Táblázatkezelés

Táblázatkezelés

Gál Zoltán

2018-11-02

Tartalomjegyzék

1. Alapfogalmak

A táblázatkezelő alkalmazások segítségével adatokat rendezhetünk el táblázatos formában, illetve ezeken az adatokon számításokat végezhetünk. A legelterjedtebb táblázatkezelő programok a *Microsoft Office* részét képező **Excel**, az *Apple* **Numbers** alkalmazása, valamint a **LibreOffice Calc**. Ezek közül ez utóbbi a nyílt forráskódú, így annak használatán fogjuk megismerni a táblázatkezelő alkalmazások használatát.

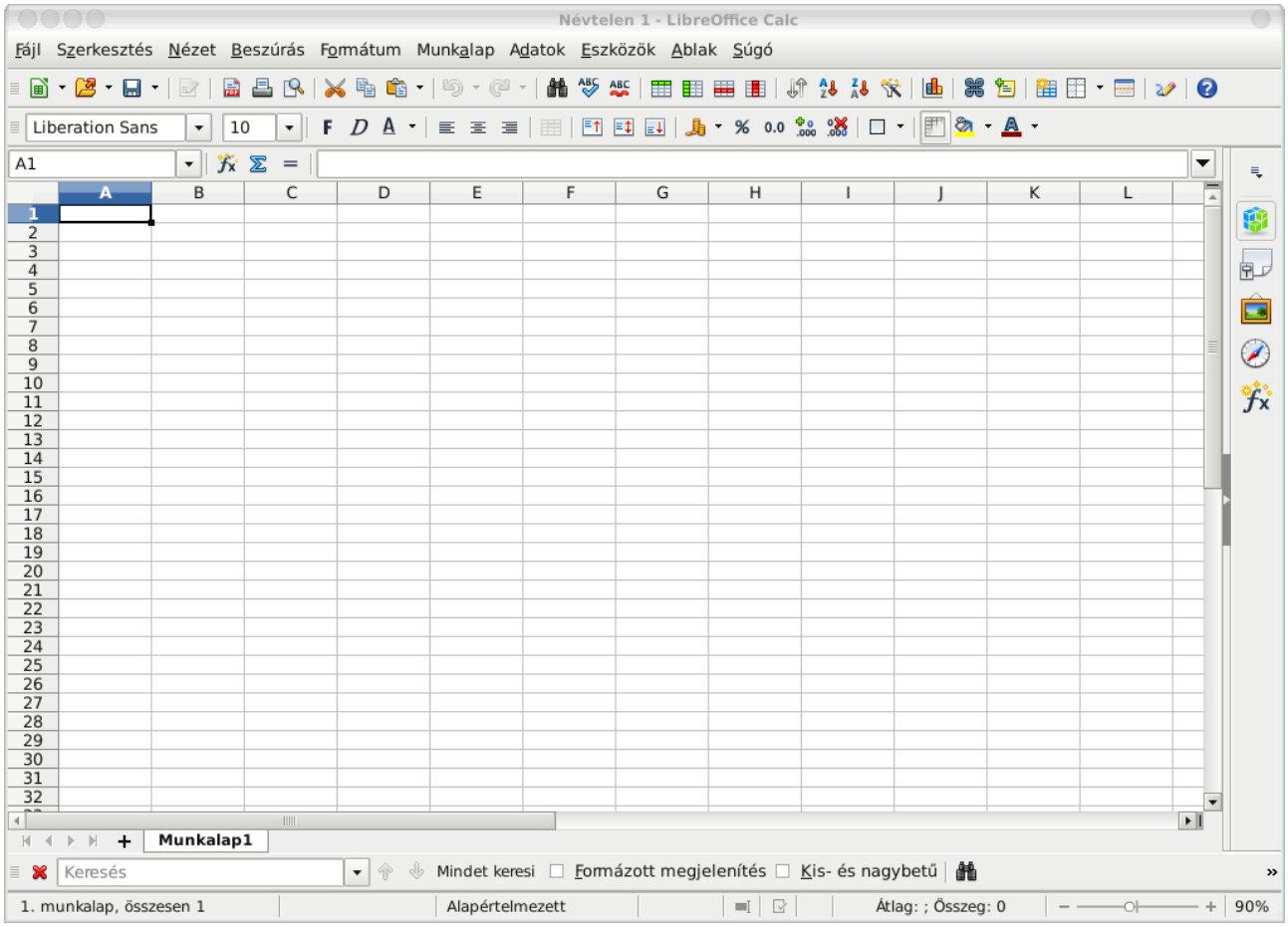
1.1. A Calc funkcionális részei

A program elindítása után a 1. ábrán látható képhez hasonló látványt kell látnunk.

