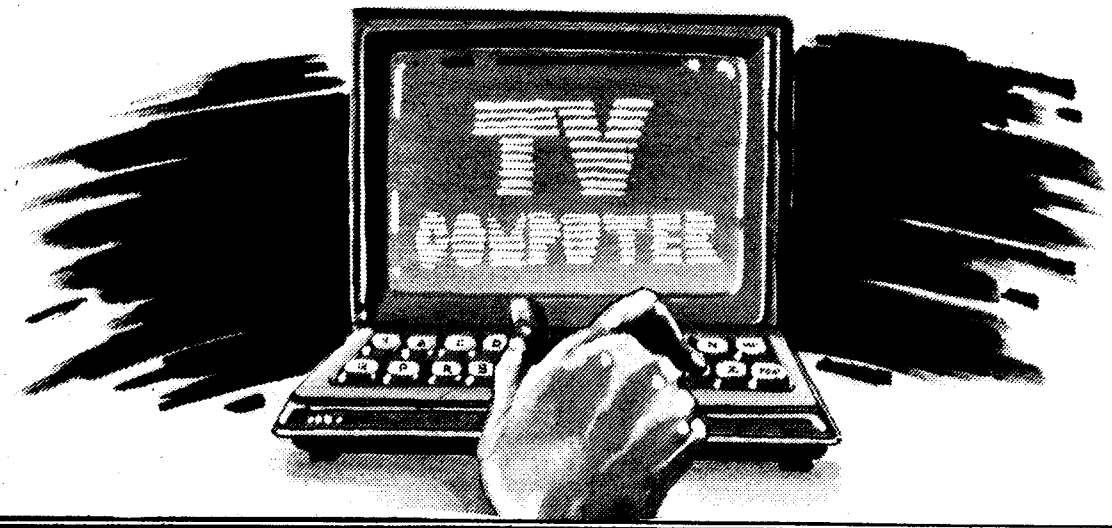



# ELSŐ KÉZBŐL

## A TV COMPUTERRŐL

## AZ EDITORRÓL

### AZ EDITOR RUTINJAI

**ED INT hívási kód: 32 vagy 160 (20h vagy A0h)**

**működés:** Az editor interrupt rutin villogtatja a kurzort, amíg az editor karakterek begépelésére vár. A felhasználó számára nem ajánlott a hívása!

**ED CHIN hívási kód: 161 (0A1h)**

**output:** C-karakterkód
A-hibakód

**működés:** Egy karakter beolvasása. Híváskor az editor belép a karakterbeolvasó-szerkesztő programhurokba: a leütött karaktereket beolvassa a billentyűzetről és kiírja a képernyőre, a szerkesztési funkciókat pedig végrehajtja. Háromféleképpen lehet a szerkesztőhurokból kilépni:

**ESC-vel:** C=27 (1Bh, ESC)
**CTRL+ESC-vel:** C=27 (1Bh, ESC)
A=245 (0F5h, STOP)
**RETURN-nel:** C=a bekezdés első karaktere

Ez a harmadik a normál kilépési mód. Ilyenkor a soronkövetkező ED-CHIN hívások a bekezdés további karaktereit kapják vissza C-ben. Ha a karakterek elfogytak, akkor C-ben 13-mal (0Dh, RETURN) tér vissza a rutin, majd egy újabb hívás hatására ismét belép a szerkesztő hurokba. A RETURN kódjának visszadásakor a kurzor a következő bekezdés, vagy üres sor elejére áll. A képernyő alján scrollozás történik.

