Nagyné Lakatos Eszter - Valentné Albert Éva

Adatbázis-kezelés FoxProban szoftverüzemeltetőknek

Nagyné Lakatos Eszter - Valentné Albert Éva

Adatbázis-kezelés FoxProban szoftverüzemeltetőknek

Bírálta:

Papp Gáborné egyetemi tanár
Nedeczky Veronika egyetemi tanár
Kincses Zoltán Ph. D.

Lektorálta:

Csernyi László mérnök-tanár

Külön írt fejezetek:

3. fejezet : Nagyné Lakatos Eszter

4., 5., 6., 7., 8. fejezet: Valentné Albert Éva

Nyomás és kötés:
Tótfalusi Tannyomda
1998
Digitalizáláshoz minimálisan átdolgozva (2012)

1. Tartalomjegyzék

2. Előszó

Könyvünk témája az **adatbázis-kezelés témaköre FoxPro programon** keresztül bemutatva. A jegyzetet az iskolánkba járó leendő 4. osztályos diákoknak készítettük, hogy majdan sikeresen válaszolhassanak **szoftverüzemeltetői** vizsgájukon az adatbázis-kezelésre vonatkozó kérdésekre is. A dolgozat alapfokú **felhasználói** ismereteket feltételez (szövegszerkesztés, táblázatkezelés, operációs rendszer) és programozási ismereteket. Mivel a szoftverüzemeltetői írásbeli vizsga kérdéseit a Feladatbank I-II. kötetéből válogatják, ezért számos feladatot átvettünk, és gyakorlásként ajánlunk oldal és feladatszám hivatkozással. Az említett feladatok alkalmasak gyakorlásra, viszont számos feladat olyan ismereteket kér számon, melyet célszerűbb volna gyakorlatban kérdezni, írásbeli helyett. Azonban iskolánk már erre a képzési formára állt rá, így a könyv ehhez kíván segítséget nyújtani. A számítástörténeti fejezetet érdekességnek, figyelemfelkeltésnek szántuk a teljesség igénye nélkül.
A Windows programban használt kezelőszervek elnevezéseit Fürtön Zsolt-Nagy Zsolt iskolánkban használt jegyzetéből vettük át, melyet a 9.2. számú melléklet tartalmaz (89. oldal).

A dolgozatban használt jelölések:

Aláhúzott, félkövér	Új fogalom
Félkövér	Definíció
Aláhúzott	Menüpontok, és egyéb kezelőszervek
`Courier`	Parancsok, file és könyvtárnevek

Jó tanulást kívánunk!

A szerzők

Köszönöm témavezetőmnek: Kincses Zoltánnak, Papp Gábornénak a dolgozat bírálójának, és a Szabóky Adolf Műszaki Szakközépiskola 1998. év 2/I-1. ill. 2/I-2. osztályába járó tanulóinak és természetesen két kedves kollégámnak: Gróf Gabriellának és Csernyi Lászlónak hasznos segítségüket. Nem hagyhatom ki Édesapámat sem, aki már második szakdolgozatom megírásában nyújtott nélkülözhetetlen segítséget.

Valentné Albert Éva

3. A számolás technikájától a számítástechnikáig

A számolás és a számítástechnika története immár olyan hatalmas területet ölel fel, hogy ilyen csekély terjedelemben csak tallózni lehet az érdekességek között. Ezért a fejezet alcíme: "Szemelvények a számítástechnika történetéből" lehetne.

A számítástechnika története nem választható el a számolási technikák fejlődésétől. A számfogalom fejlődésétől eltekintve is minden korban törekedtek a számolási eljárások egyszerű, mechanikus elvégzésére, ezért úgy érzem, ezzel is érdemes kicsit foglalkozni.

3.1. A számolás rövid története

A pitagoraszi szemlélet szerint : a dolgok természete, lényege: a szám.
- Ez természetesen évezredek folyamán alakulhatott ki, hiszen kezdetben az "egy", a "kettő" és a "sok" létezett. A nagyobb számok fogalma csak sokkal később fejlődött ki.

A kézművesség és kereskedelem elterjedése során először 5-ös, majd 10-es alapú számrendszerben számoltak.

A *Rhind Papiruszokból* és a *moszkvai papiruszból* már jelentős ismereteink vannak az egyiptomi matematikáról. Az is 10-es számrendszeren alapult, ugyanolyan elven, mint a közismert római számok rendszere.

Az egyiptomiak egy túlnyomóan additív aritmetikát alakítottak ki, ami abból állt, hogy minden szorzást többszörös összeadásra igyekeztek változtatni.

A késői sumér korszakból kitűnő szorzótáblákat találunk, amelyekben az eredeti 10-es számrendszeren kívül egy fejlett 60-as számrendszer épült fel.

A hinduk egyedülálló érdeme a tízes számrendszer, a helyi érték fogalma és a nullának mint számjegynek a bevezetése. A helyi érték-rendszert valószínűleg már ők is Alexandrián keresztül a babilóniai matematikából vették át, de ők voltak az elsők, akik a ma is használt számjegyeket bevezették.

Ezeket a számjegyeket tévesen arab számoknak nevezzük, szemben az addig használt és még sokáig használatban maradt római számokkal. Ennek az az oka, hogy az arab hódítások idején *Al-Khwarizmi* IX. századi arab matematikus egyik művén keresztül jutott el Egyiptomba ill. Európába.

Maga az algebra szó is *Al-Khwarizmi*[1] (latinosan elferdítve Algoritmi[2]) egyik művének címéből származik: "Hiszab al-dzsebr w'al mukabalah". A címben szereplő al-dzsebr helyreállítás, kiegészítés, helyrerakás értelmű szó, ami az egyenletrendezés ma is érvényes szabályait jelenti. (A szó eredetileg az eltört csontok helyre rakását is jelentette; a spanyoloknál az "algebrista" ma is használatos csontkovács értelemben.)

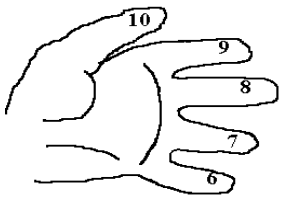
Az arab számokkal és azok használatával *Fibonacci* 1202-ben megjelent "Liber abaci" című munkáján keresztül ismerkedett meg Európa. A könyv érdekessége, hogy a címe arra utal, hogy az abakusszal végezhető számításokat tárgyalja. Ezzel szemben, tulajdonképpen ellen-propaganda éppen az arab számokkal való számolás érdekében.
Az arab számokkal való számolás minden gyakorlati előnye ellenére sem terjedt el gyorsan. (Pedig gondoljuk meg, milyen nehéz lehetett kiszámolni római számokkal a XII és LIV szorzat értékét!)
Éppen az üzletemberek, akiknek a munkáját jelentősen megkönnyítette volna, tiltakoztak a legjobban ellene. Firenze 1299-ben egyenesen megtiltotta használatukat, arra hivatkozva, hogy egy nulla beszúrásával az üzleti könyvek könnyen meghamisíthatók, hiszen az arab számok értéke 10-szeresre nő.

Az arab matematika és fizika a XV. században érte el csúcspontját. Ezt követően Európa szerte rohamosan fejlődésnek indultak a csillagászat, a hajózás és a fizika egyes ágai. A megkövetelt gyorsabb és nagyobb tömegű számítások igényei miatt mind több ötlettel álltak elő a tudósok.

Fourier 1807-ben kidolgozott egy "keresztbe szorzást", amely a részletszorzatok leírása nélkül közvetlenül a végeredményt adta meg. A német *Ferror*nak és a magyar *Frank Móric*nak hasonló közvetlen módszere volt.

Az egyszerűsített - csak az egyszeregy ismeretét igénylő - mozgó papírszeletes szorzást 1938-ban *Irhás János* szabadalmaztatta. Ennél az egyenlő helyi értékű számjegyek egymás alá kerülnek, amik azonnal az eredményt adják.

A műveletek, például a szorzás elvégzésére több módszer vált ismertté, példaként egy igen egyszerű: **szorzás két kézzel.** A módszer a 6-tól 10-ig terjedő számokra érvényes. A század elején használták helyenként az iskolázatlan emberek, főleg az angolszász területeken volt ismert.



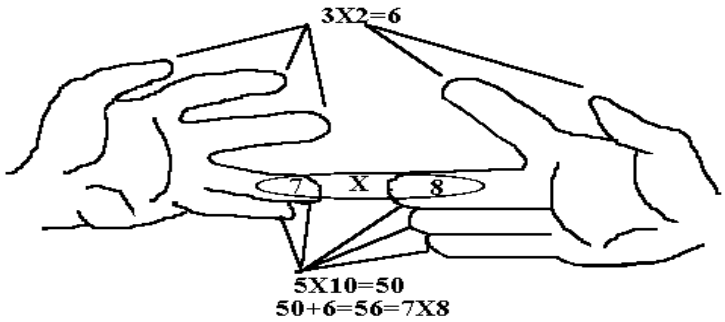
[1] Feltételezhetően az arab eredet miatt az egyes szakirodalmak átírásában neve máshogy jelenik meg, én Simonyi Károly átírását használtam.

[2] Innen ered az algoritmus szó.

Az ábrán látható módon 6-tól 10-ig megszámozzuk ujjainkat: kisujj - 6,
gyűrűsujj - 7,
középsőujj - 8,
mutatóujj - 9,
hüvelykujj -10,

A kéz ujjainak megszámozása

Ha képezni akarod pl. a 7x8 szorzatot, az egyik kéz 7-es ujját érintsd a másik kéz 8-as ujjához. A két egymáshoz érő és az alatta elhelyezkedő ujjak értéke egyenként 10. Mivel most 5 ilyen ujjunk van, ennek az értéke 50.



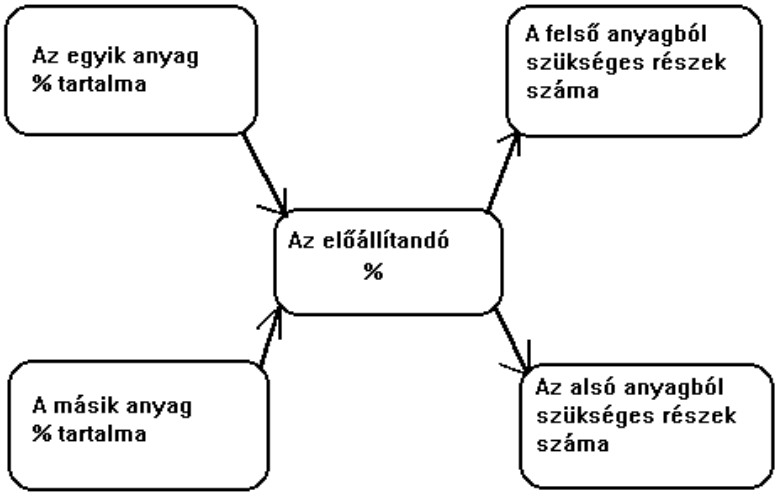
Szorzás két kézzel

Ezután szorozd össze a bal és a jobb kéz fent maradó ujjainak a számát: 3x2=6. A két szám összege: 50+6 = 56. Tehát 7x8= 56. - A módszer mindig helyes eredményt ad.

Egy másik érdekesség: a **csillagszabály**, mely a keverési feladatok megoldásában nyújt segítséget.
(A százalékszámítást általában képletek alkalmazásával oldják meg. A legtöbb iskolában így is tanítják. Azonban azok, akik így tanulták, rendszerint elfelejtik a képleteket, és végeredményben rosszul számolnak. Ezen a gondon segít a szabály alkalmazása.)

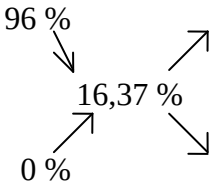
Példa: *2000 literes textilgyári kádat kell megtölteni 16,37 % lúgtartalmú oldattal. Mennyi 96 %-os lúgoldatot és mennyi vizet kell beletennünk?*

A csillagszabály nevét onnan kapta, hogy az adatokat a számolásnál csillag-alakban írjuk, így:
(A nyilak különbségképzési irányokat jelentenek.)

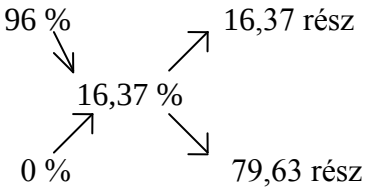


A csillagszabály

Mivel 16,37 % lúgtartalmú oldatra van szükségünk, a példát így írjuk fel:



Ezen felírás után a nyilak irányában megállapítjuk a két-két szám különbségét.
0 és 16,37 között a különbség 16,37; 96 és 16,37 között (96-16,37=) 79,63.
Ezt a két számot beírva a megfelelő helyre:



a feladat nehezebb részét már megoldottuk, mert tudjuk, hogy a 96 %-os oldatból veszünk 16,37 részt, vízből pedig 79,63 részt. A két rész együtt (16,37+79,63=) 96 rész, ez 2000 liter, tehát egy rész (2000/96=) 20,83 liternek felel meg. Ezért a lúgból (16,37*20,83=) 340,9871, kerekítve 341 liter kell, vízből pedig (2000-341=) 1659 liter.

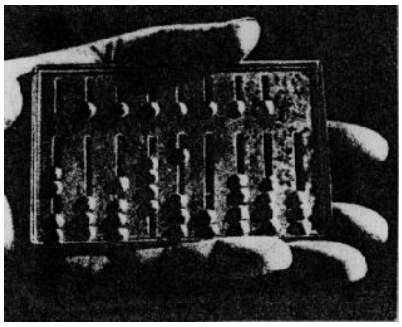
3.2. A számítógép rövid története

A számítógép a számológépekből alakult ki. A számológépeket az emberi találékonyság és lustaság szülte. Megalkotásuknál elsődleges cél volt, hogy minél egyszerűbben, a lehető leggyorsabban végezhessünk el valamilyen unalmas, hosszantartó vagy fáradságos számítási feladatot.

Az első számolási segédeszköz, "számológép" az ember 10 ujja volt, hiszen ez mindig "kéznél volt". Az ujj latinul: digitus, és innen származik a számítógépek működésével kapcsolatban napjainkban oly sokat hallott digitális szó.

A "későbbi számítógép" az ősember barlangjának bejárata előtt egy korsó és egy marék kavics formájában volt megtalálható. Reggel, amikor az ősember kiterelte a barlangból a juhokat, minden állatnál bedobott a korsóba egy kavicsot (adatbevitel, input). Napközben a korsó az állatoknak megfelelő számú kavicsot tartalmazott (adattárolás, háttértár).Este minden belépő állatnál kivett egy kavicsot (adatkivitel, output), s az eredménytől függően az adat-feldolgozás is bekövetkezett: ha kavics maradt a korsóban, néhány állatot elcsábítottak a szomszéd barlang lakói (készíteni kell a bunkót), míg ha a kavics elfogyott, de még mindig jöttek az állatok, úgy önkéntesek csatlakoztak a nyájhoz...

Az első számológép a már említett **abakusz** volt, mely a görög kultúrában jelent meg. (A szó eredetileg az oszlopok fedőlapját jelentette.) Az elnevezés onnan származik, hogy négyszögletes lapokkal könnyítették meg a számításokat. A lapokon párhuzamos vonalak jelölték az egyes "számjegyeket": jobbról balra I,V,X,L,C, stb. (egyesek, ötösök, tízesek, ötvenesek, százasok, stb.). Minden vonalra annyi kavicsot (latinul: calculus → kalkulátor) helyeztek, amennyi az egyik számnak megfelelt. Ezután úgy tologatták el balra az egyes kavicsokat, hogy a végeredményt a kavicsok helyzetéből le tudják olvasni. - Később fémkeretben elhelyezett üveggyöngyöket használtak. - Hasonló szerkezet a kínai **szuanpan**, az orosz **szcsoti** vagy a japán **szorobán**, amely újra virágkorát éli egy sor magyar iskolában is.



Forrás: Simonyi Károly: *A fizika kultúrtörténete* Gondolat, 1981.

Az abakusz

- Az ilyen számológép kezelése nem volt egyszerű, de gyorsabban lehetett számolni az eszköz használatának ismeretével.

A XVI. században sok matematikus kutatta, miként célszerű egy számtani és egy mértani sorozat elemeit egymáshoz rendelni azért, hogy segítségükkel a számolás könnyebbé váljék. A gondolat különben már *Arkhimédész*nél is megtalálható az a^0,a^1,a^2,a^3, ... sorozat felírása után az $a^k\ a^n=a^{k+n}$ azonosság megállapításában. Itt a kitevők számtani sorozatához rendelte hozzá az *a* alapú hatványok mértani sorozatát. Ha most alkalmasan választott *a* alapok hatványaival elég sűrűn lehetne előállítani a számokat, akkor például a számok **szorzását** a hozzájuk rendelt hatványkitevők **összeadásával** lehetne elvégezni. A módszer gyakorlatba való átviteléhez táblázatos előállítás szükséges.

Az első ilyen táblázatot a holland *Stevin* készítette kamatoskamat-számítás céljaira (1610 körül). Az ő táblázatát fejlesztette tovább *Bürgi*, svájci származású órásmester. Nyolc évig dolgozott a táblázatán, de kiadásával késlekedett (1620), így a publikálásban megelőzte (1614) a skót *Napier*. Táblázatának kezelése nehézkes volt, ezért barátjával megtervezték a

tízes alapú logaritmustáblázat módszerét, melyet *Briggs* ki is dolgozott és 1617-ben meg is jelentetett. - Ezt a későbbiekben tovább tökéletesítették, de közben 1620-ban egy angol professzor, *Edmond Gunter* megszerkesztette az első **logarléc**et, amely igen komoly segítséget jelentett a számítások elvégzésében, egészen az elektronikus számológépek megjelenéséig.

Az eszköz használata gyors, üzemi számításokra kiválóan megfelelt, de pontossága miatt a tudományos életben való alkalmazásra kevésbé volt alkalmas.

A középkorban a mohamedán csillagászok értek el kiemelkedő eredményeket. *Gijjád el-Din al-Kassi* (1393-1449) asztrológiai szerkezetei megdöbbentően pontosak voltak. "Konjukciós" lemezével meg lehetett állapítani, mikor lesz két bolygó egyazon hosszúsági fok mentén. Készített továbbá holdfogyatkozást kiszámító gépet, majd egy olyan planetáris számológépet, amellyel a Nap, a Hold és a látható bolygók földrajzi hosszúságát lehetett megállapítani.
"Igazi" számológépet először a német *Wilhelm Schickard* készített 1623-ban, barátjának, *Kepler*nek. Ez mechanikai szerkezetű gép volt, működése tízfogú és egyfogú fogaskerekek kapcsolódásán alapult. Volt összeadó, szorzó és részeredményt kiíró egysége.

Ugyancsak ez idő tájt készített hasonló elvű számológépet *Bürgi*.

Újabb lépcsőfokot jelentett *Blaise Pascal* számológépe 1641-ben, amely az alapvető aritmetikai műveleteket el tudta végezni, valamennyit a sorozatos pozitív vagy negatív előjelű összeadásra visszavezetve. Később (1674) *Leibniz* csak továbbfejlesztette, tökéletesítette Pascal ötletét (Leibniz kerék).

Nagy haladást jelentett a mechanikus számológépek fejlődésében *Odner* pétervári mérnök beállító szerkezete 1874-ben. Az általa alkalmazott tüskés kerekes, léptetett kocsis, hajtókaros számológéptípust még száz évig gyártották.

Digitális, azaz számjegyeken alapuló számológép készítésére csak a XIX.században tett kísérletet *Charles Babbage*."Differenciagépe" matematikai táblázatok kiszámítására és az eredmények kinyomtatására született meg.

Sokkal fejlettebb lett volna az általa tervezett Analitikus Gép. Ez már igazi számítógép! Tárral volt ellátva, (ez ma alapkövetelmény!), aritmetikai egységgel és be-/kimeneti egységekkel rendelkezett volna. Terv maradt, a kor műszaki megoldási lehetőségei nem voltak összhangban elképzeléseivel. (Később fia - apja iránti kegyeletből - a gép egy részét megvalósította, kitűnően működött.) Babbage 1871-es halála után találmánya 70 évre feledésbe merült.

Munkásságának legjobb összefoglalása *Ada Byron*tól, Lord Byron leányától maradt ránk, aki már 15 éves korában kitűnt rendkívüli matematikai érzékével. Babbage elgondolásait tökéletesen megértette, s joggal tekinthető az első női számítástechnikusnak. Róla nevezték el a nagyszámítógépekre a Pentagon megbízásából készített ADA programnyelvet.

Időközben egyre nyomasztóbb igény merült fel az egyhangú és fárasztó tevékenységek gépesítése (irányítástechnika), az egyre növekvő információmennyiség kezelése (információtechnika) és a nagy tömegű számítások elvégzése (számítástechnika) terén. E

három feladatkört folytatja és egyesíti is a mai számítástechnika. (Ráadásul a számítógép nem sztrájkol, nem kér állandóan fizetésemelést, nem beteg, sem ő, sem a gyermeke és nem jár sem esküvőre, sem a nagyszülők temetésére. Csak dolgozik!)

A hetven év elteltével felbukkan egy ma is jól csengő mozaikszó. *Jacqard* 1801-ben, a szövőszék vezérléséhez használt lyukkártyáját Babbage ötletével ötvözte az *International Business Machine Co.* (**IBM**) és *Howard Aiken* a Harward Egyetem professzora (1939). Gépük (a "**MARK I.**") 1959-ig működött, és egyes források szerint katonai kódok megfejtése volt a fejlesztés eredeti célja. - Később áttértek a lyukszalag alkalmazására.

A második világháború másik, viharos gyorsasággal kifejlesztett számítógépe az **ENIAC** volt. (1945. végére készült el az USA-ban). A név az "elektronikus numerikus integrátor és számítógép" elnevezés angol nyelvű megfelelőjének betűszava. Ez a rendkívül gyorsnak tervezett, főként elektroncsövet tartalmazó gép 1/5000 secundum alatt végzett el egy összeadást. Működtetéséhez egy kisebb erőmű kellett, a keletkezett veszteséghővel akár egy hatalmas lakótelepi házat is közmegelégedésre lehetett volna fűteni. Még a kézi telefonközpontokhoz hasonló dugaszolással működött.

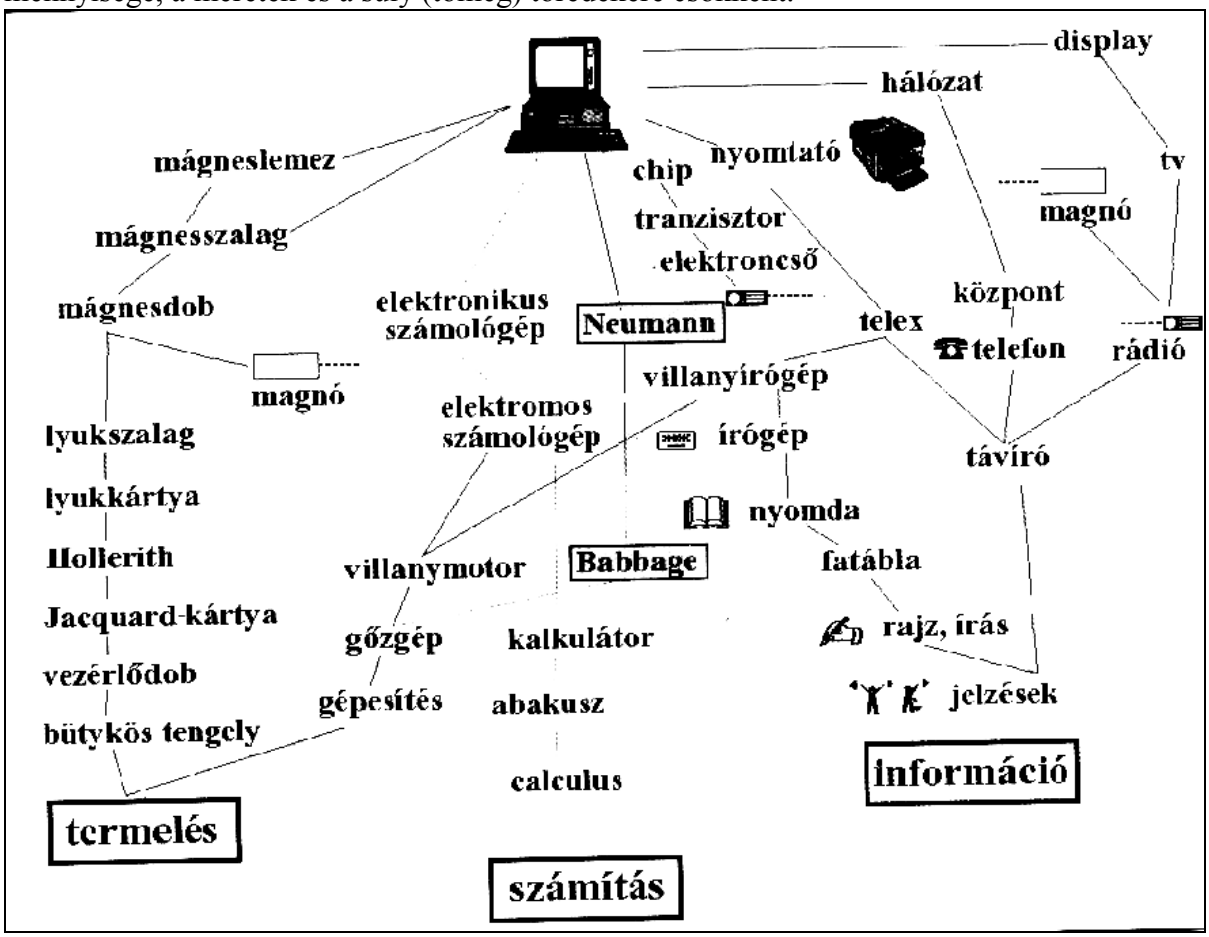
Időközben Európában *Konrad Zuse* német mérnök építette meg az első jelfogós számítógépet, mely a kettes számrendszert alkalmazta, sebessége 30-50 művelet/perc volt. (Ha Zuse megkapja a kormányától azt a támogatást, mint az USA, akkor előbb lehetett volna a németeknek számítógépük, mint az amerikaiaknak!)

A számítógépek további fejlesztéséhez *Neumann János* fogalmazott meg alapvető szempontokat: a gépben a programot és az adatokat is (ugyanolyan formában) tárolni kell, a kettes számrendszer és a belső programozás a célravezető. Az *A.W.Burks* és *H.Goldstine* társaságában 1948-ban megfogalmazott elv alapján készített számítógépeket (az első : az **EDVAC**) azóta is Neumann-gépeknek szokás titulálni. Az EDVAC (elektronikus, diszkrét változós, automatikus számítógép angol rövidítése) másodpercenként 1500 szorzást vagy 15 000 összeadást végzett.

A kezdetekben még sok elsőségre törő próbálkozás történt.
Az USA National Physical Laboratory az **ACE** (ész, ász,csúcs), a Cambridgei Egyetem az ***EDSAC***-ot fejlesztette ki (1946. - Neumann volt munkatársának, *Maurice Vincent Wilkes*nek az irányításával.). Ennek továbbfejlesztett változata a **LEO** (Lyons Electric Office) az ötvenes évektől (1953) egészen 1965-ig működött.- 1951-ben elkészült a Szovjetunió első számítógépe, a MESZM (Malaja Elektronnaja Szcsotnaja Masina = kis elektronikus számológép) *Szergej Alekszejevics Lebegyev* vezetésével. 7-8000 művelet/s sebességgel működött.

De közben Magyarországon elszabadult a pokol! Amikor mi, magyarok nem a világra figyeltünk, NOBEL-díjat kapott három fizikus: *William Bradford Shockley , John Bardeen és Walter Houser Brattain,* amerikai fizikusok 1956-ban a tranzisztor felfedezéséért része-sültek minden kutató elme legnagyobb díjában. (Az első tűs tranzisztort már 1948-ban elkészítették,

de csak ekkor kapták a díjat érte.) Innentől kezdve elmaradt a fűtés, az energiaigény roppant mennyisége, a méretek és a súly (tömeg) töredékére csökkent.



A termelés, az információkezelés és a számítás gépesítésével több ezer éven keresztül kísérleteztek. Minden korszakban volt olyan feltaláló, aki fejlesztett valamit mind a technikán, mind a módszeren. A mai számítógép ezeknek a fejlesztéseknek az eredménye
Forrás: Szűcs Ervin: *Komputervilág* Móra Könyvkiadó, 1995

Megvalósult az űrkutatás lehetősége. A két tudomány szimbiózisa lehetővé tette, hogy az egyre csökkenő tömegű és méretű számítógép felkerülhessen az űrbe, az ottani oxigén- és szennyezésmentes környezetben mesterségesen növesztett, ideális szilíciumkristályok pedig a földön, az IC-technikában lehetőséget adjanak az előállításra, és megvalósulhasson a ma már CHIP néven ismert, méretre parányi, integráltságban óriási elemek kifejlesztése.

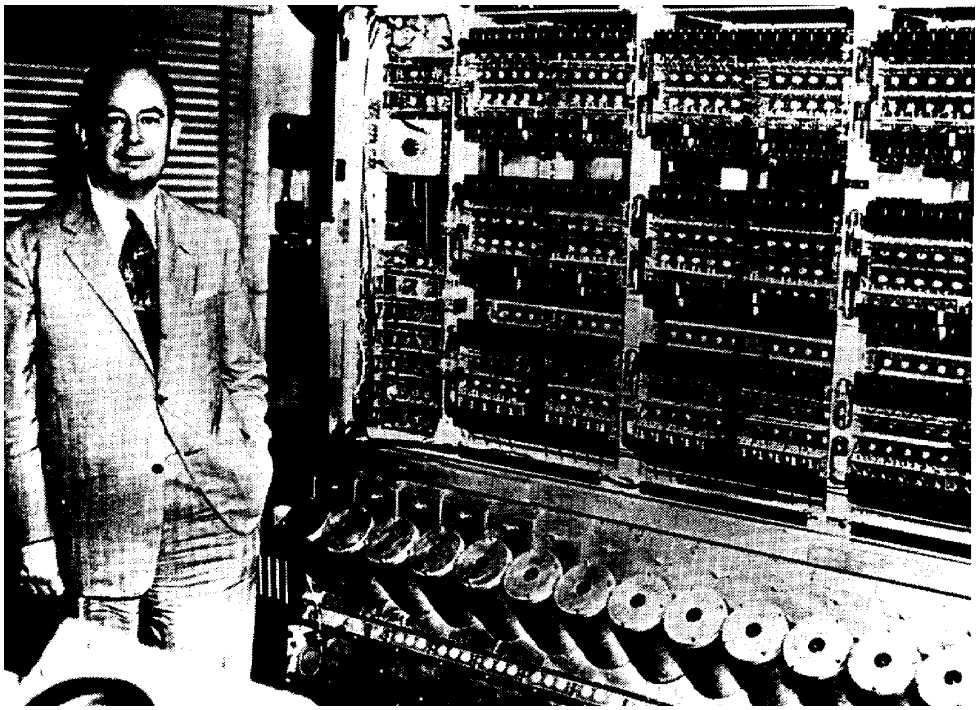
A tranzisztor felfedezésétől kezdve a számítástechnikai ipar rövid 30 év alatt a világ-ipar egyik legjelentősebb tényezőjévé vált. A miniatür méretek lehetővé tették, hogy megjelentek a kézi elektronikus kalkulátorok és rohamosan elterjedtek. (Ma már szinte minden általános iskolás diák zsebében is megtalálhatók.)

A fejlődés üteme napjainkban sem lassult le. Az áramköri elemek nagyfokú integráltsága, valamint az előállítás egyszerűsödése mindinkább lehetővé teszik, hogy a számítógép mindennapi életünk részévé váljon, bekerüljön az otthonokba, sőt az oktatásba is. Ezt a felgyorsuló folyamatot egyesek *második ipari forradalomnak* nevezik. Bizonyos, hogy a számítógépipar fejlődése előbb-utóbb gyökeres változásokat fog előidézni az emberi társadalomban is.

Relék, elektroncső, tranzisztor, integrált áramkör, chip ... Eljutottunk a számítógépek ötödik generációjához. És a következő?
A jövőben számítani lehet a fényt hasznosító (fotonikai) illetve a biológiai elven működő gépekre. (a *Biochip*-pel már fejlett kísérleteket folytatnak, elsősorban Japánban -lószív besugárzása lézerrel).

Akárhányadik generációról is van szó, Neumann János akkor sem mondott ostobaságot, amikor kijelentette: "a komputer olyan, mint egy hülyére vert fegyenc; mindent megcsinál, amit mondanak neki, de csak azt és semmi mást."

A gép intelligenciájától, attól, hogy a "fejünkre nő", kár félni. Önálló tudása nincs, amit tud, tőlünk kapta.



Neumann János a modern számítógép egyik ősével

Forrás: Simonovits Miklós: *Számítástechnika (a speciális matematikai osztályok részére)* Tankönyvkiadó, 1985.

3.3. A számítógépek elterjedése Magyarországon

Nem kizárólag a transzformátor, a cipzár, a golyóstoll, a C-vitamin, a konyhai fregoli, a karburátor; Neumann János, Kemény János, ... hirdetik a magyarok rátermettségét az élet legkülönbözőbb területein. "Névtelen" alkotók is sokat tettek a maguk szakterületein.

Kezdetben az USA-ban úgy vélték a szakértők, hogy az ezredfordulóra mintegy tucatnyi nagyteljesítményű számítógép ki fogja szolgálni az Államok igényeit. Valószínűleg ebből kiindulva a kis országok is bátorságot vettek maguknak, és nekiláttak a saját gyártmányú számítógépek kifejlesztésének.

