ax, 0003h
int 10h          ; metin moduna dön
```

Belleğe doğrudan yazma, her piksel için kesme çağırmaktan çok daha hızlıdır.

Mod 13h Belleğine Köşegen Çizmek

```
1 org 100h
▶ mov ax, 0A000h
▶ mov es, ax
4 xor di, di      ; (0,0) → offset 0
5 mov al, 0Eh     ; sarı
6 mov cx, 4       ; ilk 4 piksel
7 ciz: mov es:[di], al
8 add di, 141h    ; 320 + 1
9 loop ciz
10 ret
```

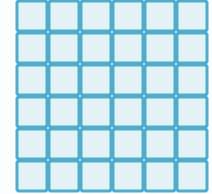
ES	DI	CX	AL
A000h	0000h	0000h	00h

DI = 0 → **y = 0, x = 0**

satır başına 320 = 140h bayt

+141h = bir alt, bir sağ

ekranın sol üstü (6 × 6 piksel)



ES = **A000h**: mod 13h görüntü belleği (320 × 200, piksel başına 1 bayt).

Adım 1 / 8

DEĞİŞKENLER

ES = A000h

DI = 0000h

CX = 0000h

Mod 13h Belleğine Köşegen Çizmek

```
1 org 100h
2 mov ax, 0A000h
3 mov es, ax
4 ► xor di, di ; (0,0) → offset 0
5 mov al, 0Eh ; sarı
6 mov cx, 4 ; ilk 4 piksel
7 ciz: mov es:[di], al
8 add di, 141h ; 320 + 1
9 loop ciz
10 ret
```

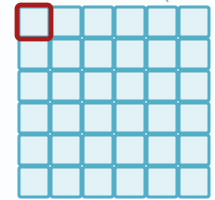
ES	DI	CX	AL
A000h	0000h	0000h	00h

DI = 0 → y = 0, x = 0

satır başına 320 = 140h bayt

+141h = bir alt, bir sağ

ekranın sol üstü (6 × 6 piksel)



DI = 0: (x, y) = (0, 0) piksel bağlı konumu $y \cdot 320 + x = 0$.

Adım 2 / 8

DEĞİŞKENLER

ES = A000h

DI = 0000h

CX = 0000h

Mod 13h Belleğine Köşegen Çizmek

```
1 org 100h
2 mov ax, 0A000h
3 mov es, ax
4 xor di, di      ; (0,0) → offset 0
5 ▶ mov al, 0Eh   ; sarı
6 ▶ mov cx, 4     ; ilk 4 piksel
7 ciz: mov es:[di], al
8 add di, 141h    ; 320 + 1
9 loop ciz
10 ret
```

ES	DI	CX	AL
A000h	0000h	0004h	0Eh

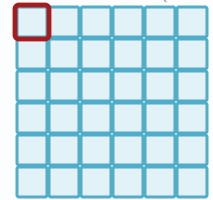
DI = 0 → y = 0, x = 0

satır başına 320 = 140h bayt

+141h = bir alt, bir sağ

AL = 0Eh (sarı), CX = 4: ilk 4 pikseli çiz.

ekranın sol üstü (6 × 6 piksel)



Adım 3 / 8

DEĞİŞKENLER

ES = A000h

DI = 0000h

CX = 0004h

Mod 13h Belleğine Köşegen Çizmek

```

1 org 100h
2 mov ax, 0A000h
3 mov es, ax
4 xor di, di      ; (0,0) → ofset 0
5 mov al, 0Eh     ; sarı
6 mov cx, 4       ; ilk 4 piksel

► ciz: mov es:[di], al
► add di, 141h    ; 320 + 1
► loop ciz

10 ret

```

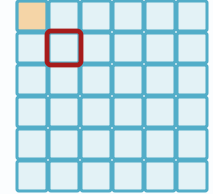
ES	DI	CX	AL
A000h	0141h	0003h	0Eh

DI = 321 → y = 1, x = 1

satır başına 320 = 140h bayt

+141h = bir alt, bir sağ

ekranın sol üstü (6 × 6 piksel)



ES:[0000h] ← 0Eh: piksel (0, 0). ADD DI, 141h → DI = 0141h
(bir alt satır + bir sağ). LOOP: CX = 3.

Adım 4 / 8

DEĞİŞKENLER

ES = A000h

DI = 0141h

CX = 0003h

Mod 13h Belleğine Köşegen Çizmek

```
1 org 100h
2 mov ax, 0A000h
3 mov es, ax
4 xor di, di      ; (0,0) → offset 0
5 mov al, 0Eh     ; sarı
6 mov cx, 4       ; ilk 4 piksel

► ciz: mov es:[di], al
► add di, 141h    ; 320 + 1
► loop ciz

10 ret
```

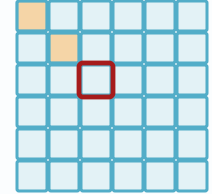
ES	DI	CX	AL
A000h	0282h	0002h	0Eh

DI = 642 → y = 2, x = 2

satır başına 320 = 140h bayt

+141h = bir alt, bir sağ

ekranın sol üstü (6 × 6 piksel)



ES:[0141h] ← 0Eh: piksel (1, 1). ADD DI, 141h → DI = 0282h
(bir alt satır + bir sağ). LOOP: CX = 2.

Adım 5 / 8

DEĞİŞKENLER

ES = A000h

DI = 0282h

CX = 0002h

Mod 13h Belleğine Köşegen Çizmek

```

1 org 100h
2 mov ax, 0A000h
3 mov es, ax
4 xor di, di      ; (0,0) → ofset 0
5 mov al, 0Eh     ; sarı
6 mov cx, 4       ; ilk 4 piksel

► ciz: mov es:[di], al
► add di, 141h    ; 320 + 1
► loop ciz

10 ret

```

ES	DI	CX	AL
A000h	03C3h	0001h	0Eh

DI = 963 → y = 3, x = 3

satır başına 320 = 140h bayt

+141h = bir alt, bir sağ



ES:[0282h] ← 0Eh: piksel (2, 2). ADD DI, 141h → DI = 03C3h
(bir alt satır + bir sağ). LOOP: CX = 1.

Adım 6 / 8

DEĞİŞKENLER

ES = A000h

DI = 03C3h

CX = 0001h

Mod 13h Belleğine Köşegen Çizmek

```
1 org 100h
2 mov ax, 0A000h
3 mov es, ax
4 xor di, di      ; (0,0) → ofset 0
5 mov al, 0Eh     ; sarı
6 mov cx, 4       ; ilk 4 piksel

► ciz: mov es:[di], al
► add di, 141h    ; 320 + 1
► loop ciz

10 ret
```

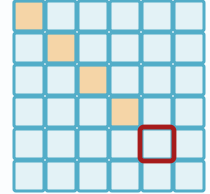
ES	DI	CX	AL
A000h	0504h	0000h	0Eh

DI = 1284 → y = 4, x = 4

satır başına 320 = 140h bayt

+141h = bir alt, bir sağ

ekranın sol üstü (6 × 6 piksel)



ES:[03C3h] ← 0Eh: piksel (3, 3). ADD DI, 141h → DI = 0504h
(bir alt satır + bir sağ). LOOP: CX = 0. Döngü biter.

Adım 7 / 8

DEĞİŞKENLER

ES = A000h

DI = 0504h

CX = 0000h

Mod 13h Belleğine Köşegen Çizmek

```

1 org 100h
2 mov ax, 0A000h
3 mov es, ax
4 xor di, di      ; (0,0) → ofset 0
5 mov al, 0Eh     ; sarı
6 mov cx, 4       ; ilk 4 piksel
7 ciz: mov es:[di], al
8 add di, 141h    ; 320 + 1
9 loop ciz
▶ ret

```

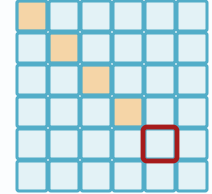
ES	DI	CX	AL
A000h	0504h	0000h	0Eh

DI = 1284 → y = 4, x = 4

satır başına 320 = 140h bayt

+141h = bir alt, bir sağ

ekranın sol üstü (6 × 6 piksel)



RET. 200 tur sürseydi son piksel (199, 199) = $199 \cdot 321 = \text{F987h}$ olurdu.

Adım 8 / 8

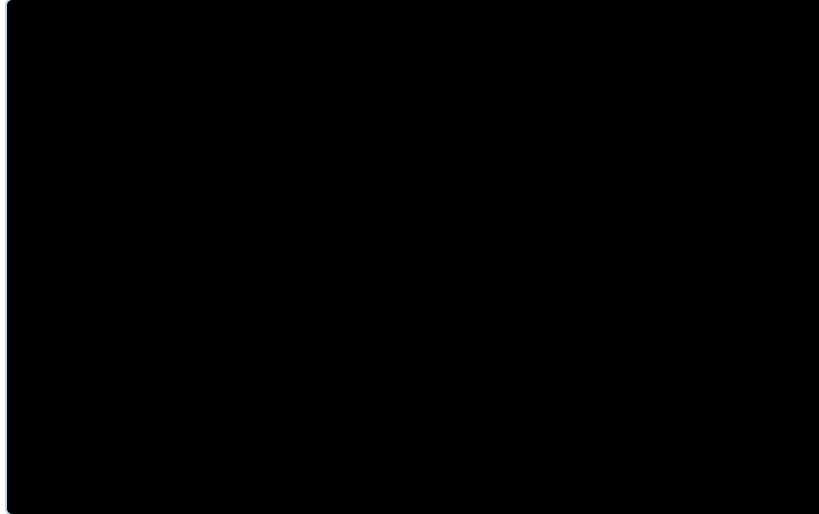
DEĞİŞKENLER

ES = A000h

DI = 0504h

CX = 0000h

Piksel Adresi Gezgini (320 × 200)



Köşegen: DI += 141h

Ters köşegen: DI += 13Fh

Temizle

$(100, 50) \rightarrow 50 \cdot 320 + 100 = 3EE4h \cdot A000:3EE4 = \text{fiziksel } A3EE4h$

Bir piksele tıklayın ya da köşegen çizdirin.

Ekrana tıklayın: (x, y) için $y \cdot 320 + x$ bağıl konumu ve fiziksel adres hesaplanır.

INT 16h: BIOS Klavye Hizmetleri

AH	Hizmet	Çıkış
00h	tuş bekle ve oku	AH = tarama kodu, AL = ASCII; tuş tampondan silinir
01h	tuş var mı? (beklemez)	ZF = 1: yok · ZF = 0: AX = tuş, tampondan silinmez
02h	Shift durumları	AL bitleri: 0 sağ Shift, 1 sol Shift, 2 Ctrl, 3 Alt, 4 Scroll, 5 Num, 6 Caps, 7 Insert

Tuş	AH (tarama)	AL (ASCII)
a / A	1Eh	61h / 41h
1	02h	31h
Enter	1Ch	0Dh
Esc	01h	1Bh
Boşluk	39h	20h

Tuş	AH (tarama)	AL
↑ / ↓	48h / 50h	00h
← / →	4Bh / 4Dh	00h
F1 ... F10	3Bh ... 44h	00h
Backspace	0Eh	08h
Ctrl+C	2Eh	03h

Tarama kodu tuşun klavyedeki **fiziksel** konumudur: 'a' ve 'A' aynı tarama kodunu verir. ASCII karşılığı olmayan (genişletilmiş) tuşlarda AL = 00h olur; tuşu AH'den tanırız.

Klavye Gezgini: INT 16h / AH = 00h Ne Döndürür?

buraya tıklayıp bir tuşa basın

AH (tarama kodu)

--

AL (ASCII)

--

AX

Alana tıklayın ve bir tuşa basın.

Harfler, rakamlar, ok tuşları, F1–F10, Enter, Esc ve Ctrl+harf tanınır. Değerler IBM PC/AT klavye düzenine (US) göreler.

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 ► mov ah, 0      ; tuş bekle
4 int 16h          ; AH=tarama, AL=ASCII
5 cmp al, 0Dh      ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h      ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h          ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

AH = 00h: klavye tamponundan bir tuş **bekle** ve al.

Adım 1 / 26

DEĞİŞKENLER

AH = 00h

AL = 00h

CF = 0

ZF = 0

ÇIKTI



Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 ► int 16h      ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

Kullanıcı **O** (Shift+o) bastı: AL = 4Fh (ASCII), AH = **18h** (tarama kodu).

Adım 2 / 26

DEĞİŖKENLER

AH = **18h**

AL = **4Fh**

CF = **0**

ZF = **0**

ÇIKTI



Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0          ; tuş bekle
4 int 16h            ; AH=tarama, AL=ASCII
5 ▶ cmp al, 0Dh      ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h        ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h            ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

4Fh ≠ 0Dh.

Adım 3 / 26

DEĞİŞKENLER

AH = 18h

AL = 4Fh

CF = 0

ZF = 0

ÇIKTI



Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

Enter değil, devam.

Adım 4 / 26

DEĞİŞKENLER

AH = 18h

AL = 4Fh

CF = 0

ZF = 0

ÇIKTI



Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 ▶ cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

'O' (4Fh) < 'a' (61h) → CF = 1.

Adım 5 / 26

DEĞİŞKENLER

AH = 18h

AL = 4Fh

CF = 1

ZF = 0

ÇIKTI

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jnb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

Küçük harf değil → doğrudan yaz .

Adım 6 / 26

DEĞİŞKENLER

AH = 18h

AL = 4Fh

CF = 1

ZF = 0

ÇIKTI

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 ► mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

AH = 0Eh (teletype). AL'de yazılacak karakter var.

Adım 7 / 26

DEĞİŞKENLER

AH = 0Eh

AL = 4Fh

CF = 1

ZF = 0

ÇIKTI



Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 ► int 10h      ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

Ekrana **O**.

Adım 8 / 26

DEĞİŞKENLER

AH = 0Eh

AL = 4Fh

CF = 1

ZF = 0

ÇIKTI

0

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 ▶ jmp oku
16 bitti:
17 ret
```

Sonraki tuşa dön.

Adım 9 / 26

DEĞİŞKENLER

AH = 0Eh

AL = 4Fh

CF = 1

ZF = 0

ÇIKTI

0

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 ► mov ah, 0      ; tuş bekle
4 int 16h          ; AH=tarama, AL=ASCII
5 cmp al, 0Dh      ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h      ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h          ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

AH = 00h: klavye tamponundan bir tuş **bekle** ve al.

Adım 10 / 26

DEĞİŞKENLER

AH = 00h

AL = 4Fh

CF = 1

ZF = 0

ÇIKTI

0

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 ► int 16h      ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

Kullanıcı **k** bastı: AL = 6Bh, AH = **25h**.

Adım 11 / 26

DEĞİŞKENLER

AH = **25h**

AL = **6Bh**

CF = **1**

ZF = **0**

ÇIKTI

0

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0          ; tuş bekle
4 int 16h            ; AH=tarama, AL=ASCII
5 ▶ cmp al, 0Dh      ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h        ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h            ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

6Bh ≠ 0Dh.

Adım 12 / 26

DEĞİŞKENLER

AH = 25h

AL = 6Bh

CF = 0

ZF = 0

ÇIKTI

0

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0          ; tuş bekle
4 int 16h            ; AH=tarama, AL=ASCII
5 cmp al, 0Dh        ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h        ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h            ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

Devam.

Adım 13 / 26

DEĞİŞKENLER

AH = 25h

AL = 6Bh

CF = 0

ZF = 0

ÇIKTI

0

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 ▶ cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h       ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

'k' ≥ 'a' → CF = 0.

Adım 14 / 26

DEĞİŖKENLER

AH = 25h

AL = 6Bh

CF = 0

ZF = 0

ÇIKTI

0

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jnb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

Küçük harf olabilir, devam.

Adım 15 / 26

DEĞİŞKENLER

AH = 25h

AL = 6Bh

CF = 0

ZF = 0

ÇIKTI

0

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

6Bh – 7Ah: ödünç → **CF = 1**, ZF = 0. Yani 'k' < 'z'.

