Példák számok kiírására

A számok kiírása is alapvetően karakterek kiírásán alapul, azonban figyelembe kell venni, hogy a számjegyeket, mint karaktereket kell kiírni.

Decimális számok kiírása

Az alábbi eljárás decimális számokat ír ki. A kiírandó számot `AX`-ben várja.

Az eljárás működésének lényege, hogy a `k1`-es címkéjű ciklus a számot mindaddig osztja 10-zel és helyezi a verembe a maradékot, amíg a maradék nem nulla. Ekkor ugyanis a legnagyobb helyiértékű számjegyhez értünk. A `k1` címkével jelzett részlet első sora megvizsgálja, hogy `AX` értéke nulla-e, és ha igen, akkor két lehetőség áll fenn:

1. `CX` nulla volt (vagyis nem tettünk semmit korábban a verembe, ami azt jelenti, hogy `AX` eredetileg is 0 volt.

2. `CX` nem nulla, vagyis a verembe a számjegyek vannak. Ezesetben a `k2` címkénél folytatódik a program futása.

A `k2` címkénél kezdődő programrészlet sorra veszi ki a számjegyeket a veremből (legfelül van a legnagyobb helyi értékű), és kiírja, mint karaktert. A szám – mint karakter – kiírása úgy történik, hogy a szám alaki értékét hozzáadjuk a '0' karakter ASCII kódjához, és az így kapott karaktert írjuk ki. A ciklus mindaddig folytatódik, amíg a legalacsonyabb helyi értékű karakterhez nem érünk.

```
;eljaras, ami kiir egy szamot decimalis alakban
;a szamot ax regiszterben varja
szamkiir  proc
          mov bx, 10d       ;szamrendszer alapja
          mov cx, 0         ;verembe tett szamok szamlaloja

k1:       cmp ax, 0         ;addig osztunk amig nulla nem lesz
          je v1             ;ha nulla ugrik
          mov dx, 0         ;a kov. ut. dx:ax-et osztja!
          div bx            ;osztas bx
          push dx           ;maradek a verembe
          inc cx
          jmp k1
v1:       jcxz cxnulla      ;ugrik, ha nem tettunk semmit
                            ;a verembe

k2:       pop ax            ;kiveszunk egy erteket a verembol
          add al, '0'       ;hozzaadjuk a '0' ASCII kodjat
          call betukiir     ;kiirjuk a szamot (mint karaktert)
          loop k2           ;johet a kovetkezo
          jmp v2

cxnulla:  mov al, '0'       ;ha ax-ben 0 volt,
          call betukiir     ;irjunk ki egy 0-t
v2:       ret
szamkiir  endp
```

Hexadecimális számok kiírása

A hexadecimális számok kiíratásának elve is ugyanaz, mint a decimálisoké, azonban jelentős különbség az, hogy a decimális számoknál nem csak számjegyek, hanem betűk is előfordulnak.

```
tobbjegyu_hexat_kiir proc   ; tobbjegyu szamot ir ki,
                            ; amely AX-ben van

          push bx           ; verembe mentjuk az ertekeket
          push cx           ; verembe mentjuk az ertekeket
          push ax           ; verembe mentjuk az ertekeket

          cmp ax, 0         ; AX == 0 ?
          je nullat_ir      ; ha igen, akkor kiirjuk a 0-t

          ; AX helyett a szamot (DX:AX)-ben taroljuk
          xor dx, dx        ; kinullazzuk DX-et
          mov cx, 0         ; CX = 0
          mov bx, 16        ; BX = 16,
                            ; hogy (DX:AX) erteket leoszthassuk
                            10-zel
vizsgal:  cmp ax, 0         ; BX == 0?
          je szam_vege      ; ha igen, akkor az elemzés
                            ; vegere ertunk
          div bx            ; DX-ben van a maradek,
                            ; AX-ben van a hanyados az osztas utan
          push dx           ; DX-ben van a szam utolso szamjegye,
                            ; ezt berakjuk a verembe
          xor dx, dx        ; kinullazzuk DX-et
          inc cx            ; CX azt szamolja,
                            ; hogy hany jegyu volt a szam
          jmp vizsgal

nullat_ir:
          mov bx, 0
          call hexat_kiir
          jmp
          elemzes_vege
```

```
; ez a programresz akkor hivodik meg,
; ha minden szamjegyet feldolgoztunk
szam_vege: pop bx           ; a verembol kiszedunk egy szamjegyet
           call hexat_kiir  ; ezt kiirjuk
           dec cx           ; egy szamjeggyel kevesebbet kell kiirni
           cmp cx, 0        ; ha nincs tobb szamjegy,
                            ; akkor vege a szamnak
           jne szam_vege

elemzes_vege:
           pop ax           ; AX eredeti erteke
           pop cx           ; CX eredeti erteke
           pop bx           ; BX eredeti erteke
           ret
tobbjegyu_hexat_kiir endp


hexat_kiir proc             ; kiirja BL-ben levo szamjegyet
                            ; egy szamjegyet ugy irunk ki, hogy
           cmp bl, 9        ; ha a szamjegy
           jle szamjegy     ; kisebb-egyenlo 9, akkor
                            ; ugras "szamjegy"-re
betu:                       ; kulonben betut ir ki
           mov al, bl       ; AL-be atmasolom a szamot
           sub al, 10       ; kivonok belole 10-et
           add al, 'A'      ; es ehhez hozzaadom az
                            ; 'A' karakter kodjat
           mov ah, 14       ; BIOS parameterezes
           int 010H         ; kiiratas
           jmp hexa_vege

szamjegy:
           mov al, '0'      ; a 0 karakterkodjahoz
                            ; hozzaadjuk a szamjegyet
           add al, bl
           mov ah, 14
           int 010H
hexa_vege: ret
hexat_kiir endp
```

Számsorozat legnagyobb eleme

Keressük meg egy bájtokból álló számsorozat legnagyobb elemét! A számsorozat hosszát a `len` változó tartalmazza. Az eredményt taroljuk a `max` változóban. Az adatszegmens pl. a következő is lehet:

```
adat    segment
        len dw 5
        max db 0
        sor db 4,5,2,7,3
adat    ends
```

```
        MOV AX, ADAT          ; elokeszites
        MOV DS, AX
        MOV ES, AX
        MOV CX, len           ; a szamsorozat hossza CX-ben van
        MOV SI, offset sor    ; az SI-t a szamsor elejere allitjuk
        MOV DI, offset max    ; ide kell irni a maximalis elemet

        XOR DX, DX            ; DL-ben lesz majd a maximum
ciklus: LODSB                 ; egy bajtot toltunk be a SI-rol
        CMP DL, AL            ; nagyobb-e a beolvasott szam
                              ; a maximumnal?
        JGE tovabb            ; nem valtozott a max
        XCHG AL,DL            ; uj max ertek
tovabb: LOOP ciklus           ; ha CX ≠ 0, akkor ugras a cilusra
        MOV AL, DL            ; a max-ot AL-be rakjuk
        STOSB                 ; AL erteket az adatszegmensbe
                              ; taroljuk el
```

Feladatok

1. Módosítsuk a maximum kereső programot hogy szavakból (dupla bájt) álló számsorozatra is működjön.

2. Módosítsuk a maximum kereső programot minimum kereső programra! Azaz keressük meg a sorozat legkisebb elemét. Figyeljünk a megfelelő kezdőértékekre!

3. Egészítsük ki a maximum kereső programot, hogy a legnagyobb szám első/utolsó előfordulási helyét a "pos" változóban tároljuk!