A műszaki fejlődés és a rohamosan csökkenő árak felborították az előzetes számításokat, így az óriási sorozatban gyártott és ma már mindenütt megtalálható személyi számítógépek határozták meg a gazdaságos gyártás helyszíneit. [3]

Magyarországon a kezdetet az MTA Kibernetikai Kutató Intézetben az 1950-es évek végén épített, első generációs, M-3 jelű, szovjet tervezésű gép jelentette. Az első hazai számítógép a MESZ-1 (1955), mely még "jelfogós" rendszerű volt. Tervezése és megépítése Kozma László (1902-1983) nevéhez fűződik, akinek munkássága a telefonközpontok területén is rendkívül jelentős. Szintén magyar tervezésű gép az 1960-as évek végén született második generációs EMG-830.

A kutatás és fejlesztés több helyen (KFKI, MTA, Számítástechnikai Koordinációs Intézet, stb.), a gyártás pedig az EMG-ben, a KFKI-ben és a VIDEOTONban indult meg, igazodva a volt KGST országok Egységes Számítástechnikai Rendszeréhez (ESZR).

A teljesség igénye nélkül néhány magyar termék: a Tárolt Programú Adatfeldolgozó (TPA), utódja a TPA-I, ahol az "I" arra utal, hogy már integrált áramköröket is tartalmaz a központi egység, a TPA 70/s (ami már egy ún. hardware stackot, azaz push-down memóriát tartalmazott), az R-10-es VIDEOTON kisgép, ami ESZR kompatibilis volt és másodpercenként 500 ezer művelet elvégzésére volt képes. A sorba tartozik a csehszlovák kutatókkal együtt kifejlesztett analóg-digitális hibrid MEDA-80/I. Ide illik a VILATI "MINICOMP"-ja is, bár belső tárral nem, csak cserélhető mágneslemezzel rendelkezett.

Ezenkívül készültek még különböző "célszámítógépek" is : pl.a Műszeripari Kutató Intézet "jelfeldolgozó célberendezése", ami EEG és EKG orvosi adatok ellenőrzésére, spektrométer - adatok mintavételezésére és egyéb, bonyolultabb feladatok elvégzésére volt alkalmas.
Természetesen a fenti sort egy seregnyi hazai gyártású periféria egészítette ki: PERFOMON-30, EP-35, PREPAMAT, READMOM, DATAVISOR, TAP és TAM sorozat a Telefongyártól, stb.
Az 1980-as évek közepére nyilvánvalóvá vált, hogy a "néhány nagy számítógép egy országnak" elméletet elsöpri az élet minden területén használható személyi számítógépek áradata. Így nyomban azon országok oldalára billent a mérleg, amelyekben hatalmas sorozatgyártásban lehetett olcsó gépeket előállítani.

[3] Épp a napokban jelentette be a Siemens cég, hogy felhagy számítástechnikai cikkeinek gyártásával, mert nem kifizetődő az évi mintegy egymillió termék előállítása. (1998. április 24.)

A Magyarországon 1979-ben megjelent **ABC80**-as személyi számítógépek jelentették az első mérföldkövet a számítástechnikai kultúra tömeges elterjedéséhez. Az ELTE TTK (akkor még) Numerikus és Gépi Matematikai Tanszéke 1981-ben beszerzett 5db személyi számítógépet, és ezeket azonnal felhasználták az oktató és a kutató munkában.

Prognosztikai vizsgálatok hazánkban még az 1980-as évek elején azt mutatták, hogy a számítógépes szakértők és a pedagógusok egyaránt az ezredforduló utánra várták a számítógépek általános oktatási alkalmazását. A felmérés azonban még a közép és nagy gépekre irányult, s nem vette figyelembe a személyi számítógépeket.

A TII (Tudományszervezési és Informatikai Intézet) 1982. januárjában pályázatot írt ki egy oktatási célú személyi számítógép tervezésére és gyártására. A zsüri a 13 pályázat közül (5 magyar) a HITEKA (Híradástechnika Szövetkezet) **HT 1080Z** típusú mikroszámítógépét választotta. 1983-ban 820 darab iskola-számítógépet vásárolt az oktatás, a következő évben további kb. ezer darabot. Így eljutott minden iskolába (létszámtól és iskolatípustól függetlenül) legalább egy számítógép. (Egy 1985-ös felmérés szerint minden 600 középiskolásra jutott egy gép.)

Az iskola-számítógép egy 8 bites, Z80 jelű mikroprocesszort tartalmazott. 16 kilobyte memóriával rendelkezett, mely "akár" 48 kilobyte-ra volt bővíthető. 4 fő egységből épült fel: a központi egységből, az ezzel egybeépített billentyűzetből, a beépített kazettás magnetofonból és a kijelző egységből, amely bármely, a kereskedelemben kapható fekete-fehér TV készülék lehetett. A gép BASIC nyelven volt programozható, s ennek nem kis szerepe volt abban, hogy a 80-as évek közepén hazánkban a BASIC programozási nyelv szinte egyeduralkodóvá vált (legalábbis a hétköznapi és az iskolai életben). - A gép kezelése egyszerű, bár a programok betöltésében kissé bizonytalan volt, de könnyen lehetett (igen kezdetleges felbontású) képeket rajzolni (koordinátarendszer-szerűen), beépített hanggenerátora segítségével zenélni.

Az iskola-számítógép azonban teljesítményéhez képest drága volt és a 80-as évek közepén egyre több külföldi számítógép került be az országba, melyek lényegesen többet tudtak. (Egy SZÁMALK felmérés szerint 1986-ban a **Commodore-64**, a HT-1080Z, a PRIMO és a Sinclair -gépek voltak Magyarországon a legelterjedtebb géptípusok.) Így az iskola-számítógépek helyett inkább ezek a gépek kerültek be az iskolákba is, hiszen szélesebb körben lehetett őket használni. A 80-as évek végére azonban az **IBM kompatibilis számítógépek** ("PC"-k) ára annyira lecsökkent, hogy ezek is bekerülhettek az oktatási folyamatba és részben könnyű bővíthetőségük, fejleszthetőségük folytán mára már szinte egyeduralkodóvá váltak.

3.4. A programozási nyelvek rövid története

Az ember és a gép kapcsolatának nagy problémája volt a nyelv kérdése. Minél kezdetlegesebb volt egy számítógép, annál több emberi energiát kellett fordítani a programozására. Az első gépeket kívülről, dugaszolással vagy más mechanikus eljárással kellett a megfelelő programra beállítani. - Az első elektronikus számítógépeknél a bitekig lebontott **gépi kód** felelt meg a fenti eljárásnak. A fejlődés második lépcsője az **assembly -nyelv**ek megjelenése.

Már Neumann János megfogalmazta 1944-ben, hogy nézzen ki egy alacsonyszintű (assembly) programozási nyelv. Az 1950-es évek elején felépített számítógépeket az **alacsonyszintű nyelvek** használata jellemzi.

Ezután kezdtek megjelenni az "emberközeli" magas szintű nyelvek, melyek kényelmesek ugyan az embernek, de nehezebben értelmezhetők a gép szempontjából, ezért különböző értelmező és fordító programokat kell közbeiktatni.

Az ún. **első generációs nyelvek**ben (1946-58.) megjelennek az elemi típusok, alapvető utasítások. Kifejezéseket még nem használtak.

Ekkor alkották meg a **FORTRAN** (az első időtálló magas szintű programozási nyelv) első verzióját, és elkészült az **ALGOL 58** (Algorithmic Language) a különböző típusú számítógépek közötti könnyebb programcserére. - Nem nagyon terjedtek el, mivel a számítógépek ekkor még igen lassúak voltak és a fordítóprogramok még jobban lelassították a működést.

A **második generációs nyelvek**ben (1958-65.) már megjelennek az összetett típusok és a kifejezés fogalma. Erre az időszakra jellemző a véges automaták és a nyelvek elméletének kidolgozása, a fordítóprogramok elméleti és gyakorlati megalapozása (BNF: Backus Norm Form, lengyelforma).

1960 körül elkészül a **FORTRAN IV**., mely már széles körben elterjedt. Ebben megjelennek az alternatív szerkezetek és a fordítási egységek.
Az időszak jelentős programnyelvei még: **ALGOL 60**, a **COBOL** (főként pénzügyi vonalon; itt jelenik meg a rekord és a file fogalom).

Az IBM **PL/I** nyelve az első kísérlet egy általános nyelvre (rengeteg utasítást használt).

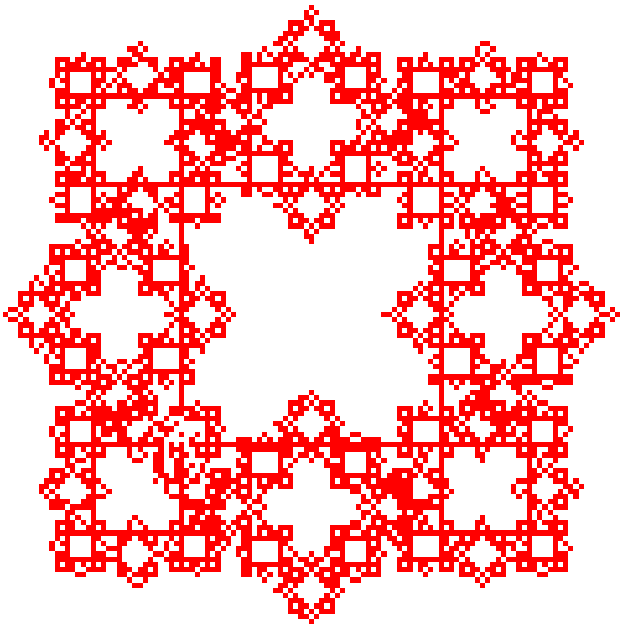
1964-ben a FORTRAN-ra alapozva megalkotják a **BASIC** programozási nyelvet. Alkotói között ott találjuk a magyar származású *Kemény Jánost*. Princetonban járt egyetemre, ahol Neumann János tanítványa volt. Ő kapta meg először az IBM 50 000 dolláros Louis Robinson-díját az időelosztásos rendszer kifejlesztéséért. - Ez volt az első oktatási célra kifejlesztett nyelv. Rengeteg "nyelvjárása" alakult ki, ezért felmerült az igény egy "minimál BASIC szabvány" létrehozására. 1980-ban el is készült egy tervezet, de nem vezették be.

A nyelv vitathatatlan előnyei közé tartozik, hogy "amatőr nyelv", könnyen, szinte hétköznapi nyelven fogalmazhatók az utasítások, de komolyabb, nagy méretű programok nem készíthetők vele (nem is ez volt a nyelv megalkotásának eredeti célja), a rendszer lassúsága miatt sem.

A **harmadik generációs** (1965-75.) programozási nyelvekre a 60-as évek végére kialakuló "szoftverkrízis" rányomta bélyegét. A szabad programírásnak köszönhetően rossz programkészítési technológiák alakultak ki. Emiatt előtérbe kerültek a struktúrált programok, típusok. Elkészült a **PASCAL**, **ALGOL 68**, **C**, ill. a nem Neumann-elvű nyelvek közül a **LISC**, **PROLOG**[4].
Ebben az időszakban dolgozták ki a programozás módszertanát. Programhelyesség-vizsgálatok és a nyelvek szemantikus leírása is ekkor történt meg.

Az oktatásban ma is széles körben elterjedt **PASCAL** nyelvet *Nicklaus Wirth*, a zürichi egyetem professzora alkotta meg 1968-ban. 1970-ben készült el a Pascal fordítóprogramja. Ennek a nyelvnek is több implementációja alakult ki, ezért 1973-ban elkészítették az első szabványát (Standard Pascal), majd ezt 10 év múlva bővítették. A nyelv rugalmasságára jellemző, hogy a személyi számítógépek megjelenésével állandóan alkalmazkodik az igényekhez: a Commodore 64 gépekre elkészült az Oxford Pascal, az IBM gépekre a Turbo Pascal 1.0 (1980-as évek eleje). A Windows rendszer megjelenését és rohamos elterjedését követve a Borland International Inc. cég 1991-ben piacra dobta a Turbo Pascal for Windows nevű programot.

1968-ban *Papert* nevéhez fűződik (a LISCre alapozva) a **LOGO** programnyelv létrehozása. Célja az volt, hogy olyan programnyelvet csináljon, amellyel már egészen kis gyerekek is tudnak programot írni. Ennek megfelelően a nyelv elsősorban rajzolásra alkalmas. Nem Neumann elvű, hanem automataelvű nyelv, többféle változata terjedt el. (Napjainkban egyre népszerűbbé válik az iskolákban a Windows felületű Comenius Logo.)



Egy fraktál képe, amelynek programja LCN LOGO nyelven íródott

[4] 1975-ben egy 10 fős magyar csapat részt vett a nyelv fejlesztésében.

A **negyedik generációs** (1975-90.) nyelveket a modularitás (elemekből való építkezés) jellemzi. Megjelennek a következő generáció csírái (objektumok, párhuzamosság).

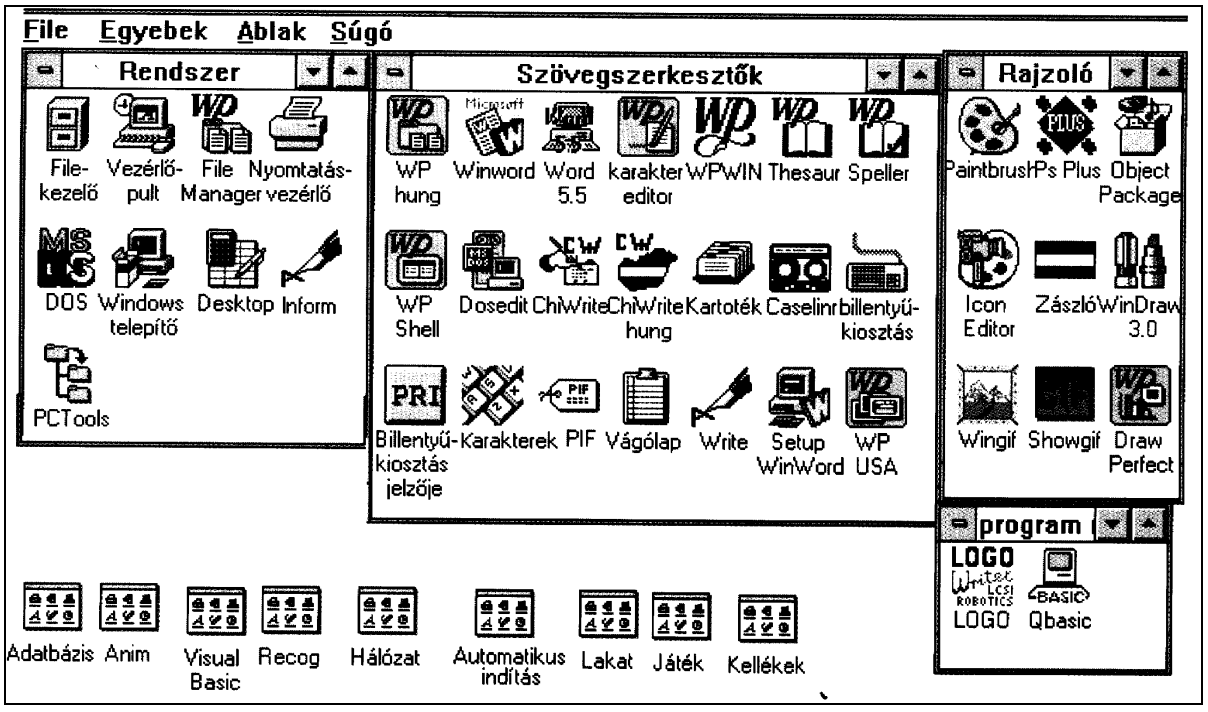
Az időszak jelentősebb programnyelvei: **MODULA-2, ADA, PASCAL 5.0, C++.**

1981-ben a Pascal logikusabbá, egyszerűbbé tételével megalkotják az **ELAN**-t (Educational Language), mely az első "tiszta" nyelv (a Pascal hiányosságait, következetlenségeit küszöbölte ki).

Az **ötödik generációs** nyelveket (1990-től) az objektum orientáltság és a párhuzamosság jellemzi. Ezek - alkalmazkodva a számítástechnikában kevésbé jártas emberek igényeihez - többnyire grafikus kezelői felülettel rendelkeznek (Visual Basic, stb.).

A mostanában megjelent **DELPHI** programnyelv a Pascalra és a Windowsra épül, melyet bővítettek az adatbázis-kezelés követelményeivel is.

A többi programnyelv is afelé a tendencia felé fejlődik, hogy egyre gyorsabbak és intelligensebbek legyenek, és használatuk minél kevesebb előképzettséget igényeljen a felhasználótól.



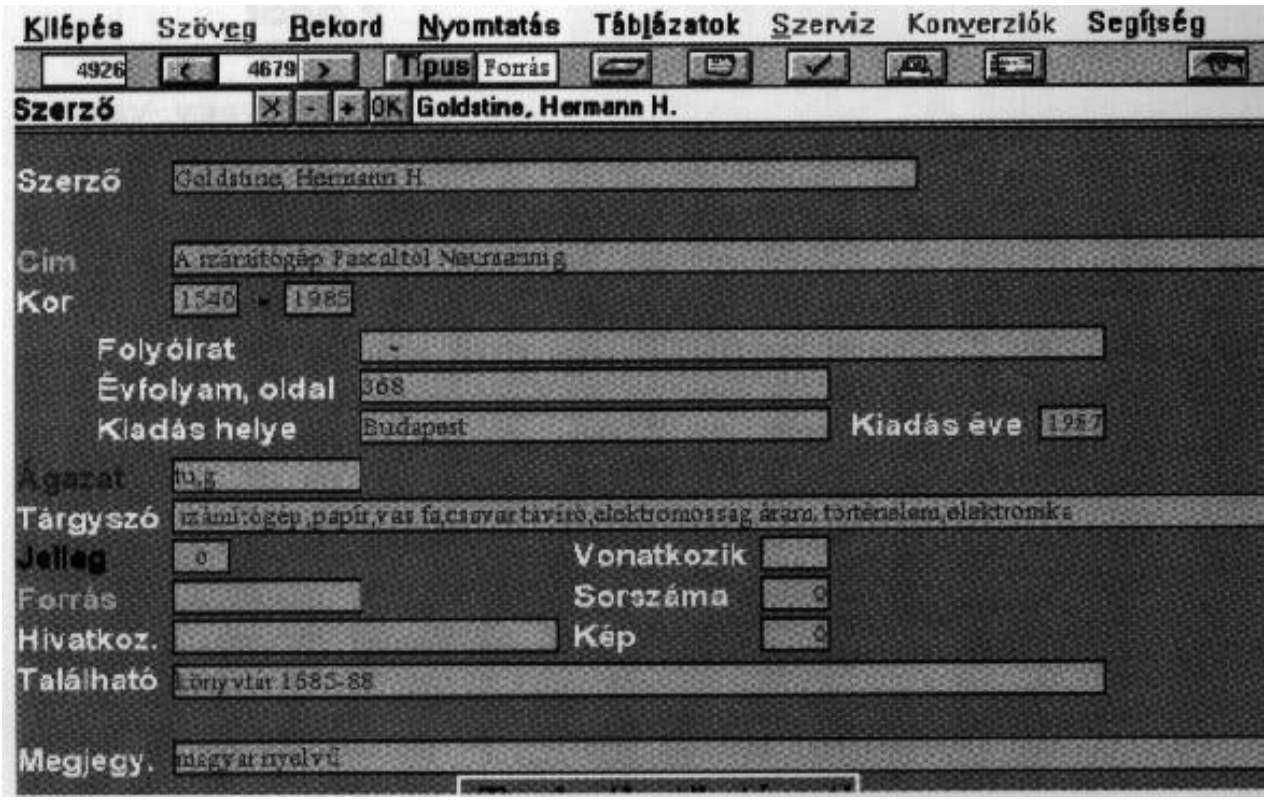
Windows ablakok (Ma már a software drágább, mint a hardware)

3.5. Utószó helyett ...

A számítógép a XX. sz. egyik legnagyobb csodája.

Amikor a múlt században a vasút berobbant az emberiség életébe, senki sem azt kérte számon, hogy történtek balesetek, vasúti tragédiák.

A számítógép is okoz bonyodalmakat: betörések, szem- és gerincbetegségek, vírusok, internet-rablások, felelőtlen gyermekek, stb. De amit a gép eddig nyújtott és ami prognosztizálható a részéről, az az eddigi csoda folytatásának ígéretét jelenti...



Technikatörténeti adatbank egy oldala

4. Adatkezelés célja, története

Adatok kezelésével már az ókori Egyiptomban is foglalkoztak, agyagtáblákra vésték fel az éves termésadatokat. Nemcsak **az adatok tárolása** volt a **cél,** hanem **az adatok visszakeresése, módosítása,** mai szóhasználattal élve **az adatok karbantartása** is. **Az adatkezelés feladata: óriási adattömegeken egyszerűbb műveleteket elvégezni: például összeadás, átlagolás, rendezés stb.**

Minden kor kialakította a maga **kézi adattárolási** rendszerét az agyagtáblától a kartotékokig. Az első forradalmian új eszköz a **gépi adattárolás** területén *Hollerith* nevéhez fűződik, aki 1890-ben megnyert egy pályázatot, melyet az Amerikai Egyesült Államokban népszámlálásra írtak ki. Az Államokban Kb. 55 millióan éltek abban az időben. Tíz évenként tartottak népszámlálást, melyen a legfontosabb személyi adatokat vették föl.(Név, születési idő, hely, lakcím stb.) Az 1880-as népszámlálás töménytelen adatainak feldolgozásával 1890-re még nem végeztek. Hollerith forradalmi újítása a **lyukkártya,** mint adattároló eszköz és a hozzá tartozó géppark volt. (kártyalyukasztó; rendező; tabulátor, mely a sornyomtató őse.) Egy ember népszámlálási adatait egy lyukkártya tartalmazta. Lényege, hogy a kartonlapot (lyukkártyát) beszámozott oszlopokra és sorokra osztották. Az azonos tulajdonságot (például a személy neme) minden kártya azonos részén, "mezőjén" levő lyukak és üres helyek pozíciója hordozta. Például bal felső sarokban levő lyuk a kártyán nőt, ha nincs lyuk, férfit jelöl. Adatok bevitele a leírtak szerint, a kártyák lyukasztásával történt A kiválasztást pedig egy erre készített mechanikus szerkezet végezte oly módon, hogy a keresett mezőben lyukakat tartalmazó kártyáknál akadálytalanul áthaladt, míg ahol nem találta a megfelelő lyukakat, azon kártyák fennakadtak. Majd a **számítógépek** megjelenését követően a lyukkártyát, mint adattároló eszközt felváltotta a **mágnesszalagos háttértároló,** mely nagyobb adatbiztonságot nyújtott és több adat tárolását tette lehetővé. Soros elérése miatt nehézkes és lassú az adatokhoz való hozzáférés vagy módosítás, mivel a kívánt adat eléréséhez az azt megelőző adatokon minden esetben végig kell futni, hacsak a kívánt adat nem a legelső a sorban. A mágnesszalagokat hamarosan leváltották a közvetlen hozzáférésű eszközök a **mágneslemezek** (diszkek), melyekben cím szerint lehet keresni ill. módosítani.

A hardverrel (alkatrész) párhuzamosan fejlődtek a különböző szoftverek (program) is. Először ún. **file-kezelő programok** jelentek meg például dBase. A FoxPro program, mellyel nemsokára megismerkedünk, szintén a file-kezelők népes családjába tartozik. Majd a komolyabb igényeket kielégítő ún. **adatbázis-kezelő programok** következtek például Oracle. A file-kezelő és adatbázis-kezelő rendszerek között a legfontosabb különbség az, hogy a file-kezelő programok adott témára több fájlt használnak, míg az adatbázis-kezelők minden adatot egy adatbázisban kezelnek. Ezenkívül az adatbázis-kezelő programok lehetővé teszik az adatok **osztott használat**át, melyben **ugyanazt az adatot egyszerre több felhasználó is elérheti.** Manapság rendelhet valaki jegyet ugyanarra a repülőgépre Washingtonban és Tokióban is, mégsem történik meg, hogy egy helyet két embernek adnának el. A továbbiakban a file-kezelő programokat is, kicsit nagyképűen, adatbázis-kezelő programoknak nevezzük.

Ma már nem kell szobányi helyen tartani egy népszámlálás adatait, elég hozzá egyetlen számítógép. Nem tart órákig egy módosítás (például valaki új címre költözik) bejegyzése, csupán másodpercekig. Az adatkezelés fejlődése a számítástechnika egyik sikertörténete. Az adatkezelés a számítógép egyik legfontosabb feladatává vált.

Feladatok:

452 oldal 104. Feladat

5. Telepítsünk egy adatbázis-kezelő programot!

Milyen adatbázis-kezelő programot válasszunk? A kereskedelemben kapható dBase, Access, Clipper, FoxPro programok közül FoxPro 2.5 verziójának magyar változata mellett döntöttünk.

A FoxPro előnyei:

- Gyors működés nagy fájlok esetén is
- Windows alatt is fut
- Magyar nyelvű változata is van
- Hálózati felhasználást is lehetővé teszi
- Rendelkezik képernyő, riport (jelentés), menü, címke, lekérdezés, alkalmazás generátorral

A képernyő, riport stb. generálás azt jelenti, hogy a felhasználó csupán megtervezi a képernyőt, jelentést (felhelyezi a megfelelő parancsgombokat, ablakokat, mezőket a képernyőre, jelentésre), de a programot (kódot) a FoxPro állítja elő (generálja), sőt a gépi kódra történő fordítást is elvégzi, jelentősen gyorsítva ezzel a felhasználó munkáját.

A FoxPro minimális hardver és szoftver igényei: 386-os processzor, 6 MB RAM, 3.1 verziójú DOS, 3.0 Windows.

A FoxPro adatbázis-kezelő programot, mint minden Windows alatt futó alkalmazást, használat előtt telepíteni (installálni) kell. A telepítés lépései:

1, A Windows elindítása után a file-kezelőt  indítjuk el.
2, Behelyezzük a megfelelő meghajtóba a FoxPro install lemezek közül az 1. számút, és beolvassuk a tartalmát.
3, Elindítjuk a `setup.exe` fájlt.
4, A megjelenő Felhasználói információk ablakban a felhasználó adatait írjuk be. Majd ellenőrizhetjük.
5, A Telepítési könyvtár és csoport párbeszédablakban megadjuk azt a könyvtárat ahová a FoxPro programot telepíteni szeretnénk. Ezután megadhatjuk annak a programcsoportnak (ablak) a nevét is, melybe a FoxPro programot és egyéb kapcsolódó alkalmazásokat helyezni szeretnénk. A Folytatás parancsgombbal folytathatjuk a telepítést.
6, A megjelenő Microsoft Windows FoxPro telepítés párbeszédablakban választhatunk a telepítés módjai közül: Teljes telepítés (minden fájlt felmásol és kicsomagol); Egyedi telepítés (csak az általunk kiválasztott fájlokat teszi föl); Minimum telepítés (csak a FoxPro futásához elengedhetetlen fájlokat teszi föl). Mi a Teljes telepítést választjuk.
7, A következő ablakban választanunk kell a DOS vagy Windows billentyű használat között. Mi a Windows-t választjuk.
8, A következő ablakban az Automatikus módosítást választva a telepítő program átírja a `config.sys` fájl `FILES` utasítás sorát `40`-re. Azaz `40` fájlt lehet megnyitni egyszerre.
9, Ezután a telepítő program elvégzi a telepítést

A telepítés befejezése után lépjünk ki a telepítőből.

A telepítés más módjai:

- Szerverre történő telepítés az `a:\setup /A` parancs segítségével történhet.
- Szerverről munkaállomásra `a:\setup /N` parancs beírásával történhet.

A FoxPro adatbázis-kezelő program indítása, a Windows-ba való belépést követően, a FoxPro ikonra  történő kétszeres kattintással tehető meg.

Feladatok:
437. oldal 17. 24. feladat
438. oldal 25.-26. feladat

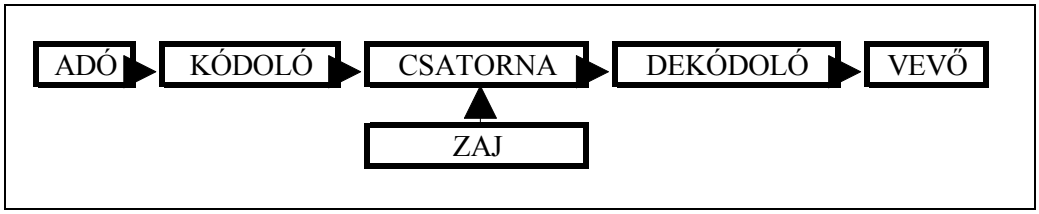
6. Adatbázis-kezelési alapfogalmak

6.1. Mire van szüksége egy adatfeldolgozó programnak?

A válasz egyszerűnek látszik: adatokra. Ez valóban olyan egyszerű?
Két régi ismerős beszélget a buszon:

-Nem volt igazi ez a tél!
-Miért?
-Nem esett a hó.
-Dehogynem, esett!
-Igen, de nem eleget.
-Nekem sok is volt!

Nos, ebből mit értsen egy számítógép, ha például az idei télről kéne adatokat rögzíteni. Semmit. A fenti adatok csupán vélemények, melyből információt tudhatunk meg arról például, hogy ki mennyire szereti a telet, ez az információ azonban nem tekinthető adatnak, az **adat** ugyanis: **lehet bármi a körülöttünk levő világból, ami jelent valamit az adatkezelő számára. Tulajdonsággal rendelkezik, és megkülönböztethető a többi adattól. Számítógépen tárolható, és általában valamilyen mérésből megfigyelésből származik.** A fenti beszélgetést egy meteorológus valószínűleg így szűrné le: Az 1997-98-as év tele enyhe volt, a hó formájában lehullott csapadék mennyisége minimális. Ezt az adatot már könnyebben tudjuk egy adatbázisba helyezni. **Az adatbázis adott céllal összeállított adatok halmaza.** Például Időjárási adatok adatbázisa.
Az információ matematikai meghatározására tudományos elméletet dolgozott ki **C. E. Shannon** 1948-ban. Szerinte **egy jel annál nagyobb információt hordoz, minél ritkábban fordul elő, minél kisebb előfordulásának valószínűsége.** Például egy szövegben az "X" betű nagyobb információt hordoz, mint egy "e" betű. **Az információ mértékegysége a bit.** Annak a hírnek az információtartalma 1 bit, melyet két azonos valószínűségű esemény közül az egyik bekövetkezéséről kapunk. Például egy szülőszoba előtt várakozó apuka megtudja, hogy születendő gyermeke kislány vagy kisfiú lett. Az információ terjedésére szintén Shannon adott modellt, melyet a hírközlésben, illetve a kommunikációelméletben stb. is alkalmaznak. **Az információt az adó hozza létre. A kódoló jelsorozattá alakítja, melyet üzenetnek hívunk. Az üzenetet csatorna továbbítja. Továbbítás közben az üzenethez zaj csatlakozik, mely torzítja az üzenetet. Az üzenet eljut a dekódolóhoz, amely átalakítja a vevő számára érthető formára.**



Shannon-modell

Mi a halmaz?
A halmaz matematikai fogalom. Egy halmazt megadhatunk, ha felsoroljuk az elemeit, vagy azt a szabályt mely alapján eldönthető valamiről, hogy **az a halmaz eleme vagy sem.** Például a Soroksári úton lakók halmaza. Bárkiről eldönthető, hogy beletartozik ebbe a halmazba vagy sem.
Mit tekinthetünk adatfeldolgozás szempontjából egyednek? Bármit amiből csak egy van? Nem. **Egyednek nevezünk bármit, bármiket ami egy másik egyedtől megkülönböztethető, valamilyen tulajdonsága alapján.** Pl. pékboltban nyilvántartják az árukészletet. Minden egyes kiflit nem tekintenek egyednek, csupán a kiflik "egészének"

számáról jegyeznek fel adatot. A kiflik összessége egy egyed, függetlenül attól, hogy hány db van belőle. Azonban ha a pékboltban többfajta kifli is van pl. sós kifli és sajtos kifli, akkor már két egyedről beszélünk. Az autókereskedésben azonban nem elég tudni, mennyi autó van összesen. Szükség van minden autó megkülönböztetésére, minden autó külön egyednek számít.

Mit nevezünk tulajdonságnak, vagy attribútumnak? Sajnos az adatkezelésben használt tulajdonság fogalom nem egyezik meg a hétköznapi életben használt fogalommal. **A tulajdonság az egyed adatkezelési szempontból fontos jellemzője, mely alapján megkülönböztethető a másik egyedtől.** Például autók esetén: a gyártási szám. Pedig ha autót vásárolok, számomra a hétköznapokban lényegtelen tulajdonság az autó gyártási száma, sokkal fontosabb a típus, életkor vagy a szín, holott adatkezelési szempontból a gyártási szám is tulajdonság akár a szín, típus stb.