Adım 16 / 26

DEĞİŞKENLER

AH = 25h

AL = 6Bh

CF = 1

ZF = 0

ÇIKTI

0

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

ja CF = 0 ve ZF = 0 ister; CF = 1 → atlama yok:
küçük harf.

Adım 17 / 26

DEĞİŞKENLER

AH = **25h**

AL = **6Bh**

CF = **1**

ZF = **0**

ÇIKTI

0

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 ► sub al, 20h  ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

Bit 5 temizlenir: $6Bh - 20h = 4Bh = 'K'$.

Adım 18 / 26

DEĞİŞKENLER

AH = 25h

AL = 4Bh

CF = 0

ZF = 0

ÇIKTI

0

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0          ; tuş bekle
4 int 16h            ; AH=tarama, AL=ASCII
5 cmp al, 0Dh        ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h        ; küçük → büyük
12 yaz:
13 ► mov ah, 0Eh
14 int 10h            ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

AH = 0Eh (teletype). AL'de yazılacak karakter var.

Adım 19 / 26

DEĞİŞKENLER

AH = 0Eh

AL = 4Bh

CF = 0

ZF = 0

ÇIKTI

0

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 ► int 10h      ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

Ekrana **K**.

Adım 20 / 26

DEĞİŞKENLER

AH = 0Eh

AL = 4Bh

CF = 0

ZF = 0

ÇIKTI

OK

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

Sonraki tuşa dön.

Adım 21 / 26

DEĞİŞKENLER

AH = 0Eh

AL = 4Bh

CF = 0

ZF = 0

ÇIKTI

OK

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 ► mov ah, 0      ; tuş bekle
4 int 16h          ; AH=tarama, AL=ASCII
5 cmp al, 0Dh      ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h      ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h          ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

AH = 00h: klavye tamponundan bir tuş **bekle** ve al.

Adım 22 / 26

DEĞİŞKENLER

AH = 00h

AL = 4Bh

CF = 0

ZF = 0

ÇIKTI

OK

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 ► int 16h      ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

Kullanıcı **Enter** bastı: AL = 0Dh, AH = **1Ch**.

Adım 23 / 26

DEĞİŖKENLER

AH = **1Ch**

AL = 0Dh

CF = 0

ZF = 0

ÇIKTI

OK

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 ▶ cmp al, 0Dh  ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h        ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

AL = 0Dh → ZF = 1.

Adım 24 / 26

DEĞİŞKENLER

AH = 1Ch

AL = 0Dh

CF = 0

ZF = 1

ÇIKTI

OK

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0          ; tuş bekle
4 int 16h            ; AH=tarama, AL=ASCII
5 cmp al, 0Dh        ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h        ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h            ; ekrana yaz
15 jmp oku
16 bitti:
17 ret
```

Enter → **bitti**.

Adım 25 / 26

DEĞİŞKENLER

AH = 1Ch

AL = 0Dh

CF = 0

ZF = 1

ÇIKTI

OK

Kod İzleme: Enter'a Kadar Oku, Büyük Harfle Yaz

```
1 org 100h
2 oku:
3 mov ah, 0      ; tuş bekle
4 int 16h        ; AH=tarama, AL=ASCII
5 cmp al, 0Dh    ; Enter?
6 je bitti
7 cmp al, 'a'
8 jb yaz
9 cmp al, 'z'
10 ja yaz
11 sub al, 20h    ; küçük → büyük
12 yaz:
13 mov ah, 0Eh
14 int 10h       ; ekrana yaz
15 jmp oku
16 bitti:
▶ ret
```

Enter'da program biter. INT 16h/00h tuşu ekrana **yansıtma**; yansıtmayı biz yaptık.

Adım 26 / 26

DEĞİŞKENLER

AH = 1Ch

AL = 0Dh

CF = 0

ZF = 1

ÇIKTI

OK

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 ▶ xor dx, dx ; iş sayacı
3 bekle:
4 inc dx ; arka plan işi
5 mov ah, 01h ; tuş var mı?
6 int 16h ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h ; tuşu tampondan al
9 int 16h
10 ret
```

DX = 0: bekleme sırasında yapılan işi sayacağız.

Adım 1 / 16

DEĞİŞKENLER

AH = 00h

AL = 00h

DX = 0000h

ZF = 1

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 ► inc dx        ; arka plan işi
5 mov ah, 01h     ; tuş var mı?
6 int 16h         ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

Arka planda iş yapılıyor: DX = 0001h.

Adım 2 / 16

DEĞİŞKENLER

AH = 00h

AL = 00h

DX = 0001h

ZF = 0

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 inc dx          ; arka plan işi
5 ► mov ah, 01h   ; tuş var mı?
6 int 16h         ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

AH = 01h: tuş durumunu sorgula (beklemez).

Adım 3 / 16

DEĞİŞKENLER

AH = 01h

AL = 00h

DX = 0001h

ZF = 0

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 inc dx          ; arka plan işi
5 mov ah, 01h     ; tuş var mı?
6 ▶ int 16h       ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

Tamponda tuş yok → **ZF = 1**.

Adım 4 / 16

DEĞİŞKENLER

AH = 01h

AL = 00h

DX = 0001h

ZF = 1

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 inc dx          ; arka plan işi
5 mov ah, 01h     ; tuş var mı?
6 int 16h         ; ZF=1 → yok
7 ► jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

ZF = 1 → **bekle** 'ye dön; program donmaz.

Adım 5 / 16

DEĞİŞKENLER

AH = **01h**

AL = **00h**

DX = **0001h**

ZF = **1**

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 ► inc dx        ; arka plan işi
5 mov ah, 01h     ; tuş var mı?
6 int 16h         ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

Arka planda iş yapılıyor: DX = 0002h.

Adım 6 / 16

DEĞİŞKENLER

AH = 01h

AL = 00h

DX = 0002h

ZF = 0

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 inc dx          ; arka plan işi
5 ► mov ah, 01h   ; tuş var mı?
6 int 16h         ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

AH = 01h: tuş durumunu sorgula (beklemez).

Adım 7 / 16

DEĞİŞKENLER

AH = 01h

AL = 00h

DX = 0002h

ZF = 0

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 inc dx          ; arka plan işi
5 mov ah, 01h     ; tuş var mı?
6 ▶ int 16h       ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

Hâlâ tuş yok → ZF = 1.

Adım 8 / 16

DEĞİŞKENLER

AH = 01h

AL = 00h

DX = 0002h

ZF = 1

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 inc dx          ; arka plan işi
5 mov ah, 01h     ; tuş var mı?
6 int 16h         ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

Yoklamaya devam.

Adım 9 / 16

DEĞİŞKENLER

AH = 01h

AL = 00h

DX = 0002h

ZF = 1

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 ► inc dx        ; arka plan işi
5 mov ah, 01h     ; tuş var mı?
6 int 16h         ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

Arka planda iş yapılıyor: DX = 0003h.

Adım 10 / 16

DEĞİŞKENLER

AH = 01h

AL = 00h

DX = 0003h

ZF = 0

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 inc dx          ; arka plan işi
5 ► mov ah, 01h   ; tuş var mı?
6 int 16h         ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

AH = 01h: tuş durumunu sorgula (beklemez).

Adım 11 / 16

DEĞİŖKENLER

AH = 01h

AL = 00h

DX = 0003h

ZF = 0

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 inc dx          ; arka plan işi
5 mov ah, 01h     ; tuş var mı?
6 ▶ int 16h       ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

Tuş geldi: **ZF = 0**, AL = 'x' (78h), AH = 2Dh. Tuş tamponda **kalır**.

Adım 12 / 16

DEĞİŞKENLER

AH = 2Dh

AL = 78h

DX = 0003h

ZF = 0

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 inc dx          ; arka plan işi
5 mov ah, 01h     ; tuş var mı?
6 int 16h         ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

ZF = 0 → döngüden çık.

Adım 13 / 16

DEĞİŞKENLER

AH = 2Dh

AL = 78h

DX = 0003h

ZF = 0

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 inc dx          ; arka plan işi
5 mov ah, 01h     ; tuş var mı?
6 int 16h         ; ZF=1 → yok
7 jz bekle
▶ mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

Tuşu tampondan almak için AH = 00h.

Adım 14 / 16

DEĞİŞKENLER

AH = 00h

AL = 78h

DX = 0003h

ZF = 0

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 inc dx          ; arka plan işi
5 mov ah, 01h     ; tuş var mı?
6 int 16h         ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
10 ret
```

Aynı tuş okunur ve tampondan silinir: AL = 'x', AH = 2Dh.

Adım 15 / 16

DEĞİŞKENLER

AH = 2Dh

AL = 78h

DX = 0003h

ZF = 0

Kod İzleme: AH = 01h ile Beklemeden Yoklama

```
1 org 100h
2 xor dx, dx      ; iş sayacı
3 bekle:
4 inc dx          ; arka plan işi
5 mov ah, 01h     ; tuş var mı?
6 int 16h         ; ZF=1 → yok
7 jz bekle
8 mov ah, 00h     ; tuşu tampondan al
9 int 16h
▶ ret
```

Oyunlarda ve animasyonlarda bu kalıp kullanılır: çalışmayı sürdür, tuşa basılınca tepki ver.

Adım 16 / 16

DEĞİŞKENLER

AH = 2Dh

AL = 78h

DX = 0003h

ZF = 0

INT 21h: Sık Kullanılan DOS Hizmetleri

AH	Hizmet	Giriş	Çıkış
01h	karakter oku, yansıt	–	AL = karakter
02h	karakter yaz	DL = karakter	AL = DL
07h / 08h	karakter oku, yansıtmadan	–	AL (08h Ctrl+C'yi denetler)
09h	dizgi yaz	DS:DX → '\$' ile biten dizgi	AL = 24h
0Ah	tamponlu satır oku	DS:DX → tampon	tampon[1] = sayı, tampon+2 = metin
2Ah	tarihi al	–	CX = yıl, DH = ay, DL = gün, AL = gün adı (0 = Pazar)
2Ch	saati al	–	CH = saat, CL = dakika, DH = saniye, DL = salise
2Bh / 2Dh	tarihi / saati ayarla	2Ah / 2Ch ile aynı yazmaçlar	AL = 00h başarılı
4Ch	programı sonlandır	AL = çıkış kodu	(dönmez)

DOS hizmetleri G/Ç yönlendirmesine uyar: `program > cikti.txt` ile 02h/09h çıktısı dosyaya gider; BIOS INT 10h çıktısı ise her zaman ekrana gider.

Kod İzleme: Karakter Oku, Yeni Satır, Karakter Yaz

```
1 org 100h
▶ 2 mov ah, 01h ; oku + yansıt
3 int 21h ; AL = karakter
4 mov bl, al ; sakla
5 mov ah, 02h ; karakter yaz
6 mov dl, 0Dh ; CR: satır başı
7 int 21h
8 mov dl, 0Ah ; LF: alt satır
9 int 21h
10 inc bl ; sonraki harf
11 mov dl, bl
12 int 21h
13 ret
```

AH = 01h: klavyeden bir karakter bekle ve **ekrana yansıt**.

Adım 1 / 12

DEĞİŞKENLER

AL = 00h

AH = 01h

BL = 00h

DL = 00h

ÇIKTI



Kod İzleme: Karakter Oku, Yeni Satır, Karakter Yaz

```
1 org 100h
2 mov ah, 01h ; oku + yansıt
3 ► int 21h ; AL = karakter
4 mov bl, al ; sakla
5 mov ah, 02h ; karakter yaz
6 mov dl, 0Dh ; CR: satır başı
7 int 21h
8 mov dl, 0Ah ; LF: alt satır
9 int 21h
10 inc bl ; sonraki harf
11 mov dl, bl
12 int 21h
13 ret
```

Kullanıcı **k** bastı: AL = 6Bh, ekranda 'k' görüldü.

Adım 2 / 12

DEĞİŞKENLER

AL = 6Bh

AH = 01h

BL = 00h

DL = 00h

ÇIKTI

k

Kod İzleme: Karakter Oku, Yeni Satır, Karakter Yaz

```
1 org 100h
2 mov ah, 01h      ; oku + yansıt
3 int 21h          ; AL = karakter
4 ► mov bl, al     ; sakla
5 mov ah, 02h      ; karakter yaz
6 mov dl, 0Dh      ; CR: satır başı
7 int 21h
8 mov dl, 0Ah      ; LF: alt satır
9 int 21h
10 inc bl          ; sonraki harf
11 mov dl, bl
12 int 21h
13 ret
```

AL sonraki INT 21h çağrılarında değişeceği için BL'de saklanır.

Adım 3 / 12

DEĞİŞKENLER

AL = 6Bh

AH = 01h

BL = 6Bh

DL = 00h

ÇIKTI

k

Kod İzleme: Karakter Oku, Yeni Satır, Karakter Yaz

```
1 org 100h
2 mov ah, 01h      ; oku + yansıt
3 int 21h          ; AL = karakter
4 mov bl, al       ; sakla
5 ► mov ah, 02h    ; karakter yaz
6 mov dl, 0Dh      ; CR: satır başı
7 int 21h
8 mov dl, 0Ah      ; LF: alt satır
9 int 21h
10 inc bl          ; sonraki harf
11 mov dl, bl
12 int 21h
13 ret
```

AH = 02h: DL'deki karakteri yaz.

Adım 4 / 12

DEĞİŞKENLER

AL = 6Bh

AH = 02h

BL = 6Bh

DL = 00h

ÇIKTI

k

Kod İzleme: Karakter Oku, Yeni Satır, Karakter Yaz

```
1 org 100h
2 mov ah, 01h      ; oku + yansıt
3 int 21h          ; AL = karakter
4 mov bl, al       ; sakla
5 mov ah, 02h      ; karakter yaz
▶ 6 mov dl, 0Dh     ; CR: satır başı
7 int 21h
8 mov dl, 0Ah      ; LF: alt satır
9 int 21h
10 inc bl          ; sonraki harf
11 mov dl, bl
12 int 21h
13 ret
```

DL = 0Dh (Carriage Return).

Adım 5 / 12

DEĞİŞKENLER

AL = 6Bh

AH = 02h

BL = 6Bh

DL = 0Dh

ÇIKTI

k

Kod İzleme: Karakter Oku, Yeni Satır, Karakter Yaz

```
1 org 100h
2 mov ah, 01h      ; oku + yansıt
3 int 21h          ; AL = karakter
4 mov bl, al       ; sakla
5 mov ah, 02h      ; karakter yaz
6 mov dl, 0Dh      ; CR: satır başı
7 ► int 21h
8 mov dl, 0Ah      ; LF: alt satır
9 int 21h
10 inc bl          ; sonraki harf
11 mov dl, bl
12 int 21h
13 ret
```

İmleç satır başına döner (görünür bir değişiklik yok).

Adım 6 / 12

DEĞİŞKENLER

AL = 0Dh

AH = 02h

BL = 6Bh

DL = 0Dh

ÇIKTI

k

Kod İzleme: Karakter Oku, Yeni Satır, Karakter Yaz

```
1 org 100h
2 mov ah, 01h      ; oku + yansıt
3 int 21h          ; AL = karakter
4 mov bl, al       ; sakla
5 mov ah, 02h      ; karakter yaz
6 mov dl, 0Dh      ; CR: satır başı
7 int 21h
▶ mov dl, 0Ah      ; LF: alt satır
9 int 21h
10 inc bl          ; sonraki harf
11 mov dl, bl
12 int 21h
13 ret
```

DL = 0Ah (Line Feed).

Adım 7 / 12

DEĞİŞKENLER

AL = 0Dh

AH = 02h

BL = 6Bh

DL = 0Ah

ÇIKTI

k

Kod İzleme: Karakter Oku, Yeni Satır, Karakter Yaz