**ED CHOUT hívási kód: 33 (21h)**

**input:** C=karakterkód

**működés:** Egy karatker kiírása. A 32-323 kódú (20h–0DFh) karaktereket kiírja a kurzor pozíciójába a képernyőre és a kurzort eggyel jobbra lépteti. A sor végéről a következő sor elejére kerül a kurzor. A képernyőn levő karaktereket felülírja mindaddig, amíg a kurzor egy bekezdés belsejében van. Ha a kiírás eléri a bekezdés utolsó sorának utolsó pozícióját, akkor egy üres sort hozzáfűz a bekezdéshez, és annak az elejére lép. Ha a soronkövetkező sor nem üres, akkor innen kezdve a sorokat eggyel lejjebb lépteti, és beszúrja az üres sort. A képernyő utolsó sorában természetesen scrollozás fog történni.
A szerkesztési funkcióval rendelkező kódok hatása olyan lesz, mintha azokat a billentyűzetről vittük volna be input során. Három kódnak van az inputtól eltérő jelentése:
10 (0Ah, LINE FEED): A kurzort a bekezdést követő első sorba viszi, az oszloppozíciót nem változtatja meg. A képernyő alján scrollozást hajt végre.
13 (0Dh, RETURN): A kurzort a képernyősor első pozíciójába viszi.
27 (1Bh, ESC): Itt hatástalan.
Az aktuális tintaszín és háttérszín nemcsak a kiírásnál, hanem a szerkesztési műveleteknél is érvényesül: a beszúrt üres sor pl. háttérszínű lesz.

**ED BKIN hívási kód: 162 (0A2h)**

**input:** BC=a beolvasandó karakterek száma
DE=pufferterület kezdőcíme

**működés:** Karaktercsoport beolvasása. Ez a funkció az ED–CHIN ismételt hívásával működik. Ha az elsőnek beolvasott bekezdés a záró RETURN-nel együtt kevesebb karaktert tartalmaz, mint a BC-ben adott érték, akkor újból belép a szerkesztő ciklusba, és további karakterekre vár. A bekezdést záró RETURN kódját is tárolja. A beolvasott karakterek a DE-ben megadott címtől kezdve találhatók meg a memóriában. Ha a szerkesztés során ESC-t nyomunk, akkor annak kódja, 27 (1Bh) egyből bekerül a pufferterületre. Ha az utolsó bekezdés több karaktert tartalmaz, mint amennyit be akarunk olvasni, akkor a következő ED–BKIN hívás először a félbehagyott bekezdést fogja beolvasni, és csak ennek végeztével lép a szerkesztési ciklusba.

**ED BKOUT hívási kód: 34 (22h)**

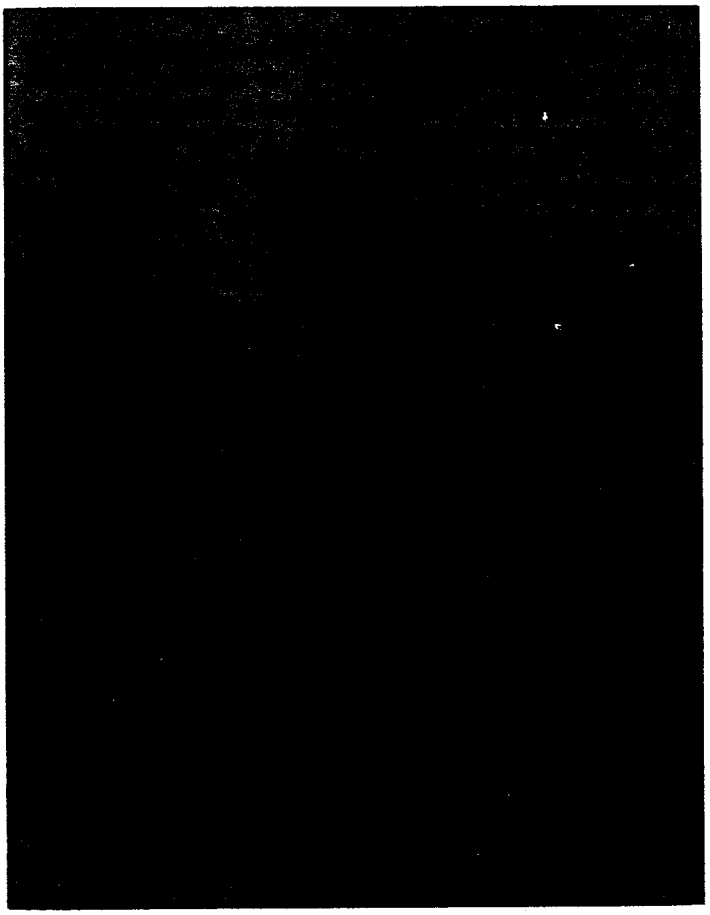
**input:** BC=kiírandó karakterek száma
DE=pufferterület kezdőcíme

**működés:** Karaktercsoport kiírása. Ez a funkció az ED–CHOUT ismételt hívásával kiírja a BC-ben megadott számú karaktert a DE-ben megadott memóriacímtől kezdve.