6.2. Hogyan tároljuk az adatokat?

Nyilvánvaló, hogy egy könyvben nagyon sok adat található, tárolható, mégsem lenne célszerű például Gárdonyi Géza Egri csillagok című művét az első sortól az utolsóig, egy adatbázis-kezelő programmal nyilvántartani. **Szövegszerű kezelést** kell használnunk, esetleg képeket csatolnunk, ahogy azt korábban, a szövegszerkesztő programok tanulásakor megismertük. Egy könyvtárban a könyvek nyilvántartásához (cím, szerző, leltári szám) érdemes **adatkezelő szoftvereket** használni, hisz azokat éppen ilyen feladatokra találták ki! Az adatokat **táblázat**os formában tároljuk.
Például

név	magyarjegy	matekjegy	átlag
Albert Veronika	5	4	4,5
Balázs Ferenc	4	4	4
Kovács Pisti	3	3	3
Zebra Zoltán	5	5	5

Táblázat

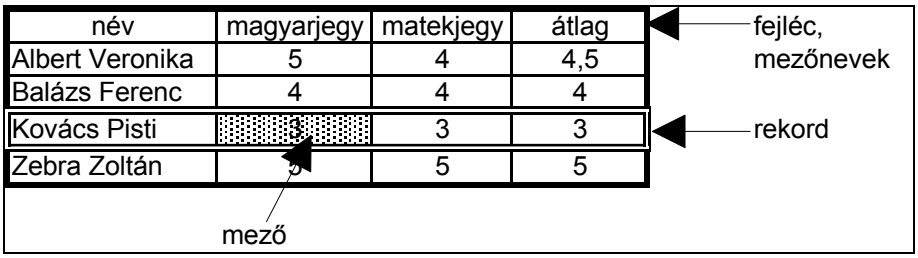
A táblázatos forma előnyei: könnyebbé válik a számomra fontos adat kikeresése. Segítség az is, ha a táblázat adatai sorba vannak rendezve. A fenti táblázat például nevek szerint van növekvő sorrendben rendezve. Amikor viszont az iskolai tanulmányi versenyre vagyunk kíváncsiak, akkor a jegyek átlaga szerint kell csökkenő sorrendbe rendezni a táblázatot. **A gyorsabb keresés érdekében fontos, hogy egy adatbázis-kezelő program tudjon**, és lehetőleg gyorsan tudjon, **különböző rendezési feltételeknek eleget tevő táblázatokat készíteni.** Ennek sajátos trükkjei vannak, melyekre a későbbiekben még visszatérünk.

6.3. A táblázat részei

A táblázatnak mindig **van neve**. Ugyanúgy, ahogy egy Word szövegszerkesztővel írt dokumentumnak is adtunk nevet `.doc` kiterjesztéssel, vagy egy Excel táblázatkezelő programmal készített munkalapnak is `.xls` kiterjesztéssel. Az adatbázis-kezelő programmal létrehozott táblázat egy fájlban tárolódik. A fájl neve az angol ABC kis és nagy betűit tartalmazhatja és az aláhúzás jelet (_), hossza maximum 8 karakter. A táblázatot tartalmazó fájl azonosítójának **kiterjesztése mindig `.dbf`**. Például `tanulo.dbf`

A táblázat fejléce tartalmazza az egyedek tulajdonságainak nevét, az ún. mezőneveket.
A táblázat sorokból és oszlopokból áll. A táblázat egy sora tárolja egy egyed adatait. Például, Kovács Pistiét. (Ahogy a lyukkártya idejében egy kártya tartalmazta egy személy adatait.) **A táblázat sorait rekordoknak nevezzük.**
A táblázat sorok és oszlopok által határolt legkisebb egysége a cella. A táblázat egy celláját mezőnek nevezzük. **A mező az egyed egy tulajdonságáról, jellemzőjéről tartalmaz adatot például Kovács Pisti magyar jegyét.** A táblázat egy oszlopa**, vagyis egymás alatt levő mezők,** a különböző egyedek azonos tulajdonságairól, jellemzőiről árulkodik. Például magyar jegyek.

A `tanulo.dbf` tábla



A táblázat részei

6.4. Egy táblázat készítésének első lépései, mezőtípusok

A táblázat tervezését, készítését mindig a táblázat fejlécének elkészítésével kezdjük. A táblázat fejlécében szereplő mezőnevek mindig egy szóba írandók, a magyar abc betűit tartalmazhatják, számokat, és aláhúzás (_) jelet. Számmal nem kezdődhetnek, és hosszuk maximum 10 karakter lehet. A mezőnevek legyenek "értelmesek", ún. beszélő nevek, hogy más felhasználó is értse őket. (Lehet **becenevet** (**ALIAS**) is használni később, mely olyan szabadon választott név, amelyet utasításokban mezőknek, vagy kifejezéseknek adhatunk, azért hogy tömörebbek legyenek.)
A mezőnevek megadása után, meg kell adni a mezőkben szereplő adatok típusát és méretét is. Mindegyik típushoz tartozik egy művelethalmaz is, ami azon műveletek csoportja, mely az adott típusú adattal elvégezhető. A táblázat tervezésekor fontos szempont a helytakarékosság, mivel a számítógép memóriája véges. A jobb helykihasználás egyik módja a lehető legkisebb méretű mezők létrehozása.

A típusok neve, szélessége (mérete), és a velük való műveletek:

Karakteres mező (**C**haracter)
Maximális hosszúság: 254 karakter.
Karakteres adatnak tekinthető minden olyan adat, amely egy sorba leírható, és nem tartalmaz vezérlő karaktereket (tabulátort, soremelést (ENTERt)).Műveletek során a karakteres mezők tartalmára mindig idézőjellel hivatkozunk (például "Kovács Pisti"). Olyan számadatokat is érdemes karakteres mezőbe írni, mellyel később nem végzünk aritmetikai műveleteket (összeadás, kivonás stb.) Erre jó példa a postai irányító szám.
A karakterkezelő függvényeket a **Hiba! A hivatkozási forrás nem található.** sz. melléklet tartalmazza.(**Hiba! A könyvjelző nem létezik.**.oldal)

Numerikus mező (**N**umeric)
Maximális hosszúság: 20 karakter előjellel (+,-) és tizedesvesszővel együtt.
Beírható adat: számjegy, tizedesvessző, előjel. Számolási pontosság maximum 16 tizedesjegyre.
A numerikus mező tartalmával aritmetikai műveletek végezhetők (+,-,*,/,** vagy ^ azaz négyzetreemelés). A numerikus mező tartalmát kezelő függvényeket a **Hiba! A hivatkozási forrás nem található.** sz. melléklet tartalmazza. (**Hiba! A könyvjelző nem létezik.**.oldal)

Lebegőpontos mező (**F**loat)
Maximális hosszúság: 20 karakter.
Nagypontosságú tudományos adatok tárolására való.
Matematikai, aritmetikai műveletek végezhetők vele. Műveletei és függvényei a numerikus mező műveleteivel és függvényeivel egyeznek meg.

Dátum mező (**D**ate)
Hosszúság: 8 karakter. (A mező hossza nem változtatható)
Dátum típusú adatok 8 jegyű pozitív egész számként vannak tárolva a memóriában. Így összeadás és kivonási műveletek is végezhetők rajtuk. Például ha egy dátumhoz hozzáadunk egy pozitív számot, akkor annyival későbbi dátumot kapunk eredményül. A dátum megjelenési formáját a Windows beállításai szabályozzák. (Windows 3.1-nél a Vezérlőpult Nemzetközi ikonjánál levő értékek.) Dátumra való hivatkozáskor a dátum mező tartalmát kapcsos zárójelek közé kell tenni például {98.04.11}. A dátumkezelő függvényeket a **Hiba! A hivatkozási forrás nem található.** sz. melléklet tartalmazza. (**Hiba! A könyvjelző nem létezik.**.oldal)

Logikai mező (**L**ogical)
Hosszúság: 1 karakter. (A mező hossza nem változtatható)
A logikai mező a logikai értékek (igaz/hamis) tárolását szolgálja. Az igaz értéknek .T. betű (true magyarul igaz), a hamis értéknek .F. betű (false, magyarul hamis) felel meg.
Logikai műveletek: .NOT. .AND. .OR. Logikai feltételekben a tagadás (NOT) helyett felkiáltó jelet (!) alkalmaznak. Új logikai művelet a .XOR. művelet, igazság táblázatát a 9.1 sz. melléklet tartalmazza. (89.oldal)

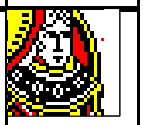
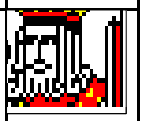
Feljegyzés mező (**M**emo)
Hossza:10 karakter (A mező hossza nem változtatható)
Ajánlott minimális hosszúság: 255 karakter.
A feljegyzés mezőbe hosszabb szöveges információ írható. Például egy videokölcsönző adatbázisában a film témája, tartalma kerülhetne memo típusú mezőbe. A leírt szöveg külön fájlban (`.fpt`) tárolódik. A 10 hosszúságú mezőben csak egy mutató található, amely leírja, hogy az aktuális rekordhoz található e feljegyzés a feljegyzéseket tartalmazó `.fpt` fájlban. Tehát ha adatbázis-kezelővel készített táblázatunkban memo típusú mező van és szeretnénk pl. floppyra másolni, akkor a `.dbf` kiterjesztésű fájlok mellett az `.fpt` kiterjesztésű fájlt is másolnunk kell.

Általános mező (**G**eneral)
Hossza:10 karakter (A mező hossza nem változtatható)
Kép és hanganyagok, valamint Windows objektumok tárolására szolgál.
A tárolás módja megegyezik a feljegyzés mezőtípusával. Újdonság a Feljegyzés mezőhöz képest, hogy a mező tartalma nemcsak beágyazással kerülhet a táblázatba, hanem csatolással is. Ekkor nemcsak a `.dbf`, `.fpt` fájlokat hanem a csatolni kívánt fájlt is le kell másolnunk.

Kidolgozott példa:

Milyen típusú és hosszú mezőket tervezne a `tanulo` nevű táblában, melyben név, magyarjegy, matekjegy, németjegy, átlag, születési_idő, szakkörös, jellemzés, arckép nevű mezők szerepelnek? Rövidebben: Tanulo(név, magyarjegy, matekjegy, németjegy, átlag, születési_idő, szakkörös, jellemzés, arckép)

TANULO.DBF

név	magyarjegy	matekjegy	németjegy	átlag	születési_idő	szakkörös	jellemzés	arckép
Albert Veronika	5	4	5	4,67	79.09.21	i	Jól tanul, segítõkész társaival. Vezéregyéniség a közösségben.	
Balázs Ferenc	4	4	5	4,33	79.09.21	i	Szorgalmasan tanul, ám társaival ellenséges.	
Kovács Pisti	3	3	2	2,67	80.01.01	n	Lusta, ha a tanulásról van szó, de nagyon szeret sportolni.	
Zebra Zoltán	5	5	5	5,00	79.02.20	n	Nagyon jól tanul, szívesen magyaráz társainak	

Táblázat a tanulókról

Megoldás:

Mező neve	**Mező típusa**	**Mező hossza**	**Tizedes**	***Indoklás***
név	karakteres C	35		*Ide nevek kerülnek, tehát karakterek, ügyelni kell arra, hogy a nagyon hosszú nevek is kiférjenek*
magyarjegy	numerikus N	1		*Mivel ide 1-től 5-ig egyjegyű számok kerülnek a hossz 1*
matekjegy	numerikus N	1		*Mivel ide 1-től 5-ig egyjegyű számok kerülnek*
németjegy	numerikus N	1		*Mivel ide 1-től 5-ig egyjegyű számok kerülnek*
átlag	numerikus N	4	2	*1 db számjegy (1-től 5-ig)+1 db tizedesvessző+2 db tizedesjegy= 4 jegy*
születési_ idő	dátum D	8		*mivel ez dátum, és hossza mindig 8 karakter (a hossz nem változtatható)*
szakkörös	logikai L	1		*Vagy szakkörös vagy nem (a logikai mező hossza nem változtatható)*
jellemzés	memo M	10		*Osztályfőnök hosszabb jellemzést is írhat, több sornyit (a mező hossza nem változtatható)*
arckép	általános G	10		*A tanuló arcképe egy képfájl (a mező hossza nem változtatható)*

6.5. A kulcs fogalma

Hogyan lehet egy adatbázis egyedeit megkülönböztetni? Emberek esetén erre leggyakrabban a nevet használjuk. Ez nem mindig elég. Járhat pl. egy osztályba két ugyanolyan nevű diák. Pl. két Szabó Mária. Ilyenkor valami mesterséges megkülönböztetést teszünk közöttük. Egyik Szabó Mária 1, másik Szabó Mária 2 lesz. Ha nagyon sok személy adatait tartják nyilván pl. egy tüdőszűrő állomáson, ott a fenti módszer nem alkalmazható az egyforma nevek kivédésére. Ilyenkor az ápolónő további adatokat is kér. Kérdezi a lakcímet, anyja nevét stb. Ebben az esetben a konkrét személynek több adatot meg kellett adni (név, lakcím, anyja neve). **Azon tulajdonságot, vagy tulajdonságokat, mely alapján meg lehet különböztetni egyértelműen egyik egyedet a másiktól, azonosítónak, vagy elsődleges kulcsnak nevezzük.** Tehát **az elsődleges kulcs olyan oszlopa vagy oszlopai a táblázatnak, melynél minden sor különbözik. A kulcs szerepe a rekordok azonosítása. A kulcsnak egy sorban szereplő értéke a kulcsérték.**

A fenti példában elsődleges kulcs: név, lakcím, anyja neve együttesen. Ezt a fajta kulcsot **összetett kulcsnak nevezzük, mivel 2 vagy több tulajdonságból áll.** Mivel az így kapott elsődleges kulcs túl hosszú, sok esetben ún. **mesterséges azonosító**t alkalmaznak, mégpedig úgy, hogy **sorszámozzák az egyedeket.** Ilyenkor a legegyszerűbb elsődleges kulcs maga a sorszám. **Mindig tervezzünk kulcsot adattáblánkba! Idegen kulcsnak nevezzük azt az azonosítót, mely egy másik táblában elsődleges kulcsként szerepel.**

Táblázat kulcsértékkel

Kidolgozott Példa
461. oldal 167.
Feladat
Az alábbi táblákban aláhúzással jelölje meg az elsődleges kulcsnak alkalmas mezőket!
A. DOLGOZO(Torzsszam, Nev, Fizetes, Datum_be, Nyelv_kod, Lakcim)
B. NYELV (Nyelv_kod, Megnevezes)
C. NYELVTUD (Torzsszam, Nyelv_kod, Minosites)

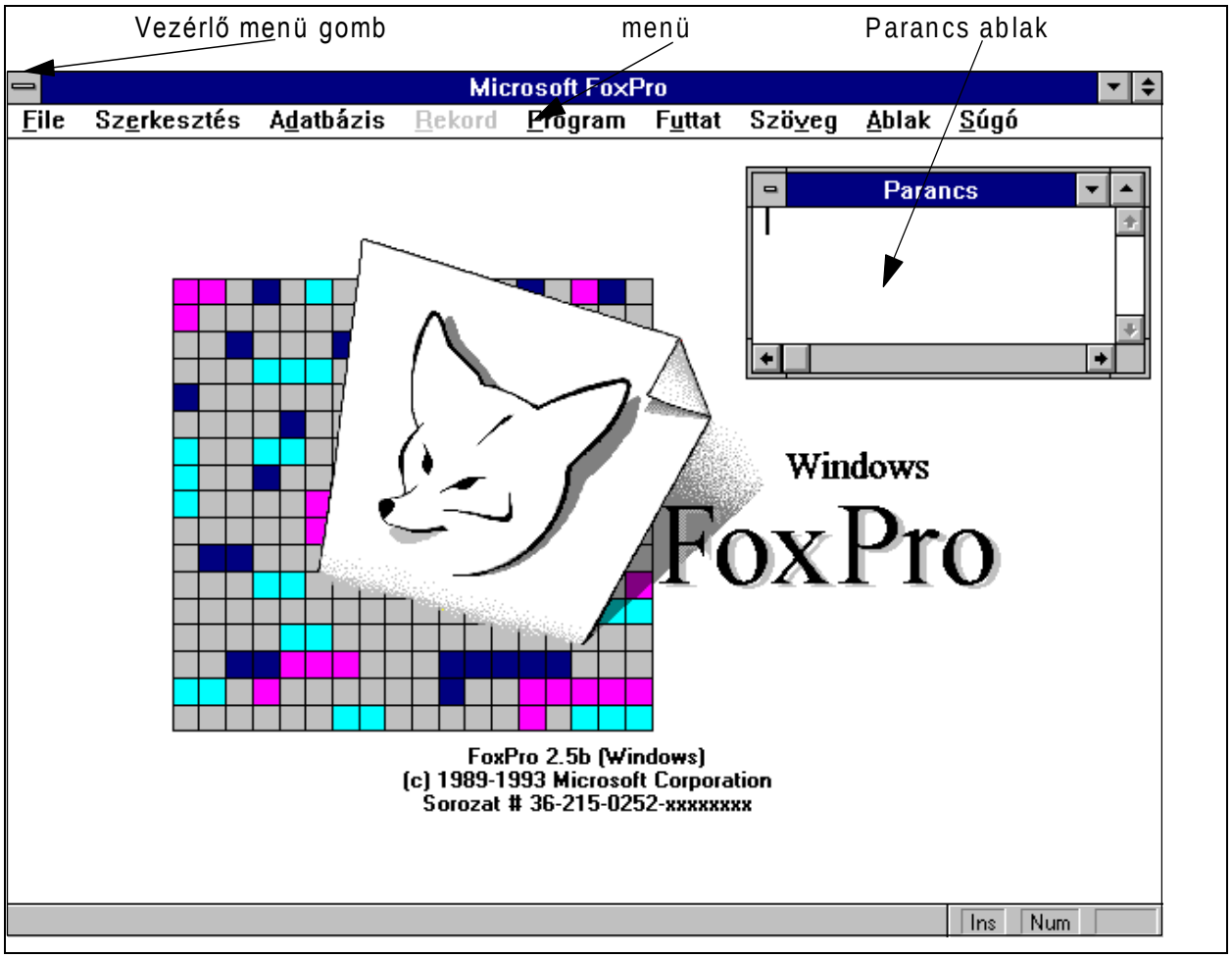
Megoldás:
Az elsődleges kulcsot keressük, vagyis azt a mezőnevet, tulajdonságot, mely alapján egyértelműen azonosíthatjuk az egyedet.
A. A dolgozókról fel van jegyezve számuk (Torzsszam), nevük (Nev), fizetésük (Fizetes), belépésük dátuma (Datum_be), nyelvtudások kódja (Nyelv_kod), lakcímük (Lakcim). Ezek közül olyan adat, mely biztosan nem egyezik senki máséval sem, csak a Torzsszám lehet, tehát ez az elsődleges kulcs.
B. Tároljuk a nyelvekhez tartozó sorszámot (Nyelv_kod), és elnevezésüket (Megnevezes). Ami biztosan különböző az a Nyelv_kod.
C. Tárolva van egy dolgozóról sorszáma (Torzsszam), milyen nyelven tud (Nyelv_kod), és mennyire (Minosites). Nem elég elsődleges kulcsnak a Nyelv_kod, mint az imént, hisz sok dolgozó beszélheti ugyanazt a nyelvet. Nem elég elsődleges kulcsnak Torzsszam sem, hiszen egy dolgozó több nyelvet is beszélhet más és más szinten. Így itt összetett kulcs lesz az elsődleges kulcs Torzsszam, Nyelv_kod együtt.
A válasz tehát:
A. DOLGOZO(Torzsszam, Nev, Fizetes, Datum_be, Nyelv_kod, Lakcim)
B. NYELV (Nyelv_kod, Megnevezes)
C. NYELVTUD (Torzsszam, Nyelv_kod, Minosites)

6.6. Hozzunk létre adattáblát FoxPro programmal!

A FoxPro 2.5 for Windows egy Windows alatt futó adatbázis-kezelő program magyar változata. A programban hasonló kezelőszervek találhatók, mint a korábban tanult Windows alkalmazásokban. A FoxPro programot ikonjára történő kétszeres kattintással indíthatjuk el. Indítása után megjelenő munkafelületen a következő kezelőszervek jelennek meg:

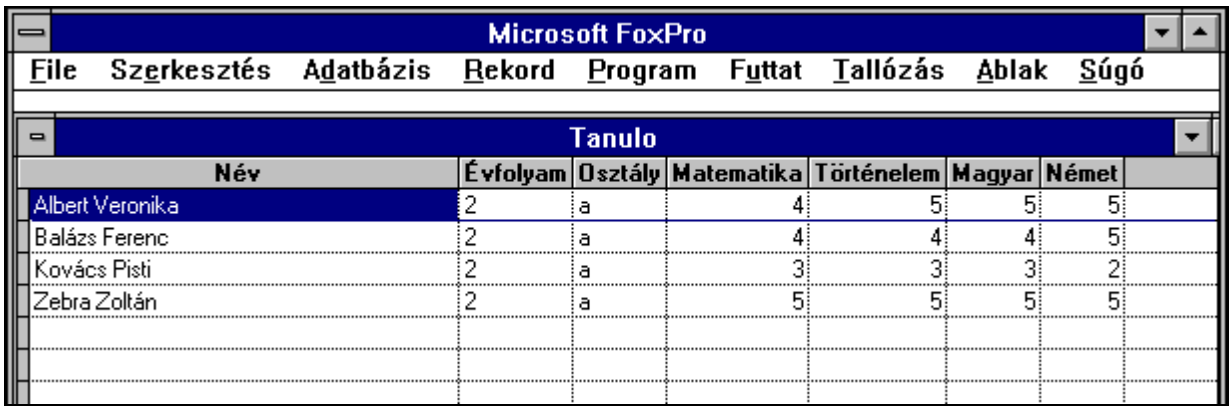
menü: mely parancsok kiadását teszi lehetővé

parancsablak: melybe parancsokat gépelhetünk



A FoxPro kezelőszervei

Készítsük el első adattáblánkat!
A feladat a következő:
Hozzuk létre az alábbi táblázatot `tanulo.dbf` néven a `G:` meghajtónkra `ADATOK` könyvtárba.



Név	Évfolyam	Osztály	Matematika	Történelem	Magyar	Német
Albert Veronika	2	a	4	5	5	5
Balázs Ferenc	2	a	4	4	4	5
Kovács Pisti	2	a	3	3	3	2
Zebra Zoltán	2	a	5	5	5	5

A `tanulo.dbf` tábla

A táblázat elkészítéséhez két út vezet, a; **menüvel, vagy** b; **parancsablakon keresztül**.

a; Nézzük először a menüvel történő megoldást!

Válasszuk a FILE főmenüpont Új almenüpontját. A megjelenő párbeszédablakban válasszuk ki a Táblázat opciógombot. Ezután kattintsunk az Új feliratú parancsgombra.

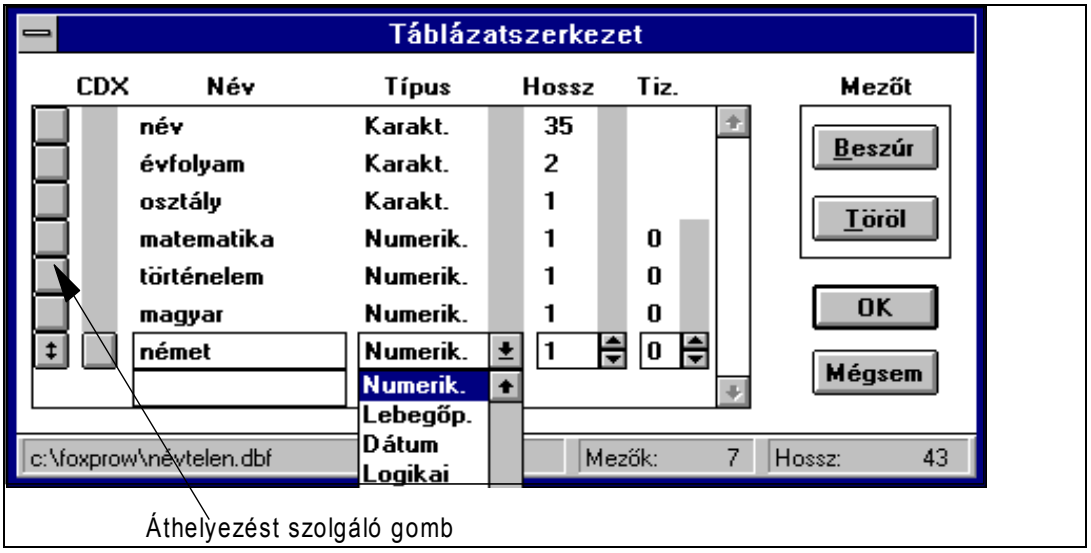
File főmenü Új almenüpontja

A táblázat készítését fejlécének megadásával, azaz a mezőnevek, típusok és méretek megadásával kell kezdeni, melyet a Táblázatszerkezet feliratú párbeszédpanelben tehetünk meg.
A Táblaszerkezet ablak első oszlopában Áthelyezést szolgáló gombot találunk. E gomb húzásával lehet a mezők sorrendjét megváltoztatni utólagosan. A második oszlopban a CDX felirat alatt adhatjuk meg, hogy adott mezőhöz készítünk indextáblát, vagy nem. Ennek a későbbiekben nagy jelentősége lesz, most azonban nem töltjük ki. A Név oszlopban levő

szövegmezőbe kell beírni a mező nevét. A Típus oszlopban a mező típusát kell kiválasztanunk egy listamezőből, a Hossz felirat alatt a mező hosszát, és végül a Tíz. oszlopba pedig a tizedesjegyek számát választhatjuk ki számlistamezőből.

Az alábbi mezőtípusokat választjuk:

- • név: karakteres, és 35 karakter hosszúságú, mivel ügyelnem kellett arra, hogy a leghosszabb név is kiférjen
- • évfolyam: karakteres, és 2 hosszúságú, mivel ide számokat írok ugyan, de sosem fogom összeadni vagy kivonni őket, vagyis nem érdemes numerikus mező típust választanom, hiszen nem végzek matematikai műveletet a mezők tartalmával. Maximum 12 évfolyamú lehet ma egy iskola, tehát 2 jegynél többre nincs szükségem.
- • osztály: karakteres és 1 hosszúságú, mivel az osztályokat általában 1 betűvel jellemzik. Pl. b osztály
- • matematika, történelem, magyar, német mezőket numerikusnak választom, és 1 hosszúságúnak, mivel ide 1-től 5-ig kerülnek majd számok, és velük végezhetek matematikai műveletet pl. átlagszámítást



Táblaszerkezet létrehozása

A Beszúr parancsgombbal a kurzor előtti sorba szúrhatunk be egy újabb mezőtípus megadást. A Töröl gombbal azt a definíciót törölhetjük, ahol a kurzor áll. A Mégsem gomb hatására félbeszakíthatjuk a táblaszerkezet megadását. Mi az OK parancsgombot nyomjuk le.

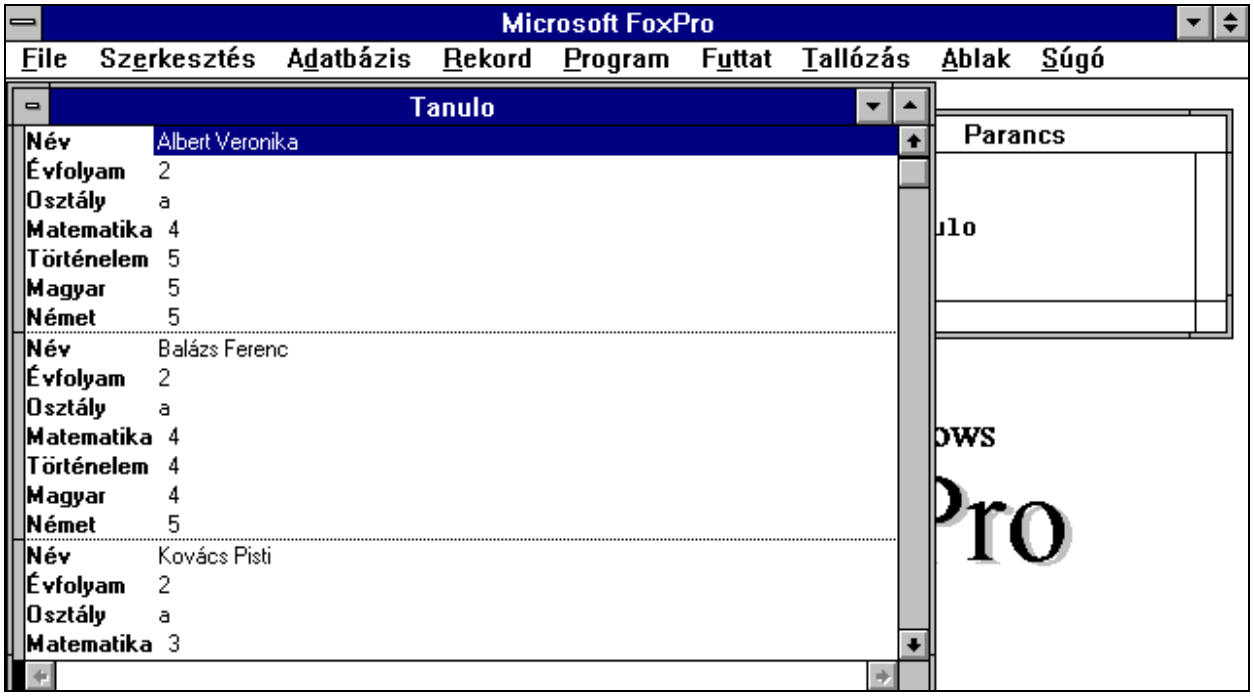
A megjelenő Mentés új néven ablakban a kívánt meghajtó (`G:`) és könyvtárnév (`ADATOK`) megadás után, az Adja meg a táblázat új nevét feliratú szövegmezőbe írjuk be a táblázat nevét: `tanulo`. A program a `.dbf` kiterjesztést automatikusan hozzáilleszti.

Mentés új néven ablak

A Létrehoz gombot választva a program végrehajtja a mentést, majd a következő ablak jelenik meg:

Párbeszédablak adatok beírása előtt

Azaz módunk nyílik az imént létrehozott táblázatot kitölteni adatokkal, ha az Igen gombot választjuk. Ha a Nem gombot választanánk a kurzor a parancsablakba kerülne, és táblázatunk szerkezete, fejléce közben eltűnne a képernyőről, de nem veszne el, mert el lenne mentve az előbb megadott fájlba. Az Igen parancsgombot választva, a megjelenő ablakban kitöltjük a táblázatot adatokkal.
Az ablak megnyílásakor az első rekordnak, azaz a táblázat első sorának adatait gépelhetjük be. Mihelyt beírtunk valamit, azonnal megjelenik a következő üres rekord képe, tehát az **új rekord hozzáfűzése automatikus**. A mezők között ENTER, kurzormozgató nyílak vagy egér segítségével közlekedhetünk. Minden beírt rekord azonnal az előbb kijelölt fájlba, táblázatba lesz kimentve, **nem kell külön elmentenünk**. Ha egy mezőt ESC billentyűvel hagyunk el úgy az adott mező tartalma elvész, és az ablak bezárul.

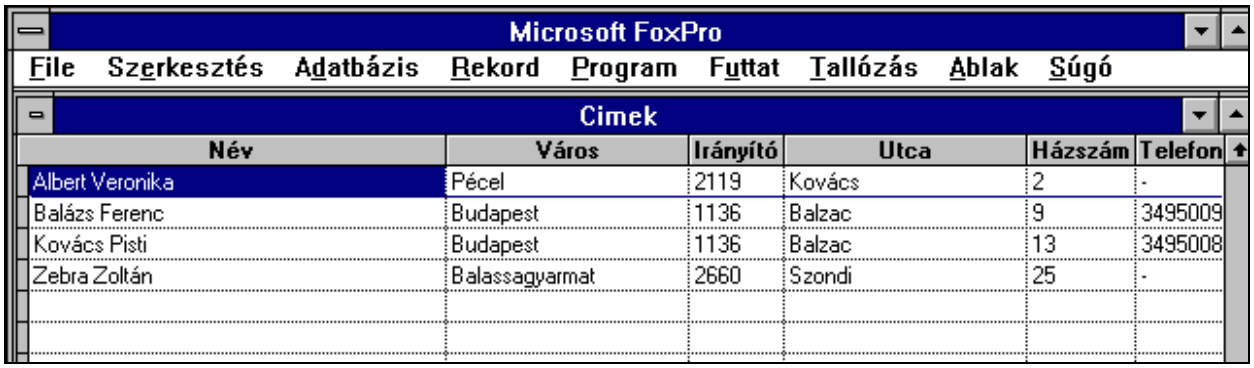


Adatbeírás

Ha begépeltük az összes adatot, befejezéskor kattintsunk kétszer a vezérlőmenü gombjára. Közben természetesen **a FoxPro automatikusan ment az előbb megadott fájlba**. Ezek után kiléphetünk a FoxPro programból a File főmenü Kilép almenüpontjával.

b; Táblázat készítése parancsablakban megadott parancsok segítségével.

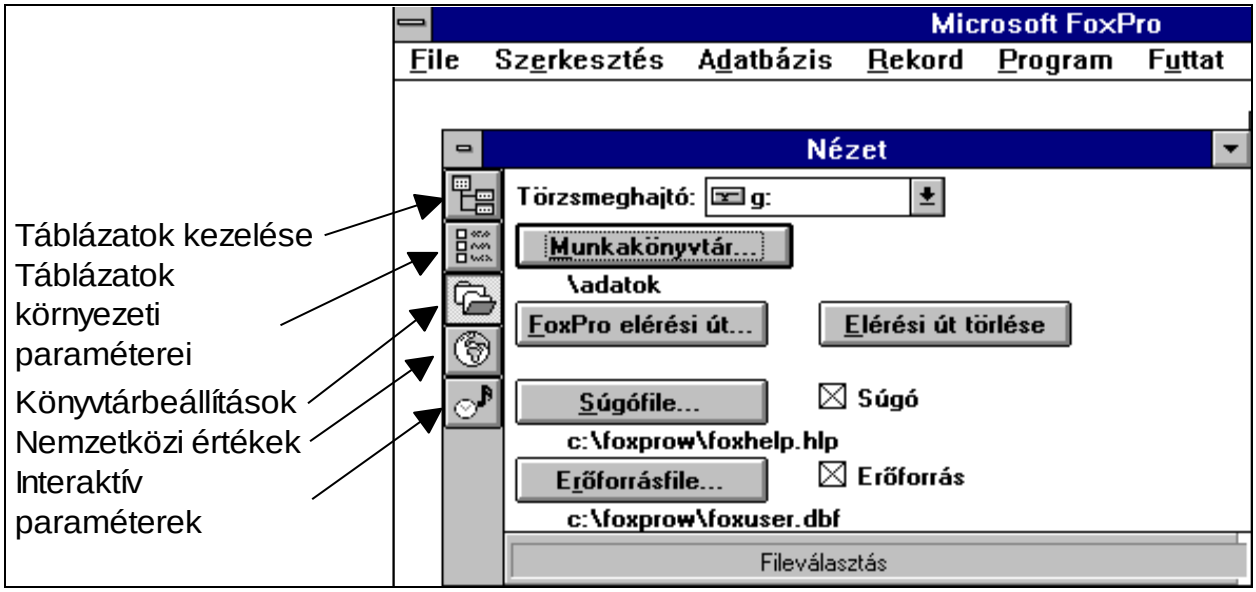
Készítsük el az alábbi táblázatot `cimek.dbf` néven a `G:` meghajtóra `ADATOK` nevű könyvtárba!