```
1 org 100h
2 mov ah, 01h      ; oku + yansıt
3 int 21h          ; AL = karakter
4 mov bl, al       ; sakla
5 mov ah, 02h      ; karakter yaz
6 mov dl, 0Dh      ; CR: satır başı
7 int 21h
8 mov dl, 0Ah      ; LF: alt satır
9 ► int 21h
10 inc bl          ; sonraki harf
11 mov dl, bl
12 int 21h
13 ret
```

İmleç bir alt satıra iner. Yeni satır = **CR** + **LF**.

Adım 8 / 12

DEĞİŞKENLER

AL = 0Ah

AH = 02h

BL = 6Bh

DL = 0Ah

ÇIKTI

k

Kod İzleme: Karakter Oku, Yeni Satır, Karakter Yaz

```
1 org 100h
2 mov ah, 01h      ; oku + yansıt
3 int 21h          ; AL = karakter
4 mov bl, al       ; sakla
5 mov ah, 02h      ; karakter yaz
6 mov dl, 0Dh      ; CR: satır başı
7 int 21h
8 mov dl, 0Ah      ; LF: alt satır
9 int 21h
▶ inc bl           ; sonraki harf
11 mov dl, bl
12 int 21h
13 ret
```

$BL = 6Bh + 1 = 6Ch = 'l'$.

Adım 9 / 12

DEĞİŞKENLER

AL = 0Ah

AH = 02h

BL = 6Ch

DL = 0Ah

ÇIKTI

k

Kod İzleme: Karakter Oku, Yeni Satır, Karakter Yaz

```
1 org 100h
2 mov ah, 01h      ; oku + yansıt
3 int 21h          ; AL = karakter
4 mov bl, al       ; sakla
5 mov ah, 02h      ; karakter yaz
6 mov dl, 0Dh      ; CR: satır başı
7 int 21h
8 mov dl, 0Ah      ; LF: alt satır
9 int 21h
10 inc bl          ; sonraki harf
11 ► mov dl, bl
12 int 21h
13 ret
```

DL = 'I'.

Adım 10 / 12

DEĞİŞKENLER

AL = 0Ah

AH = 02h

BL = 6Ch

DL = 6Ch

ÇIKTI

k

Kod İzleme: Karakter Oku, Yeni Satır, Karakter Yaz

```
1 org 100h
2 mov ah, 01h      ; oku + yansıt
3 int 21h          ; AL = karakter
4 mov bl, al       ; sakla
5 mov ah, 02h      ; karakter yaz
6 mov dl, 0Dh      ; CR: satır başı
7 int 21h
8 mov dl, 0Ah      ; LF: alt satır
9 int 21h
10 inc bl          ; sonraki harf
11 mov dl, bl
12 int 21h
13 ret
```

Ekrana **l** yazılır. 02h sonrası AL de yazılan karakteri içerir.

Adım 11 / 12

DEĞİŞKENLER

AL = 6Ch

AH = 02h

BL = 6Ch

DL = 6Ch

ÇIKTI

k
l

Kod İzleme: Karakter Oku, Yeni Satır, Karakter Yaz

```
1 org 100h
2 mov ah, 01h      ; oku + yansıt
3 int 21h          ; AL = karakter
4 mov bl, al       ; sakla
5 mov ah, 02h      ; karakter yaz
6 mov dl, 0Dh      ; CR: satır başı
7 int 21h
8 mov dl, 0Ah      ; LF: alt satır
9 int 21h
10 inc bl          ; sonraki harf
11 mov dl, bl
12 int 21h
▶ ret
```

Program biter.

Adım 12 / 12

DEĞİŞKENLER

AL = 6Ch

AH = 02h

BL = 6Ch

DL = 6Ch

ÇIKTI

k
l

Kod İzleme: AH = 09h ile Dizgi Yazdırma

```
1 org 100h
2 ► mov dx, offset msg1 ; DS:DX → dizgi
3 mov ah, 09h
4 int 21h
5 mov dx, offset msg2
6 int 21h ; AH hâlâ 09h
7 ret
8 msg1 db 'Merhaba, 8086!', 0Dh, 0Ah, '$'
9 msg2 db 'Güle güle$'
```

DX = 010Dh: dizginin bağlı konumu. Kesim DS'dir.

Adım 1 / 6

DEĞİŞKENLER

AX = 0000h

DX = 010Dh

ÇIKTI

Kod İzleme: AH = 09h ile Dizgi Yazdırma

```
1 org 100h
2 mov dx, offset msg1 ; DS:DX → dizgi
3 ► mov ah, 09h
4 int 21h
5 mov dx, offset msg2
6 int 21h ; AH hâlâ 09h
7 ret
8 msg1 db 'Merhaba, 8086!', 0Dh, 0Ah, '$'
9 msg2 db 'Güle güle$'
```

AH = 09h: '\$' ile biten dizgiyi yaz.

Adım 2 / 6

DEĞİŞKENLER

AX = 0900h

DX = 010Dh

ÇIKTI



Kod İzleme: AH = 09h ile Dizgi Yazdırma

```
1 org 100h
2 mov dx, offset msg1 ; DS:DX → dizgi
3 mov ah, 09h
4 ► int 21h
5 mov dx, offset msg2
6 int 21h ; AH hâlâ 09h
7 ret
8 msg1 db 'Merhaba, 8086!', 0Dh, 0Ah, '$'
9 msg2 db 'Güle güle$'
```

'\$' görülene kadar yazılır; 0Dh 0Ah satırı bitirir.
Dönüşte AL = 24h ('\$').

Adım 3 / 6

DEĞİŞKENLER

AX = 0924h

DX = 010Dh

ÇIKTI

Merhaba, 8086!

Kod İzleme: AH = 09h ile Dizgi Yazdırma

```
1 org 100h
2 mov dx, offset msg1 ; DS:DX → dizgi
3 mov ah, 09h
4 int 21h
▶ mov dx, offset msg2
6 int 21h ; AH hâlâ 09h
7 ret
8 msg1 db 'Merhaba, 8086!', 0Dh, 0Ah, '$'
9 msg2 db 'Güle güle$'
```

İkinci dizgi: DX = 011Eh.

Adım 4 / 6

DEĞİŞKENLER

AX = 0924h

DX = 011Eh

ÇIKTI

Merhaba, 8086!

Kod İzleme: AH = 09h ile Dizgi Yazdırma

```
1 org 100h
2 mov dx, offset msg1 ; DS:DX → dizgi
3 mov ah, 09h
4 int 21h
5 mov dx, offset msg2
6 int 21h ; AH hâlâ 09h
7 ret
8 msg1 db 'Merhaba, 8086!', 0Dh, 0Ah, '$'
9 msg2 db 'Güle güle$'
```

AH değişmediği için yeniden yüklemek gerekmez (yine de yüklemek iyi alışkanlıktır).

Adım 5 / 6

DEĞİŞKENLER

AX = 0924h

DX = 011Eh

ÇIKTI

```
Merhaba, 8086!
Güle güle
```

Kod İzleme: AH = 09h ile Dizgi Yazdırma

```
1 org 100h
2 mov dx, offset msg1 ; DS:DX → dizgi
3 mov ah, 09h
4 int 21h
5 mov dx, offset msg2
6 int 21h ; AH hâlâ 09h
7 ► ret
8 msg1 db 'Merhaba, 8086!', 0Dh, 0Ah, '$'
9 msg2 db 'Güle güle$'
```

Çıktı iki satır. '\$' karakterinin kendisi bu hizmetle yazdırılamaz.

Adım 6 / 6

DEĞİŞKENLER

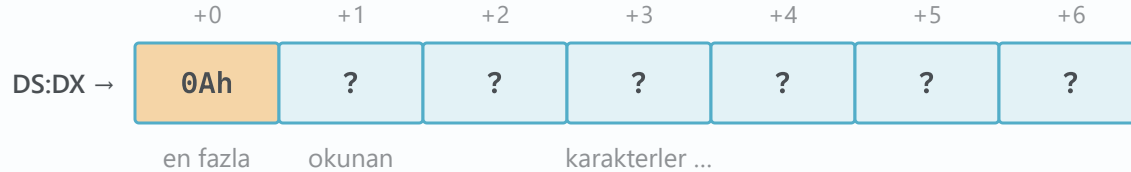
AX = 0924h

DX = 011Eh

ÇIKTI

```
Merhaba, 8086!
Güle güle
```

AH = 0Ah: Tamponlu Satır Girişi

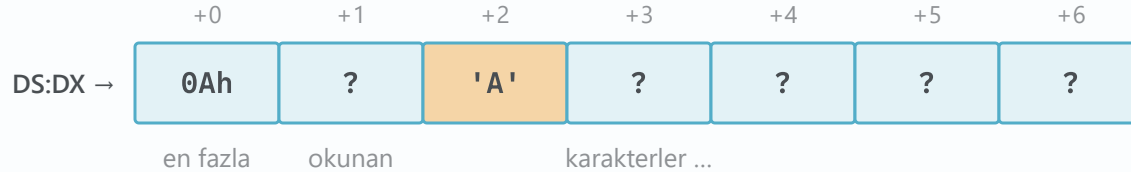


Program tamponu hazırlar: `tampon db 10, ?, 10 dup(?)`. Bayt 0 = **en fazla 10** karakter (Enter dahil).

Adım 1 / 5

- DOS satır düzenlemeyi kendisi yapar (Backspace, Esc); Enter'a basılana kadar program beklemez, çalışmaz.
- Tampon dolunca fazla tuşlar kabul edilmez ve zil çalar. İlk bayt 0 olamaz.

AH = 0Ah: Tamponlu Satır Girişi

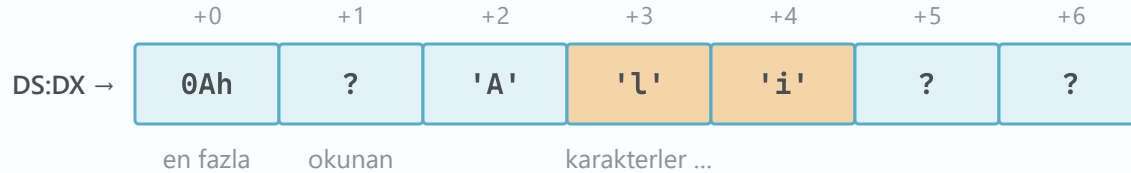


Kullanıcı **A** yazar: karakter 2. konuma gelir ve ekrana yansıtılır. Backspace ile düzeltme yapılabilir.

Adım 2 / 5

- DOS satır düzenlemeyi kendisi yapar (Backspace, Esc); Enter'a basılana kadar program beklemez, çalışmaz.
- Tampon dolunca fazla tuşlar kabul edilmez ve zil çalar. İlk bayt 0 olamaz.

AH = 0Ah: Tamponlu Satır Girişi

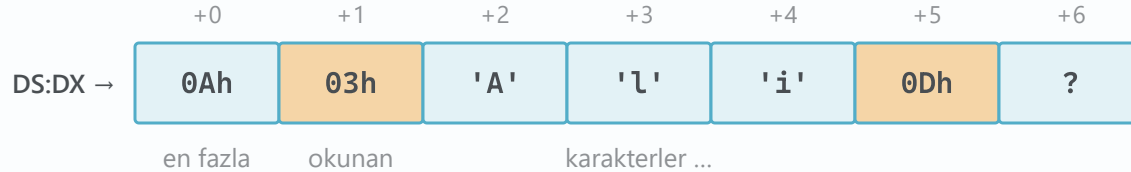


ı ve **i** de eklenir.

Adım 3 / 5

- DOS satır düzenlemeyi kendisi yapar (Backspace, Esc); Enter'a basılana kadar program beklemez, çalışmaz.
- Tampon dolunca fazla tuşlar kabul edilmez ve zil çalar. İlk bayt 0 olamaz.

AH = 0Ah: Tamponlu Satır Girişi

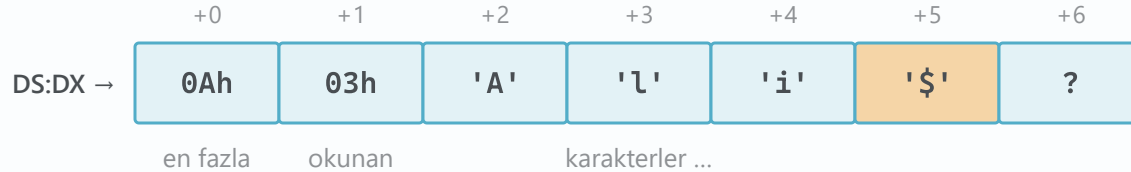


Enter: **0Dh** sona yazılır; DOS bayt 1'e okunan sayıyı (**3**, CR hariç) yazar ve döner.

Adım 4 / 5

- DOS satır düzenlemeyi kendisi yapar (Backspace, Esc); Enter'a basılana kadar program beklemez, çalışmaz.
- Tampon dolunca fazla tuşlar kabul edilmez ve zil çalar. İlk bayt 0 olamaz.

AH = 0Ah: Tamponlu Satır Girişi



Metin **tampon+2**'den başlar, uzunluğu **[tampon+1]**'dedir. '\$' ile bitmez; 09h ile yazdırmak için 0Dh yerine '\$' konur.

Adım 5 / 5

- DOS satır düzenlemeyi kendisi yapar (Backspace, Esc); Enter'a basılana kadar program beklemez, çalışmaz.
- Tampon dolunca fazla tuşlar kabul edilmez ve zil çalar. İlk bayt 0 olamaz.

Kod İzleme: İsim Oku, Selam Ver

```
1 org 100h
2 ► mov dx, offset tampon
3 mov ah, 0Ah          ; tamponlu giriş
4 int 21h
5 mov bl, [tampon+1]   ; okunan sayı
6 mov bh, 0
7 mov byte ptr tampon[bx+2], '$'
8 mov dx, offset selam
9 mov ah, 09h
10 int 21h
11 lea dx, [tampon+2]
12 int 21h
13 ret
14 tampon db 10, ?, 10 dup(?)
15 selam db 0Dh, 0Ah, 'Merhaba $'
```

DS:DX tamponu gösterir: DX = 0120h.

Adım 1 / 12

DEĞİŞKENLER

BX = 0000h

DX = 0120h

tampon = 0Ah

tampon+1 = 00h

tampon+2 = '.....'

ÇIKTI



Kod İzleme: İsim Oku, Selam Ver

```
1 org 100h
2 mov dx, offset tampon
3 ► mov ah, 0Ah          ; tamponlu giriş
4 int 21h
5 mov bl, [tampon+1]    ; okunan sayı
6 mov bh, 0
7 mov byte ptr tampon[bx+2], '$'
8 mov dx, offset selam
9 mov ah, 09h
10 int 21h
11 lea dx, [tampon+2]
12 int 21h
13 ret
14 tampon db 10, ?, 10 dup(?)
15 selam db 0Dh, 0Ah, 'Merhaba $'
```

AH = 0Ah: Enter'a kadar satır oku.

Adım 2 / 12

DEĞİŞKENLER

BX = 0000h

DX = 0120h

tampon = 0Ah

tampon+1 = 00h

tampon+2 = '.....'

ÇIKTI



Kod İzleme: İsim Oku, Selam Ver

```
1 org 100h
2 mov dx, offset tampon
3 mov ah, 0Ah          ; tamponlu giriş
4 ▶ int 21h
5 mov bl, [tampon+1]    ; okunan sayı
6 mov bh, 0
7 mov byte ptr tampon[bx+2], '$'
8 mov dx, offset selam
9 mov ah, 09h
10 int 21h
11 lea dx, [tampon+2]
12 int 21h
13 ret
14 tampon db 10, ?, 10 dup(?)
15 selam db 0Dh, 0Ah, 'Merhaba $'
```

Kullanıcı **Ali** yazdı. DOS: tampon[1] = 3, tampon[2..4] = 'Ali', tampon[5] = 0Dh. Enter yalnızca CR basar: imleç satır başında.

Adım 3 / 12

DEĞİŞKENLER

BX = 0000h

DX = 0120h

tampon = 0Ah

tampon+1 = 03h

tampon+2 = 'Ali..'

ÇIKTI

Ali

Kod İzleme: İsim Oku, Selam Ver