Mint látható, a program hasonlít is meg nem is a Writerhez. A legfeltűnőbb különbség, hogy itt a fő munkaterületen egy téglalapokból álló rács jelenik meg. Ez a téglalapokból álló rács az a táblázat, amelyben dolgozni lehet. A téglalapok valójában a táblázat *cellái*.

Mint látható, a táblázat *oszlopai* balról jobbra haladva az angol ábécé betűivel vannak jelölve, míg a *sorai* fentről lefelé növekvő sorszámokat kaptak.

A program két eszköztára is ismerős lehet a szövegszerkesztőből. Elnevezésük itt is „*Standard*” és „*Formázás*”. A rajtuk levő gombok azonban már kissé eltérnek a szövegszerkesztésben használttól. A képen láthatóhoz képest az eszköztár gombjai azért is eltérhetnek, mert az egyes



1. ábra: A LibreOffice Calc program ablaka

programverziókban a fejlesztők hajlamosak azokat össze-vissza keverni – másrészt a kinézetüket a használt gombstílus is meghatározza, a képen a *Galaxy* stílus látható. A megjelenő gombokat az *Eszközök → Testreszabás* menüben lehet beállítani.

A „*Standard*” és a „*Formázás*” eszköztárak alatt van egy „*Képlet eszköztár*” nevű sor is, amely kiemelve látható a 2. ábrán.

2. ábra: A képlet-eszköztár

Ennek az eszköztárnak a baloldalán levő legördülő lista az **aktív cella** nevét mutatja. Mint a fönti ábrából látható, a cellák egyike – alaphelyzetben az **A oszlop** és az **1. sor** találkozásánál levő cella – vastagabb szegéllyel jelenik meg. Ez a cella az **aktív cella**. Jelentése olyan, mint a beviteli kurzoré: ide történhet az adatbevitel a billentyűzetről. A cella nevét mutató listát *névdoboz*nak hívja a Calc. Az itt látható név mutatja, hogy a cellák neve hogyan alakul ki, hogyan lehet rájuk a későbbiekben például a képletekben hivatkozni: először az oszlop betűjele, majd a sor sorszáma következik, amelyek metszéspontjában a cella található.

***Próbáld ki**: Kattints rá valamelyik másik cellára az egér bal gombjával! Ekkor ez a cella lesz az aktuális cella. A cella nevét a névdobozban leolvashatod. Próbáld meg aktuálissá tenni a **D8** cellát!*

Az aktuális cella kiválasztása több módon is megtörténhet:

- Az egér bal gombjával rákattintva.
- A kurzorbillentyűkkel, az *Enter* vagy a *Tabulátor billentyűkkel mozogva. Ez utóbbi esetben az Enter illetve a Tabulátor billentyűket a SHIFT* billentyűvel együtt használva ellenkező irányú mozgás érhető el.
- A névdobozba beírva a cella nevét, azonnal ez lesz az aktív cella.

A névdoboznak van még más funkciója is, amiről később lesz szó.

A *Képlet eszköztár* (2. ábra) jobb szélén látható hosszú, fehér téglalap Excelbeli neve *szerkesztőléc*, a Calc „*Beviteli mező*” néven ismeri. Ez mutatja az aktuális cella tartalmát (ha van neki), és egyben itt módosítani is lehet a tartalmat. Amennyiben csak simán gépelni kezdünk, akkor felülírjuk az aktuális cella pillanatnyi tartalmát, míg a *beviteli mezőn* keresztül módosíthatjuk is azt annak teljes törlése nélkül. Amennyiben a cella helyén szeretnénk módosítani a tartalmát, azt megtehetjük a cellára történő dupla kattintással...