Név	Város	Irányító	Utca	Házszám	Telefon
Albert Veronika	Pécel	2119	Kovács	2	-
Balázs Ferenc	Budapest	1136	Balzac	9	3495009
Kovács Pisti	Budapest	1136	Balzac	13	3495008
Zebra Zoltán	Balassagyarmat	2660	Szondi	25	-

`cimek.dbf` tábla

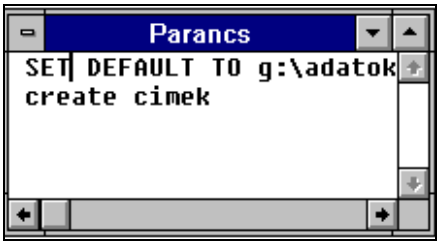
Mindenek előtt **állítsuk be a munkakönyvtárunkat** (`G:\ADATOK`), ahová dolgozni, menteni szeretnénk! Az Ablak főmenü Nézet almenüpontját válasszuk ki. Ez egy roppant hasznos menüpont, ismerkedjünk meg vele.



Nézet ablak (Ablak főmenü Nézet almenüpontja)

Kattintsunk a Könyvtárbeállítások gombra. Válasszuk a Munkakönyvtár gombot. A megjelenő párbeszédablakban állítsuk be a kívánt meghajtót és könyvtárat.(`G:\ADATOK`)

A táblázatot a `CREAT` paranccsal hozzuk létre. A parancs után megadjuk a létrehozandó táblázat nevét.



Ábra Hiba! A kapcsoló argumentuma érvénytelen.**. `cimek.dbf` tábla létrehozása paranccsal**

Ettől kezdve hasonló a művelet folytatása, mint az előbbi menüvel történő táblázat létrehozásnál. ENTER billentyű lenyomása után a Táblaszerkezet ablakban kell megadnunk a táblázat fejlécét, azaz a mezők nevét, típusát, méretét.

A `cimek.dbf` tábla szerkezete

Az OK parancsgombot használva lehetőségünk nyílik az adatok begépelésére, ha az “Akar most adatokat megadni ?” kérdésre igennel válaszolunk. Az **adatok hozzáfűzéséhez** a parancsablakba is **begépelhetjük az `APPEND` parancsot**, és ezután is kitölthetjük adatokkal a táblázatunkat

`USE` parancs beírásával bezárhatjuk a táblát.

`QUIT` parancs hatására kiléphetünk a FoxPro programból.

6.7. A táblázat sorainak listázása, nyomtatása

Korábban készített `tanulo.dbf` adattáblánk megtekintéséhez állítsuk be munkakönyvtárnak a `G:\ADATOK` könyvtárunkat. (Ezt az Ablak főmenü Nézet almenüjénél tehetjük meg.) A képernyőre történő listázáshoz először meg kell nyitni adattáblánkat, majd ki kell adni a listázásra vonatkozó parancsot.

Táblázat megnyitása parancsokkal:

`USE` filenév

Egy tábla sorainak megjelenítése parancsokkal:

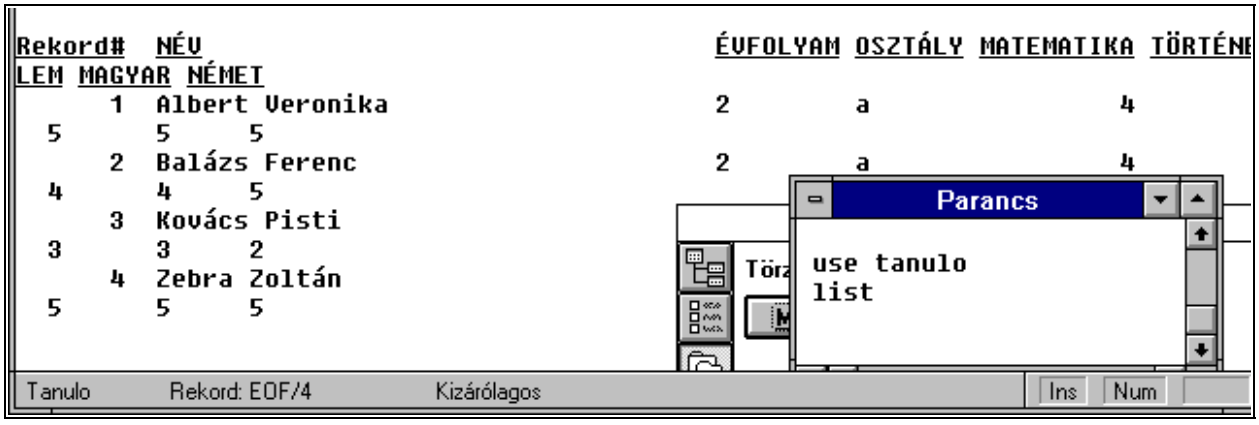
`USE` filenév

`LIST`

Példánknál be kell gépelni a parancsablakba:

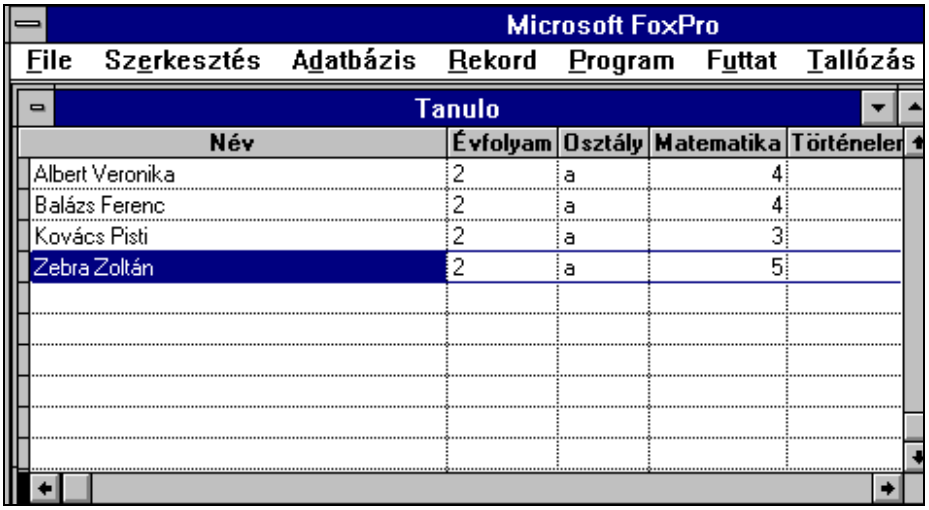
```
USE tanulo
LIST
```

ENTER billentyű leütése után megjelenik a táblázat, a rekordokon belül az egyes mezők egymás alatt.



Táblázat megjelenítése `LIST` paranccsal

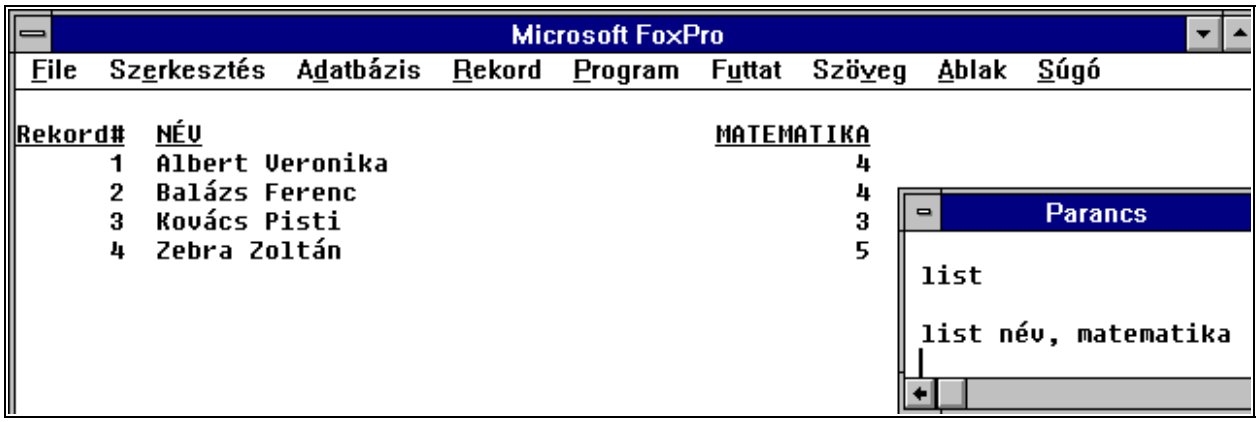
Ha a szokásos **táblázatos formá**t szeretnénk látni, akkor a Adatbázis főmenü Tallózás almenüjét kell választanunk, vagy be kell gépelnünk a parancsablakba a **`BROWSE`** parancsot. (A módosító ablakhoz való visszatérés a Tallózás főmenü Módosítás almenüjével lehetséges.)



Tallózás ablak

Ha csupán a tanulók nevére és matekjegyére vagyunk kíváncsiak, akkor a **`LIST` parancs után fel kell sorolnunk a megjeleníteni kívánt mezők neveit vesszővel elválasztva.**

```
USE tanulo
LIST név, matematika
```



Kiválasztott mezők megjelenítése

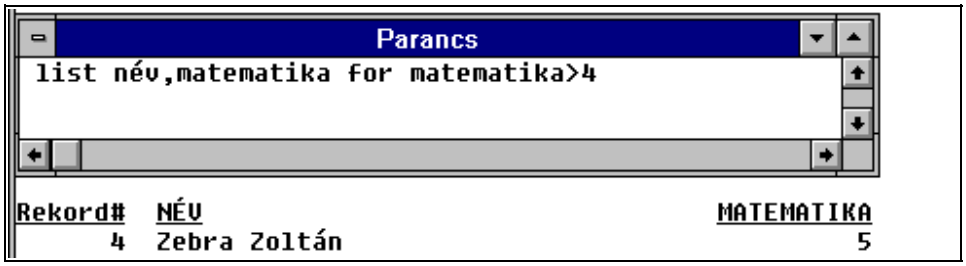
Hogyan jeleníthetnénk meg azon gyerekek listáját, amelyeknek 4 vagy annál jobb jegyük van matematikából? **Egy tábla adott feltételnek eleget tevő rekordjait, sorait a következő parancsokkal jeleníthetjük meg:**

`USE filenév`

`LIST  FOR feltétel`

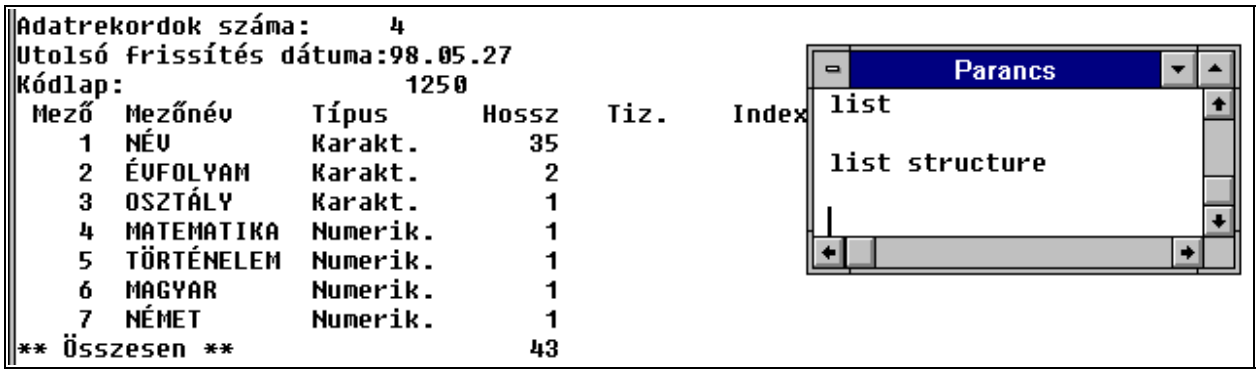
A fenti példánál, a 4-nél jobb matematika jeggyel rendelkező gyerekek listázásához az alábbi parancsokat kell kiadni:

```
USE tanulo
LIST  név, matematika FOR matematika>4
```



Lista feltétellel

Táblázat szerkezetének, azaz fejlécének listázására
`LIST STRUCTURE`
parancs használható.



Táblaszerkezet listázása

Lista kinyomtatása:
`USE filenév`
`LIST TO PRINT`

Feladat
460. oldal 158, 159,
461. oldal 169

6.8. A táblázat egy sorának megjelenítése:

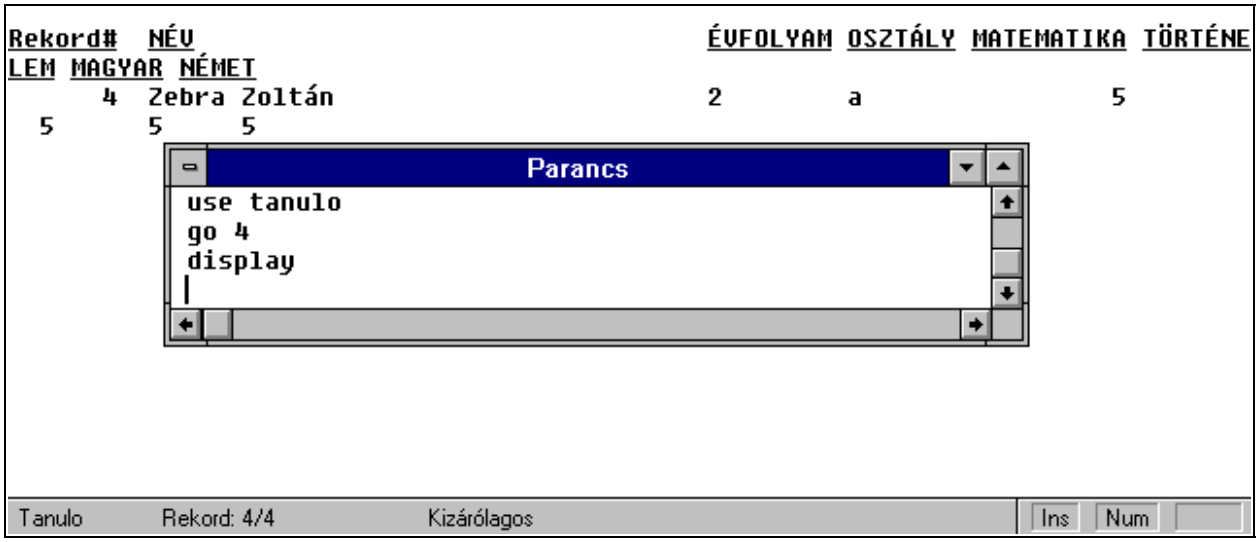
Nemcsak a táblázat egészét jeleníthetem meg, hanem egyetlen sorát, rekordját is. Először azonban meg kell ismernünk a rekordmutató fogalmát. **A rekordmutató mindig a táblázat aktuális sorára, rekordjára mutat.**
A rekordmutató által kijelölt sort megjeleníthetjük:
DISPLAY
utasítással.

A rekordmutatót a `GO` paranccsal mozgathatjuk.

`GO n`	n. rekordra lép
`GO TOP`	logikailag első rekordra lép
`GO BOTTOM`	logikailag utolsó rekordra lép
`SKIP n`	n-esével lépteti a rekordmutatót
`NEXT n`	az aktuális rekordmutatótól n-ig csinál valamit,
pl	LIST parancs után írva n db-ot listáz

Példa Listázzuk ki a `tanulo.dbf` táblázatfájl 4. sorát:

```
USE tanulo
GO 4
DISPLAY
```



Sor listázása

Feladat
435 oldal 1.
461 oldal 168.

Kidolgozott példa
463 oldal 177.
Feladat: Az ARU tábla az alábbi oszlopokat tartalmazza:
ARU(Cikkszam, Megnevezes, AR, Keszlet, Afa)
Hogyan készítene listát az áruk nevével és 10%-kal megemelt árakkal?
Megoldás:
A tábla megnyitása után először ki kell listázni a Megnevezés mezőt, majd Ar mezőt megszorozva 1,1-gyel. (Parancsoknál a tizedesvessző helyett alkalmazzunk pontot, hisz a vessző a mezők elválasztására szolgál. Adatbevitelnél használhatjuk a tizedesvesszőt.)
A válasz:

```
USE  aru
LIST megnevezes,ar*1.1
```

Feladat
462 oldal 175.
463 oldal 176-179.

6.9. Táblázat adatainak módosítása, szerkesztése, javítása, cseréje

Ha táblázatunk adatait utólagosan módosítani szeretnénk, akkor a táblázat megnyitása után ki kell adnunk a megfelelő parancsot:
Táblázat módosítása parancsokkal:

```
USE filenév
EDIT
```

vagy más módon:

```
USE filenév
BROWSE
```

Az ENTER gomb lenyomása után táblázatunkban egér vagy kurzormozgató billentyűk segítségével mozoghatunk, és a kívánt módosítást végrehajthatjuk. **Ha egy teljes oszlopot kell kicserélni**, akkor azt a következőképpen tehetjük meg:

```
USE filenév
REPLACE  ALL mezőnév WITH újérték
```

A mezőnév helyére a módosítani kívánt oszlop nevét kell írni, az újérték helyére pedig a mezők új tartalmát.

Kidolgozott példa:
454. oldal 117.
Feladat: Hogyan emelné meg minden áru árát 8 %-kal az alábbi táblában?
ARU(Cikkszam, Megnevezes, AR, Keszlet, Afa)

Válasz:

```
USE aru
REPLACE  ALL ar WITH ar*1.08
```

Feltételtől függő csere:

```
USE filenév
REPLACE ALL  mezőnév WITH újérték FOR feltétel
```

Kidolgozott példa:
455. oldal 127.
Feladat: A 10%-nál kisebb ÁFÁkat emelje meg 4%-kal az alábbi táblában:
ARU(Cikkszam, Megnevezes, AR, Keszlet, Afa)
Megoldás, válasz:

```
USE aru
REPLACE ALL afa WITH afa+4 FOR afa<10
```

Feladat:
454. oldal 118.-120.
455. oldal 121.-126.
456. oldal 128. 129.

6.10. Új sor felvétele a táblázatba parancsablakon keresztül

Új sor bevitelét, azaz új rekord fölvételét a táblázat megnyitása után az `APPEND` paranccsal is hajthatunk végre. Más módon is **fűzhetünk új rekordot táblázatunkhoz**:

```
USE filenév
APPEND BLANK
```

Ilyenkor az új rekord a többi rekord mögé íródik be.

Gyakorlati példa: új diák jön az osztályba: Tóth Imre.

Címe: Tóth Imre Budapest Zöld utca 9. 1173
Fel kell vennünk az adatait a `G:\ADATOK\cimek.dbf` fájlba. A módosítás végrehajtásához állítsuk be a munkakönyvtárat, majd gépeljük be a parancsablakba:

```
USE cimek
APPEND BLANK
```

ENTER leütése után begépelhetjük Tóth Imi adatait.
Hasonlóképpen módosítsuk a `tanulo.dbf` fájlt is. Tóth Imi minden jegye: 4.

6.11. Szerkezet módosítása:

Ha nem gondoltuk át tervezéskor az adattáblánk szerkezetét, vagy közben változtak a tényleges adatok pl. telefonszám 6 jegyűről 7 jegyűre változik, kénytelenek vagyunk megváltoztatni az adattáblánk szerkezetét, hiszen a 6 hosszúságú telefon nevű mezőbe nem férnek el a 7 jegyű számok. Tehát a szerkezet módosításakor megváltoztathatjuk a már meglévő táblázat oszlopait (név, mezőtípus, méret), törölhetünk belőle, vagy újat vehetünk fel. Törlésnél persze elveszítjük a törlendő oszlopban tárolt adatokat.

A szerkezet megváltoztatását

```
USE filenév
MODIFY STRUCTURE
```

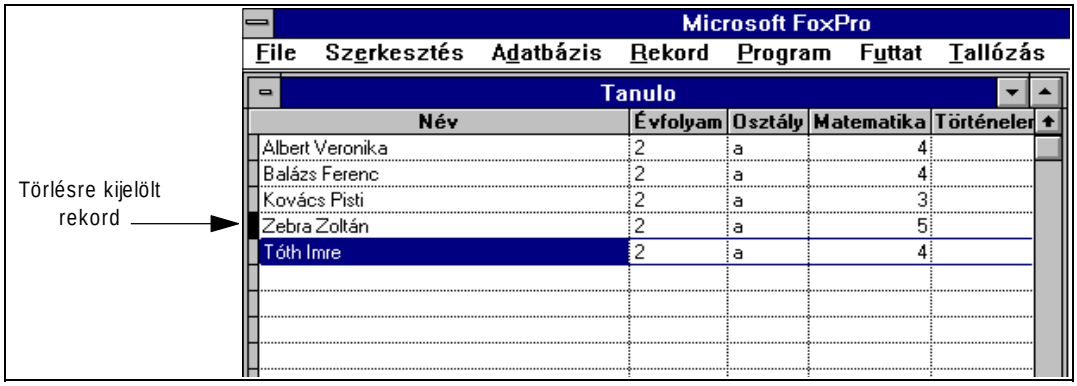
paranccsal tehetjük meg. ENTER lenyomása után Táblázatszerkezet ablak jelenik meg, ahol a kívánt módosításokat végrehajthatjuk.

6.12. Táblázat adatainak törlése

A törlés két lépésből áll. Először kijelöljük a törölni kívánt adatokat. (Ilyenkor az adat még visszaállítható.) Majd a kijelölés után fizikailag is törlünk. Gyakorlatilag az adatbázis-kezelő program átmásolja az egész adattáblát az üressé vált rekordok nélkül. Úgy ahogy karácsonyi vásárláskor Marcsi fölírja egy nagy lapra, hogy kinek mit akar venni, majd vásárlás közben kihúzza amit már megvett. Ha nem sikerült mindent megvenni, akkor a nagy lapról egy kisebbre írja a meg nem vásárolt árukat, a folytatáshoz.
a, A FoxPro programban a törlést a következő módon lehet végrehajtani a menü segítségével:

6.12.1. **Kijelölés:**

Aktuális rekordot (azaz azt a rekordot melyen a kurzor áll) törlésre kijelölni a Tallózás főmenü Törlés ki/be almenüjével (vagy CTRL +T billentyűparanccsal) lehet. **A kijelölést az első mező előtti oszlop fekete sávja jelzi.** A kijelölést visszavonni szintén a fenti menüpontoknál lehet. Egér segítségével is ki tudunk jelölni, ha az első mező előtti oszlopra kattintunk.

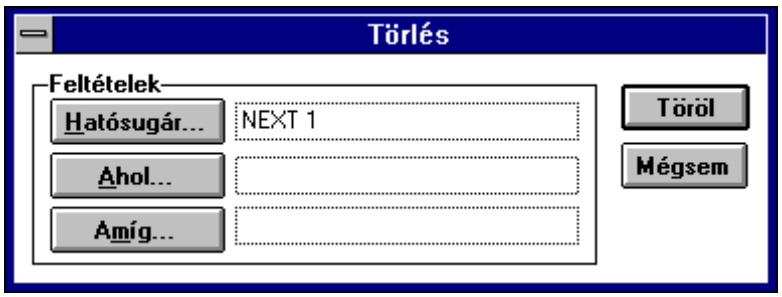


Kijelölés törlésre

6.12.2. **Törlés fizikailag:**

A kijelölt rekordokat az Adatbázis főmenü Tömörítés almenüjével lehet letörölni, ha a megjelenő "Tömöríti az alábbi táblázatot?" kérdésre Igennel válaszolunk. **A törlés visszafordíthatatlan, a letörölt rekordok végleg elvesznek.**

Ha nagyon sok rekordot szeretnénk törölni, melyek valamilyen feltételnek tesznek eleget, akkor nem érdemes egyesével kijelölni őket, hanem a Rekord főmenü Törlés almenüjénél



Rekord főmenü Törlés almenü Törlés ablaka

Hatósugár parancsgomb lenyomása után a Minden opciógombot válasszuk, majd az OK gomb lenyomása után visszatérünk az előző ablakba. A törlés feltételeit az Ahol nyomógomb lenyomása után írhatjuk be. Pl törölni szeretnénk azokat a tanulókat a `cimek.dbf` fájlból, akik Budapesten laknak, azaz a város mezőben Budapest szerepel. Ahol nyomógomb lenyomása után megjelenő ablakban, a középen levő Szerkesztési tartományba be kell gépelni Város="Budapest" sort majd OK gombot nyomjuk le. A Törlés nyomógombbal a kijelölés

megtörténik. (A Rekord főmenü Visszahívás almenüjénél még visszavonható a kijelölés.) Az Adatbázis főmenü Tömörítés almenüjénél a kijelölt rekordok fizikailag is törölhetők a fájlból.
b, Aktuális rekord törlésre jelölése parancsokkal:
`DELETE`

A törlésre való kijelölés csak a rekordmutató mozgatása után válik láthatóvá.
Aktuális rekord törlésből való visszahívása parancsokkal:
`RECALL`

Fizikai, végleges törlés:
`PACK`

Minden rekordot töröl
`ZAP`

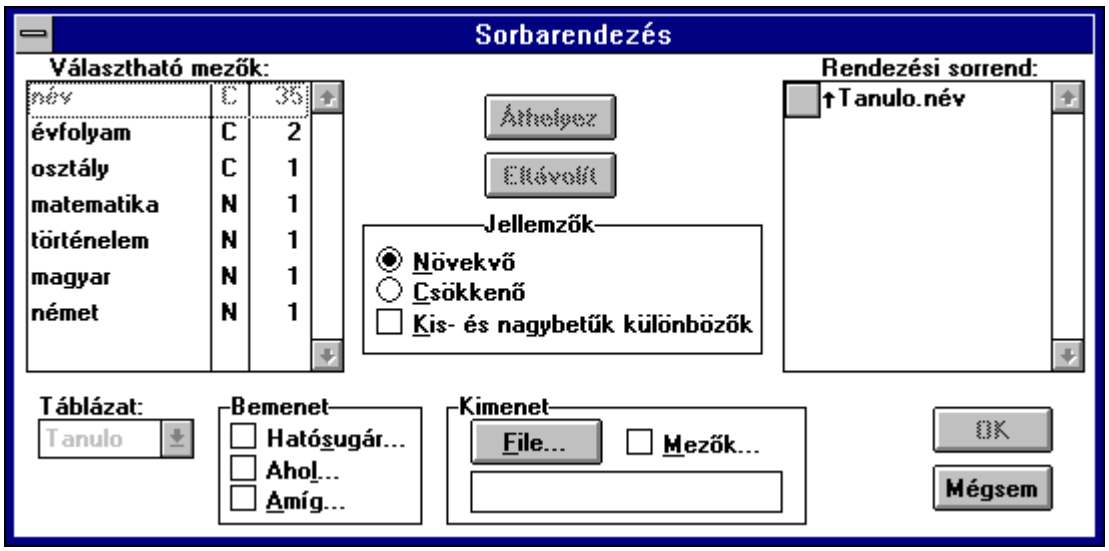
Ha az összes rekordot eltávolítjuk a fájlból, akkor csak a táblázat fejléce marad, mintha most hoztuk volna létre.
Feladat
453. oldal 112. 113.
456. oldal 130.

6.13. Táblázat sorainak rekordjainak sorba rendezése

Két módja van a rendezésnek: fizikai, logikai. A sorba rendezett táblázat áttekinthetőbbé válik, gyorsabb a lekérdezés és a keresés. Bizonyos szempontok szerint **lehet növekvően ill. csökkenően rendezni egy táblázatot.** Ha pl. valamilyen karakteres mező szerint rendezünk növekvően, akkor az az ABC-nek megfelelő rendezés lesz. Csökkenő sorrend esetén viszont pont a fordítottja ZS, Z, Y stb. sorrendű.

6.13.1. **Fizikai rendezés**

Pl. névsor szerint rendezett táblázat, pontosabban mondva, név mező szerint növekvően rendezett.(Pl. matekjegy szerint csökkenően.) Tudnunk kell azonban, hogy **egy táblázatot tartalmazó fájl csak egy sorrend szerint lehet fizikailag rendezett. Ha új rendezést szeretnénk, akkor az egész táblázatot át kell másolni a rekordok elmozgatásával egy új fájlba a megfelelő sorrendnek megfelelően.** Így meglehetősen hosszú idő alatt áll elő egy új sorrend. Tehát új fájl jön létre, más néven. **Előnye viszont, hogy a rekordmutató gyorsabban mozog, mint a 2. módszerben.**
a, A megnyitott táblázatfájl sorba rendezéséhez az Adatbázis főmenü Sorbarendezés almenüpontjánál megjelenő ablakban állítjuk be, hogy mi szerint rendezzünk. A Választható mezők listamezőből kiválaszthatjuk, azt a mezőt, mely szerint rendezni szeretnénk.
Gyakorlati példa: Rendezzük a `tanulo.dbf` fájlt név mező szerint növekvően.



Sorbarendezés ablak (Adatbázis főmenü Sorbarendezés almenüénél)

A táblázatfájl megnyitása után az Adatbázis menü Sorbarendezés almenüjénél a Választható mezők listájából válasszuk a név mezőt, majd az Áthelyez parancsgomb lenyomásával áttesszük a Rendezési sorrend panelbe. Ezek után válasszuk a Növekvő opciót. Majd a mentéshez nyomjuk le a Kimenet panelben levő File parancsgombot, és a megjelenő Mentés Új néven ablakban adjuk meg a file nevét, melybe az új sorrendű táblázat belekerül. Pl `tanulnev.dbf`

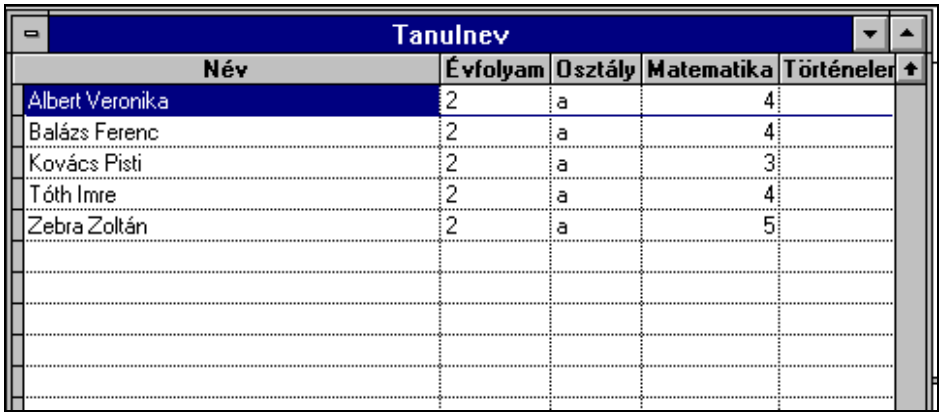
b, Mindezt parancsokon keresztül is megadhatjuk. A táblázat megnyitása után a **rendezéshez:**

`SORT ON kulcs TO filenév`

kulcs helyére azon mezőnév kerül, mely szerint rendezni szeretnénk.
filenév helyére pedig az új sorrendet tartalmazó fájl neve.

A fenti példában:

```
USE tanulo
SORT ON név TO tanulnev
```

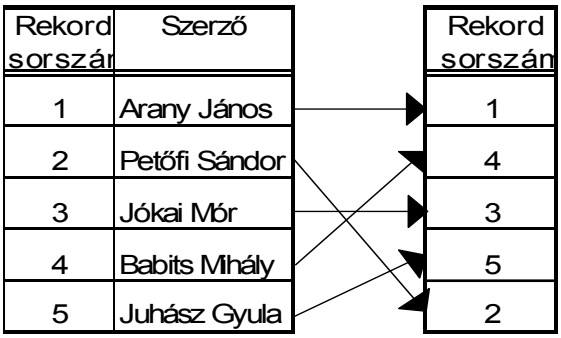


Fizikailag rendezett táblázat név mező szerint növekvően

6.13.2. **Logikai rendezés, indexelés avagy a gyors rendezés trükkje.**

Peti rendet akart tenni a család könyvszekrényében. Szépen névsorba rakta a magyar irodalom képviselőit, külön polcra helyezte a világirodalom remekeit. Sajnos túl sok volt a könyv, a szekrény pedig kicsi, így kénytelen volt a könyveket több sorba rakni. Viszont így csak a könyvek felét látta, hisz egyik eltakarta a másikat. 1 napi munka után nem volt elégedett, újból kiszedegette a könyveket, megpróbálta a nagyobbakat hátra tenni, a kisebbeket pedig előre. Azonban kevés nagy könyv volt, annál több kicsi, nem sikerült úgy rendezni, hogyha keresett valamit gyorsan megtalálja. Úgy döntött katalógust készít. Minden könyvről 1-1 cédulát írt. A cédulára ráírta a szerzőt, könyv címét, kiadásának évét, témáját és helyét a könyvszekrényben. A cédulákat sorba rakta szerzők szerint. Így később, amikor adott szerzőtől keresett könyvet, hamar megtalálta. Előfordult azonban, hogy csak egy könyv címére emlékezett, a szerzőt viszont elfelejtette, így csak hosszas keresgélés után akadt rá a kívánt könyvre. Ekkor úgy határozott, hasonlóképpen jár el, mint ahogy a könyvtárakban szokás. Lemásolta a cédulákat, és egy másik dobozba cím szerint rendezte őket. Később újabb igényei támadtak. Látni szerette volna, hogy egy adott témában milyen könyvekkel rendelkezik ...
Ez ugyan elég fáradságos munka volt, mégis könnyebb, mint a könyveket átrendezni. **A logikai rendezésnek, más néven indexelésnek, az a lényege, hogy a rekordok tényleges fizikai sorrendjétől függetlenül kialakíthatunk tetszőleges sorrendet, oly módon hogy létrehozunk egy ún. indexet, melyben a rekordok, sorok a kívánt sorrendben szerepelnek. Valójában az ún. indextábla 1 oszlopból áll, ez az oszlop a rekord fizikai sorszámát tárolja a logikai sorrendnek megfelelően.**

Fizikai sorrend Index, szerző szerint növekvően



Az indexelés lényege

Az indexelés előnyei:

- **Kisebb helyen tárolható egy indextábla, mintha az egész állományt le kéne másolni egy új fájlba, mint a fizikai rendezésnél.**
- **Egy fájlhoz több index is készíthető**
- **Egy index elkészítése gyorsabb mintha fizikailag rendeznénk a fájlt**

Indextáblát új vagy meglévő táblázathoz is készíthetünk. Új táblázat esetén a File főmenü Új almenüjénél Táblázat opciónál, vagy meglévő táblázat esetén a

```
USE filenév
MODIFY STRUCTURE
```

Táblázat szerkezet ablak

parancsok kiadása után a Táblázat szerkezet ablakban.