```
1 org 100h
2 mov dx, offset tampon
3 mov ah, 0Ah          ; tamponlu giriş
4 int 21h
5 ► mov bl, [tampon+1] ; okunan sayı
6 mov bh, 0
7 mov byte ptr tampon[bx+2], '$'
8 mov dx, offset selam
9 mov ah, 09h
10 int 21h
11 lea dx, [tampon+2]
12 int 21h
13 ret
14 tampon db 10, ?, 10 dup(?)
15 selam db 0Dh, 0Ah, 'Merhaba $'
```

BL = okunan karakter sayısı = **3** (CR sayılmaz).

Adım 4 / 12

DEĞİŞKENLER

BX = 0003h

DX = 0120h

tampon = 0Ah

tampon+1 = 03h

tampon+2 = 'Ali··'

ÇIKTI

Ali

Kod İzleme: İsim Oku, Selam Ver

```
1 org 100h
2 mov dx, offset tampon
3 mov ah, 0Ah          ; tamponlu giriş
4 int 21h
5 mov bl, [tampon+1]   ; okunan sayı
▶ mov bh, 0
7 mov byte ptr tampon[bx+2], '$'
8 mov dx, offset selam
9 mov ah, 09h
10 int 21h
11 lea dx, [tampon+2]
12 int 21h
13 ret
14 tampon db 10, ?, 10 dup(?)
15 selam db 0Dh, 0Ah, 'Merhaba $'
```

BX = 0003h; adres hesaplamasında kullanılacak.

Adım 5 / 12

DEĞİŞKENLER

BX = 0003h

DX = 0120h

tampon = 0Ah

tampon+1 = 03h

tampon+2 = 'Ali...'

ÇIKTI

Ali

Kod İzleme: İsim Oku, Selam Ver

```
1 org 100h
2 mov dx, offset tampon
3 mov ah, 0Ah          ; tamponlu giriş
4 int 21h
5 mov bl, [tampon+1]    ; okunan sayı
6 mov bh, 0
▶ 7 mov byte ptr tampon[bx+2], '$'
8 mov dx, offset selam
9 mov ah, 09h
10 int 21h
11 lea dx, [tampon+2]
12 int 21h
13 ret
14 tampon db 10, ?, 10 dup(?)
15 selam db 0Dh, 0Ah, 'Merhaba $'
```

$\text{tampon}[3 + 2] = \text{tampon}[5]$: 0Dh yerine '\$' yazılır →
dizgi AH = 09h ile yazılabilir.

Adım 6 / 12

DEĞİŞKENLER

BX = 0003h

DX = 0120h

tampon = 0Ah

tampon+1 = 03h

tampon+2 = 'Ali\$.'

ÇIKTI

Ali

Kod İzleme: İsim Oku, Selam Ver

```
1 org 100h
2 mov dx, offset tampon
3 mov ah, 0Ah          ; tamponlu giriş
4 int 21h
5 mov bl, [tampon+1]   ; okunan sayı
6 mov bh, 0
7 mov byte ptr tampon[bx+2], '$'
▶ mov dx, offset selam
9 mov ah, 09h
10 int 21h
11 lea dx, [tampon+2]
12 int 21h
13 ret
14 tampon db 10, ?, 10 dup(?)
15 selam db 0Dh, 0Ah, 'Merhaba $'
```

DX = selam.

Adım 7 / 12

DEĞİŞKENLER

BX = 0003h

DX = 012Ch

tampon = 0Ah

tampon+1 = 03h

tampon+2 = 'Ali\$.'

ÇIKTI

Ali

Kod İzleme: İsim Oku, Selam Ver

```
1 org 100h
2 mov dx, offset tampon
3 mov ah, 0Ah          ; tamponlu giriş
4 int 21h
5 mov bl, [tampon+1]   ; okunan sayı
6 mov bh, 0
7 mov byte ptr tampon[bx+2], '$'
8 mov dx, offset selam
9 ► mov ah, 09h
10 int 21h
11 lea dx, [tampon+2]
12 int 21h
13 ret
14 tampon db 10, ?, 10 dup(?)
15 selam db 0Dh, 0Ah, 'Merhaba $'
```

AH = 09h.

Adım 8 / 12

DEĞİŞKENLER

BX = 0003h

DX = 012Ch

tampon = 0Ah

tampon+1 = 03h

tampon+2 = 'Ali\$.'

ÇIKTI

Ali

Kod İzleme: İsim Oku, Selam Ver

```
1 org 100h
2 mov dx, offset tampon
3 mov ah, 0Ah          ; tamponlu giriş
4 int 21h
5 mov bl, [tampon+1]   ; okunan sayı
6 mov bh, 0
7 mov byte ptr tampon[bx+2], '$'
8 mov dx, offset selam
9 mov ah, 09h
▶ int 21h
11 lea dx, [tampon+2]
12 int 21h
13 ret
14 tampon db 10, ?, 10 dup(?)
15 selam db 0Dh, 0Ah, 'Merhaba $'
```

CR LF + 'Merhaba ' yazılır (yeni satırda).

Adım 9 / 12

DEĞİŞKENLER

BX = 0003h

DX = 012Ch

tampon = 0Ah

tampon+1 = 03h

tampon+2 = 'Ali\$.'

ÇIKTI

Ali
Merhaba

Kod İzleme: İsim Oku, Selam Ver

```
1 org 100h
2 mov dx, offset tampon
3 mov ah, 0Ah          ; tamponlu giriş
4 int 21h
5 mov bl, [tampon+1]    ; okunan sayı
6 mov bh, 0
7 mov byte ptr tampon[bx+2], '$'
8 mov dx, offset selam
9 mov ah, 09h
10 int 21h
11 ▶ lea dx, [tampon+2]
12 int 21h
13 ret
14 tampon db 10, ?, 10 dup(?)
15 selam db 0Dh, 0Ah, 'Merhaba $'
```

DX = tampon + 2: girilen metnin başlangıcı (0122h).

Adım 10 / 12

DEĞİŞKENLER

BX = 0003h

DX = 0122h

tampon = 0Ah

tampon+1 = 03h

tampon+2 = 'Ali\$.'

ÇIKTI

Ali
Merhaba

Kod İzleme: İsim Oku, Selam Ver

```
1 org 100h
2 mov dx, offset tampon
3 mov ah, 0Ah          ; tamponlu giriş
4 int 21h
5 mov bl, [tampon+1]   ; okunan sayı
6 mov bh, 0
7 mov byte ptr tampon[bx+2], '$'
8 mov dx, offset selam
9 mov ah, 09h
10 int 21h
11 lea dx, [tampon+2]
12 ► int 21h
13 ret
14 tampon db 10, ?, 10 dup(?)
15 selam db 0Dh, 0Ah, 'Merhaba $'
```

Çıktı: **Merhaba Ali.**

Adım 11 / 12

DEĞİŞKENLER

BX = 0003h

DX = 0122h

tampon = 0Ah

tampon+1 = 03h

tampon+2 = 'Ali\$.'

ÇIKTI

Ali

Merhaba Ali

Kod İzleme: İsim Oku, Selam Ver

```
1 org 100h
2 mov dx, offset tampon
3 mov ah, 0Ah          ; tamponlu giriş
4 int 21h
5 mov bl, [tampon+1]   ; okunan sayı
6 mov bh, 0
7 mov byte ptr tampon[bx+2], '$'
8 mov dx, offset selam
9 mov ah, 09h
10 int 21h
11 lea dx, [tampon+2]
12 int 21h
13 ► ret
14 tampon db 10, ?, 10 dup(?)
15 selam db 0Dh, 0Ah, 'Merhaba $'
```

tampon[0] = 10: en çok 9 karakter + CR. Fazlası kabul edilmez (DOS zil sesi çalar).

Adım 12 / 12

DEĞİŞKENLER

BX = 0003h

DX = 0122h

tampon = 0Ah

tampon+1 = 03h

tampon+2 = 'Ali\$.'

ÇIKTI

Ali

Merhaba Ali

AH = 2Ah / 2Ch: Tarih ve Saat

AH = 2Ah (tarih)

CX	yıl (ör. 07EAh = 2026)
DH	ay (1–12)
DL	gün (1–31)
AL	haftanın günü (0 = Pazar ... 6 = Cumartesi)

AH = 2Ch (saat)

CH	saat (0–23)
CL	dakika (0–59)
DH	saniye (0–59)
DL	salise (0–99)

; gün adını yazdır

```
mov ah, 2Ah
```

```
int 21h ; AL = 0...6
```

```
mov ah, 0
```

```
mov bl, 4
```

```
mul bl ; AX = AL x 4
```

```
mov dx, offset gunler
```

```
add dx, ax
```

```
mov ah, 09h
```

```
int 21h
```

```
ret
```

```
gunler db 'Paz$Pzt$Sal$Car$'
```

```
db 'Per$Cum$Cmt$'
```

Değerler **ikili** sayıdır, BCD değil: saat 14 → CH = 0Eh. Ekrana yazmak için AAM ya da 10'a bölme kullanılır. BIOS'taki INT 1Ah/02h ise saati BCD olarak verir.

Tablodan Gün Adı: AL × 4

```
1 org 100h
```

```
► mov ah, 2Ah
```

```
3 int 21h ; AL = 0...6
```

```
4 mov ah, 0
```

```
5 mov bl, 4
```

```
6 mul bl ; AX = AL × 4
```

```
7 mov dx, offset gunler
```

```
8 add dx, ax
```

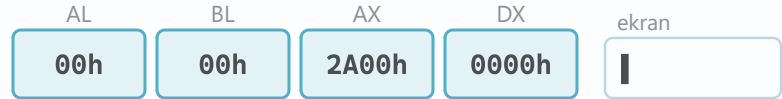
```
9 mov ah, 09h
```

```
10 int 21h
```

```
11 ret
```

```
12 gunler db 'Paz$Pzt$Sal$Car$'
```

```
13 db 'Per$Cum$Cmt$'
```



gunler (0114h): 7 × 4 bayt



AH = 2Ah: DOS tarih hizmeti.

Adım 1 / 8

DEĞİŞKENLER

AL = 00h

AX = 2A00h

DX = 0000h

Yorumlayıcıda INT 21h/2Ah olmadığından çalıştırılan kopyada bu satır, aynı uzunluktaki

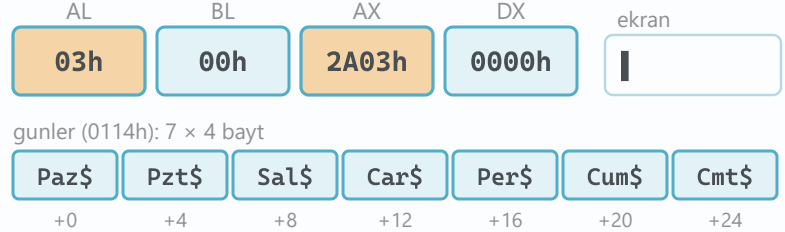
`mov al, 3` ile değiştirildi (örnek gün Çarşamba).

Tablodan Gün Adı: AL × 4

```

1 org 100h
2 mov ah, 2Ah
3 ► int 21h ; AL = 0...6
4 mov ah, 0
5 mov bl, 4
6 mul bl ; AX = AL × 4
7 mov dx, offset gunler
8 add dx, ax
9 mov ah, 09h
10 int 21h
11 ret
12 gunler db 'Paz$Pzt$Sal$Car$'
13         db 'Per$Cum$Cmt$'

```



INT 21h dönüşte CX = yıl, DH = ay, DL = gün ve AL = haftanın günü verir. Örnek gün Çarşamba: **AL = 3** (0 = Pazar).

Adım 2 / 8

DEĞİŞKENLER

AL = 03h

AX = 2A03h

DX = 0000h

Yorumlayıcıda INT 21h/2Ah olmadığından çalıştırılan kopyada bu satır, aynı uzunluktaki

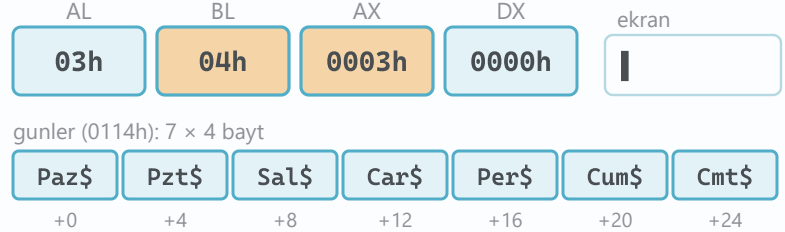
`mov al, 3` ile değiştirildi (örnek gün Çarşamba).

Tablodan Gün Adı: AL × 4

```

1 org 100h
2 mov ah, 2Ah
3 int 21h          ; AL = 0...6
▶ mov ah, 0
▶ mov bl, 4
6 mul bl          ; AX = AL × 4
7 mov dx, offset gunler
8 add dx, ax
9 mov ah, 09h
10 int 21h
11 ret
12 gunler db 'Paz$Pzt$Sal$Car$'
13         db 'Per$Cum$Cmt$'

```



AH = 0 (AX = 0003h), BL = 4: her gün adı '\$' dahil 4 bayt.

Adım 3 / 8

DEĞİŞKENLER

AL = 03h

AX = 0003h

DX = 0000h

Yorumlayıcıda INT 21h/2Ah olmadığından çalıştırılan kopyada bu satır, aynı uzunluktaki

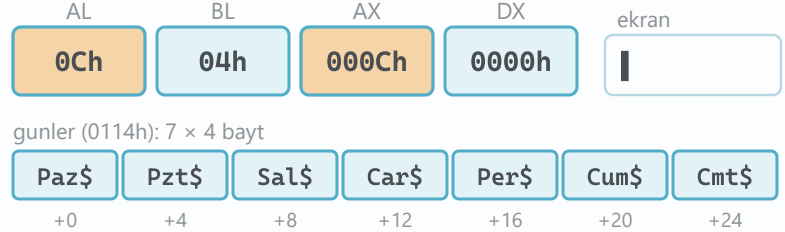
`mov al, 3` ile değiştirildi (örnek gün Çarşamba).

Tablodan Gün Adı: AL × 4

```

1 org 100h
2 mov ah, 2Ah
3 int 21h          ; AL = 0...6
4 mov ah, 0
5 mov bl, 4
6 mul bl          ; AX = AL × 4
7 mov dx, offset gunler
8 add dx, ax
9 mov ah, 09h
10 int 21h
11 ret
12 gunler db 'Paz$Pzt$Sal$Car$'
13         db 'Per$Cum$Cmt$'

```



MUL BL: AX = 3 × 4 = **12** = 000Ch: tablodaki bağıl konum.

Adım 4 / 8

DEĞİŞKENLER

AL = 0Ch

AX = 000Ch

DX = 0000h

Yorumlayıcıda INT 21h/2Ah olmadığından çalıştırılan kopyada bu satır, aynı uzunluktaki

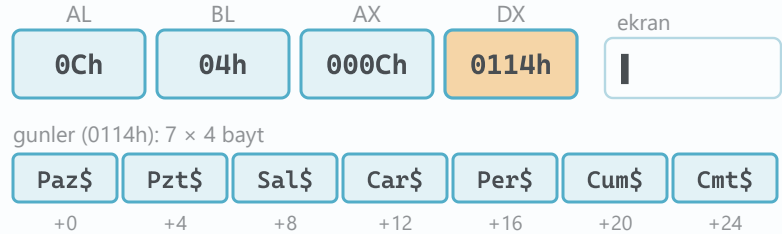
`mov al, 3` ile değiştirildi (örnek gün Çarşamba).

Tablodan Gün Adı: AL × 4