3. Ábra

Bármilyen módon kezdünk is gépelni a cellába, a beviteli mező bal oldalán a szumma és az egyenlőségjel gombja helyén egy piros kereszt és egy zöld pipa jelenik meg (3. ábra) – *Galaxy* gombstílus esetén –, amelyek az adatbevitel befejezésére alkalmasak: A piros keresztre kattintva – vagy az ESC billentyűt lenyomva – semmissé lehet tenni a bevitelt. Ekkor a cella tartalma az marad, ami eredetileg is volt. A zöld pipára kattintva érvényesíteni lehet a bevitelt olyan módon, hogy közben a cella marad az aktív cella. Ezzel szemben az *Enter* billentyűt lenyomva a következő sorban, a *Tabulátort* lenyomva a következő oszlopban levő cella lesz az adatbevitelt követően az aktuális cella.

***Próbáld ki**: Legyen az A1 az aktív cella, majd gépeld be az „egy” szót a cellába! Különböző módszerekkel írd át a cella értékét a „kettő”, „három” stb. értékekre, végül írd vissza az „egy” értékre! Próbáld ki közben az adatbevitel visszavonását is!*

Még nem esett szó az ablak alján található elemekről.

4. Ábra: Munkalapok

A vízszintes gördítősáv alatt található fülek[1] jelzik, hogy egynél több táblázatunk is lehet a doku-

1 *Ezek kezdetbeli számát szintén a program beállításai határozzák meg, ahogyan az eszköztár gombjainak kinézetét*

mentumban. Ezek neve az ábrán láthatóan alapértelmezésben „*Munkalap1*”, „*Munkalap2*”, stb. Azonban ezeket a munkalapokat átnevezhetjük majdnem tetszőleges névre. A „*majdnem*” oka, hogy a használható karakterek halmaza korlátozva van a betűkre, számokra, az aláhúzásjelre és a szóközre.

Az átnevezés nagyon egyszerű: kattints duplán az aktuális (a többitől eltérő megjelenésű) munkalap fülére és az előbukkanó párbeszédablakba írd be az új nevet! A fülre jobb egér gombbal kattintva további lehetőségeket kaphatunk: új munkalapot lehet beszúrni, törölni lehet munkalapot, át lehet helyezni (az egérrel megfogva szintén át lehet helyezni a munkalapot), sőt akár még ki is lehet színezni, amely egy sok munkalapból álló dokumentum esetén a munkalapok egyfajta csoportosítására is alkalmas.

Mielőtt tovább mennénk, tulajdonképpen hány munkalapja, hány sora és hány oszlopa van a dokumentumnak és vajon mennyi helyet igényel mindezek tárolása?

Nos, magát a dokumentumot, amit lemezre menthetünk **munkafüzet**nek hívjuk. A munkafüzet legalább egy munkalapot tartalmaz. Eredetileg legfeljebb 256 munkalapból állhatott egy munkafüzet, de azóta ez a korlát valószínűleg nagyobb lett. Egy munkafüzet sorainak és oszlopainak száma programverziónként változhat. Aki akarja, kiszámolhatja, figyelembe véve, hogy az angol ábécé 26 betűje amikor elfogy, akkor betűpárokkal folytatódik a jelölés: **AA**, **AB**, **AZ**, **BA**, **BB**, **BZ**, **CA**, … Egészen az *OpenOffice.org 2.x* sorozatáig és az *Excel* 2003-as verziójáig **IV** volt az utolsó oszlop neve, ami 256 oszlopt jelentett. Bár ez sem szokott elfogyni, a *3.1-es OpenOffice.org* utolsó oszlopának neve: **AMJ** lett! A sorok száma azonban még mindig megmaradt a $2^{16} = 65\,536$ (a kettő hatványok használatának a cellahivatkozások belső tárolásánál van jelentősége, hiszen azok binárisan kerülnek tárolásra). A LibreOffice 5.4-es verziójában már a sorok száma is megemelkedett – az Excelben az oszlopok száma is tovább nőtt – egészen $2^{20} = 1\,048\,576$-ra.

Ha csak azt számolnánk, hogy egy-egy cellában négy byte méretű adatokat helyezünk el, akkor is megtelne egyetlen munkafüzettel egy egész merevlemez! Hogy ez mégsem így van, az annak köszönhető, hogy a program mindig csak azokat a cellákat tárolja el, amelyekben található valamilyen tartalom, vagy valamely jellemzője a cellának eltér az alapértelmezettől (például formázás történt rajta).