A CDX feliratú oszlop alatt kis négyzetek jelölik az indexet. Ha egyszer rákattintunk egérrel a négyzetre, felfelé mutató nyíl (↑) jelenik meg, mely növekvő sorrendet jelöl. Kétszeres kattintás hatására lefelé mutató nyíl (↓) jelenik meg, mely csökkenő sorrendet jelöl. Ha újra rákattintunk, akkor a négyzet üressé válik, jelezve, hogy azon mező szerint nem készítünk indexelést.

Ha eldöntöttük, mely mezők szerint kívánunk indexelni, az OK gomb lenyomása után a táblázatfájl nevével megegyező nevű ún. **összetett indexfájl keletkezik, `.cdx` kiterjesztéssel. Ebben az egy fájlban több indextábla is szerepelhet.**

A kulcsmezőre mindig készítsünk indexelést.

Gyakorlati Feladat:

Készítsünk indexet a `tanulo.dbf` állományhoz, melyben a rekordok név mező szerinti növekvő sorrendben szerepelnek.

Megoldás:

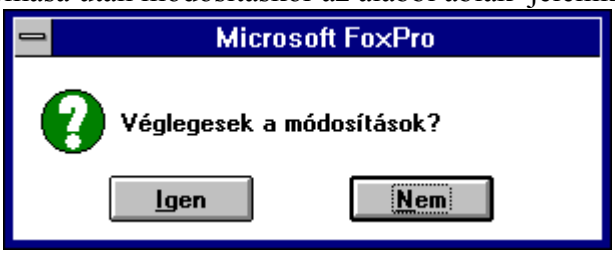
a, A munkakönyvtár beállítása után (Ablak főmenü, Nézet almenü, Könyvtárbeállítások ikon), gépeljük be a parancsablakba:

```
USE tanulo
MODIFY STRUCTURE
```

majd ENTER billentyű leütése után állítsuk be a következőket:

Ábra Hiba! A kapcsoló argumentuma érvénytelen.**. Indexelés a Táblázatszerkezet ablakban**

OK parancsgomb lenyomása után módosításkor az alábbi ablak jelenik meg:



Párbeszédablak

A kérdésre válaszoljunk Igennel. Érdemes az indexelést mindig a táblázat létrehozásakor készíteni.

b, Indextábla készítése parancsokon keresztül:

`INDEX ON kulcs TAG indexnév`

kulcs helyére azon mezőt vagy mezőket írjuk, mely szerint létrejön az indexelés,
indexnév helyére az indextábla neve kerül, mely lehetőleg hasonlít a mező nevére.

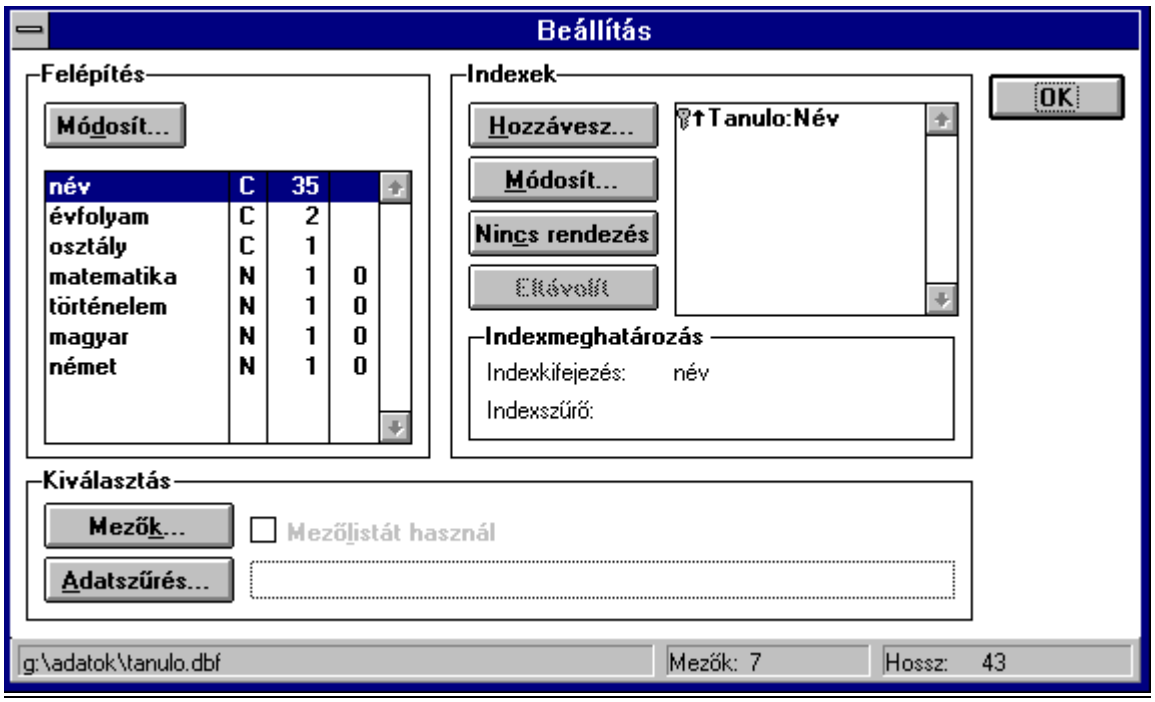
A fenti gyakorlati példa megoldása:

```
USE tanulo
INDEX ON név TAG tannev
USE
```

Ezáltal létrejön egy `tanulo.cdx` fájl, benne egy indextábla, melyben a rekordok sorszáma a név mező szerinti sorrendet tükrözi.
Van egy fontos tudnivaló az indexállománnyal kapcsolatban. **Nemcsak 1 mező szerint indexelhetünk, hanem több szerint is.**

Adat módosításkor (pl. új rekord fölvétele) a megnyitott indexállomány automatikusan frissítődik, nem kell róla külön gondoskodnunk. **Amikor egy táblázatfájlt megnyitunk automatikusan megnyitódik a vele azonos nevű indexfájl is. Egyszerre csak 1 féle indexelés lehet aktív.** Hisz amikor használunk egy táblázatot, annak sorai csak egyféle sorrendbe sorakozhatnak. **Az aktív indexet mesterindexnek nevezzük**. Természetesen egy index aktívvá tétele után megváltozhat a logikai sorrend a fizikai sorrendhez képest, pl. más lesz a logikailag első rekord, ennek megfelelően a rekordmutató is máshová mutat. **Egy táblázatfájl megnyitásakor nem aktív egyik index sem, erről külön kell gondoskodnunk.**

Nyissuk meg a kívánt táblázatot az Ablak főmenü Nézet almenüjénél. Először a munkakönyvtárat állítsuk be a Könyvtárbeállítások ikonnál. Majd kattintsunk a Táblázatok kezelése ikonra. A Megnyitás parancsgomb lenyomása után válasszuk ki a megnyitandó fájlt, és lenyomjuk a Megnyit parancsgombot. Majd visszatérve az előző ablakhoz, nyomjuk le a Beállítás parancsgombot.



Mesterindex beállítása Ablak főmenü Nézet almenü Beállítás parancsgombnál

Indexek panelen belül kiválasztjuk a megfelelő indexet mesterindexnek, és lenyomjuk a Rendezés parancsgombot. Ezek után OK parancsgombbal távozhatunk az ablakból. A mesterindexet a neve előtti kis kulcs jelzi.

Kidolgozott példa
463. oldal 181.
Hogyan rendezné a VEVOK táblát, ha városonkénti névsorra lenne szüksége?
VEVOK(Azonosító, Utca, Nev, Varos, Utca, Szamlaszam)
Megoldás:

```
INDEX ON varos+nev TAG vanev
```

Mivel városonként kell rendezni, a város mező szerepel először és ehhez kell + jellel kapcsolni a második mezőt, esetünkben a név mezőt.
Egyre azonban ügyelnünk kell. **Ha a kulcskifejezésben szereplő mezők nem mindegyike karakteres, akkor azokat a megfelelő függvénnyel karakteressé kell alakítani.**

Feladat
463. oldal 180.
463. oldal 182.
464. oldal 183.-185.
464. oldal 187.-188.

Kidolgozott példa
464. oldal 190.
Feladat Milyen műveletet kell elvégeznie, ha egy cég dolgozói közül a tíz legjobban keresőt kell kilistáznia?

Megoldás:
Először megnyitjuk a táblázatot Ablak főmenü Nézet almenüjénél a Táblázatok kezelése ikonnál, Megnyit parancsgombbal. Majd a Beállít parancsgomb benyomása után mesterindexnek választjuk a Fizetés mezőt és Csökkenő sorrendbe rendezzük. Ezek után a logikailag első rekordra állítjuk a rekordmutatót, és kilistázzuk az első tíz rekordot. Ennek érdekében begépeljük a parancsablakba:

```
GO TOP
LIST NEXT 10
```

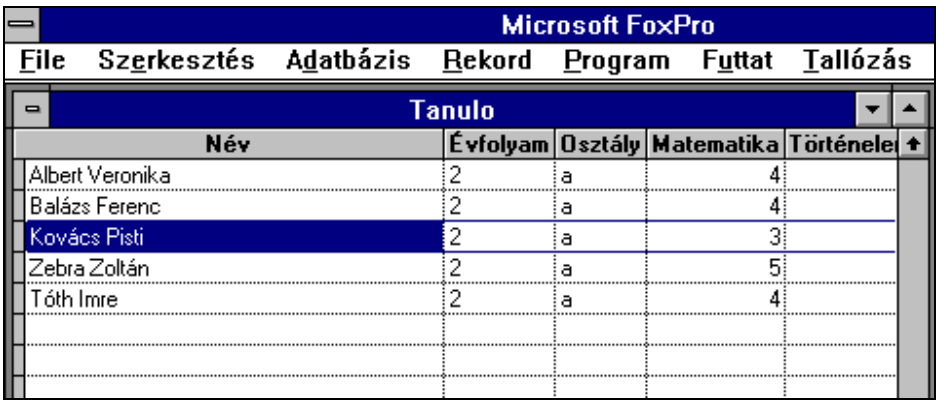
Feladat
465. oldal 191.-195.

6.14. Keresés

Peti rendkívül gyorsan megkeresi pl. Arany János műveit, hisz előveszi azt a dobozt, melyben a könyvekről szóló cédulák szerző szerint vannak sorba rendezve. Ha az Aranyember c. könyvet keresné, az sem volna nehezebb, hisz azt a dobozt kell elővennie, ahol könyvcímek szerint rakta sorba a könyvcédulákat. Amikor viszont Vercsi átjön a szomszédból, és nézegeti a könyvszekrényt, Peti dobozait azonban nem látja, csak hosszas keresgetés után találja meg a fenti könyveket. Hisz a könyvek rendezetlenül vannak a szekrényben, legalábbis Vercsi számára. Természetesen a számítógép is könnyebben keres egy rendezett táblázatban, mint egy rendezetlenben. Persze ez sem reménytelen.

6.14.1. **Keresés rendezetlen táblázatban:**

A keresés rendezetlen táblázatban azt jelenti, hogy a rekordmutató az általunk megadott feltételnek eleget tevő első rekordra, sorra áll, ha van ilyen, ha pedig nincs, akkor a file-vége jelre. A számítógép összehasonlítja az általunk megadott feltételt a mező tartalmával, a feltétel első karakterétől az utolsóig, nagy és kisbetűket is megkülönböztetve. Ha egyezést talál valamelyik rekordnál, a rekordmutató ott marad a keresett rekordon.
A keresési feltételben meg kell adni a keresendő mező nevét, majd egyenlőség jel után (=) meg kell adni a keresett értéket.
Példa: Olyan személyeket keresünk, melyek neve "Ko" karakterekkel kezdődik az alábbi táblázatban.
Keresési feltétel: NÉV="Ko"

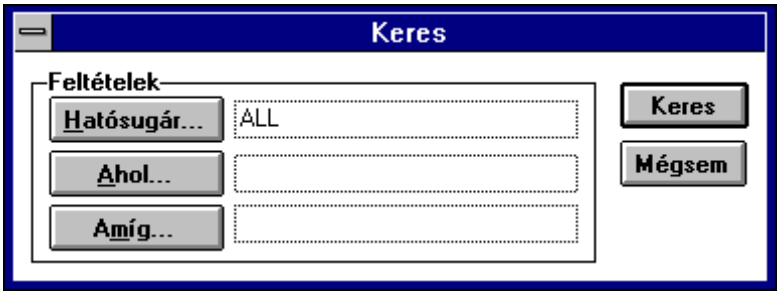


Név	Évfolyam	Osztály	Matematika	Történele
Albert Veronika	2	a	4	
Balázs Ferenc	2	a	4	
Kovács Pisti	2	a	3	
Zebra Zoltán	2	a	5	
Tóth Imre	2	a	4	

Keresés eredménye

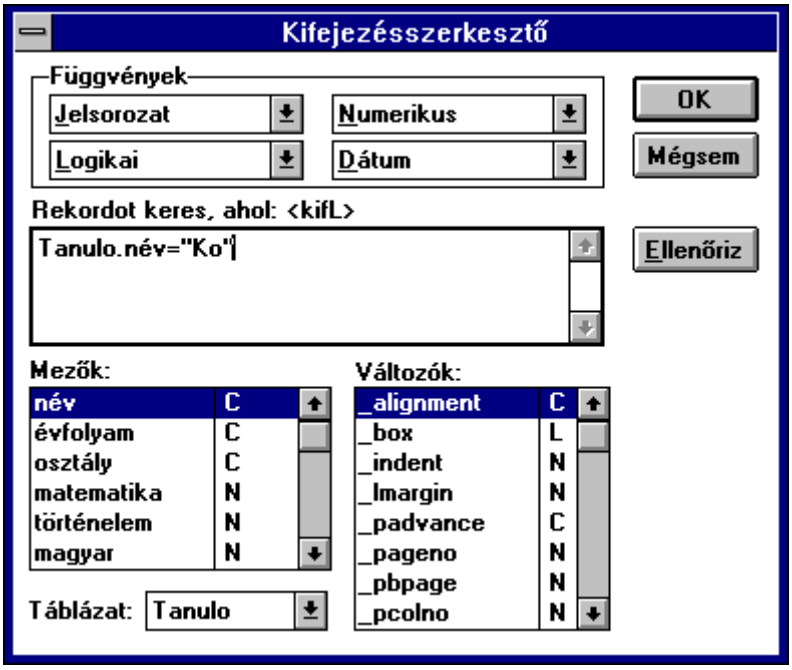
A keresés eredménye az lesz, hogy a rekordmutató a 3. rekordra fog mutatni, hisz Kovács Pisti megfelel a feltételnek. Természetesen a keresést lehet folytatni, egészen a file-vége jelig.

Gyakorlati példa:
Végezzük el a fenti keresést a `G:\ADATOK\tanulo.dbf` fájlra.
a. Megoldás menüvel: A munkakönyvtár beállítása és a fájl megnyitása után listázzuk ki táblázatunkat táblázatos formába a `BROWSE` paranccsal. A Rekord főmenü Keresés almenüpontját



Rekord főmenü Keresés almenüpont Keres ablak

választva a megjelenő párbeszédablakban válasszuk az Ahol parancsgombot.



Kifejezésszerkesztő ablak

A megjelenő ablakban a Rekordot keres ahol panelbe gépeljük be a keresési feltételt, azaz Név="Ko" szöveget. (A Mezők listából is válogathatunk mezőket kétszeres kattintással.) Aztán nyomjuk le az OK parancsgombot, és visszatérve az előző ablakba, a Keres parancsgombbal indíthatjuk a keresést. A rekordmutató a kívánt rekordara lép.
Ha több rekord van mely eleget tesz a keresési feltételnek, folytathatjuk a további keresést a Rekord főmenü Folytat almenüjénél. A rekordmutató a következő keresési feltételnek eleget tevő rekordra áll. Ha nincs több rekord, a státus sorban a "Keresési hatósugár vége" felirat jelenik meg mert a rekordmutató elérte a file-vége jelet.

b. Keresés parancsokon keresztül
USE file-név
LOCATE FOR feltétel

Fenti példa parancsokkal megadva:
```
USE tanulo
LOCATE FOR Név="Ko"
```

Folytatás parancsa:
CONTINUE

6.14.2. **Gyors keresés rendezett, indexelt táblában**

A keresés rendezett, indexelt táblában azt jelenti, hogy a rekordmutató az általunk megadott feltételnek eleget tevő első rekordra, sorra áll, ha van ilyen, ha pedig nincs, akkor a hozzá leginkább hasonlóra. Így keres pl a szótárgép is: Ha beírjuk a "lenni" szót, de ilyet nem talál a gép, akkor a hozzá legközelebb állót fordítja le, esetünkben a "lenn" szót németre, az eredmény: "unten".
Ezt a fajta keresést nem lehet folytatni. Nem is kell, hisz rendezett táblázatról lévén szó, az összes, feltételnek eleget tevő rekord ott látható a kijelölt sor után. Ezek után kurzormozgató billentyűk segítségével léphetünk át a következő rekordra. **Mivel ez a keresés kihasználja a tábla rendezettségét, nagyságrendekkel gyorsabb az előzőnél.**

a. Oldjuk meg az előző példát gyorskereséssel, menün keresztül:

A táblázat megnyitása után aktívvá kell tennünk a név mező szerinti indexelést. Aztán megjelenítjük a táblázatot `BROWSE` paranccsal, majd a Rekord főmenü Gyorskeresés almanüjénél a szerkesztési tartományba írjuk be a keresési feltételt:
"Ko"
OK parancsgomb használata után a rekordmutató a keresett rekordra áll.

b. Gyorskeresés parancsa:

SEEK

7. Adatmodellek, adatbázis szerkezetek

7.1. Több tábla használata

Egy középiskolában számon tartják a tanulók korábbi iskoláit, általános iskoláit is. Pl. azért hogy a felvételi értesítéseket elküldjék az iskolákba. Ki kell tehát bővítenünk a `cimek.dbf` táblázatot 5 oszloppal, mégpedig a korábbi iskola nevével, helységnévvel, és címével.

File Szerkesztés Adatbázis Rekord Program Futtat Tallózás Ablak Súgó

Név	Város	Irányító	Utca	Házsz	Telefon	Isknév	Iskváros	Iskutca	Isk	Iskirs
Albert Veronik	Pécel	2119	Kovács	2	-	Szemere Pál Ált. Isk	Pécel	Szemere	15	2119
Balázs Ferenc	Budapest	1136	Balzac	9	3495009	252. Ált. Isk.	Budapest	Balzac	3	1136
Kovács Pisti	Budapest	1136	Balzac	13	3495008	252. Ált. Isk.	Budapest	Balzac	3	1336
Zebra Zoltán	Balassagyarma	2660	Szondi	25	-	Dózsa György Ált.Isk.	Balassagyarmat	Dózsa	3	1136
Tóth Imre	Budapest	1173	Zöld	9	-	Zöld Ált. Isk.	Budapest	Zöld	5	1173

Tábla túl sok adattal

Képzeljük el, hogy az iskolába 900 tanuló jár, és ebből 150 ugyanabból az iskolából jött. A táblázatunk sok adatismétlődést tartalmaz. Roppant fárasztó 150-szer ugyanazt leírni. Sokkal célszerűbb volna egy másik táblázatot létrehozni, melyben az iskolák adatait tárolnánk (isknév, iskváros, iskutca, iskházszám, iskirszám). Pl így:

Microsoft FoxPro - Iskola

File Szerkesztés Adatbázis Rekord Program Futtat Tallózás

Isknév	Iskváros	Iskutca	Iskházszám	Iskirszam
Szemere Pál Ált. Isk	Pécel	Szemere	15	2119
252. Ált. Isk.	Budapest	Balzac	3	1336
Dózsa György Ált.Isk.	Balassagyarmat	Dózsa	3	1136
Zöld Ált. Isk.	Budapest	Zöld	5	1173

A hiányos `iskola.dbf` tábla

Miért pont ezeket az adatokat tettük egy új táblázatba? Milyen kapcsolat van az első táblázatbeli adatok között? Egy gyermeknek közvetlen, szoros jellemzője, tulajdonsága a neve, sőt a lakcíme is. Az is fontos, hogy melyik iskolából jött. Az iskola címe azonban nem a gyermek jellemzője, hanem az iskoláé. **Igyekeznünk kell olyan táblázatokat létrehozni, ahol egy rekordban csak az egyed közvetlen tulajdonságai vannak.** Egyik táblázatban a gyerekek jellemzői, másikban az iskolák jellemzői. Így az az ötletünk támadhat, hogy a címek táblázatból hagyjuk el az iskolák adatait, csak a nevük szerepeljen. Ez azonban nem jó, hisz lehet Petőfi Sándor Általános Iskola Budapesten és Pécelen is, emellett még hosszú is ez a mező, túl sok helyet foglal. De honnan tudjuk így, hogy melyik gyermek melyik iskolából jött? A legegyszerűbb megoldás, ha az `iskola.dbf` táblát kibővítjük még egy oszloppal,

mellyel mesterséges kulcsként sorszámozzuk az egyes iskolákat. A `cimek.dbf` táblázatban, melynek mostantól legyen inkább `cimisk` a neve. (Adatbázisfájlt nem lehet átnevezni, vagy más néven kimenteni. Ha ugyanannak a táblának más nevet kívánunk adni, akkor át kell másolnunk pl. Norton Commander programmal egy más nevű fájlba!) Ne tároljunk mást az iskolákról, csupán a megfelelő iskolák sorszámát. Így a két táblázat sokkal kevesebb helyet foglal, a létrehozása is egyszerűbb.

Microsoft FoxPro - Cimisk

File Szerkesztés Adatbázis Rekord Program Futtat Talló

Név	Város	Irányító	Utca	Házszám	Telefon	Iskkód
Albert Veronika	Pécel	2119	Kovács	2	-	1
Balázs Ferenc	Budapest	1136	Balzac	9	3495009	2
Kovács Pisti	Budapest	1136	Balzac	13	3495008	2
Zebra Zoltán	Balassagyarmat	2660	Szondi	25	-	3
Tóth Imre	Budapest	1173	Zöld	9	-	4

`cimisk.dbf`

Microsoft FoxPro - Iskola

File Szerkesztés Adatbázis Rekord Program Futtat Tallózás

Iskkód	Isknév	Iskváros	Iskutca	Iskházszám	Iskirszam
1	Szemere Pál Ált. Isk	Pécel	Szemere	15	2119
2	252. Ált. Isk.	Budapest	Balzac	3	1336
3	Dózsa György Ált.Isk.	Balassagyarmat	Dózsa	3	1136
4	Zöld Ált. Isk.	Budapest	Zöld	5	1173

Az optimális `iskola.dbf` tábla

Mostantól **egyetlen oszlop, tulajdonság van, mely mindkét táblázatban szerepel, és az egyik táblázatban kulcsmező. Ezt a mezőt kapcsoló mezőnek nevezzük, hisz biztosítja a kapcsolatot egy adatbázis két táblázata között.**
Vérszemet kapunk és korábbi táblázatainkat is helyre hozzuk.

Microsoft FoxPro - Tanulo

File Szerkesztés Adatbázis Rekord Program Futtat Ta

Név	Évfolyam	Osztály	Matematika	Történelem	Magyar	Német
Albert Veronika	2	a	4	5	5	5
Balázs Ferenc	2	a	4	4	4	5
Kovács Pisti	2	a	3	3	3	2
Zebra Zoltán	2	a	5	5	5	5
Tóth Imre	2	a	4	4	4	4

`tanulo.dbf`

A `cimisk.dbf` és a `tanulo.dbf` táblázatban is szerepel a gyerekek neve. Ez nemcsak adatismétlődés, de hiba lehetőség is lehet. Pl. egyik táblába rosszul gépeljük be a nevet. Próbálkozzunk megint a jól bevált mesterséges azonosítóval. A `tanulo.dbf` fájlban hozzunk létre egy új oszlopot, ahol sorszámozzuk a gyerekeket. A tulajdonság neve legyen tankód, a tanuló kód rövidítéseként. A `cimisk.dbf` táblázatnak módosítsuk a szerkezetét,

illesszünk hozzá egy oszlopot tankód névvel. Gépeljük be a tanulók megfelelő kódjait, majd szerkezet módosítással töröljük a tanulók nevét tartalmazó oszlopot.

Microsoft FoxPro - Cimisk

File Szerkesztés Adatbázis Rekord Program Futtat Tall

Tankód	Város	Irányító	Utca	Házszám	Telefon	Iskkód
1	Pécel	2119	Kovács	2	-	1
2	Budapest	1136	Balzac	9	3495009	2
3	Budapest	1136	Balzac	13	3495008	2
4	Balassagyarmat	2660	Szondi	25	-	3
5	Budapest	1173	Zöld	9	-	4

Az optimális `cimisk.dbf` tábla

Microsoft FoxPro - Tanulo

File Szerkesztés Adatbázis Rekord Program Futtat Tallózás A

Tankód	Név	Évfolyam	Osztály	Matematika	Történelem	Magyar	Német
1	Albert Veronika	2	a	4	5	5	5
2	Balázs Ferenc	2	a	4	4	4	5
3	Kovács Pisti	2	a	3	3	3	2
4	Zebra Zoltán	2	a	5	5	5	5
5	Tóth Imre	2	a	4	4	4	4

Az optimális `tanulo.dbf` tábla

Ezt a két táblázatot a tankód kapcsoló mezővel tudom **összekapcsolni, más szóval összeilleszteni.**
Elértük tehát, hogy **adataink több táblázatban, a lehető legtakarékosabban vannak tárolva, így a legkevesebb helyet foglalják el.**
A táblázat használatakor azonban, amikor valamilyen műveletet végzek az adatokkal, a megfelelő kapcsolat létrehozása után, úgy látom majd, mintha egyetlen táblázatom lenne. Természetesen ez csak logikailag egy táblázat, fizikailag több adatfájl. Ezt az adatbázist relációs adatbázisnak nevezzük.
Már csak egy feladatunk van, a gyakorlatban létrehozni a kapcsolatokat.

Feladat:
443. oldal 54. feladat
460. oldal 154. feladat

Gyakorlati példa: A fenti módosításokat hajtsuk végre a `tanulo.dbf` táblázaton, és a `cimek.dbf` táblázaton is, a megváltozott `cimek` táblázat neve legyen `cimisk.dbf` . Hozzuk létre az `iskola.dbf` fájlt is. Készítsük el a megfelelő indextáblákat is. A `tanulo.dbf` fájlhoz név mező szerint, a `cimisk.dbf` táblához tankód szerint, az `iskola.dbf` táblázathoz pedig iskkód szerint.

7.2. Relációs adatbázis létrehozása a gyakorlatban

Egyszerre több tábla megnyitására van szükségem. Mivel egyszerre csak egy tábla lehet nyitva, más megoldást kell találni. Ún. munkaterületeket használok. **Egy munkaterület megnyitása a**
`SELECT n`

paranccsal történik. n a munkaterület hivatkozási száma. Összesen 225 munkaterület nyitható meg a FoxProban egyszerre. Lépkedni közöttük a `SELECT` paranccsal lehet. **Egyszerre csak egy munkaterület lehet aktív, azonban kapcsolatot tudunk teremteni egy másik munkaterülettel is. A kapcsolatnak az a feltétele, hogy a kapcsolt táblázat a kapcsolómező szerint indexelve legyen.**

Egyes

Valami	Kód
ez	1
az	2
amaz	3

Kettes

Kód	Bármi
1	itt
2	ott
3	amott

`Egyes` és a `Kettes` tábla

Esetünkben az `Egyes` táblához kapcsoljuk a `Kettes` táblát. Tehát a `Kettes` tábla a kapcsolt tábla. **Azt a táblázatot, melyhez a másikat kapcsoljuk <u>dinamikus</u> <u>táblának</u> nevezzük.** Most dinamikus tábla az `Egyes` tábla.

Ábra Hiba! A kapcsoló argumentuma érvénytelen.**. A kapcsolat sematikus rajza:**

A viszonyuk azt fejezi ki, hogy majdan az `Egyes` táblában, a dinamikus táblában lépkedünk végig a rekordokon, és csak a kód mezőn keresztül kapcsolódunk a `Kettes` táblához, mintegy odamásolva a `Kettes` táblázat megfelelő elemeit az `Egyes` táblázat mellé.

Lépések:
1. **Megnyitjuk a 2. munkaterületet**
2. **A kapcsolt táblázatot megnyitjuk és rendezzük a kapcsolómező szerint.**
3. **Megnyitjuk az 1. munkaterületet**
4. **Megnyitjuk a dinamikus táblát**
5. **Létrehozzuk a kapcsolatot**

Kapcsolat létrehozásának parancsa:
`SET RELATION TO kapcsolómező INTO kapcsolttábla`

Tehát, ha az `Egyes` táblához kapcsolom a `Kettes` táblát:

```
SELECT 2
USE Kettes ORDER kód
```

```
SELECT 1
USE Egyes
SET RELATION TO kód INTO Kettes
```

Az aktuális munkaterület mezőit a már jól ismert LIST paranccsal lehet megnézni, **a másik munkaterületre pedig annak számával, vagy a tábla nevével lehet hivatkozni, majd kötőjel és nagyobb jel után(->) lehet írni a megfelelő mezőt. A -> jel helyett pontot is írhatunk.**

Mindhárom megoldás egyformán jó pl. ha a `Kettes` tábla bármi nevű mezőjére hivatkozunk:

LIST 2->bármi
LIST Kettes->bármi
LIST Kettes.bármi

1. Kidolgozott példa
472. oldal 221. feladat
Feladat:
Hogyan listázná ki az alábbi táblákból a vevők nevét, címét és számláikon szereplő tartozást?
VEVOK(Azonosito, Utca, Nev, Irszam, Varos)
SZAMLAK(Szamlaszam, Datum, Osszeg, Azonosito, Tartozas)
Megoldás:
A VEVOK táblázat a vevők címét tartalmazza, a SZAMLAK táblázat az összes számla adatait. Egy vevőnek több számlája is lehet, és mi az összeset szeretnénk látni. Ehhez végig kell lépkednünk a SZAMLAK táblázaton. Tehát ez lesz a dinamikus tábla. Innen majd a Tartozas mezőt kell listázni. A SZAMLAK táblához kell kapcsolni a VEVOK táblát az Azonosito mezőn keresztül. A listában tehát a Tartozás mellett szerepel majd a vevő neve és címe is.