```

1 org 100h
2 mov ah, 2Ah
3 int 21h           ; AL = 0...6
4 mov ah, 0
5 mov bl, 4
6 mul bl           ; AX = AL × 4
7 ► mov dx, offset gunler
8 add dx, ax
9 mov ah, 09h
10 int 21h
11 ret
12 gunler db 'Paz$Pzt$Sal$Car$'
13         db 'Per$Cum$Cmt$'

```



DX = gunler = 0114h.

Adım 5 / 8

DEĞİŞKENLER

AL = 0Ch

AX = 000Ch

DX = 0114h

Yorumlayıcıda INT 21h/2Ah olmadığından çalıştırılan kopyada bu satır, aynı uzunluktaki

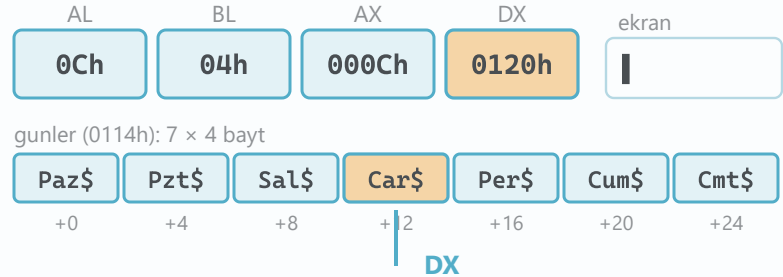
`mov al, 3` ile değiştirildi (örnek gün Çarşamba).

Tablodan Gün Adı: AL × 4

```

1 org 100h
2 mov ah, 2Ah
3 int 21h          ; AL = 0...6
4 mov ah, 0
5 mov bl, 4
6 mul bl           ; AX = AL × 4
7 mov dx, offset gunler
▶ add dx, ax
9 mov ah, 09h
10 int 21h
11 ret
12 gunler db 'Paz$Pzt$Sal$Car$'
13         db 'Per$Cum$Cmt$'

```



ADD DX, AX → DX = 0120h: 'Car\$' dizgisinin başı.

Adım 6 / 8

DEĞİŞKENLER

AL = 0Ch

AX = 000Ch

DX = 0120h

Yorumlayıcıda INT 21h/2Ah olmadığından çalıştırılan kopyada bu satır, aynı uzunluktaki

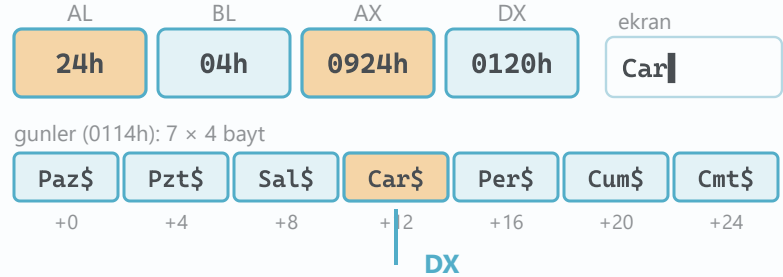
`mov al, 3` ile değiştirildi (örnek gün Çarşamba).

Tablodan Gün Adı: AL × 4

```

1 org 100h
2 mov ah, 2Ah
3 int 21h           ; AL = 0...6
4 mov ah, 0
5 mov bl, 4
6 mul bl           ; AX = AL × 4
7 mov dx, offset gunler
8 add dx, ax
9 ▶ mov ah, 09h
10 ▶ int 21h
11 ret
12 gunler db 'Paz$Pzt$Sal$Car$'
13         db 'Per$Cum$Cmt$'

```



AH = 09h, INT 21h: DS:DX'ten '\$'a kadar → ekranda **Car**.

Adım 7 / 8

DEĞİŞKENLER

AL = 24h

AX = 0924h

DX = 0120h

Yorumlayıcıda INT 21h/2Ah olmadığından çalıştırılan kopyada bu satır, aynı uzunluktaki

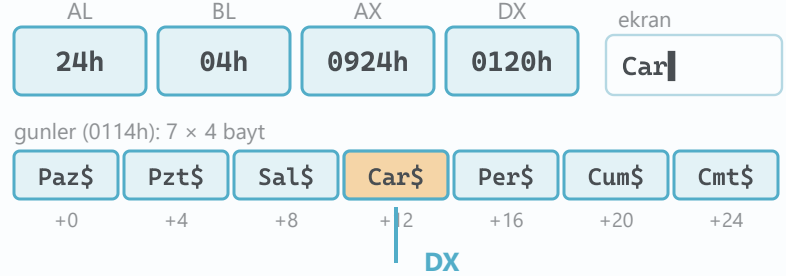
`mov al, 3` ile değiştirildi (örnek gün Çarşamba).

Tablodan Gün Adı: AL × 4

```

1 org 100h
2 mov ah, 2Ah
3 int 21h          ; AL = 0...6
4 mov ah, 0
5 mov bl, 4
6 mul bl           ; AX = AL × 4
7 mov dx, offset gunler
8 add dx, ax
9 mov ah, 09h
10 int 21h
11 ► ret
12 gunler db 'Paz$Pzt$Sal$Car$'
13         db 'Per$Cum$Cmt$'

```



RET: program biter.

Adım 8 / 8

DEĞİŞKENLER

AL = 24h

AX = 0924h

DX = 0120h

Yorumlayıcıda INT 21h/2Ah olmadığından çalıştırılan kopyada bu satır, aynı uzunluktaki

`mov al, 3` ile değiştirildi (örnek gün Çarşamba).

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 ▶ mov ah, 2Ch      ; sistem saatini al
3 int 21h           ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:            ; AL (0-99) → 2 hane
13 aam              ; AH=onlar AL=birler
14 add ax, 3030h    ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

AH = 2Ch: sistem saatini oku.

Adım 1 / 28

DEĞİŞKENLER

AX = 2C00h

BX = 0000h

CX = 0000h

DX = 0000h

SP = FFFEh

ÇAĞRI YİĞİNİ

boş

ÇIKTI

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 ► int 21h        ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

Dönüş: **CH = 14** (0Eh) saat, **CL = 35** (23h) dakika, DH = 7 saniye, DL = 42 salise.

Adım 2 / 28

DEĞİŞKENLER

AX = 2C00h

BX = 0000h

CX = 0E23h

DX = 072Ah

SP = FFEh

ÇAĞRI YİĞİNİ

boş

ÇIKTI

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
▶ 4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

AL = saat = 0Eh.

Adım 3 / 28

DEĞİŞKENLER

AX = 2C0Eh

BX = 0000h

CX = 0E23h

DX = 072Ah

SP = FFFEh

ÇAĞRI YİĞİNİ

boş

ÇIKTI

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 ▶ call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

Prosedür çağrılır: dönüş adresi yığında.

Adım 4 / 28

DEĞİŞKENLER

AX = 2C0Eh

BX = 0000h

CX = 0E23h

DX = 072Ah

SP = FFFCh

ÇAĞRI YIĞINI

FFFCh: 0109h

ÇIKTI

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 ► aam           ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

aam : 14 → AH = 1, AL = 4.

Adım 5 / 28

DEĞİŞKENLER

AX = 0104h

BX = 0000h

CX = 0E23h

DX = 072Ah

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0109h

ÇIKTI

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch          ; sistem saatini al
3 int 21h              ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:                ; AL (0-99) → 2 hane
13 aam                  ; AH=onlar AL=birler
14 ► add ax, 3030h       ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

Her iki basamak ASCII'ye: AX = 3134h.

Adım 6 / 28

DEĞİŞKENLER

AX = 3134h

BX = 0000h

CX = 0E23h

DX = 072Ah

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0109h

ÇIKTI

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 ► mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

BX'te sakla; INT 21h/02h AL'yi değiştirir.

Adım 7 / 28

DEĞİŞKENLER

AX = 3134h

BX = 3134h

CX = 0E23h

DX = 072Ah

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0109h

ÇIKTI



Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 ► mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

DL = onlar basamağı.

Adım 8 / 28

DEĞİŞKENLER

AX = 3134h

BX = 3134h

CX = 0E23h

DX = 0731h

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0109h

ÇIKTI

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 ► mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

AH = 02h.

Adım 9 / 28

DEĞİŞKENLER

AX = 0234h

BX = 3134h

CX = 0E23h

DX = 0731h

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0109h

ÇIKTI

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

Ekrana '1'.

Adım 10 / 28

DEĞİŞKENLER

AX = 0231h

BX = 3134h

CX = 0E23h

DX = 0731h

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0109h

ÇIKTI

1

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 ► mov dl, bl
20 int 21h
21 ret
```

DL = birler basamağı.

Adım 11 / 28

DEĞİŞKENLER

AX = 0231h

BX = 3134h

CX = 0E23h

DX = 0734h

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0109h

ÇIKTI

1

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

Ekrana '4' → **14**.

Adım 12 / 28

DEĞİŞKENLER

AX = 0234h

BX = 3134h

CX = 0E23h

DX = 0734h

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0109h

ÇIKTI

14

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
▶ ret
```

Prosedürden dönülür.

Adım 13 / 28

DEĞİŞKENLER

AX = 0234h

BX = 3134h

CX = 0E23h

DX = 0734h

SP = FFFEh

ÇAĞRI YİĞİNİ

boş

ÇIKTI

14

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 ► mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

DL = ':'.

Adım 14 / 28

DEĞİŞKENLER

AX = 0234h

BX = 3134h

CX = 0E23h

DX = 073Ah

SP = FFFEh

ÇAĞRI YİĞİNİ

boş

ÇIKTI

14

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 ► mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

AH = 02h.

Adım 15 / 28

DEĞİŞKENLER

AX = 0234h

BX = 3134h

CX = 0E23h

DX = 073Ah

SP = FFFEh

ÇAĞRI YİĞİNİ

boş

ÇIKTI

14

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
▶ int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

Ekrana ':\.

Adım 16 / 28

DEĞİŞKENLER

AX = 023Ah

BX = 3134h

CX = 0E23h

DX = 073Ah

SP = FFFEh

ÇAĞRI YİĞİNİ

boş

ÇIKTI

14:

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 ► mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

AL = dakika = 23h (35). CL hiçbir çağrıda bozulmadı.

Adım 17 / 28

DEĞİŞKENLER

AX = 0223h

BX = 3134h

CX = 0E23h

DX = 073Ah

SP = FFFEh

ÇAĞRI YİĞİNİ

boş

ÇIKTI

14:

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
▶ call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

Dakika için aynı prosedür.

Adım 18 / 28

DEĞİŞKENLER

AX = 0223h

BX = 3134h

CX = 0E23h

DX = 073Ah

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0114h

ÇIKTI

14:

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 ► aam           ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

35 → AH = 3, AL = 5.

Adım 19 / 28

DEĞİŞKENLER

AX = 0305h

BX = 3134h

CX = 0E23h

DX = 073Ah

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0114h

ÇIKTI

14:

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 ► add ax, 3030h ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

Her iki basamak ASCII'ye: AX = 3335h.

Adım 20 / 28

DEĞİŞKENLER

AX = 3335h

BX = 3134h

CX = 0E23h

DX = 073Ah

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0114h

ÇIKTI

14:

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 ► mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

BX'te sakla; INT 21h/02h AL'yi değiştirir.

Adım 21 / 28

DEĞİŞKENLER

AX = 3335h

BX = 3335h

CX = 0E23h

DX = 073Ah

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0114h

ÇIKTI

14:

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 ► mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

DL = onlar basamağı.

Adım 22 / 28

DEĞİŞKENLER

AX = 3335h

BX = 3335h

CX = 0E23h

DX = 0733h

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0114h

ÇIKTI

14:

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 ► mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

AH = 02h.

Adım 23 / 28

DEĞİŞKENLER

AX = 0235h

BX = 3335h

CX = 0E23h

DX = 0733h

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0114h

ÇIKTI

14:

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

Ekrana '3'.

Adım 24 / 28

DEĞİŞKENLER

AX = 0233h

BX = 3335h

CX = 0E23h

DX = 0733h

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0114h

ÇIKTI

14:3

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 ► mov dl, bl
20 int 21h
21 ret
```

DL = birler basamağı.

Adım 25 / 28

DEĞİŞKENLER

AX = 0233h

BX = 3335h

CX = 0E23h

DX = 0735h

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0114h

ÇIKTI

14:3

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

Ekrana '5' → **14:35**.

Adım 26 / 28

DEĞİŞKENLER

AX = 0235h

BX = 3335h

CX = 0E23h

DX = 0735h

SP = FFFCh

ÇAĞRI YİĞİNİ

FFFCh: 0114h

ÇIKTI

14:35

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch      ; sistem saatini al
3 int 21h          ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ret
12 yaz2:           ; AL (0-99) → 2 hane
13 aam             ; AH=onlar AL=birler
14 add ax, 3030h   ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
▶ ret
```

Prosedürden dönülür.

Adım 27 / 28

DEĞİŞKENLER

AX = 0235h

BX = 3335h

CX = 0E23h

DX = 0735h

SP = FFFEh

ÇAĞRI YİĞİNİ

boş

ÇIKTI

14:35

Kod İzleme: Saati SS:DD Biçiminde Yazdırma