**CPOS hívási kód: 35 vagy 163 (23h vagy 0A3h)**
**input:** B=oszloppozíció (0,1–16,1–32,1–64)
C=képernyősor (0,1–24)
**output:** A=hibakód
**működés:** Pozícionálja a kurzort a B és C regiszterek értéke alapján. Ha a regiszterbe írt érték zérus, akkor a megfelelő pozíció változatlan marad. A maximális oszloppozíció a grafikus felbontástól függően 16, 32 vagy 64.

**CFIX hívási kód: 36 vagy 164 (24h vagy 0A4h)**
**működés:** Kurzorpozíció megjegyzése. Közvetlenül az ED–CHIN vagy ED–BKIN hívások előtt kell meghívni. Hatására az editor megjegyzi az aktuális kurzorpozíciót, és az input során ezt tekinti a paragrafus első beolvasható karakterének. Ezt használja az INPUT PROMPT utasítás is. Csak akkor érvényesül a hatása, ha az input szerkesztő ciklusában nem kerül a kurzor a paragrafusban a megjegyzett pozíció elé, emellett a képernyőn nem mozdult el a megjegyzett pozíció (pl. sortörlés vagy sorbeszúrás esetén is elmozdulhat az egész bekezdéssel együtt). Természetesen az input lezárásakor (RETURN) a kurzornak a megjelölt bekezdésben kell lennie. Csupán az első szerkesztési ciklusban veszi az input figyelembe, tehát ED–BKIN esetén a második paragrafus bekérésekor már nem. A szerkesztőciklusban leütött ESC is megszünteti a hatását.

# MEGHÍVÓ



## EGY APRÓ FOGÁS: PROGRAMSOROK EGYBESZERKESZTÉSE

Az editor ismertetésénél kiemeltük, hogy két bekezdést, például két kilistázott programsort, nem tudunk normál módon összefűzni. Normál módon nem, de egy trükkel igen! Ehhez a következőket kell tudni:
Az editor a munkaterületén nyilvántartja minden képernyősorról, hogy az hány karaktert tartalmaz. Erre 24 byte-ot használ. Sőt itt jelzi azt is, hogy a sornak van-e folytatása vagy sem. Ha a hosszbyte legfelső bitje 1-es, akkor a bekezdés a következő sorban folytatódik. Ezek alapján egyszerűen össze tudunk fűzni két bekezdést, csak a fenti táblázat kezdőcímét kell ismerni:
**LINE TAB 24 byte, címe 3664=0E50h**
Négyszínű módban a sorhossz 32 karakter, a táblázatba irandó érték: 32+128 (tele sor+folytatás):
**POKE 3663+SOR, 160**
A SOR változó helyére a kívánt sor számát (1–24) kell írni. Például BASIC programunk tartalmazza a következő két sort:
**100 Y=960–40*Y:X=32*X–32**
**110 PLOT ,X,Y;X+28,Y;X+28,Y+36;X,Y+36;X,Y**
Ebből akarunk egyetlen 100-as sort készíteni. Töröljük le a képernyőt és listázzuk ki ezt a két sort:
**CLS:LIST 100,110**
Most a képernyő első sorában van a 100-as sor. Az összefűzés
**POKE 3663+1,160**
Ezután a kurzort az első sor végére visszük, odaírunk egy kettőspontot (:), majd addig ejtjük ki innen a karaktereket (SHIFT+DEL), amíg a PLOT utasítás P-je a kurzor pozíciójába kerül. A sort RETURN-nel lezárjuk, és kitöröljük a feleslegessé vált 110-es sort:
**DELETE 110**
Végül javaslom mindenkinek kipróbálásra, hogy mi történik akkor, ha az eredeti sorhosszhoz adjuk hozzá a folytatást jelentő 128-at:
**POKE 3663+SOR,PEEK(3663+SOR)**

***Cseh Tibor***