A kapcsolat sematikus rajza

Lépések:
1. Megnyitom a 2-es munkaterületre a VEVOK táblázatot és rendezem Azonosito mező szerint.
2. Majd megnyitom az 1-es munkaterületre a SZAMLAK táblát
3. Létrehozom a kapcsolatot
4. Listát készítek

Válasz:

```
SELECT 2
USE  vevok ORDER TO Azonosito
SELECT 1
USE szamlak
SET RELATION TO Azonosito INTO vevok
LIST vevok->Nev, vevok->Irszam,vevok->Varos, vevok->Utca, tarozas
```

2. Kidolgozott példa:

A feladat egy olyan lista készítése, melyben szerepel a tanulók neve és címe is. A `tanulo.dbf` táblázaton fogunk végig menni és ehhez kapcsoljuk a `cimisk.dbf` fájlt. A két táblázat a tankód mezőn keresztül kapcsolódik.
a, Megoldás parancsokkal:

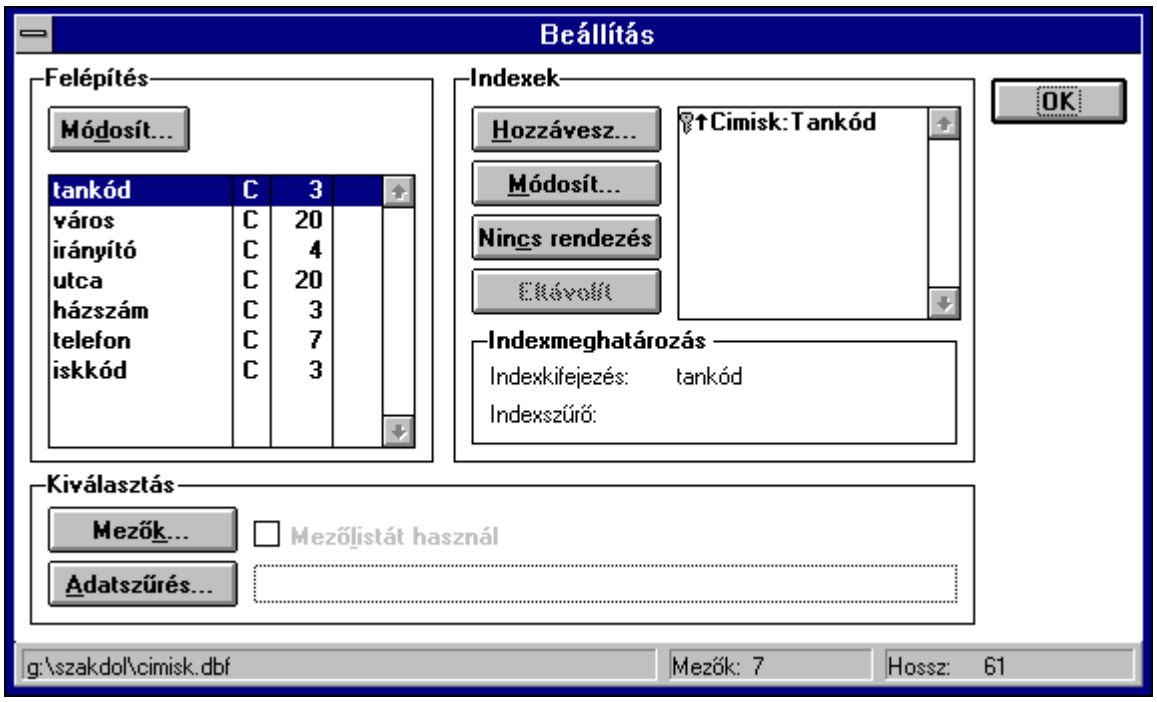
```
SELECT 2
USE cimisk ORDER tankód
SELECT 1
USE tanulo
SET RELATION TO tankód ORDER cimisk
LIST név, cimisk->Irszam, cimisk->város, cimisk->Utca, cimisk->Házszám
```

b, Kapcsolat létrehozása menüvel:

Az Ablak főmenü Nézet almenüjénél a Táblázatok kezelése ikont válasszuk A megjelenő ablak bal oldalán levő Munkaterületek lista most még üres.

Ablak főmenü Nézet almenüjénél a Táblázatok kezelése

Nyissuk meg az 2. munkaterületre a `cimisk.dbf` fájlt úgy hogy először a 2. munkaterületre kattintunk az egérrel, majd a Megnyitás parancsgombot használjuk. `A cimisk.dbf` fájl megnyitása után a Megnyit parancsgomb használatával térhetünk ismét az előző ablakhoz vissza. A Beállítás parancsgomb lenyomásánál megjelenő ablakban állíthatjuk be a tankód szerinti

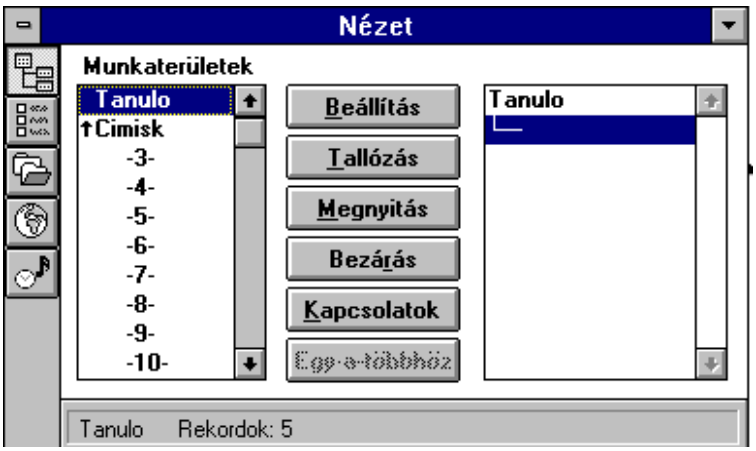


Beállítás ablak

rendezettséget, mégpedig úgy, hogy a kívánt mezőre kattintunk kétszer, és a megjelenő Táblázat szerkezet ablakban beállítjuk az indexelést. Esetünkben a tankód mező szerint. Majd OK gomb lenyomásával visszatérünk a Beállítás ablakhoz. Az Indexek blokkon belül a Rendezés parancsgombbal, majd OK gomb lenyomásával visszatérünk az Nézet ablakhoz. Jelöljük ki az 1. munkaterületet és nyissuk meg a `tanulo.dbf` fájlt a Megnyitás parancsgombbal. Mostantól mindkét táblánk nyitva van.

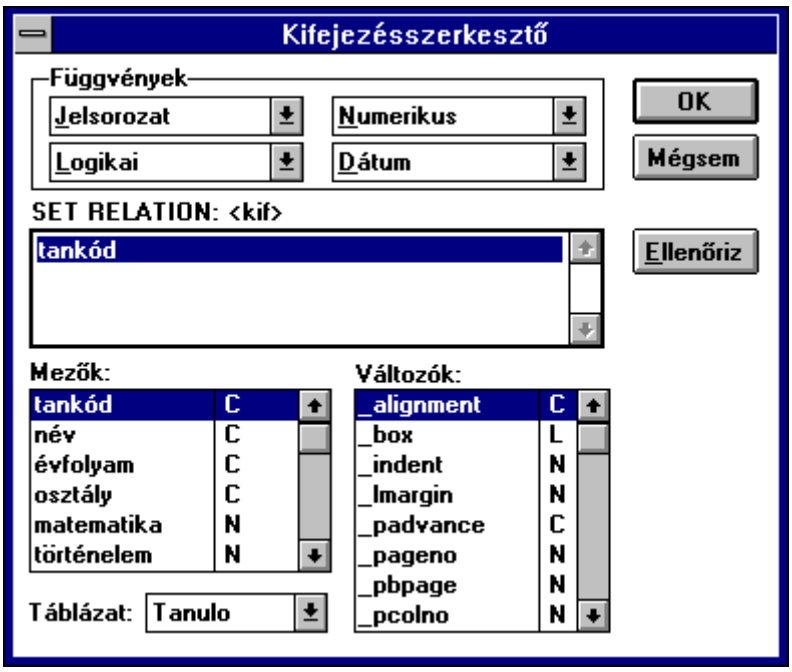
Nézet ablak két megnyitott táblával

Következhet a kapcsolat létrehozása. **Mindig a dinamikus állományt jelöljük ki a kapcsolat létrehozásához,** esetünkben a `tanulo.dbf` fájlt, majd nyomjuk le a Kapcsolatok gombot. Az ablak jobb oldalán megjelenik a kapcsolatot jelző vonal.



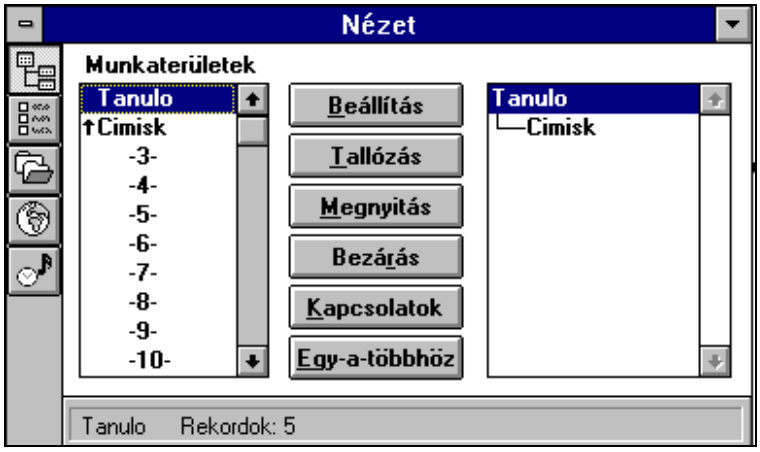
Dinamikus tábla kapcsolattal

Majd kattintsunk rá a kapcsolt táblát tartalmazó munkaterületre, azaz a `cimisk`-re. Ezután az Kifejezésszerkesztő megadása ablakban megjelenik a kapcsolómező, esetünkben a tankód.



Kifejezésszekesztő

Hagyjuk jóvá az OK nyomógombbal. A kapcsolat készen van.



Kapcsolat

A lista előállításához gépeljük be a parancsablakba a

```
LIST név, cimisk->Irányító, cimisk->város, cimisk->Utca, cimisk->Házszám
```

utasítás sorozatot.

Az eredmény:

Rekord#	NÉV	Cimisk.IRÁNYÍTÓ	Cimisk.VÁROS	Cimisk.UTCA	Cimisk.HÁZSZÁM
1	Albert Veronika	2119	Pécel	Kovács	2
2	Balázs Ferenc	1136	Budapest	Balzac	9
3	Kovács Pisti	1136	Budapest	Balzac	13
4	Zebra Zoltán	2660	Balassagyarmat	Szondi	25
5	Tóth Imre	1173	Budapest	Zöld	9

Lista kapcsolt táblázatokból

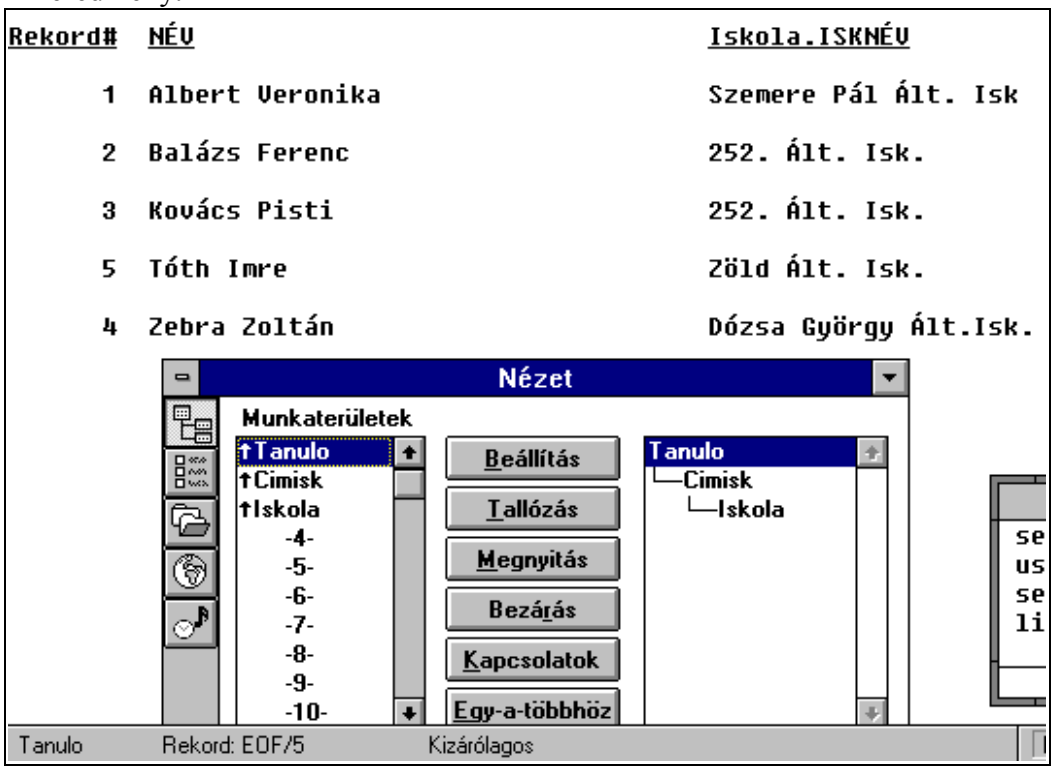
A táblázatok bezárását a
`CLOSE DATABASES`
paranccsal oldhatjuk meg.

3. Kidolgozott példa:

A feladat egy olyan lista készítése, ahol a tanulók nevei névsorba vannak rendezve, és minden tanuló neve mellett szerepel a régi iskolája neve. A tanulók nevét a `tanulo.dbf` fájl tartalmazza, mely a Tankód mezővel kapcsolódik a `cimisk` táblázathoz. Ez a tábla a gyerekek címét tartalmazza, és az Iskkód mezőn keresztül kapcsolódik az `iskola.dbf` fájlhoz. A dinamikus állomány, fájl a `tanulo.dbf` fájl lesz és ahhoz kapcsolódik a `cimisk.dbf` fájl, a Tankód kapcsolómezőn keresztül. A `cimisk.dbf` szintén dinamikus állománya lesz az `iskola.dbf` fájlnak mely az Iskkód mezőn keresztül kapcsolódik hozzá. Ügyeljünk arra, hogy a szükséges indexeket előre hozzuk létre.
a, Parancsok:

```
SELECT 3
USE iskola ORDER iskkód
SELECT 2
USE cimisk ORDER  tankód
SET RELATION TO iskód INTO iskola
SELECT 1
USE tanulo ORDER név
SET RELATION TO tankód INTO cimisk
LIST név, iskola->isknév
```

Az eredmény:



Lista kapcsolt táblázatokból

b, Kapcsolat létrehozása menüvel:

Az Ablak főmenü Nézet almenüjénél a Táblázatok kezelése ikont válasszuk. Nyissuk meg az 3. munkaterületre a `iskola.dbf` fájlt úgy hogy először a 3. munkaterületre kattintunk az egérrel, majd a Megnyitás parancsgombot használjuk. Az `iskola.dbf` fájl megnyitása után a Megnyit parancsgomb használatával térhetünk ismét az előző ablakhoz vissza. A Beállítás parancsgomb lenyomásánál megjelenő ablakban állíthatjuk be az iskkód szerinti rendezettséget az Indexek blokkon belül a Rendezés parancsgombbal, majd OK gomb lenyomásával visszatérünk az előző ablakhoz. Jelöljük ki az 2. munkaterületet és nyissuk meg a `cimisk.dbf` fájlt a Megnyitás parancsgombbal. Mostantól két táblánk van nyitva. Ezt is rendezzük tankód szerint a Beállítás parancsgombbal. Következhet a kapcsolat létrehozása. Mindig a dinamikus állományt jelöljük ki a kapcsolat létrehozásához, azaz `cimisk`-et majd nyomjuk le a Kapcsolatok gombot. Az ablak jobb oldalán megjelenik a kapcsolatot jelző vonal. Majd kattintsunk rá a kapcsolt táblát tartalmazó munkaterületre, azaz az `iskola`-ra . Ezután az Kifejezésszerkesztő megadása ablakban megjelenik a kapcsolómező, esetünkben a iskkód. Hagyjuk jóvá az OK nyomógombbal. A kapcsolat egy része készen van. Lépjünk az 1. munkaterületre. Nyissuk meg a `tanulo.dbf` fájlt a Megnyitás parancsgombnál. Rendezzük ezt név mező szerint a Beállítás parancsgombnál. Hozzuk létre a kapcsolatot a Kapcsolatok parancsgombra való kattintással. Majd jelöljük ki a `cimisk`-et tartalmazó munkaterületet. Hagyjuk jóvá a tankód mezőt, mint kapcsolómezőt az OK nyomógombbal. A kapcsolat készen van.

A lista előállításához gépeljük be a parancsablakba a

```
LIST nev, iskola->isknév
```

A táblázatok bezárását a

```
CLOSE DATABASES
```

paranccsal oldhatjuk meg.

Feladat:
435. oldal 4. feladat
470. oldal 212.-215. feladat
471. oldal 216.-220. feladat

7.3. Relációs adatbázis tulajdonságai, adatmodellek

A relációs adatbázis elnevezést az indokolja , hogy a táblázatok matematikai értelemben relációk. A relációs adatbázis **jellemzői:**

- **Az adatokat táblázatos formában tárolja, mindegyik táblának van külön neve.**
- **Egy táblán belül minden sorban ugyanannyi oszlop van. Egy oszlopon belül azonos típusú adatok vannak. Pl. mind numerikus típusú.**
- **Az oszlopok, tulajdonságok (attribútumok) sorrendje tetszőleges, de minden oszlop neve különböző kell legyen.**
- **A relációs adatbázis nem tartalmazhat 2 azonos sort. Mindig létezik kulcs, azaz olyan oszlop vagy oszlophalmaz, mely alapján 2 egyed megkülönböztethető.**
- **A táblák közötti kapcsolatot a kapcsolómező biztosítja**.

Ha egy táblázat tudja a fentieket **normalizált tábláz**atnak nevezzük. Továbbá normalizált táblára igaz:

Ha relációs adatbázisban módosítjuk, vagy töröljük a dinamikus tábla egy rekordját, akkor a kapcsolódó táblákban is módosul az összes kapcsolódó rekord. Ezt az eljárást kaszkádolt frissítésnek, vagy törlésnek nevezzük. Ilyenkor az adatbázisban ún. hivatkozási integritás van érvényben. A tábla konzisztens.

Korábban használtak más adatbázis szerkezeteteket, modelleket is.

A valóság leképezésére, ábrázolására modelleket, adatbázis struktúrákat alkotunk. Egyik ilyen modell a relációs adatbázis modell is.

Modellek:

- **Hierarchikus**
- **Hálós**
- **Relációs**

A hierarchikus modellt fa szerkezettel jellemezhetjük, a csomópontok az egyedek, az élek pedig a kapcsolatok. Az adatok alá-fölé rendelten helyezkednek el. **A hálós modellt gráffal jellemezhetjük,** ahol a csomópontok az egyedek, az élek pedig a kapcsolatok. Itt nincs alá-fölé rendelt viszony. **Relációs modell nem kezeli külön a kapcsolatokat és külön az egyedeket. A kapcsolatot egyszerűen a táblák közös oszlopai, a kapcsolómezők biztosítják.**

Feladat:
436. oldal 12. feladat
437. oldal 15. feladat
439. oldal 33. feladat
441. oldal 42. feladat
475. oldal 236.-237. feladat
483. oldal 283.-285. feladat

7.4. Kapcsolatok típusai, viszonyok

Kapcsolatnak nevezzük az adatbázis táblázatainak oszlopai közti viszonyt. Más szóval a tulajdonságok közti viszonyt.

7.4.1. Egy az egyhez kapcsolat (1:1)

Ha egy táblázat egy sorához a másik táblázat egy és csakis egy sora kapcsolható egy az egyhez kapcsolatról (1:1) beszélünk. Ilyenkor mindegy, melyik táblázatot választjuk dinamikusnak, és melyiket kapcsoljuk hozzá, hisz mindkettő ugyanannyi rekordból áll. Ha az egyik táblázatból választunk 1 rekordot, biztosan megtaláljuk a másik táblázatban is a hozzá tartozó adatokat. Ilyen kapcsolat van az óvodában a gyerekek és a jeleik között. Ha kiválasztunk a jelek közül egyet, pl boríték, biztosan megtaláljuk a hozzátartozó gyereket is.

7.4.2. Egy a többhöz kapcsolat (1:N)

Mikor az első tábla egy rekordjához a másik táblában több rekord is tartozhat. Viszont a másik tábla 1 rekordjához az első táblából csak 1 rekord tartozhat, elsődleges kulcsérték szerinti kapcsolatnál. Pl.
1. VEVOK(Azonosito, Utca, Nev, Irszam, Varos)
2. SZAMLAK(Szamlaszam, Datum, Osszeg, Azonosito, Tartozas)

1 vevőnek több számlája is lehet, 1 számla viszont nem tartozhat több vevőhöz.

7.4.3. Több az egyhez viszony (N:1)

Mikor az első tábla több rekordja a másik tábla 1 rekordjához illeszkedik.
1. SZAMLAK(Szamlaszam, Datum, Osszeg, Azonosito, Tartozas)
2. VEVOK(Azonosito, Utca, Nev, Irszam, Varos)

7.4.4. Több a többhöz (N:M) kapcsolat

Pl. nyelvek és emberek viszonya. 1 ember több nyelvet is beszélhet, viszont több ember beszélheti ugyanazt a nyelvet. A nyelvek és emberek viszonya szimmetrikus.
Feladat:
473. oldal 229.-230. feladat

8. Programozási bevezető

A FoxPro nyelve az XBase nyelvhez áll legközelebb. 1 sor 1 utasítást jelent. Nincs külön deklarációs rész.

8.1. Alap utasítás készlet

Értékadás:
változó = kifejezés

Mezőváltozónál:
```
REPLACE WITH kifejezés
```

Elágazás:
```
IF feltétel
        utasítás
        utasítás
        -
        ELSE
        utasítás
        utasítás
        -
ENDIF
```

A feltétel lehet egyszerű, de logikai műveletekkel (.AND. .OR. .NOT.) összetett feltételeket is megadhatunk.

Többirányú elágazás
```
DO CASE
        CASE feltétel
           utasítás
           utasítás
           -
        CASE feltétel
           utasítás
           -
        -
        OTHERWISE
           utasítás
           utasítás
           -
ENDCASE
```

Ciklus:

```
FOR ettől TO eddig
        utasítás
        utasítás
        -
NEXT
```

Elöltesztelős ciklus:

```
DO WHILE feltétel
        utasítás
        utasítás
        -
ENDDO
```

Ciklus magon belül LOOP utasítással visszaléphetünk a ciklus elejére, vagy EXIT utasítással feltétel nélkül kiléphetünk a ciklusból.

Változók:

Lokális változók: csak abban az eljárásban használhatók, melyben deklaráltuk őket.

```
PRIVATE vált1, vált2,...
```

Globális változók: minden eljárásban használhatók a programon belül.

```
PUBLIC vált3, vált4...
```

Eljárás hívás:

```
DO eljárásnév WITH paraméterek
```

Eljárás megadás:

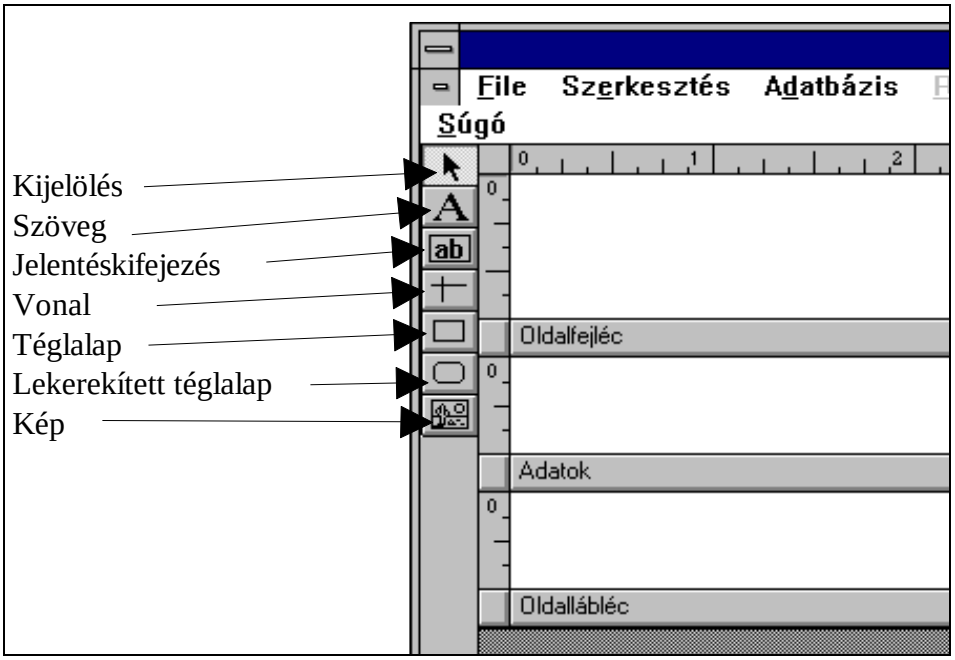
```
PROCEDURE eljárásnév
        PARAMETERS (paraméter1,paraméter2,...)
        -
        -
RETURN
```

Függvény megadás:

```
FUNCTION függvénynév
        PARAMETERS (paraméter1,paraméter2,...)
        -
        -
RETURN (visszaadott érték)
```

8.2. Jelentés készítés jelentésgenerátorral

Készíthetünk jelentést (riportot) a `LIST` parancs használatával is, de sokkal kényelmesebb jelentés-generátor használatával. **A jelentés-generátor használatának az a lényege, hogy megtervezzük milyen (akár adatot akár számított értéket tartalmazó) mezőket, objektumokat szeretnénk látni a jelentésünkben, listánkban. A többi már a FoxPro dolga. A kódot ő generálja, nem mi.** Jelentés, lista készítéséhez meg kell nyitnunk azt a táblát, táblákat, melyekből mezőket szeretnénk megjeleníteni. A táblák közötti kapcsolatot is meg kell határoznunk. Mi most a `tanulo.dbf` fájlt nyitjuk meg. Ezek után az Új főmenü Jelentés opcióját kell választanunk. Ekkor megjelenik a Jelentésformátum-tervező ablak, mely 3 részből, sávból áll.



Jelentésformátum-tervező ablak eszköztárral

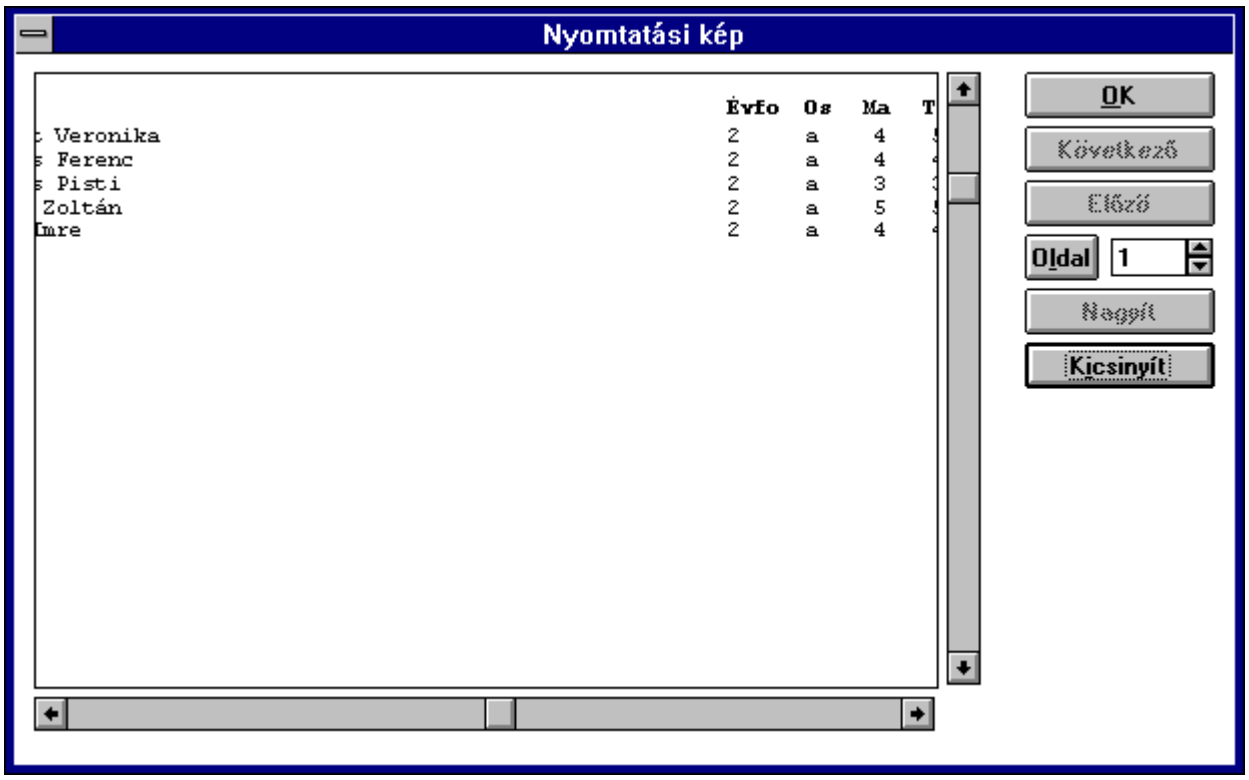
A legfelső sávba (Oldalfejléc) kerülnek azok az objektumok, mezőnevek, stb., melyek minden lap tetején látszódnak majd. A középső sávba (Adatok) kerülnek majd azok az objektumok, mezők, melyek a táblázat 1 sorát alkotják majd. A legalsó sávba (Oldallábléc) olyan objektumok kerülnek, melyek minden lap alján jelennek majd meg. Az egyes sávok mérete a sávok aljára történő rámutatással és az egér húzásával változtatható. Válasszuk a Jelentés főmenü Gyorsjelentés almenüpontját.

Gyorsjelentés ablak

A Gyorsjelentés ablakban megválaszthatjuk a mezők elrendezését, (vízszintes, vagy függőleges) a Mezőelrendezés ábránál. Mi a vízszintes elrendezést válasszuk. Majd a mezők fölhelyezéséhez ikszeljük be a Mezők igazológombot. Ekkor megjelenik a Mezőválasztás párbeszédablak.

Mezőválasztás párbeszédablak

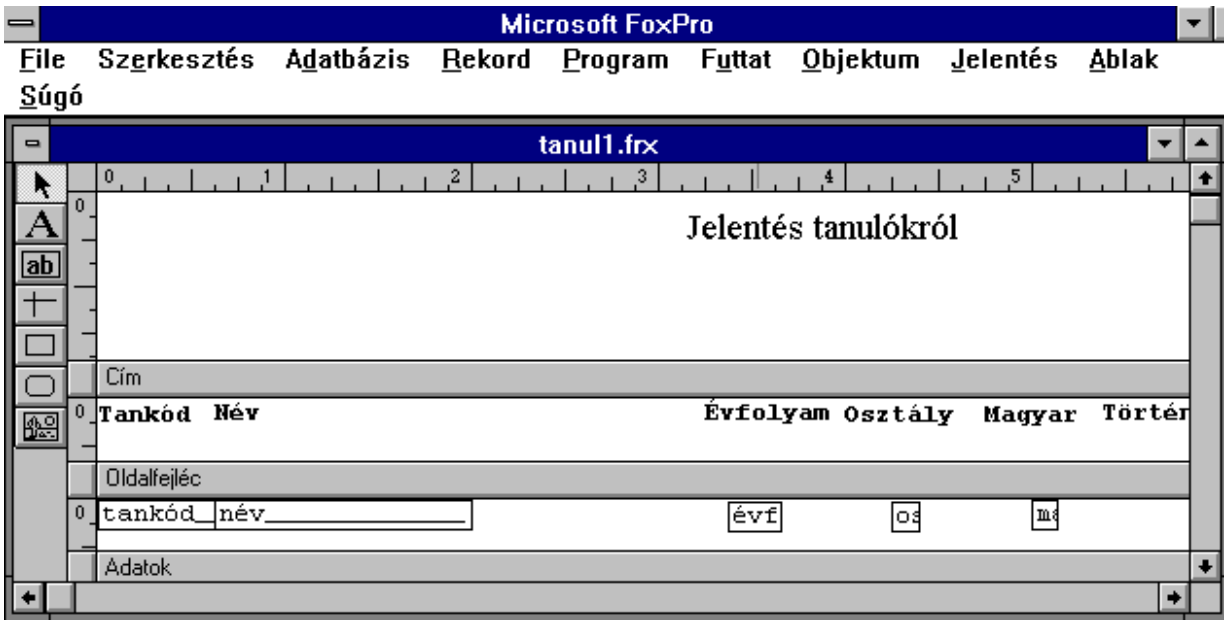
A Választható mezők listájából kiválaszthatjuk a jelentésben szereplő mezőket, és az Áthelyez parancsgombbal a Kiválasztott mezők listájába helyezhetjük át őket. A Mind parancsgombbal az összes felkínált mezőt áthelyezhetjük Kiválasztott mezőkhöz. Mi is ezt tesszük. OK gombot használva visszatér az előző ablak, itt ismét OK-ra kattintva a Jelentésformátum-tervező ablakban leszünk
A jelentés nyomtatási képét a Jelentés főmenü Nyomtatási kép almenüpontnál lehet megnézni.



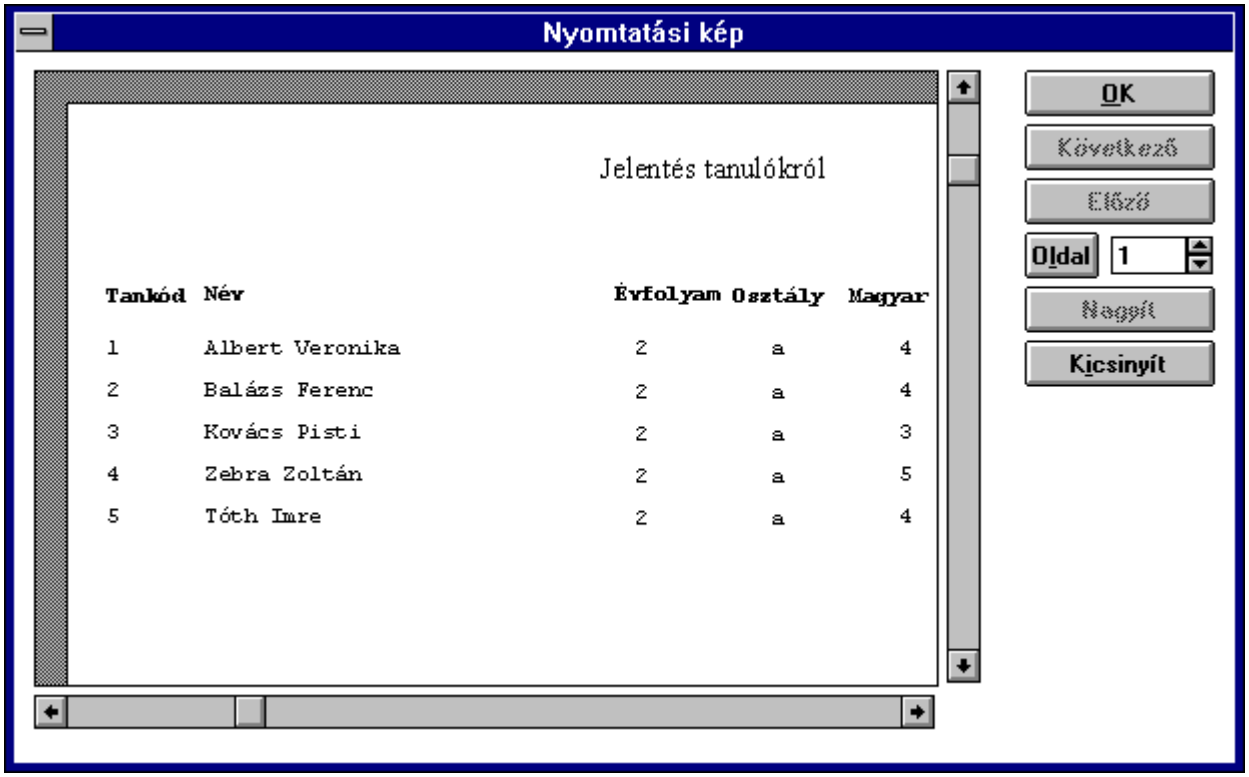
Gyorsjelentés

Látható, hogy még hiányos a listánk. Nem megfelelőek a betűméretek, nincs címe, stb. Térjünk vissza a Jelentésformátum-tervező ablakhoz az OK gombbal. A cím beírásához kattintsunk a Jelentés főmenü Cím/összegzés almenüjére, és ott kapcsoljuk be a Címsáv igazológombot, majd kattintsunk az OK gombra. Új sávként megjelent a Címsáv is a Jelentésformátum-tervező ablakban Bármilyen fajta szöveg írására használható a Szöveg gomb. Kattintsunk rá, majd arra a sávra ahová írni szeretnénk, esetünkben a címsávba. A cím legyen Jelentés tanulókról.