```
1 org 100h
2 mov ah, 2Ch          ; sistem saatini al
3 int 21h              ; CH=saat CL=dakika
4 mov al, ch
5 call yaz2
6 mov dl, ':'
7 mov ah, 02h
8 int 21h
9 mov al, cl
10 call yaz2
11 ► ret
12 yaz2:               ; AL (0-99) → 2 hane
13 aam                 ; AH=onlar AL=birler
14 add ax, 3030h       ; ASCII
15 mov bx, ax
16 mov dl, bh
17 mov ah, 02h
18 int 21h
19 mov dl, bl
20 int 21h
21 ret
```

Ekranda **14:35**. Tarih için AH = 2Ah: CX = yıl, DH = ay, DL = gün, AL = haftanın günü.

Adım 28 / 28

DEĞİŞKENLER

AX = 0235h

BX = 3335h

CX = 0E23h

DX = 0735h

SP = FFFEh

ÇAĞRI YİĞİNİ

boş

ÇIKTI

14:35

INT 1Ah: Saat Sayacı ve Gerçek Zaman Saati

AH	İşlev	Dönüş
00h	tık sayacını oku	CX:DX = gece yarısından beri tık, AL = gün döndü bayrağı
02h	RTC saatini oku	CH = saat, CL = dakika, DH = saniye (BCD), DL = yaz saati; CF = 1 : hata
04h	RTC tarihini oku	CH = yüzyıl, CL = yıl, DH = ay, DL = gün (BCD); CF = 1: hata

Paketli BCD → ASCII (CL = 35h, yani dakika 35)

35h = 0011 0101b

Her 4 bit bir onluk basamaktır: yüksek nibble 3, düşük nibble 5.

Adım 1 / 4

RTC, pille beslenen CMOS saatidir (AT ve sonrası); bilgisayar kapalıyken de çalışır.

emu8086'nın desteklediği kesmeler arasında INT 1Ah için yalnızca **00h** (tık sayacı) bulunur; 02h gerçek DOS ya da DOSBox'ta çalışır. emu8086'da saati INT 21h/**2Ch** ile okuyun (INT 21h DOS kısmı).

INT 1Ah: Saat Sayacı ve Gerçek Zaman Saati

AH	İşlev	Dönüş
00h	tık sayacını oku	CX:DX = gece yarısından beri tık, AL = gün döndü bayrağı
02h	RTC saatini oku	CH = saat, CL = dakika, DH = saniye (BCD), DL = yaz saati; CF = 1 : hata
04h	RTC tarihini oku	CH = yüzyıl, CL = yıl, DH = ay, DL = gün (BCD); CF = 1: hata

Paketli BCD → ASCII (CL = 35h, yani dakika 35)

`shr dl, cl` (CL = 4) → **03h**

Üst basamak sağa kaydırılır. 8086'da 1'den büyük kaydırma sayısı CL ile verilir.

Adım 2 / 4

RTC, pille beslenen CMOS saatidir (AT ve sonrası); bilgisayar kapalıyken de çalışır.

emu8086'nın desteklediği kesmeler arasında INT 1Ah için yalnızca **00h** (tık sayacı) bulunur; 02h gerçek DOS ya da DOSBox'ta çalışır. emu8086'da saati INT 21h/**2Ch** ile okuyun (INT 21h DOS kısmı).

INT 1Ah: Saat Sayacı ve Gerçek Zaman Saati

AH	İşlev	Dönüş
00h	tık sayacını oku	CX:DX = gece yarısından beri tık, AL = gün döndü bayrağı
02h	RTC saatini oku	CH = saat, CL = dakika, DH = saniye (BCD), DL = yaz saati; CF = 1 : hata
04h	RTC tarihini oku	CH = yüzyıl, CL = yıl, DH = ay, DL = gün (BCD); CF = 1: hata

Paketli BCD → ASCII (CL = 35h, yani dakika 35)

`and dl, 0Fh` → **05h**

Alt basamak maskelenir.

Adım 3 / 4

RTC, pille beslenen CMOS saatidir (AT ve sonrası); bilgisayar kapalıyken de çalışır.

emu8086'nın desteklediği kesmeler arasında INT 1Ah için yalnızca **00h** (tık sayacı) bulunur; 02h gerçek DOS ya da DOSBox'ta çalışır. emu8086'da saati INT 21h/**2Ch** ile okuyun (INT 21h DOS kısmı).

INT 1Ah: Saat Sayacı ve Gerçek Zaman Saati

AH	İşlev	Dönüş
00h	tık sayacını oku	CX:DX = gece yarısından beri tık, AL = gün döndü bayrağı
02h	RTC saatini oku	CH = saat, CL = dakika, DH = saniye (BCD), DL = yaz saati; CF = 1 : hata
04h	RTC tarihini oku	CH = yüzyıl, CL = yıl, DH = ay, DL = gün (BCD); CF = 1: hata

Paketli BCD → ASCII (CL = 35h, yani dakika 35)

+30h → '**3**' 33h, '**5**' 35h

Her basamak ASCII rakama çevrilir. INT 21h/2Ch ise aynı dakikayı ikili verir: CL = 23h; orada AAM gerekir.

Adım 4 / 4

RTC, pille beslenen CMOS saatidir (AT ve sonrası); bilgisayar kapalıyken de çalışır.

emu8086'nın desteklediği kesmeler arasında INT 1Ah için yalnızca **00h** (tık sayacı) bulunur; 02h gerçek DOS ya da DOSBox'ta çalışır. emu8086'da saati INT 21h/**2Ch** ile okuyun (INT 21h DOS kısmı).

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 ▶ mov ah, 02h
3 int 1Ah      ; CH=saat CL=dk DH=sn
4 jc son      ; CF=1: saat okunamadı
5 mov bl, cl   ; dakikayı sakla
6 mov cl, 4    ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz  ; "14"
9 mov bh, bl
10 call ikinokta ; ":35"
11 mov bh, dh
12 call ikinokta ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:      ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl   ; üst hane
20 add dl, 30h  ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh  ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

AH = 02h: BIOS gerçek zaman saatini (RTC) oku.

Adım 1 / 37

DEĞİŞKENLER

AX = 0200h

BX = 0000h

CX = 0000h

DX = 0000h

CF = 0

ÇAĞRI YİĞİNİ

boş

ÇIKTI

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 ► int 1Ah ; CH=saat CL=dk DH=sn
4 jc son ; CF=1: saat okunamadı
5 mov bl, cl ; dakikayı sakla
6 mov cl, 4 ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz ; "14"
9 mov bh, bl
10 call ikinokta ; ":35"
11 mov bh, dh
12 call ikinokta ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz: ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl ; üst hane
20 add dl, 30h ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Dönüş **paketli BCD**: CH = **14h** (saat 14), CL = **35h** (dakika 35), DH = **07h** (saniye). DL = 00h (yaz saati yok).

Adım 2 / 37

DEĞİŞKENLER

AX = 0200h

BX = 0000h

CX = 1435h

DX = 0700h

CF = 0

ÇAĞRI YİĞİNİ

boş

ÇIKTI

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah      ; CH=saat CL=dk DH=sn
4 jc son      ; CF=1: saat okunamadı
5 mov bl, cl   ; dakikayı sakla
6 mov cl, 4    ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz  ; "14"
9 mov bh, bl
10 call ikinokta ; ":35"
11 mov bh, dh
12 call ikinokta ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:      ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl   ; üst hane
20 add dl, 30h  ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh  ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

CF = 0: okuma başarılı. CF = 1 olsaydı saat çalışmıyor demektir; program yazdırmadan biterdi.

Adım 3 / 37

DEĞİŞKENLER

AX = 0200h

BX = 0000h

CX = 1435h

DX = 0700h

CF = 0

ÇAĞRI YİĞİNİ

boş

ÇIKTI

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 ▶ mov bl, cl     ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

CL'ye birazdan kaydırma sayısı yazılacak; dakika önce BL'ye kopyalanır.

Adım 4 / 37

DEĞİŞKENLER

AX = 0200h

BX = 0035h

CX = 1435h

DX = 0700h

CF = 0

ÇAĞRI YİĞİNİ

boş

ÇIKTI

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl       ; dakikayı sakla
▶ mov cl, 4        ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz      ; "14"
9 mov bh, bl
10 call ikinokta   ; ":35"
11 mov bh, dh
12 call ikinokta   ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:         ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl      ; üst hane
20 add dl, 30h     ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh     ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

CL = 4: SHR DL, CL 4 bit kaydıracak (8086'da 1'den büyük sabit kaydırma yok).

Adım 5 / 37

DEĞİŞKENLER

AX = 0200h

BX = 0035h

CX = 1404h

DX = 0700h

CF = 0

ÇAĞRI YİĞİNİ

boş

ÇIKTI

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl       ; dakikayı sakla
6 mov cl, 4        ; SHR kaydırma sayısı
7 ► mov bh, ch
8 call bcdyaz      ; "14"
9 mov bh, bl
10 call ikinokta    ; ":35"
11 mov bh, dh
12 call ikinokta    ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:          ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl       ; üst hane
20 add dl, 30h      ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh      ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

BH = saat = 14h. Prosedür değeri BH'de bekler; INT 21h/02h BH'yi bozmaz.

Adım 6 / 37

DEĞİŞKENLER

AX = 0200h

BX = 1435h

CX = 1404h

DX = 0700h

CF = 0

ÇAĞRI YİĞİNİ

boş

ÇIKTI

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta   ; ":35"
11 mov bh, dh
12 call ikinokta   ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:         ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl      ; üst hane
20 add dl, 30h     ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh     ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Saat için prosedür çağrılır (ikinokta atlanır, doğrudan bcdyaz).

Adım 7 / 37

DEĞİŞKENLER

AX = 0200h

BX = 1435h

CX = 1404h

DX = 0700h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 010Fh

ÇIKTI

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl       ; dakikayı sakla
6 mov cl, 4        ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz      ; "14"
9 mov bh, bl
10 call ikinokta    ; ":35"
11 mov bh, dh
12 call ikinokta    ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:          ; BH = paketli BCD
18 ► mov dl, bh
19 shr dl, cl       ; üst hane
20 add dl, 30h      ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh      ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

DL = BH = 14h = 0001 0100b.

Adım 8 / 37

DEĞİŞKENLER

AX = 0200h

BX = 1435h

CX = 1404h

DX = 0714h

CF = 0

ÇAĞRI YİĞİNİ

FFFCh: 010Fh

ÇIKTI

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl       ; dakikayı sakla
6 mov cl, 4        ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz      ; "14"
9 mov bh, bl
10 call ikinokta   ; ":35"
11 mov bh, dh
12 call ikinokta   ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:         ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl       ; üst hane
20 add dl, 30h      ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh      ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

`shr dl, cl` : 0001 0100b → 0000 0001b. DL = **01h**, yani üst basamak.

Adım 9 / 37

DEĞİŞKENLER

AX = 0200h

BX = 1435h

CX = 1404h

DX = 0701h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 010Fh

ÇIKTI

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 ► add dl, 30h  ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

+30h: DL = 31h = '1'.

Adım 10 / 37

DEĞİŞKENLER

AX = 0200h

BX = 1435h

CX = 1404h

DX = 0731h

CF = 0

ÇAĞRI YİĞİNİ

FFFCh: 010Fh

ÇIKTI

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta   ; ":35"
11 mov bh, dh
12 call ikinokta   ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 ► mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

AH = 02h: DL'deki karakteri yaz.

Adım 11 / 37

DEĞİŞKENLER

AX = 0200h

BX = 1435h

CX = 1404h

DX = 0731h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 010Fh

ÇIKTI

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h   ; ASCII rakam
21 mov ah, 02h
22 ► int 21h
23 mov dl, bh
24 and dl, 0Fh   ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Ekrana '1'. DOS, AL'ye DL'yi koyar (AL = 31h).

Adım 12 / 37

DEĞİŞKENLER

AX = 0231h

BX = 1435h

CX = 1404h

DX = 0731h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 010Fh

ÇIKTI

1

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl       ; dakikayı sakla
6 mov cl, 4        ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz      ; "14"
9 mov bh, bl
10 call ikinokta   ; ":35"
11 mov bh, dh
12 call ikinokta   ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:         ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl       ; üst hane
20 add dl, 30h      ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 ► mov dl, bh
24 and dl, 0Fh      ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

DL = BH = 14h (ikinci basamak için yeniden).

Adım 13 / 37

DEĞİŞKENLER

AX = 0231h

BX = 1435h

CX = 1404h

DX = 0714h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 010Fh

ÇIKTI

1

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl       ; dakikayı sakla
6 mov cl, 4        ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz      ; "14"
9 mov bh, bl
10 call ikinokta   ; ":35"
11 mov bh, dh
12 call ikinokta   ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:         ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl      ; üst hane
20 add dl, 30h     ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh     ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

and dl, 0Fh : üst 4 bit silinir: 0000 0100b = **04h**.

Adım 14 / 37

DEĞİŞKENLER

AX = 0231h

BX = 1435h

CX = 1404h

DX = 0704h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 010Fh

ÇIKTI

1

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4      ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 ► add dl, 30h
26 int 21h
27 ret
```

+30h: DL = 34h = '4'.

Adım 15 / 37

DEĞİŞKENLER

AX = 0231h

BX = 1435h

CX = 1404h

DX = 0734h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 010Fh

ÇIKTI

1

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl       ; dakikayı sakla
6 mov cl, 4        ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz      ; "14"
9 mov bh, bl
10 call ikinokta   ; ":35"
11 mov bh, dh
12 call ikinokta   ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:         ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl      ; üst hane
20 add dl, 30h     ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh     ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Ekrana '4' → **14.** AH hâlâ 02h olduğundan yeniden yüklemek gerekmez.

Adım 16 / 37

DEĞİŞKENLER

AX = 0234h

BX = 1435h

CX = 1404h

DX = 0734h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 010Fh

ÇIKTI

14

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
▶ ret
```

Prosedürden dönülür.

Adım 17 / 37

DEĞİŞKENLER

AX = 0234h

BX = 1435h

CX = 1404h

DX = 0734h

CF = 0

ÇAĞRI YİĞİNİ

boş

ÇIKTI

14

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 ► mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

BH = dakika = 35h (BL'de saklanmıştı).

Adım 18 / 37

DEĞİŞKENLER

AX = 0234h

BX = 3535h

CX = 1404h

DX = 0734h

CF = 0

ÇAĞRI YİĞİNİ

boş

ÇIKTI

14

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Bu kez **ikinokta** girişi çağrılır: Önce ':' yazar, sonra bcdyaz'a düşer.

Adım 19 / 37

DEĞİŞKENLER

AX = 0234h

BX = 3535h

CX = 1404h

DX = 0734h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0114h

ÇIKTI

14

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta   ; ":35"
11 mov bh, dh
12 call ikinokta   ; ":07"
13 son: ret
▶ ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

DL = ':' (3Ah).

Adım 20 / 37

DEĞİŞKENLER

AX = 0234h

BX = 3535h

CX = 1404h

DX = 073Ah

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0114h

ÇIKTI

14

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl       ; dakikayı sakla
6 mov cl, 4        ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz      ; "14"
9 mov bh, bl
10 call ikinokta    ; ":35"
11 mov bh, dh
12 call ikinokta    ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 ► mov ah, 02h
16 int 21h
17 bcdyaz:          ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl       ; üst hane
20 add dl, 30h      ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh      ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

AH = 02h.

Adım 21 / 37

DEĞİŞKENLER

AX = 0234h

BX = 3535h

CX = 1404h

DX = 073Ah

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0114h

ÇIKTI

14

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl       ; dakikayı sakla
6 mov cl, 4        ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz      ; "14"
9 mov bh, bl
10 call ikinokta   ; ":35"
11 mov bh, dh
12 call ikinokta   ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:         ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl      ; üst hane
20 add dl, 30h     ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh     ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Ekrana ':' → **14:**. Ardından komut bcdyaz etiketine kendiliğinden geçer (fall-through).

Adım 22 / 37

DEĞİŞKENLER

AX = 023Ah

BX = 3535h

CX = 1404h

DX = 073Ah

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0114h

ÇIKTI

14:

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl       ; dakikayı sakla
6 mov cl, 4        ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz      ; "14"
9 mov bh, bl
10 call ikinokta    ; ":35"
11 mov bh, dh
12 call ikinokta    ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:          ; BH = paketli BCD
18 mov dl, bh
19 ► shr dl, cl     ; üst hane
20 add dl, 30h      ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh      ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

35h → **03h** (üst basamak).

Adım 23 / 37

DEĞİŞKENLER

AX = 023Ah

BX = 3535h

CX = 1404h

DX = 0703h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0114h

ÇIKTI

14:

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta   ; ":35"
11 mov bh, dh
12 call ikinokta   ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 ► int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Ekrana '3' (33h).

Adım 24 / 37

DEĞİŞKENLER

AX = 0233h

BX = 3535h

CX = 1404h

DX = 0733h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0114h

ÇIKTI

14:3

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4      ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

35h AND 0Fh = **05h**.

Adım 25 / 37

DEĞİŞKENLER

AX = 0233h

BX = 3535h

CX = 1404h

DX = 0705h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0114h

ÇIKTI

14:3

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh   ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Ekrana '5' → **14:35**.

Adım 26 / 37

DEĞİŞKENLER

AX = 0235h

BX = 3535h

CX = 1404h

DX = 0735h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0114h

ÇIKTI

14:35

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
▶ ret
```

Dönülür.

Adım 27 / 37

DEĞİŞKENLER

AX = 0235h

BX = 3535h

CX = 1404h

DX = 0735h

CF = 0

ÇAĞRI YİĞİNİ

boş

ÇIKTI

14:35

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 ► mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

BH = saniye = 07h. DH, INT 21h/02h tarafından değiştirilmedi.

Adım 28 / 37

DEĞİŞKENLER

AX = 0235h

BX = 0735h

CX = 1404h

DX = 0735h

CF = 0

ÇAĞRI YIĞINI

boş

ÇIKTI

14:35

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 ► call ikinokta ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Saniye için yine ikinokta.

Adım 29 / 37

DEĞİŞKENLER

AX = 0235h

BX = 0735h

CX = 1404h

DX = 0735h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0119h

ÇIKTI

14:35

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta   ; ":35"
11 mov bh, dh
12 call ikinokta   ; ":07"
13 son: ret
▶ ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:         ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl      ; üst hane
20 add dl, 30h     ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh     ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

DL = ':'.

Adım 30 / 37

DEĞİŞKENLER

AX = 0235h

BX = 0735h

CX = 1404h

DX = 073Ah

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0119h

ÇIKTI

14:35

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta   ; ":35"
11 mov bh, dh
12 call ikinokta   ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Ekrana ':' → **14:35:**

Adım 31 / 37

DEĞİŞKENLER

AX = 023Ah

BX = 0735h

CX = 1404h

DX = 073Ah

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0119h

ÇIKTI

14:35:

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 ► shr dl, cl   ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

07h → **00h** (üst basamak).

Adım 32 / 37

DEĞİŞKENLER

AX = 023Ah

BX = 0735h

CX = 1404h

DX = 0700h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0119h

ÇIKTI

14:35:

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 ► int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Ekrana '0' (30h).

Adım 33 / 37

DEĞİŞKENLER

AX = 0230h

BX = 0735h

CX = 1404h

DX = 0730h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0119h

ÇIKTI

14:35:0

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4      ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

07h AND 0Fh = **07h**.

Adım 34 / 37

DEĞİŞKENLER

AX = 0230h

BX = 0735h

CX = 1404h

DX = 0707h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0119h

ÇIKTI

14:35:0

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Ekrana '7' → **14:35:07**.

Adım 35 / 37

DEĞİŞKENLER

AX = 0237h

BX = 0735h

CX = 1404h

DX = 0737h

CF = 0

ÇAĞRI YİĞİNİ

FFCh: 0119h

ÇIKTI

14:35:07

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
▶ ret
```

Dönülür.

Adım 36 / 37

DEĞİŞKENLER

AX = 0237h

BX = 0735h

CX = 1404h

DX = 0737h

CF = 0

ÇAĞRI YİĞİNİ

boş

ÇIKTI

14:35:07

Kod İzleme: RTC Saatini SS:DD:ss Yazdırma

```
1 org 100h
2 mov ah, 02h
3 int 1Ah          ; CH=saat CL=dk DH=sn
4 jc son          ; CF=1: saat okunamadı
5 mov bl, cl      ; dakikayı sakla
6 mov cl, 4       ; SHR kaydırma sayısı
7 mov bh, ch
8 call bcdyaz     ; "14"
9 mov bh, bl
10 call ikinokta  ; ":35"
11 mov bh, dh
12 call ikinokta  ; ":07"
13 ► son: ret
14 ikinokta: mov dl, ':'
15 mov ah, 02h
16 int 21h
17 bcdyaz:        ; BH = paketli BCD
18 mov dl, bh
19 shr dl, cl     ; üst hane
20 add dl, 30h    ; ASCII rakam
21 mov ah, 02h
22 int 21h
23 mov dl, bh
24 and dl, 0Fh    ; alt hane
25 add dl, 30h
26 int 21h
27 ret
```

Ekranında **14:35:07**. Aynı saat INT 21h/2Ch ile ikili (CH = 0Eh, CL = 23h) gelirdi; BCD'de onaltılık basamaklar doğrudan onluk basamaklardır.

Adım 37 / 37

DEĞİŞKENLER

AX = 0237h

BX = 0735h

CX = 1404h

DX = 0737h

CF = 0

ÇAĞRI YİĞİNİ

boş

ÇIKTI

14:35:07

INT 1Ah VE BIOS VERİLERİ

BIOS Veri Alanı (Kesim 0040h)

Adres	Boyut	İçerik
0040:0010	sözcük	donanım listesi (INT 11h'nin döndürdüğü değer)
0040:0013	sözcük	taban bellek, KB (INT 12h)
0040:0017	bayt	klavye bayrakları: bit 0 sağ Shift, 1 sol Shift, 2 Ctrl, 3 Alt, 4 Scroll Lock, 5 Num Lock, 6 Caps Lock, 7 Insert
0040:001A	sözcük	tampon başı : sıradaki okunacak tuşun bağlı konumu
0040:001C	sözcük	tampon sonu : sıradaki boş yerin bağlı konumu (baş = son → tampon boş)
0040:001E– 003D	32 bayt	klavye tamponu: 16 sözcük (ASCII + tarama kodu), dairesel; en çok 15 tuş
0040:0049	bayt	geçerli video modu (03h, 13h, ...)
0040:004A	sözcük	ekrandaki sütun sayısı (mod 03h'te 80)
0040:0050	8 sözcük	8 sayfanın imleç konumu (düşük bayt sütun, yüksek bayt satır)
0040:006C	dword	tık sayacı : gece yarısından beri $\approx 18,2$ Hz'lik kesme sayısı
0040:0070	bayt	gece yarısı geçti bayrağı (INT 1Ah/00h AL'de verir ve sıfırlar)

Fiziksel adres = 00400h + bağlı konum: 0040:006C = 0046Ch = 0000:046C. INT 16h ve INT 1Ah bu alanı kullanır; program da doğrudan okuyabilir.

BIOS Veri Alanını Okumak

```
; yaklaşık 1 saniye bekle
mov ax, 0040h
mov es, ax
mov bx, es:[006Ch] ; başlangıç tıklı
bekle:
mov ax, es:[006Ch] ; şimdiki tıklı
sub ax, bx          ; geçen tıklı
cmp ax, 18          ; 18 tıklı ≈ 0,99 s
jb  bekle

; Caps Lock açık mı? Bekleyen tuş var mı?
test byte ptr es:[0017h], 40h
jnz caps_acik       ; bit 6 = 1
mov ax, es:[001Ah]  ; tampon başı
cmp ax, es:[001Ch]  ; tampon sonu
je  tampon_bos
```

$$1\ 193\ 182\ \text{Hz} \div 65\ 536 = \mathbf{18,2065\ \text{Hz}}$$

8253/8254 zamanlayıcısı bölen 65 536 iken
IRQ 0'ı bu sıklıkta üretir; her kesmede
0040:006C bir artar.

Adım 1 / 4

emu8086 BIOS veri alanının tamamını gerçek PC'deki gibi güncellemez; tıklı sayacı ve klavye tamponu örnekleri DOSBox'ta tam çalışır.

BIOS Veri Alanını Okumak

```
; yaklaşık 1 saniye bekle
mov ax, 0040h
mov es, ax
mov bx, es:[006Ch] ; başlangıç tıklı
bekle:
mov ax, es:[006Ch] ; şimdiki tık
sub ax, bx          ; geçen tık
cmp ax, 18          ; 18 tık ≈ 0,99 s
jb  bekle

; Caps Lock açık mı? Bekleyen tuş var mı?
test byte ptr es:[0017h], 40h
jnz caps_acik       ; bit 6 = 1
mov ax, es:[001Ah]  ; tampon başı
cmp ax, es:[001Ch]  ; tampon sonu
je  tampon_bos
```

1 tık = $65\,536 \div 1\,193\,182 \text{ s} \approx \mathbf{54,9 \text{ ms}}$

18 tık $\approx 0,989 \text{ s} \cdot 5 \text{ s} \approx 91 \text{ tık}$

Bekleme süresi tık sayısı ile ölçülür;
çözünürlük yaklaşık 55 ms'dir.

Adım 2 / 4

emu8086 BIOS veri alanının tamamını gerçek PC'deki gibi güncellemez; tık sayacı ve klavye tamponu örnekleri DOSBox'ta tam çalışır.

BIOS Veri Alanını Okumak

```
; yaklaşık 1 saniye bekle
mov ax, 0040h
mov es, ax
mov bx, es:[006Ch] ; başlangıç tıklı
bekle:
mov ax, es:[006Ch] ; şimdiki tıklı
sub ax, bx          ; geçen tıklı
cmp ax, 18          ; 18 tıklı ≈ 0,99 s
jb  bekle

; Caps Lock açık mı? Bekleyen tuş var mı?
test byte ptr es:[0017h], 40h
jnz caps_acik       ; bit 6 = 1
mov ax, es:[001Ah]  ; tampon başı
cmp ax, es:[001Ch]  ; tampon sonu
je  tampon_bos
```

Günde 1800B0h = **1 573 040** tıklı

Sayaç bu değere ulaştınca sıfırlanır ve
0040:0070 bayrağı kurulur.

Adım 3 / 4

emu8086 BIOS veri alanının tamamını gerçek PC'deki gibi güncellemez; tıklı sayacı ve klavye tamponu örnekleri DOSBox'ta tam çalışır.

BIOS Veri Alanını Okumak

```
; yaklaşık 1 saniye bekle
mov ax, 0040h
mov es, ax
mov bx, es:[006Ch] ; başlangıç tıklı
bekle:
mov ax, es:[006Ch] ; şimdiki tıklı
sub ax, bx          ; geçen tıklı
cmp ax, 18          ; 18 tıklı ≈ 0,99 s
jb  bekle

; Caps Lock açık mı? Bekleyen tuş var mı?
test byte ptr es:[0017h], 40h
jnz caps_acik       ; bit 6 = 1
mov ax, es:[001Ah]  ; tampon başı
cmp ax, es:[001Ch]  ; tampon sonu
je  tampon_bos
```

`sub ax, bx` ile fark

Yalnız düşük sözcük okunur. Fark alındığı için düşük sözcük FFFFh → 0000h dönse de sonuç doğrudur; yalnız gece yarısındaki sıfırlama hesabı bozar.

Adım 4 / 4

emu8086 BIOS veri alanının tamamını gerçek PC'deki gibi güncellemez; tık sayacı ve klavye tamponu örnekleri DOSBox'ta tam çalışır.

PROGRAM SONLANDIRMA

Programı Sonlandırmanın Üç Yolu

Yöntem	Nasıl çalışır?	Nerede?
<code>ret</code>	COM programında yığının üstünde 0000h vardır; RET, PSP:0000h'a döner, orada <code>int 20h</code> (CD 20) bulunur.	yalnız COM, yığın dengeli ise
<code>int 20h</code>	DOS programı sonlandırır; CS'nin PSP kesimini göstermesi gerekir.	COM
<code>mov ax, 4C00h</code> <code>int 21h</code>	AL'deki çıkış kodu ile sonlandırır, açık dosyaları kapatır.	COM ve EXE · önerilen

Çıkış kodu

AL = 0 başarı, ≠ 0 hata anlamına gelir. Toplu iş dosyasında `IF ERRORLEVEL 1` ile sınanır.

Sık hata

Sonlandırma komutu unutulursa işlemci programdan sonraki baytları (verileri!) komut gibi yürütür ve sistem kilitlenebilir.

PSP (Program Segment Prefix): DOS'un her programın önüne koyduğu 256 baytlık alan; COM'larda kod bu yüzden ORG 100h'tan başlar.

emu8086 Şablonları: COM ve EXE

COM (tek kesim)

```
org 100h
mov dx, offset msg
mov ah, 09h
int 21h
ret          ; ya da int 20h
msg db 'Merhaba$'
```

- CS = DS = ES = SS; ayar gerekmez.
- En fazla 64 KB (kod + veri + yığın).

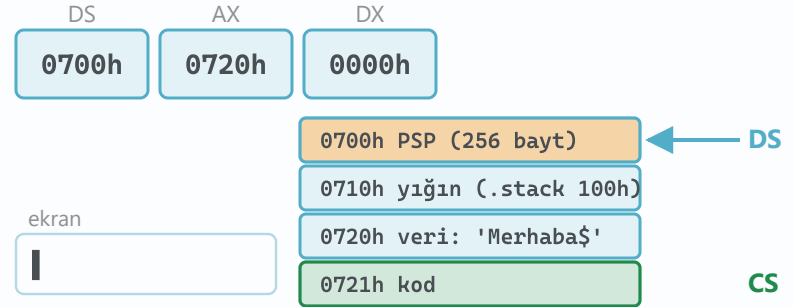
EXE (ayrı kesimler)

```
.model small
.stack 100h
.data
msg db 'Merhaba$'
.code
basla:
    mov ax, @data
    mov ds, ax      ; DS'yi ayarla!
    mov dx, offset msg
    mov ah, 09h
    int 21h
    mov ax, 4C00h
    int 21h
end basla
```

EXE'de DOS başlangıçta DS'yi PSP'ye ayarlar; `mov ax, @data / mov ds, ax` yapılmazsa DS:DX yanlış yeri gösterir ve ekrana çöp basılır.

EXE Programında DS Ayarı

```
1 .model small
2 .stack 100h
3 .data
4 msg db 'Merhaba$'
5 .code
6 basla:
7 ► mov ax, @data
8 mov ds, ax      ; DS'yi ayarla!
9 mov dx, offset msg
10 mov ah, 09h
11 int 21h
12 mov ax, 4C00h
13 int 21h
14 end basla
```



Başlangıçta DS = ES = **0700h**: DOS onları PSP'ye ayarlar. CS = 0721h (kod), SS = 0710h (yığın). `@data` veri kesiminin değeridir → AX = 0720h.

Adım 1 / 6

DEĞİŞKENLER

DS = 0700h

AX = 0720h

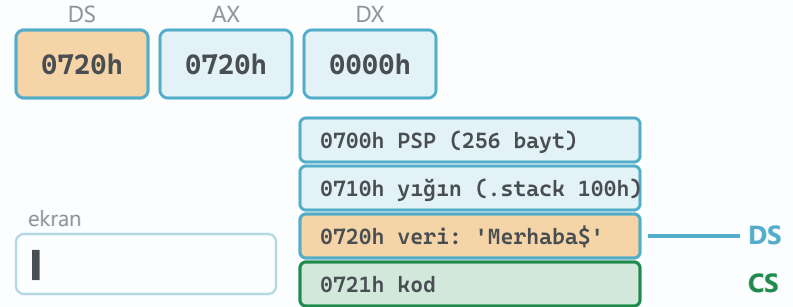
DX = 0000h

CS = 0721h

Kesim değerleri yorumlayıcının yerleşimidir (PSP 0700h); gerçek DOS'ta sayılar farklı olabilir, sıra ve DS ayarının gereği aynıdır.

EXE Programında DS Ayarı

```
1 .model small
2 .stack 100h
3 .data
4 msg db 'Merhaba$'
5 .code
6 basla:
7 mov ax, @data
8 ► mov ds, ax ; DS'yi ayarla!
9 mov dx, offset msg
10 mov ah, 09h
11 int 21h
12 mov ax, 4C00h
13 int 21h
14 end basla
```



MOV DS, AX: artık DS = 0720h, veri kesimini gösteriyor. Bu satır unutulsaydı DS:DX PSP'yi gösterir, ekrana çöp basılırdı.

Adım 2 / 6

DEĞİŞKENLER

DS = 0720h

AX = 0720h

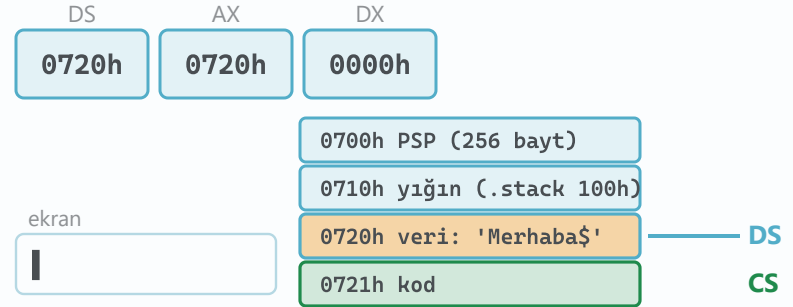
DX = 0000h

CS = 0721h

Kesim değerleri yorumlayıcının yerleşimidir (PSP 0700h); gerçek DOS'ta sayılar farklı olabilir, sıra ve DS ayarının gereği aynıdır.

EXE Programında DS Ayarı