A Címsávot növeljük meg, és változtassunk a cím betűinek méretén és igazítsuk középre. Jelöljük ki a szöveget egyszeres kattintással, majd az Objektum főmenü megfelelő almenüjével végezzük el a kívánt változtatásokat. Most a Betűtípus és a Középre almenüket kell használnunk. Ha más objektumon szeretnénk változtatni az áttérés előtt használjuk a Kijelölés gombot az eszköztárról. (A mezők méretének megváltoztatását, kijelölés után az objektumot körülvevő keret kis négyzeteire állva, az egér bal gombját nyomva tartva, mozgatással tehetjük meg.)



tanul1.frx jelentés



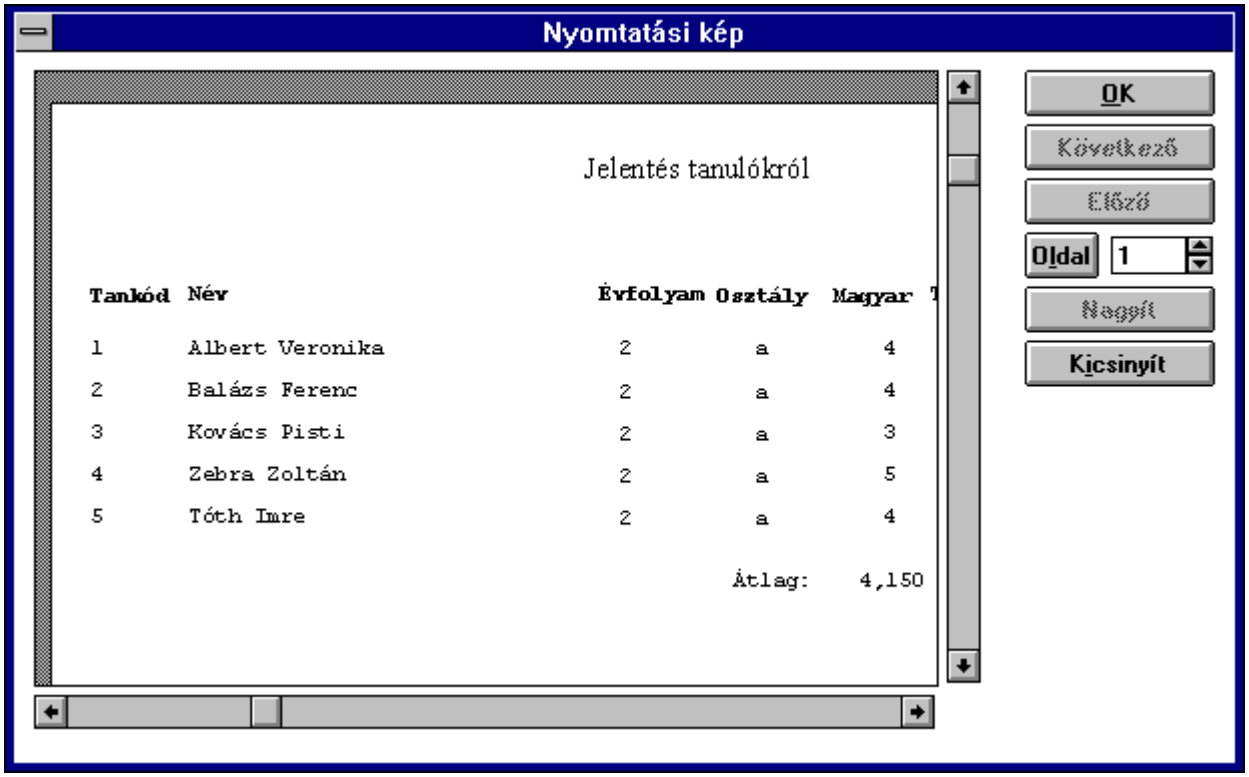
A jelentés nyomtatási képe

A jelentést mentsük el a File főmenü Ment almenüjénél, `tanul1.frx` néven. Első mentéskor a FoxPro megkérdezi, hogy mentjük e a környezeti információkat. (mely táblázatok vannak megnyitva, milyen kapcsolatok vannak érvényben stb.). Erre válaszoljunk Igenel, mert ez azt jelenti, hogy a jelentés megnyitásakor nem kell külön megnyitnunk a táblákat, kapcsolatokat létrehoznunk, mert ezeket a FoxPro automatikusan megcsinálja. A jelentésfájl kiterjesztése: `.frx`. Ha változtatok egy korábban elmentett jelentésen, azt érdemes új néven

elmenteni, mert a környezeti beállítások nem módosulnak a jelentés módosításakor csak a létrehozáskor.
A táblázat bezárásához írjuk be a parancsablakba a `USE` parancsot.

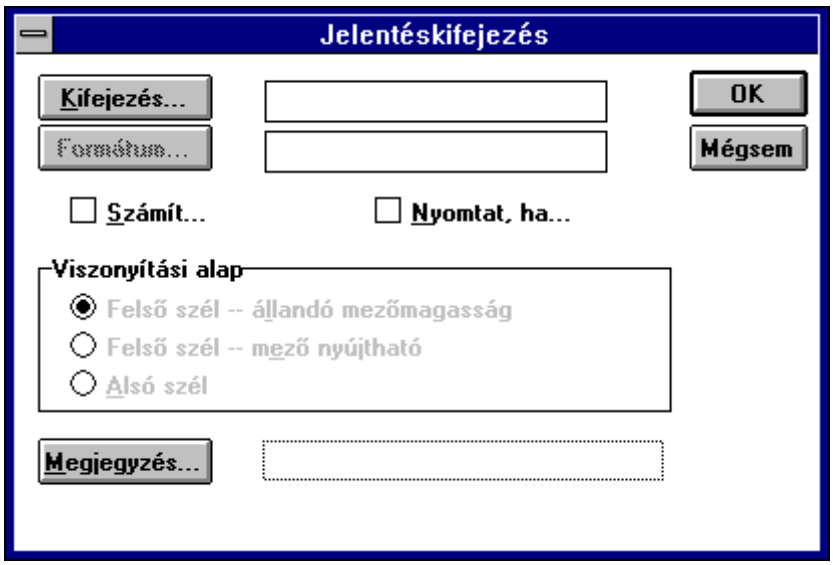
8.2.1. Jelentés készítése számított mezőkkel

A számított mező olyan mező, amely nem tárol adatokat, csupán kifejezések eredményét jeleníti meg. A kifejezésekben matematikai, logikai műveleteken túl függvényeket is használhatunk. Gyakori feladat numerikus mezők átlagát, összegét, darabszámát, maximumát, minimumát kiszámolni.
Gyakorlati példa:
Készítsük el az alábbi jelentést:



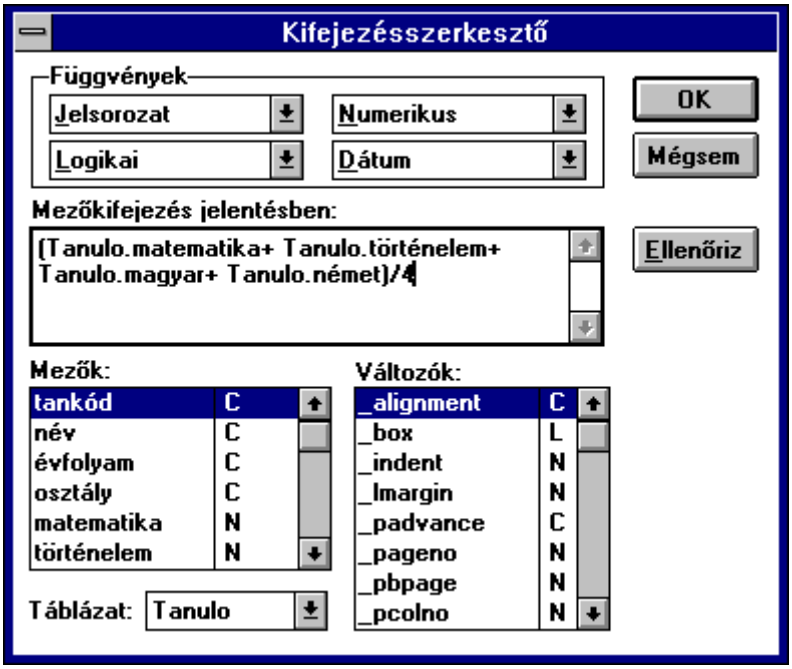
Jelentés készítése számított mezőkkel (átlaggal)

Látható, hogy a fenti táblázat csupán az átlag számítás miatt különbözik az előzőtől, ezért módosítsuk azt. Nyissuk meg, majd a Jelentésformátum-tervező ablakban hozzuk létre az ún. összegző sávot, a Jelentés főmernüpont Cím/összegzés almenüpontjánál az Összegzősáv igazológomb beikszelésével, és OK gombra kattintással. Az összegző sávban lehet megjelentetni azokat az adatokat melyek a lap alján, vagy a jelentés végén szerepelnek majd. Esetünkben ide a jegyek átlaga kerül majd, de kerülhetne összegzés, minimum, darabszámolás stb. eredménye is.
Az átlagszámításhoz, és egyéb műveletekhez a Jelentés-kifejezés gombot használjuk. Először a Jelentés-kifejezés gombra való kattintás után elhelyezzük a Jelentésformátum-tervező ablakban a számolás eredményének helyét. Most ez az Összegző sávban lesz. Ezek után megjelenik a Jelentéskifejezés ablak.



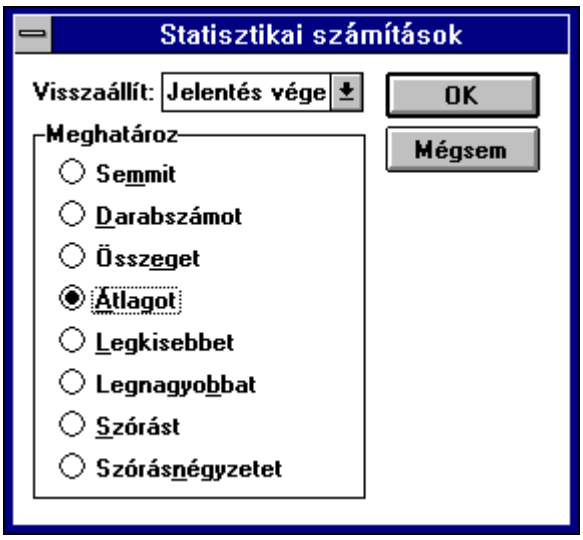
Jelentéskifejezés ablak

A Kifejezés parancsgomb lenyomása után kiválaszthatjuk a kifejezésben résztvevő mezőket kétszeres kattintással. Mi a német, magyar, matematika, történelem mezőkre kattintunk duplán.



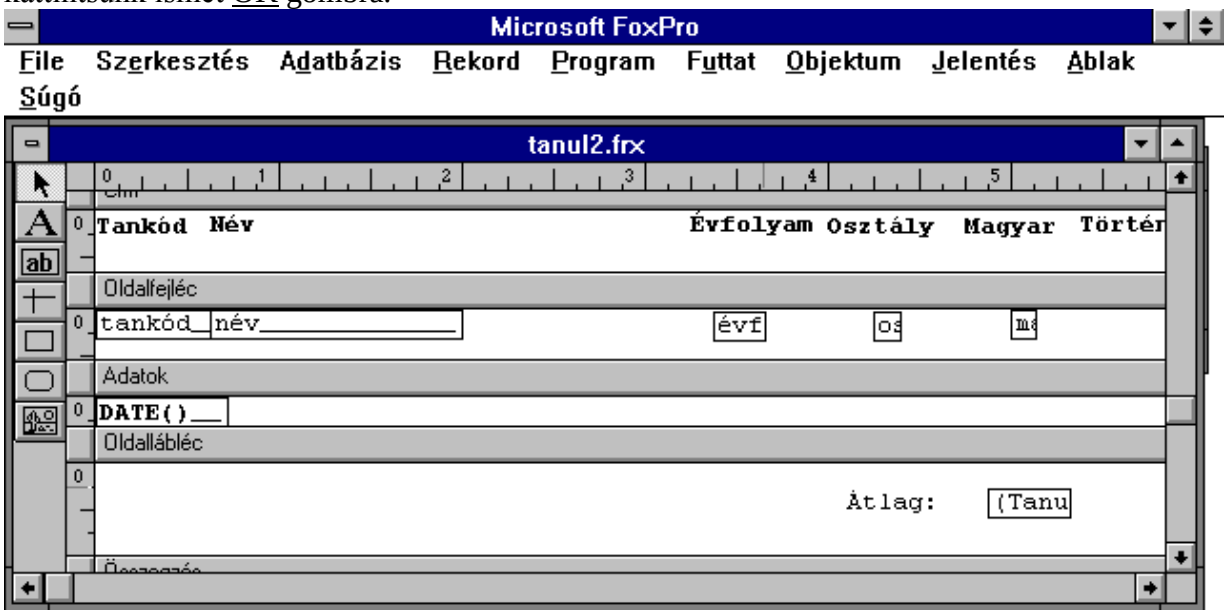
Kifejezésszerkesztő

Az OK gomb lenyomása után állítjuk be a mező formátumát. Állítsuk be numerikusnak, és 2 tizedesjegyet tartalmazónak. OK parancsgomb lenyomása után a Számít igazológomb bejelölésével megjelenő ablakban választhatjuk ki, hogy milyen műveletet kívánunk elvégezni (Semmit, Darabszámot, Összeget, Átlagot, Legkisebbet, Legnagyobbat, Szórást, Szórásnégyzetet). Mi az Átlagot válasszuk.



Statisztikai számítások

A Visszaállít listamezőnél kiválaszthatjuk, hogy hol nullázza le a kiszámított értéket a program:
-Jelentés végénél, ha az egész jelentésről szeretnék összegzést
-Oldal végénél, ha oldalanként szeretnék összegzést.
-Hasáb végénél
-Csoport végénél.
Mi a Jelentés végénél elemet válasszuk. Majd OK-val visszatértünk az előző ablakhoz, és kattintsunk ismét OK gombra.



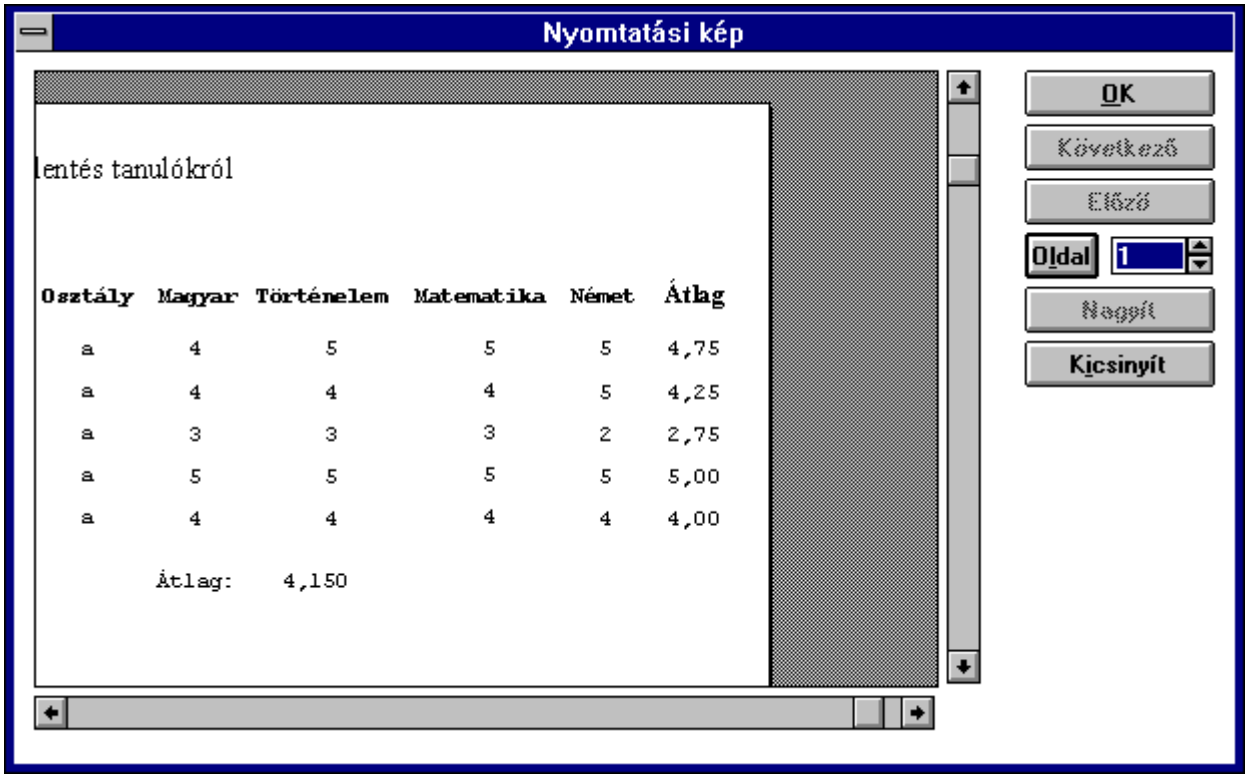
A Jelentéskifejezés ablak képe átlagszámításnál

A Jelentés főmenü Nyomtatási kép parancsával nézhetjük meg a jelentést.

Mentsük el `tanul2.frx` néven a File főmenü Új néven ment almenüjénél. A táblázat bezárásához írjuk be a parancsablakba a `USE` parancsot.

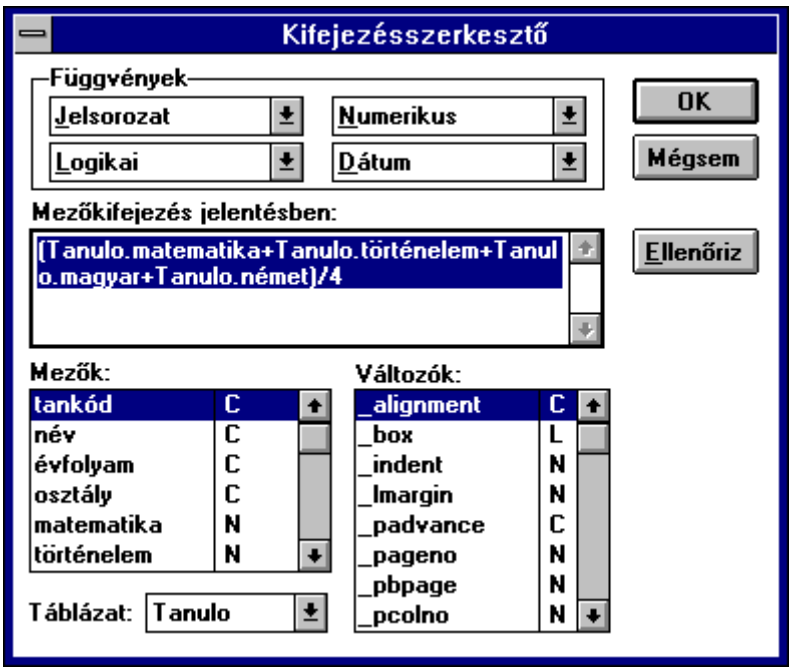
8.2.2. **Jelentés soronként számított mezőkkel**

Gyakorlati példa:
Készítsük el az alábbi jelentést:



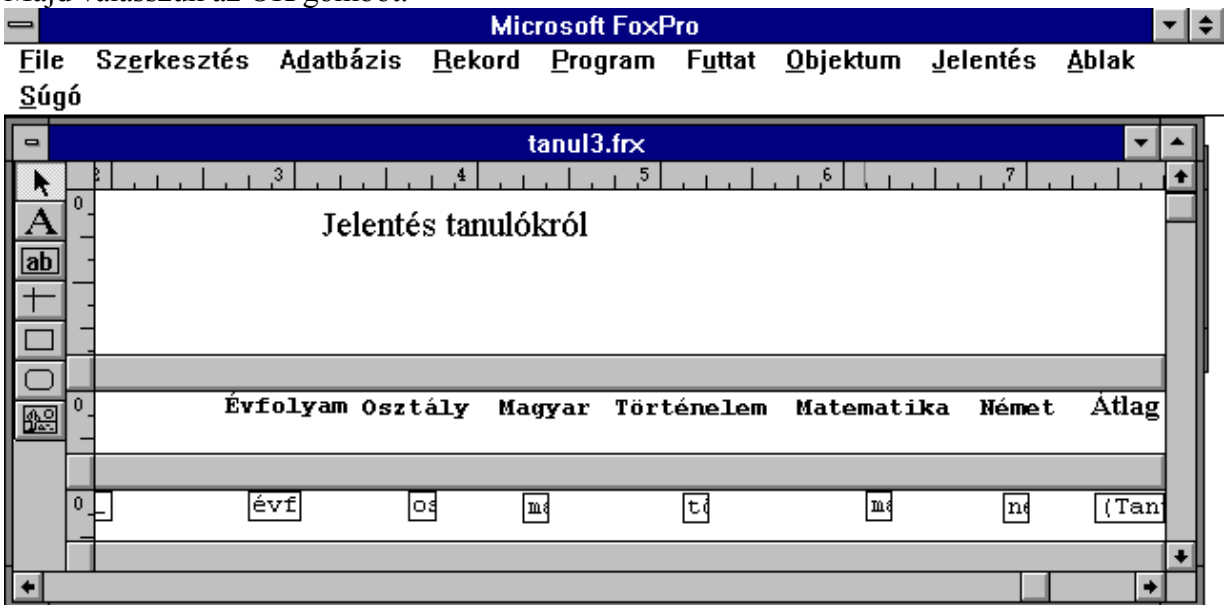
Táblázat átlaggal és soronkénti átlaggal

Látható, hogy csak soronkénti átlagolással gyarapodott előző jelentésünk. Ezért nyissuk meg a `tanul2` nevű fájlt és a Jelentésformátum-tervező ablak középső (Adatok) sávjába, és többi mező mellé utolsónak helyezzünk el egy kifejezést a Jelentés-kifejezés gombbal. A megjelenő párbeszédablakban a Kifejezés parancsgomb lenyomása után szerkesszük meg a megfelelő numerikus függvények felhasználásával az alábbi kifejezést:



Kifejezésszerkesztő

Majd válasszuk az OK gombot.

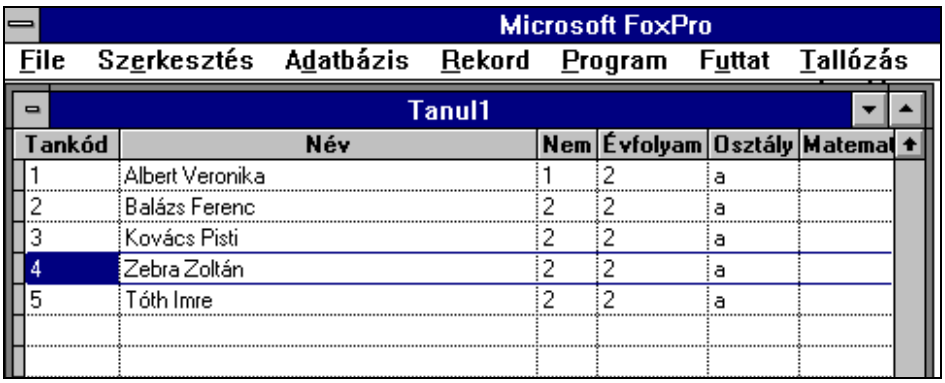


Jelentéskifejezés ablak soronkénti átlagok számításánál

Majd nézzük meg jelentésünk nyomtatási képét. Ezután végezzük el a mentést `tanul3.frx` néven és zárjuk be a táblát a `USE` paranccsal.

8.2.3. **Jelentés csoportokkal**

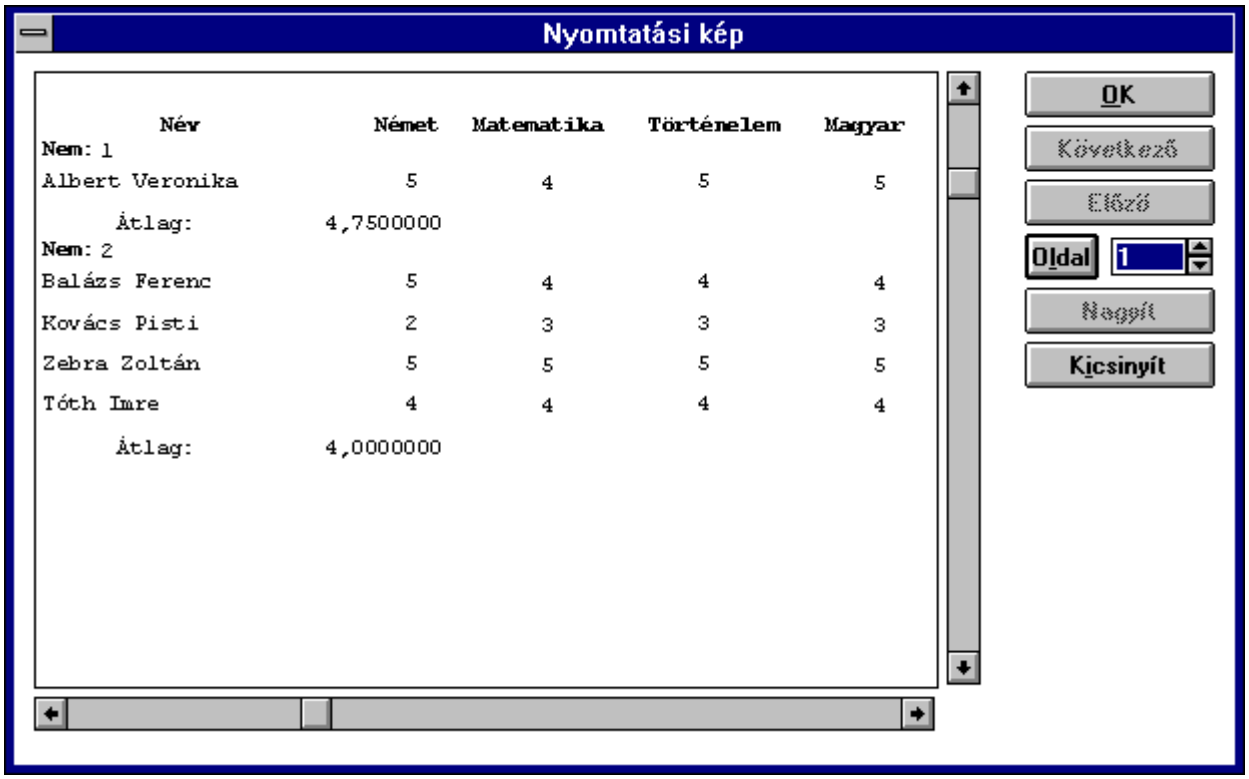
Hozzuk létre az alábbi `tanul1.dbf` táblázatot, melyben szerepel egy új karakteres mező is, ez pedig a tanulók nemét tartalmazza. A lányok jele: 1, a fiúké: 2.



Tankód	Név	Nem	Évfolyam	Osztály	Matemat
1	Albert Veronika	1	2	a	
2	Balázs Ferenc	2	2	a	
3	Kovács Pisti	2	2	a	
4	Zebra Zoltán	2	2	a	
5	Tóth Imre	2	2	a	

`tanul1.dbf`

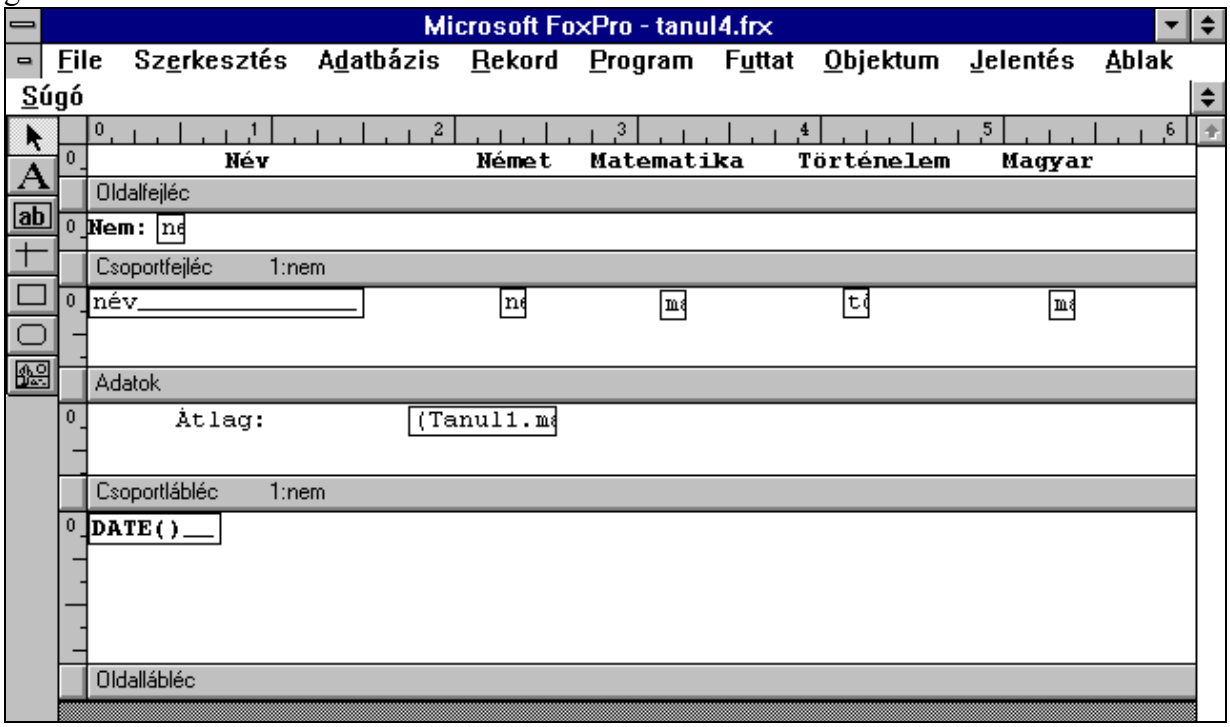
Minderre azért volt szükség, mert szeretnénk külön a lányok, és külön a fiúk átlagát megnézni. Azaz külön csoportot képezünk a lányokból és egy másikat a fiúkból. **A csoport az egyedhalmaz (táblázat) olyan részhalmaza, melyben az egyedek valamilyen közös tulajdonsággal rendelkeznek.** Pl mindegyik neme lány. Célunk az alábbi jelentés elkészítése:



Jelentés csoportokkal

A csoportképzés előtt a táblázatot indexelni kell a csoportosítási szempontot adó mezőre, és aktívvá kell ezt tenni. Esetünkben a nem mezőre. Majd gyorsjelentést készítünk. Ezek után hozzuk létre a csoportokat, a Jelentés főmenü Adatcsoportosítás almenüjénél. A

megjelenő ablakban a Hozzávesz parancsgomb lenyomása után a Csoportinformáció ablakban levő Csoport... gomb melletti szövegmezőbe gépeljük be a csoportosítási szempontot adó mező nevét, esetünkben a nem mezőnevet. OK gomb lenyomásával visszatértünk az előző ablakhoz. Innen OK gombbal távozunk. A Jelentésformátum-tervező ablakban új sávként megjelent a Csoportfejléc és Csoportlábléc sáv. A sávok méretét növeljük meg, végül a nem feliratot és a nem feliratú mezőt húzzuk át a Csoportfejléc sávba. Az átlagoláshoz ismét a Jelentéskifejezés gombra kell kattintanunk, majd a Csoportlábléc sávba. A megjelenő ablakban a Kifejezés parancsgomb lenyomása után válasszuk ki a megfelelő mezőket, dupla kattintással, és írjuk be a használt függvényeket, vagy műveleti jeleket. Majd OK gomb lenyomása után a kívánt formátumot állítsuk be a Formátum parancsgomb lenyomása után. Ha beállítottuk a kívánt formát OK gombbal visszatérünk az előző ablakhoz, ahol a Számít igazológomb beikszelése után kiválaszthatjuk az Átlagot opciót. Majd kattintsunk az OK gombra ebben és a következő ablakban is.



Jelentéskifejezés ablak csoportonkénti számítással

A Jelentés főmenü Nyomtatási kép parancsával nézzük meg a jelentést.

Mentsük el `tanul4.frx` néven. A táblázat bezárásához írjuk be a parancsablakba a `USE` parancsot.