```
1 .model small
2 .stack 100h
3 .data
4 msg db 'Merhaba$'
5 .code
6 basla:
7 mov ax, @data
8 mov ds, ax      ; DS'yi ayarla!
9 ► mov dx, offset msg
10 mov ah, 09h
11 int 21h
12 mov ax, 4C00h
13 int 21h
14 end basla
```



DX = offset msg = 0000h: msg, veri kesiminin başında.

Adım 3 / 6

DEĞİŞKENLER

DS = 0720h

AX = 0720h

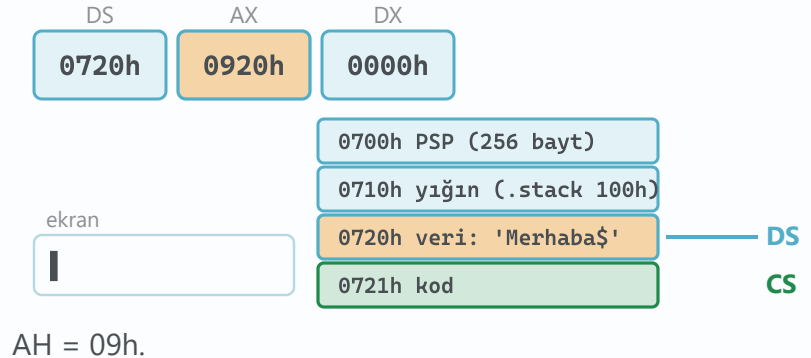
DX = 0000h

CS = 0721h

Kesim değerleri yorumlayıcının yerleşimidir (PSP 0700h); gerçek DOS'ta sayılar farklı olabilir, sıra ve DS ayarının gereği aynıdır.

EXE Programında DS Ayarı

```
1 .model small
2 .stack 100h
3 .data
4 msg db 'Merhaba$'
5 .code
6 basla:
7 mov ax, @data
8 mov ds, ax      ; DS'yi ayarla!
9 mov dx, offset msg
10 ► mov ah, 09h
11 int 21h
12 mov ax, 4C00h
13 int 21h
14 end basla
```



Adım 4 / 6

DEĞİŞKENLER

DS = 0720h

AX = 0920h

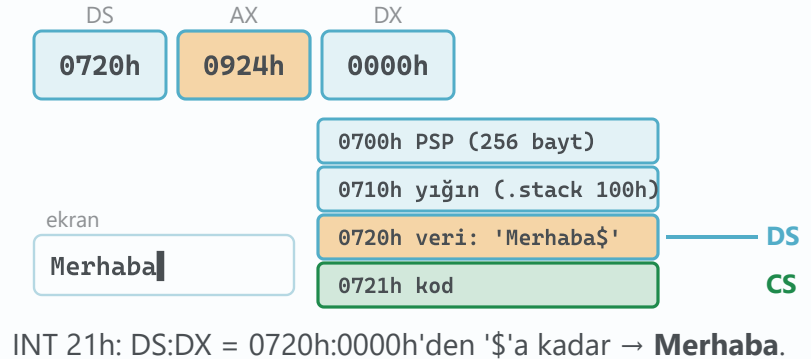
DX = 0000h

CS = 0721h

Kesim değerleri yorumlayıcının yerleşimidir (PSP 0700h); gerçek DOS'ta sayılar farklı olabilir, sıra ve DS ayarının gereği aynıdır.

EXE Programında DS Ayarı

```
1 .model small
2 .stack 100h
3 .data
4 msg db 'Merhaba$'
5 .code
6 basla:
7 mov ax, @data
8 mov ds, ax      ; DS'yi ayarla!
9 mov dx, offset msg
10 mov ah, 09h
11 ► int 21h
12 mov ax, 4C00h
13 int 21h
14 end basla
```



Adım 5 / 6

DEĞİŞKENLER

DS = 0720h

AX = 0924h

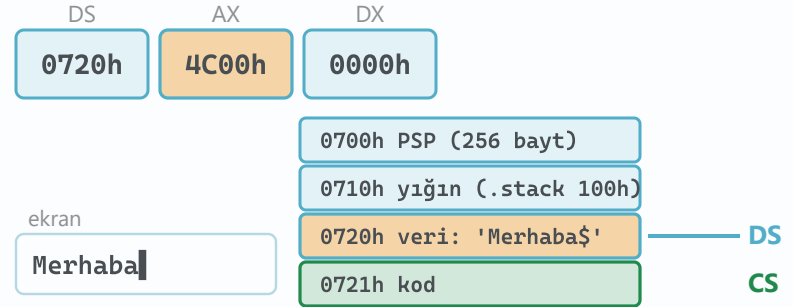
DX = 0000h

CS = 0721h

Kesim değerleri yorumlayıcının yerleşimidir (PSP 0700h); gerçek DOS'ta sayılar farklı olabilir, sıra ve DS ayarının gereği aynıdır.

EXE Programında DS Ayarı

```
1 .model small
2 .stack 100h
3 .data
4 msg db 'Merhaba$'
5 .code
6 basla:
7 mov ax, @data
8 mov ds, ax      ; DS'yi ayarla!
9 mov dx, offset msg
10 mov ah, 09h
11 int 21h
12 ▶ mov ax, 4C00h
13 ▶ int 21h
14 end basla
```



AX = 4C00h, INT 21h: çıkış kodu 0 ile DOS'a dön. EXE'de RET kullanılamaz (yığının üstünde PSP adresi yok).

Adım 6 / 6

DEĞİŞKENLER

DS = 0720h

AX = 4C00h

DX = 0000h

CS = 0721h

Kesim değerleri yorumlayıcının yerleşimidir (PSP 0700h); gerçek DOS'ta sayılar farklı olabilir, sıra ve DS ayarının gereği aynıdır.

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 ► mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

DX = menü metninin adresi.

Adım 1 / 22

DEĞİŞKENLER

AH = 00h

AL = 00h

DX = 0121h

ZF = 0

ÇIKTI



Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 ▶ mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

AH = 09h.

Adım 2 / 22

DEĞİŞKENLER

AH = 09h

AL = 00h

DX = 0121h

ZF = 0

ÇIKTI



Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 ► int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

Menü yazılır.

Adım 3 / 22

DEĞİŞKENLER

AH = 09h

AL = 24h

DX = 0121h

ZF = 0

ÇIKTI

1:Selam q:Cikis

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 ► mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h      ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

AH = 01h: yansıtılmalı tek karakter oku.

Adım 4 / 22

DEĞİŞKENLER

AH = 01h

AL = 24h

DX = 0121h

ZF = 0

ÇIKTI

1:Selam q:Cikis

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 ► int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

Kullanıcı **1** bastı: AL = 31h (ekranda görünür).

Adım 5 / 22

DEĞİŞKENLER

AH = 01h

AL = 31h

DX = 0121h

ZF = 0

ÇIKTI

1:Selam q:Cikis 1

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 ► cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

31h ≠ 'q' → ZF = 0.

Adım 6 / 22

DEĞİŞKENLER

AH = 01h

AL = 31h

DX = 0121h

ZF = 0

ÇIKTI

1:Selam q:Cikis 1

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

Çıkış değil.

Adım 7 / 22

DEĞİŞKENLER

AH = 01h

AL = 31h

DX = 0121h

ZF = 0

ÇIKTI

1:Selam q:Cikis 1

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 ▶ cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

AL = '1' → ZF = 1.

Adım 8 / 22

DEĞİŞKENLER

AH = 01h

AL = 31h

DX = 0121h

ZF = 1

ÇIKTI

1:Selam q:Cikis 1

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

Eşit → atlama yok (başka tuşa menü yeniden gösterilirdi).

Adım 9 / 22

DEĞİŞKENLER

AH = 01h

AL = 31h

DX = 0121h

ZF = 1

ÇIKTI

1:Selam q:Cikis 1

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 ► mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

DX = selam.

Adım 10 / 22

DEĞİŞKENLER

AH = 01h

AL = 31h

DX = 0134h

ZF = 1

ÇIKTI

1:Selam q:Cikis 1

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 ► mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

AH = 09h.

Adım 11 / 22

DEĞİŞKENLER

AH = 09h

AL = 31h

DX = 0134h

ZF = 1

ÇIKTI

1:Selam q:Cikis 1

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

Yeni satıra **Selam!** yazılır.

Adım 12 / 22

DEĞİŞKENLER

AH = 09h

AL = 24h

DX = 0134h

ZF = 1

ÇIKTI

```
1:Selam q:Cikis 1
Selam!
```

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 ► jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

Menüye dön.

Adım 13 / 22

DEĞİŞKENLER

AH = 09h

AL = 24h

DX = 0134h

ZF = 1

ÇIKTI

```
1:Selam q:Cikis 1
Selam!
```

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 ► mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

DX = menü metninin adresi.

Adım 14 / 22

DEĞİŞKENLER

AH = 09h

AL = 24h

DX = 0121h

ZF = 1

ÇIKTI

```
1:Selam q:Cikis 1
Selam!
```

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 ▶ mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

AH = 09h.

Adım 15 / 22

DEĞİŞKENLER

AH = 09h

AL = 24h

DX = 0121h

ZF = 1

ÇIKTI

```
1:Selam q:Cikis 1
Selam!
```

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 ▶ int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

Menü yeniden yazılır.

Adım 16 / 22

DEĞİŞKENLER

AH = 09h

AL = 24h

DX = 0121h

ZF = 1

ÇIKTI

```
1:Selam q:Cikis 1
Selam!
1:Selam q:Cikis
```

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 ► mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h      ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

AH = 01h: yansıtılmalı tek karakter oku.

Adım 17 / 22

DEĞİŞKENLER

AH = 01h

AL = 24h

DX = 0121h

ZF = 1

ÇIKTI

```
1:Selam q:Cikis 1
Selam!
1:Selam q:Cikis
```

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

Kullanıcı **q** bastı: AL = 71h.

Adım 18 / 22

DEĞİŞKENLER

AH = 01h

AL = 71h

DX = 0121h

ZF = 1

ÇIKTI

```
1:Selam q:Cikis 1
Selam!
1:Selam q:Cikis q
```

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 ▶ cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

AL = 'q' → ZF = 1.

Adım 19 / 22

DEĞİŞKENLER

AH = 01h

AL = 71h

DX = 0121h

ZF = 1

ÇIKTI

```
1:Selam q:Cikis 1
Selam!
1:Selam q:Cikis q
```

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 ► je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

→ cikis .

Adım 20 / 22

DEĞİŞKENLER

AH = 01h

AL = 71h

DX = 0121h

ZF = 1

ÇIKTI

```
1:Selam q:Cikis 1
Selam!
1:Selam q:Cikis q
```

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 ► mov ax, 4C00h   ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

AH = 4Ch (sonlandır), AL = 00h (çıkış kodu).

Adım 21 / 22

DEĞİŞKENLER

AH = 4Ch

AL = 00h

DX = 0121h

ZF = 1

ÇIKTI

```
1:Selam q:Cikis 1
Selam!
1:Selam q:Cikis q
```

Kod İzleme: Mini Menü Programı

```
1 org 100h
2 goster:
3 mov dx, offset menu
4 mov ah, 09h
5 int 21h
6 mov ah, 01h      ; seçimi oku
7 int 21h
8 cmp al, 'q'
9 je cikis
10 cmp al, '1'
11 jne goster
12 mov dx, offset selam
13 mov ah, 09h
14 int 21h
15 jmp goster
16 cikis:
17 mov ax, 4C00h    ; çıkış kodu 0
18 int 21h
19 menu db 0Dh, 0Ah, '1:Selam q:Cikis $'
20 selam db 0Dh, 0Ah, 'Selam!$'
```

Program DOS'a döner; çıkış kodu toplu iş dosyalarında ERRORLEVEL ile okunur.

Adım 22 / 22

DEĞİŞKENLER

AH = 4Ch

AL = 00h

DX = 0121h

ZF = 1

ÇIKTI

```
1:Selam q:Cikis 1
Selam!
1:Selam q:Cikis q
```

Sık Yapılan Hatalar

'\$' unutuldu

INT 21h/09h belleği '\$' bulana kadar yazar: ekrana çöp dolar.

DS ayarlanmadı (EXE)

DX doğru ama kesim yanlış: `mov ax, @data /
mov ds, ax.`

AL'nin bozulması

02h ve 09h AL'yi değiştirir; okunan karakteri yazmadan önce başka yazmaçta saklayın.

09h sonrası imleç

INT 10h/09h imleci ilerletmez; sonraki yazı üstüne gelir.

13h'te DS:DX

BIOS dizgi hizmeti adresi **ES:BP**'de bekler.

0Ah tamponu

İlk bayt (kapasite) ayarlanmalı; metin tampon+2'den başlar, '\$' ile bitmez.

Yalnız LF

Yeni satır için CR + LF (0Dh, 0Ah) gerekir.

Sonlandırma yok

RET/4Ch unutulursa veriler komut olarak yürütülür.

ÖZET

Özet: Hızlı Başvuru

İş	Çağrı	Parametreler
Ekranı temizle	INT 10h / 00h	AL = 03h
İmleci konumla	INT 10h / 02h	BH = 0, DH = satır, DL = sütun
Renkli karakter	INT 10h / 09h	AL, BH, BL = öznitelik, CX = sayı
Karakter (daktilo)	INT 10h / 0Eh	AL
Renkli dizgi	INT 10h / 13h	AL = mod, BL, CX, DH:DL, ES:BP
Tuş bekle / yokla	INT 16h / 00h, 01h	→ AH = tarama, AL = ASCII; 01h: ZF
Karakter oku / yaz	INT 21h / 01h, 02h	→ AL / DL
Dizgi yaz / satır oku	INT 21h / 09h, 0Ah	DS:DX ('\$' / tampon)
Tarih / saat	INT 21h / 2Ah, 2Ch	→ CX DH DL AL / CH CL DH DL
RTC saati (BCD)	INT 1Ah / 02h	→ CH CL DH, CF = 1 hata
Piksel (mod 13h)	INT 10h / 0Ch	AL = renk, CX = x, DX = y; ya da A000:(y·320 + x)
Programı bitir	INT 21h / 4Ch, INT 20h, RET	AL = çıkış kodu

ÖZET

Alıştırmalar

1. Ekranın tam ortasına (12, 38) mavi zemin üzerine beyaz "8086" yazan programı INT 10h/13h ile yazın.
2. Kullanıcıdan iki rakam okuyup (01h) toplamalarını tek basamaklı olarak yazdırın (toplam ≤ 9).
3. Esc'ye basılana kadar basılan tuşların tarama kodunu onaltılık yazdıran bir program yazın.
4. 0Ah ile okunan metni tersten yazdırın.
5. Tarihi GG.AA.YYYY biçiminde yazdırın. Yıl için hangi yöntemi kullanırsınız?

► İpuçları / cevaplar

ÖZET

Alıştırmalar (devam)

6. `mov ah, 09h` / `int 21h` ile '\$' karakterini nasıl yazdırırsınız?
7. Mod 13h'te (160, 100) pikselinin A000h kesimindeki bağıl konumu nedir?
8. INT 1Ah/02h DH = 59h döndürdü. Ekrana ne yazılır? Aynı saniye INT 21h/2Ch ile hangi değerle gelir?
9. 0040:006C tık sayacıyla yaklaşık 5 saniye beklemek için kaç tık beklenmelidir?

► İpuçları / cevaplar

Kaynaklar

- Ralf Brown, *Ralf Brown's Interrupt List* (INT 10h, 16h, 21h ayrıntılı başvuru).
- IBM, *Personal Computer AT Technical Reference*, BIOS bölümü, 1984.
- Peter Abel, *IBM PC Assembly Language and Programming*, 5. baskı, Prentice Hall, 2001.
- Barry B. Brey, *The Intel Microprocessors*, 8. baskı, Pearson, 2009.
- emu8086 yardım belgeleri: *supported interrupt functions*.
- Ş. Ç. Öztürk, O. Ergin, *Intel 8086 ile Mikroişlemci Programlamaya Giriş*, 2009.

Son

Bölüm 11: Kesme Fonksiyonları

Sonraki bölüm: Bölüm 12: Vize Çalışma Soruları · [Terim sözlüğü](#)