Feladat

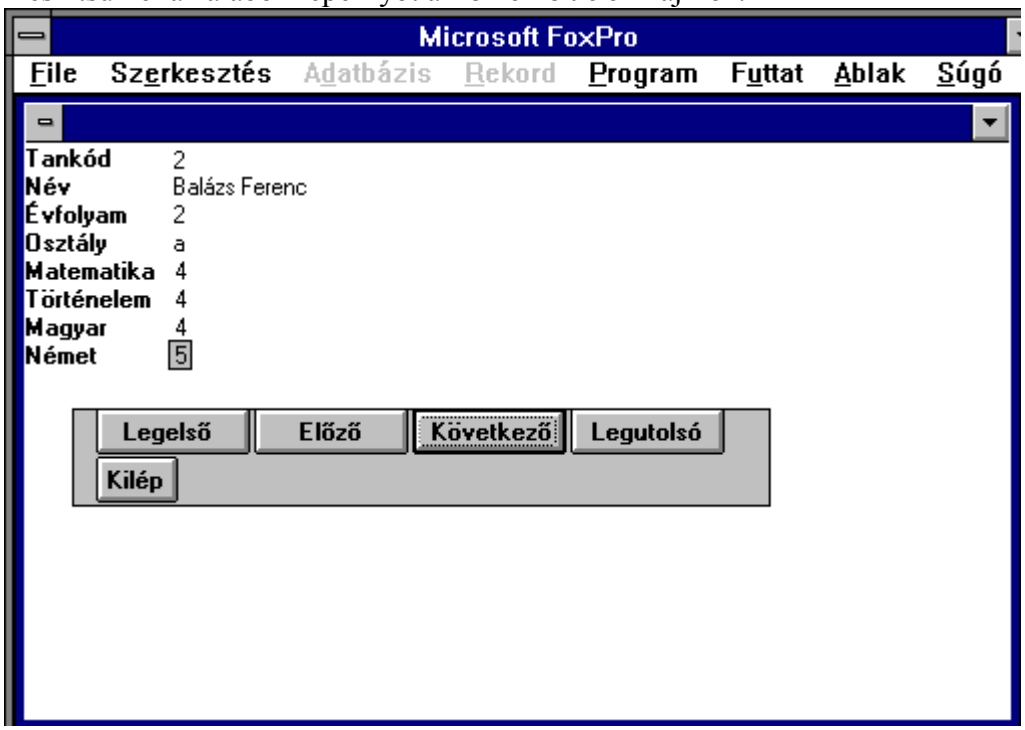
440. oldal 35. feladat

8.3. Képernyő generálás

A képernyő generálás a jelentés generáláshoz hasonlóan nagyon kényelmes, felhasználóbarát eszköz. Mi felhasználók **csak megtervezzük és felrakjuk a** Képernyőformátum-tervező **ablakra a kívánt objektumokat, majd a FoxPro készít, generál hozzá egy kódot, melyet aztán futtatni lehet.**

8.3.1. Mozgás a fájlban parancsgombok segítségével

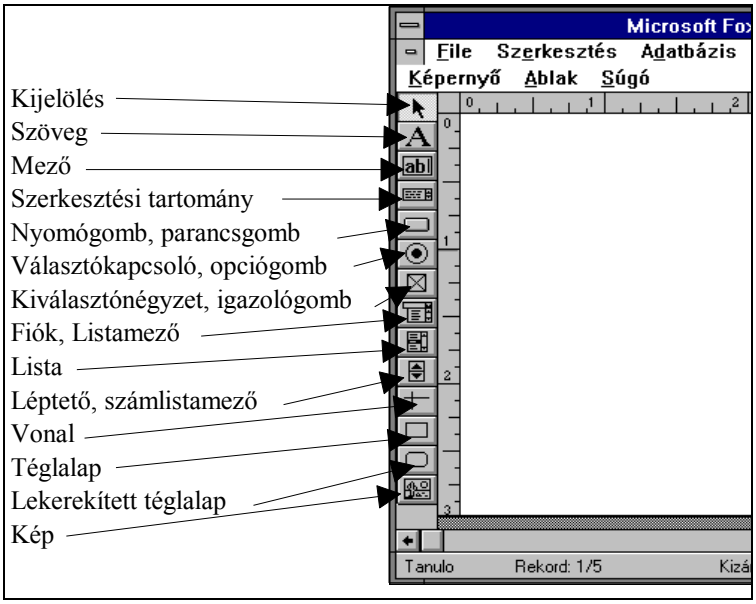
Készítsük el az alábbi képernyőt a `Tanulo.dbf` fájlhoz.



Output (Kiíró) képernyő a fájlban való mozgáshoz és az adatok megtekintéséhez

A Legelső, Előző, Következő, Legutolsó parancsgombokkal lépkedhetünk a `tanulo.dbf` táblázatban. A Kilép gombbal kiléphetünk az ablakból.
A feladat megoldását mindig a környezeti beállításokkal kell kezdeni, azaz meg kell nyitni a megfelelő táblázatokat, indexeket aktívvá kell tenni, és a kapcsolatokat létre kell hozni. Mi is ezzel kezdjük. Először meg kell nyitni a `tanulo.dbf` táblázatot az Ablak főmenü Nézet almenüjénél a Táblázatok kezelése gombnál. Érdemes név mező szerint rendezni, a Beállítás gombnál. Ezek után lássunk neki a képernyő készítésnek.

A File főmenü Új almenüpontjánál ki kell választani a Képernyő opciót. Majd az Új parancsgombot választjuk. Ekkor megjelenik a Képernyőformátum-tervező ablak.

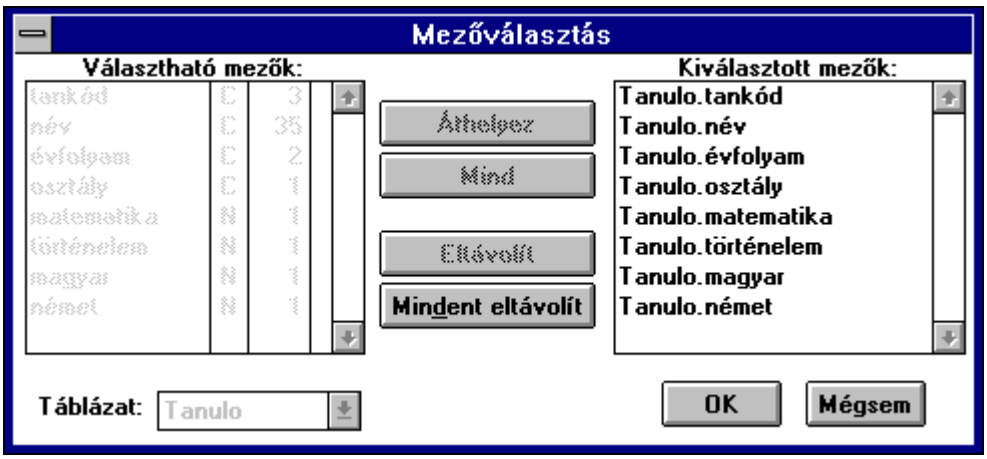


Képernyő-formátum tervező ablak eszköztárral

A megjelenítendő mezők fölrakásához a Képernyő főmenü Gyorsképernyő almenüpontját válasszuk.

Gyorsképernyő

A megjelenő ablakban ikszeljük be a Mezők igazológombot. A Mezőválasztás ablakban a kiválaszthatjuk, mely mezőket akarjuk majd látni a képernyőn, kijelöléssel, majd az Áthelyez parancsgombbal. Mi helyezzük át az összes mezőt a Mind parancsgombbal.



Mezőválasztás

OK parancsgombokkal térhetünk vissza az előző ablakhoz, ismét OK gombot használva pedig a Képernyőformátum-tervező ablakhoz.

Fontos tudni, hogy alapjában véve 2 féle mezőt lehet felhelyezni a képernyőre. Kiíró mezőt, vagy beviteli mezőt. A kiíró mező tartalmát a felhasználó nem módosíthatja, a beviteli mező tartalmát viszont igen. Érdemes az általunk felrakott mezőket kiíró mezőkké alakítani, hiszen a majdani felhasználó csak nézegetni fogja az adatokat, nem módosít, változtat rajtuk. Oly módon, hogy kijelöljük a kívánt mezőt majd duplán kattintunk az egérrel, és a megjelenő Mező párbeszédablakban beállíthatjuk a szükséges változtatást.

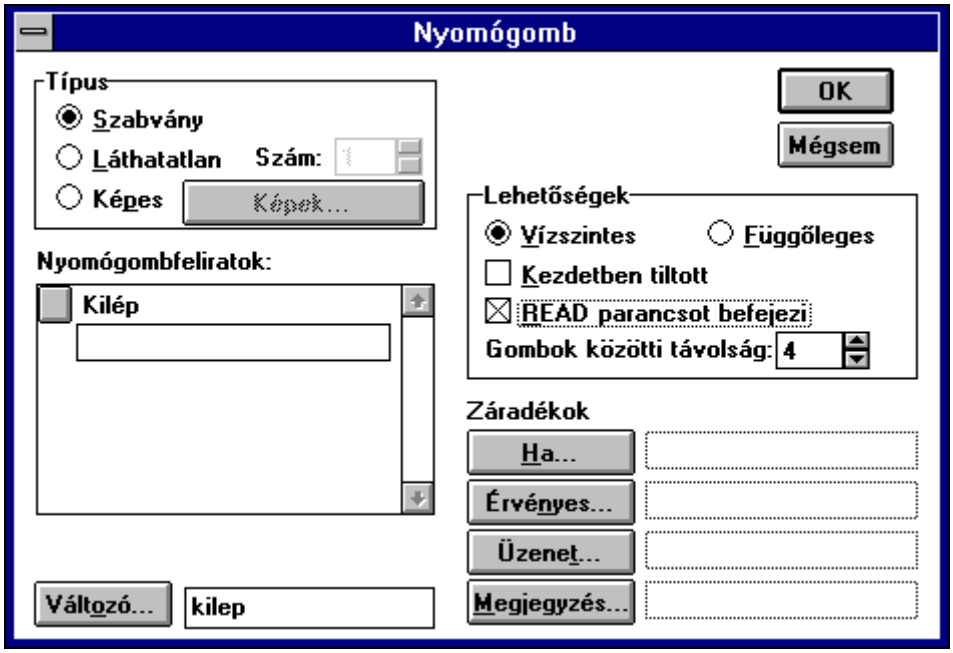
Mező párbeszédablak kiíró mezőnél

A Mező ablak bal felső sarkánál levő Mező felirat alatt válasszuk ki a Kiíró mező opciót. Az Egyebek felirat alatt pedig bekapcsoljuk a Kimeneti mező frissítése igazológombot. Ezt ismételjük meg a többi mezőre is. Az átlépést a Kijelölés gombra kattintva tehetjük meg.

Finomítsunk kicsit a képernyőn. Rendezzük el a mezőket, hogy minél jobban hasonlítsanak a mintához.

Helyezzük fel a Kilép parancsgombot a képernyőre. Kattintsunk a Nyomógomb gombra, majd arra a helyre ahová a nyomógombot tesszük.

A megjelenő Nyomógomb párbeszédpanelben a Nyomógombfeliratok alatti mezőbe írjuk be a gombon levő szöveget: Kilép, a Változó felirat alatti mezőbe írjuk be a kilep nevet. Ikszeljük be a Read parancsot befejezi igazológombot.



Nyomógomb ablak Kilép parancsgomb felhelyezéséhez

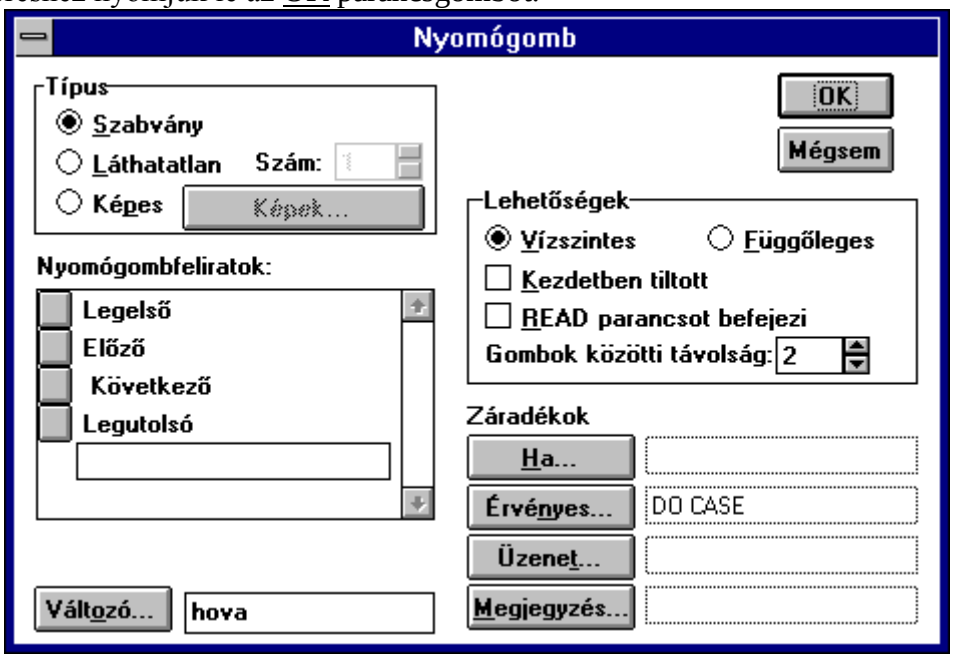
Nyomjuk le az OK parancsgombot.
Helyezzük fel a Legelső, Előző, Következő, Legutolsó parancsgombokat. Ehhez kattintsunk a Nyomógomb gombra, majd arra a helyre ahová a gombokat helyezni szeretnénk. A megjelenő Nyomógomb ablakba írjuk be a Nyomógombfeliratok mezőbe, egymás alá:
Legelső
Előző
Következő
Legutolsó

Majd írjuk be a változó felirat mellé: hova
A Lehetőségek felirat alatti részben a Gombok közti távolság számlistamezőt állítsuk 2-re.
Az Érvényes parancsgomb lenyomása után gépeljük be a megjelenő ablakba az alábbi programot, mely a rekordmutató mozgatásáról gondoskodik.

```
DO CASE
     CASE hova =1
          GO   TOP
     CASE hova =2
          IF !BOF()
               SKIP -1
               IF BOF()
                    GO TOP
               ENDIF
          ENDIF
     CASE hova =3
          IF !EOF()
               SKIP
               IF EOF()
                GO BOTTOM
               ENDIF
          ENDIF
     CASE hova =4
          GO BOTTOM
ENDCASE
SHOW GETS
RETURN
```

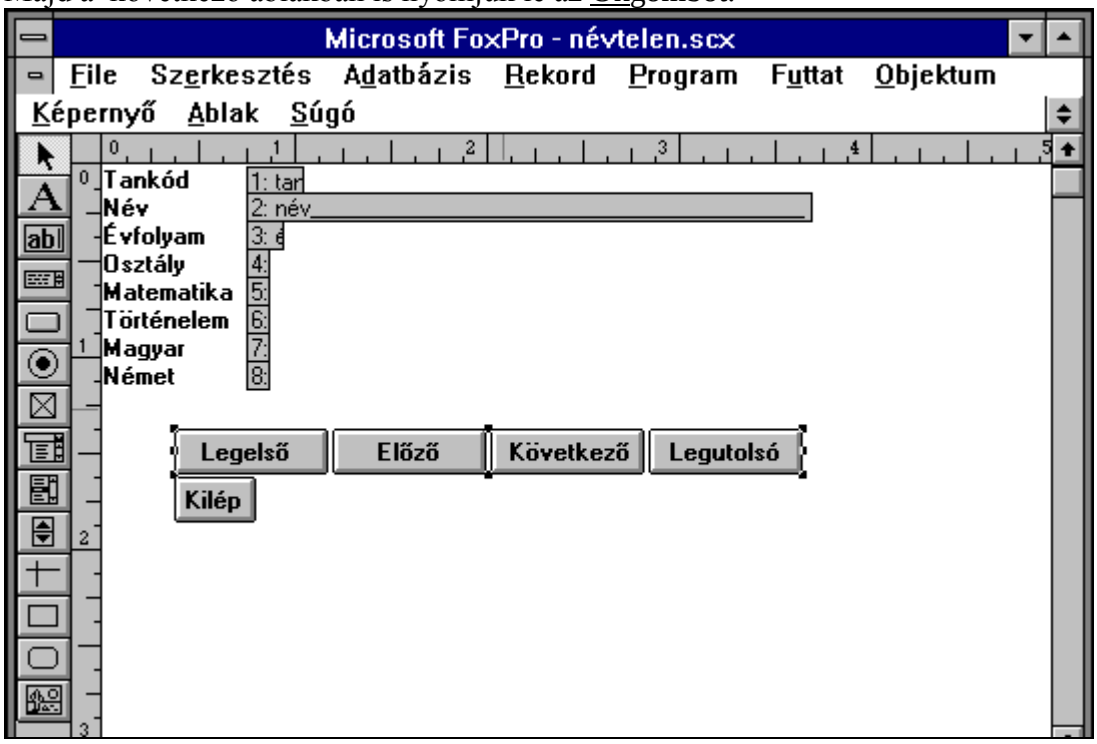
Magyarázat:
Ha hova=1 akkor
    ugorj a legelső rekordra
Ha hova=2 akkor
    ha nem fájl eleje
        lépj vissza 1-et,
        ha fájl eleje akkor
                ugorj az első rekordra
        Elágazás vége
    Elágazás vége
Ha hova=3 akkor
    ha nem fájl vége
        lépj 1-et előre,
        ha fájl eleje akkor
                ugorj az utolsó rekordra
        Elágazás vége
    Elágazás vége
Ha hova=4 akkor
    ugorj az utolsó rekordra
Elágazás vége
Frissít
Eljárás vége

A visszatéréshez nyomjuk le az OK parancsgombot.



Nyomógomb ablak tartalma a rekordmutató mozgatását segítő parancsgombok felhelyezéséhez

Majd a következő ablakban is nyomjuk le az Okgombot.



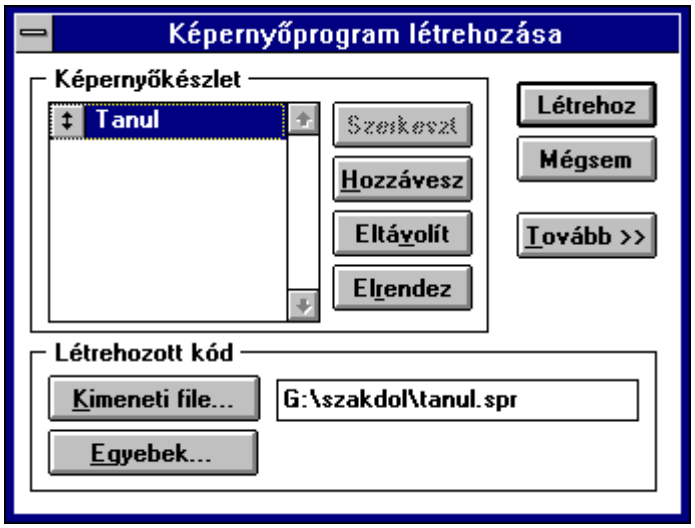
A képernyőformátum-tervező ablak tartalma a rekordmutató mozgatását segítő parancsgombok felhelyezése után

Keretek készítéséhez a Téglalap eszközt válasszuk, majd kattintsunk arra a helyre ahová a keret sarkát szeretnénk elhelyezni, és az egér húzásával alakíthatjuk ki a kívánt kereteket, szegélyeket. A színbeállításhoz a keret kijelölése után az Objektum főmenü Kitöltés színe almenüpontot kell választanunk, ahol a kívánt színt állíthatjuk be. A szöveg láthatóvá tételéhez válasszuk ismét az Objektum főmenü Takarásba almenüpontját. Így a szöveg láthatóvá válik, és a szöveg kerül előtérbe.

A képernyő elkészült. Ahhoz hogy megnézhessük, le kell generálni a forrásnyelvi állományt

8.3.1.1. Generálás:

A Program főmenü Létrehoz almenüpontját válasszuk. Majd a Menti az alábbi fájl változásait? kérdésre válaszoljunk Igennel. Majd a Menti a környezeti információt? kérdésre is.



A Program főmenü Létrehoz almenüpontjánál megjelenő Képernyőprogram létrehozása ablak

A megjelenő ablakban válasszuk a Tovább parancsgombot.

Jelöljük be a Modális képernyő igazológombot. A Létrehoz parancsgombbal létrehozzuk a forrásnyelvű kódot.

Képernyőprogram létrehozása ablak

Ezek után zárjuk be a képernyőformátum-tervező ablakot a File főmenü Bezár almenüpontjával, és a táblázatfájl bezárásához gépeljük be a parancsablakba a

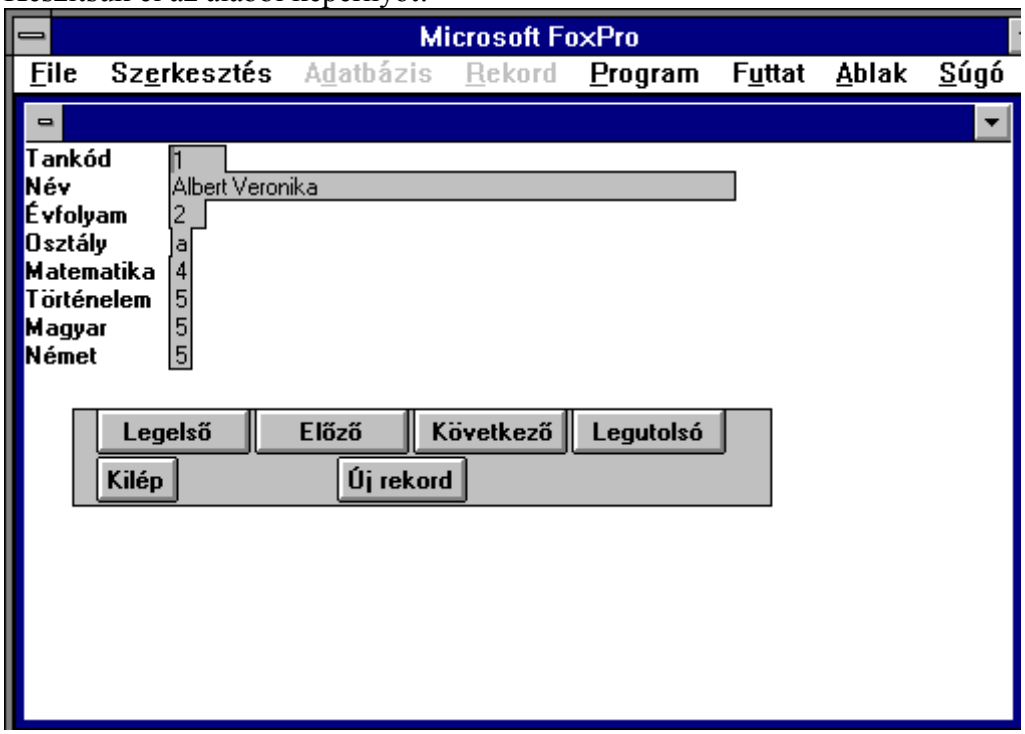
```
CLOSE DATABASES
```

parancsot.

A képernyőprogram futtatásához a Futtat főmenü Képernyő almenüpontját válasszuk. A Futtatandó képernyő listában válasszuk ki a `tanulo.spr` fájlt. és nyomjuk le a Futtat parancsgombot.
Ha hibás a programunk, akkor megjelenik a forráskód, de javítanunk nem itt kell, hanem a képernyőformátum-tervező ablakban, az `.scx` kiterjesztésű képernyő fájlban!

8.3.2. **Képernyő új rekord fölvételéhez**

Készítsük el az alábbi képernyőt!



Képernyő új rekord fölvételéhez (Input mezőkkel)

Nyissuk meg a `tanulo.scx` fájlt és illesszünk hozzá egy Új rekord feliratú parancsgombot, mely lehetővé teszi az Új rekord fölvételét. Mivel most módosítani szeretnénk a `tanulo.dbf` fájlon, hiszen új rekordot szeretnénk hozzáadni, meg kell változtatnunk a mezők jellegét. Ehhez kattintsunk a változtatandó mezőre és a megjelenő Mező ablakban állítsuk be a Mező felirat alatt a Beviteli mező opciót.

Mező ablak(Input mezőkkel)

A Tartomány felirat alatti Alsó ill. Felső parancsgomboknál beírhatnánk a mezőbe bevihető értékek alsó ill. felső határát. Most nincs ilyen.

A Ha parancsgomb mellé belépési feltételt határozhatunk meg, az Érvényes (Valid) parancsgombnál kilépési feltételt. Az Üzenet gombnál segítséget írhatunk ki a status sorba,

mely akkor jelenik meg, ha belép a mezőbe a felhasználó. Hiba gombnál azt az üzenetet írhatjuk be, mely akkor teljesül, ha nem érvényes a mezőbe bevitt érték.
Ha minden mezőt beviteli mezővé alakítottunk, helyezzük fel az Új rekord feliratú nyomógombot. Ehhez kattintsunk a Nyomógomb gombra majd helyezzük el a Képernyőformátum-tervező ablakban a gombot. A Nyomógombfelirat mellé írjuk be: Új rekord
Változó mellé: ujrek
Kattintsunk az Érvényes parancsgombra, és a megjelenő ablakba gépeljük be az alábbi parancsokat:

`APPEND BLANK`	Hozzáfűz egy rekordot
`SHOW GETS`	Frissít
`_CUROBJ=1`	A kurzor az első mezőben villog

Eljárás a Kódcsipet ablakban új rekord hozzáfűzéséhez

Kattintsunk az OK gombra ebben és az előző ablakban is. Ezek után már csak a mentés van hátra és a generálás.
Feladat
456. oldal 131.-136. feladat
507. oldal 404.-406. feladat
508. oldal 407.-409. feladat
508. oldal 412.-413. feladat

8.3.3. **Néhány szó az SQL-ről**

Az SQL strukturált lekérdező nyelv. Adatbázisok kezelésére és lekérdezésére használható szabványos programnyelv. Segítségével adatokat csatolhatunk, importálhatunk, exportálhatunk. A lekérdezés a táblákban tárolt adatokra vonatkozó lekérdezés ill. művelet. Szűrésnek, vagy filternek nevezzük azt amikor csak bizonyos feltételeknek eleget tevő rekordokat jelenítünk meg. Szűrés során joker (helyettesítő) karaktereket is alkalmazhatunk:

?	1 karaktert helyettesít
!	tagadást fejez ki
*	bármennyi karaktert, számot helyettesít
#	1 számot

Több táblából történő lekérdezések egyszerűsítésére ún. nézeteket használunk. A lekérdezés eredményét eredményhalmaznak nevezzük. Értelmezési tartománynak nevezzük a rekordok azon csoportját, melyet SQL kifejezésekkel választottunk ki.

Feladat
440. oldal 36.- 37. feladat
474. oldal 231. feladat
476. oldal 244. feladat
480. oldal 265.-268. feladat
481. oldal 269.-273. feladat
508. oldal 410.-411. feladat

9. Mellékletek

9.1. XOR

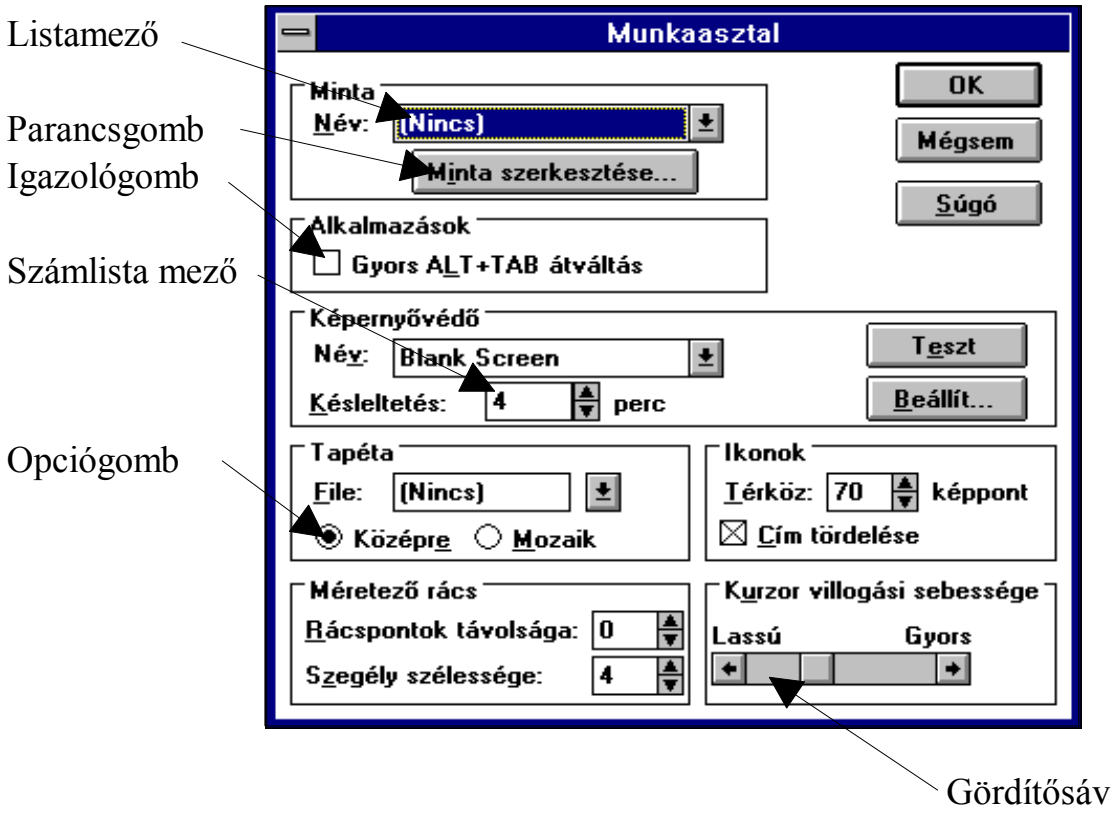
A XOR művelet igazságtáblázata

A	B	A XOR B
1	1	1
1	0	0
0	1	0
0	0	1

A XOR művelet igazságtáblázata

9.2. Windows kezelőszervei

Windowns kezelőszervek elnevezése



Windows kezelőszervek nevei

9.3. Irodalomjegyzék

Bradbeer, R.- DeBono, P. - Laurie, P.: *Műsoron a számítógép*
Műszaki Könyvkiadó, 1984.

Brown, Clement: *Beszélgessünk a számítógépről!*
Műszaki Könyvkiadó, 1975.

Csató István: *A számítógép az ön szolgálatában*
Kossuth Könyvkiadó, 1975.

Erdős Nándor: *A számolás technikája*
Műszaki Könyvkiadó, 1961.

Dr. Fercsik János: *A PASCAL programozási nyelv*
Műszaki Könyvkiadó, 1995.

Farkas Csaba - Benkovics Viktor: *Számítástechnika jegyzet*
Jedlik Oktatási Stúdió, Budapest, 1996.

Fürtön Zsolt - Nagy Zsolt: *Számítógépkezelői ismeretek 1995.*

Gazsó Zoltán: *Adatbáziskezelés FoxPro-ban*
Computerbooks, Budapest, 1996.

Halassy Béla: *Adatmodellezés, adatbázis tervezés*
Központi Statisztikai Hivatal
Nemzetközi Számítástechnikai Oktató és Tájékoztató központ, Budapest, 1980

Halassy Béla: *Adatbázisok kezelésének alapvető kérdései 1987*
Központi Statisztikai Hivatal
Nemzetközi Számítástechnikai Oktató és Tájékoztató központ, Budapest, 1980

Irhás János: *A fejszámolás titkai*
Pannon Nyomda, Vesztprém, 1941.

Dr. Kocsis András: *Programozás BASIC nyelven*
SZÁMALK, 1983.

Makszimovics, Gennagyij: *A csodálatos számítógép*
Kárpáti Kiadó, Budapest-Uzsgorod, 1981.

Cary N. Prague - James E. Hagmit: *Adatbázis - kezelés programozás dBASE II.*
Műszaki Könyvkiadó, Budapest, 1986.

Sain Márton: *Matematikatörténeti ABC*
Tankönyvkiadó, 1980.

Simonovits Miklós: *Számítástechnika (a speciális matematikai osztályok részére)*
Tankönyvkiadó, 1985.

Simonyi Károly: *A fizika kultúrtörténete*
Gondolat, 1981.

Szelezsán János: *Adatbázisok*
LSI Oktatóközpont
A mikroelektronika alkalmazásának kultúrájáért alapítvány.

Szerkesztő: Szenes Katalin: *dBASE III Plus Novell netware kapcsolat*
Foxbase + Clipper (I-II kötet)

Szikszai Csaba: *Barátunk a számítógép*
Móra Könyvkiadó, 1985.

Szlávi Péter: *A számítógépről népszerű stílusban*
ELTE TTK Általános Számítástudományi Tanszék, μlógia 5. 1995.

Szűcs Ervin: *Komputervilág*
Móra Könyvkiadó, 1995.

Szűcs Pál szerkesztésében: *Mikroszámítógépek a tanítási-tanulási folyamatban Magyarországon*
Országos Oktatástechnikai Központ, 1985.
Vágó Árpád: *FoxPro For Windows*
Gyakorlati programozás
LSI Oktatóközpont A mikroelektronika alkalmazásának kultúrájáért alapítvány.
Varga Kornél: *Az adatbázis-kezelés alapjai*
Nemzeti Tankönyvkiadó, Budapest, 1996. 74530
Varga Kornél: *Bevezetés az adatfeldolgozásba*
Nemzeti Tankönyvkiadó, Budapest, 1996. 73530

Feladatbank II.
Számítógépkezelő, számítástechnikai szoftverüzemeltető
Országos Közoktatási Szolgáltató Iroda, 1995.