1.2. Munkafüzetek formátumai

Az egyes táblázatkezelő programok mind más-más formátumokat használnak a munkafüzetek tárolására. Az *Excel* eredetileg az `.xls` kiterjesztésű formátumot használta. Ez nem nyílt formátum, ennek ellenére egy idő elteltével az *OpenOffice.org* is képessé vált a használatára. Talán ez volt az eredeti indítéka, talán nem – a *Microsoft* a 2007-es Office-tól kezdve új formátumot vezetett be, amely az `.xlsx` kiterjesztést kapta. Az OpenOffice.org és az abból létrejött LibreOffice elég hamar ennek használatára is képessé vált.

Az *OpenOffice.org* és a *LibreOffice* az **Open Document Format** specifikációját használja minden dokumentumfajta esetében. Ez a táblázatkezelő esetén az *Open Document Spreadsheet*, amelyet `.ods` kiterjesztésű állományokban találhatunk.

Az *Apple* szoftvereket használók megszokhatták, hogy az egyes programok általában a dokumentumok tárolására olyan formátumot használnak, amelynek állományai a program nevét viselik a kiterjesztésükben, így a *Numbers* által használt formátumú állományok kiterjesztése: `.numbers`. A `.numbers` állományokat a LibreOffice nem tudja kezelni.

A mentésre szolgáló párbeszédablak operációs rendszertől függ, azonban a főbb működési elv minden esetben azonos: a *File → Mentés (másként)* menüponttal hívható elő az általánosan erre el-

és a ténylegesen megjelenő gombokat is.

terjedt *Ctrl+S*, illetve *Ctrl+Shift+S* billentyűkombinációk mellett. Az állomány nevének megadása alatt az állomány típusánál mindig figyeljünk a formátumra! Ha nincs másra okunk, akkor mindig az *ODF-formátumot* (`.ods` kiterjesztés) válasszuk és ügyeljünk arra, hogy az automatikus állománynév kiterjesztés be legyen kapcsolva, különben nehezen fogjuk legközelebb megnyitni a dokumentumot! A jelszóval történő mentést csak különösen indokolt esetben használjuk, hiszen ha elfelejtjük a jelszót, akkor soha többé nem leszünk képesek megnyitni a dokumentumot!

1.3. Adatbevitelt segítő lehetőségek

A cellákba történő adatbevitelt több eszközzel is segíti a táblázatkezelő program. Ezeket vesszük most sorra.

1.3.1. Automatikus kitöltés

Ilyen eszköz például az *automatikus kitöltés*. Az aktív cella jobb alsó sarkában egy kis négyzetet lehet találni, ezzel lehet elérni ezt a funkciót. Amennyiben ezt az egérrel megfogva lefelé vagy balra kezdjük húzni, akkor az automatikus kitöltés segítségével lehet azokat a cellákat kitölteni valamilyen értékkel, amelyeket érint a művelet. Balra vagy felfelé húzva a tartalom törlése érhető el az érintett cellákból.

Hogy az automatikus kitöltés pontosan hogyan működik, az több mindentől függ:

1. Amennyiben egy cella van kijelölve és abban egy szám található, akkor a kitöltés hatására egyesével növekvő sorozattal tölthető ki az oszlop vagy a sor. A CTRL billentyű egyidejű lenyomása mellett a cellákban a kiinduló cellában levő érték fog megismétlődni.
2. Szöveget tartalmazó cella esetén, ha a megadott érték egy *rendezett lista* eleme, akkor annak a *rendezett listá*nak a soron következő elemei következnek – szükség esetén elölről kezdve a listát. CTRL esetén ekkor is a kiinduló cella értéke fog ismétlődni.
3. Szöveget tartalmazó cella esetén, ha a megadott érték nem eleme egy *rendezett listá*nak, akkor a CTRL billentyű lenyomott állapotától függetlenül ezt az értéket ismétli meg az új cellákban a program.
4. Ha olyan szöveget tartalmaz a kiinduló cella, amelynek a végén szám található, akkor az előtte levő szöveg ugyan változatlan marad, de a számérték folyamatosan nő. CTRL esetén ekkor is a megadott érték ismétlődik módosítás nélkül.
5. Amennyiben két szomszédos cella van kijelölve a másolni kívánt irányban – tehát oszlopba kitöltés esetén két egymás alatti cella, sorba kitöltés esetén két egymás melletti cella –, akkor a kijelölés jobb alsó sarkában megjelenő kitöltő négyzetet használva számok esetén a két kezdő érték különbségével növekvő számtani sorozatot lehet előállítani. CTRL billentyűvel a két számot fogja a program ismételgetni. Szöveg esetén a két érték ismétlődik mindenképpen, kivéve, ha a két érték ugyanannak a *rendezett listá*nak az egymást követő két eleme. Ez utóbbi esetben ha nem szomszédos elemek, akkor a másodiknak szereplő értéktől folytatódik a lista.
6. Ha két olyan cella van kijelölve, amelyben ugyan szöveg van, de a végén számérték található, akkor ebben az esetben is úgy változik a szöveg utáni szám, hogy számtani sorozat jöjjön létre.
7. Három vagy több cellából álló kijelölés esetén a számtani sorozat különbségeként a program a megadott kiinduló értékek közti különbség átlagaként kapott értéket használja. A CTRL billentyű hatására ekkor is a megadott cellaértékek ismétlődnek.
8. Amennyiben a kijelölés és a kívánt kitöltés iránya egymásra merőleges – azaz két egymás melletti cella kijelölését lefelé vagy két egymás alatti cella kijelölését jobbra akarjuk kitöl-

teni –, akkor a táblázatkezelő az egyes kijelölt cellákra egyenként alkalmazza az automatikus kitöltést.

Automatikus kitöltés egyszerre mindig csak oszlopra vagy sorra alkalmazható. Amennyiben a másik irányban szeretnénk a kitöltést folytatni, akkor el kell engedni a kis négyzetet, majd újra megfogni.

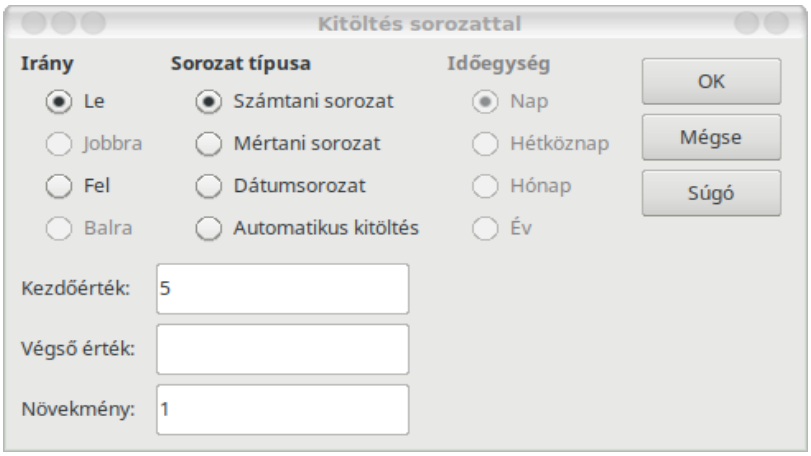
Amennyiben lefelé kell kitölteni egy oszlopot, amelynek a baloldalán levő oszlopban vannak értékek, akkor elég a kitöltő négyzetre kattintani és a program elvégzi az automatikus kitöltést az utolsó olyan sorig, amelyben az egyel balra levő oszlopban van érték.

További automatikus kitöltési lehetőségeket adnak a rendezett listák és a sorozattal kitöltés.

1.3.2. Sorozattal kitöltés

A *Munkalap* menü *Cellák kitöltése* menüpont is használható az automatikus kitöltés eléréséhez, néhány olyan lehetőséggel kiegészítve, amely csak itt érhető el. Ilyen lehetőség a sorozattal kitöltés. Ehhez először ki kell jelölni azt a tartományt, amelynek a celláiba a kitöltés eredményének kerülnie kell. Ezután kell a *Munkalap→Cellák kitöltése→Sorozatok* menüpontot kiválasztani.

A 5. ábrán látható párbeszédablakot adja a menüpont. Mint látható, így nem csak számtani sorozatot, hanem akár mértani sorozatot is gyorsan elő lehet állítani. Az ábrán a kezdőérték már azért van kitöltve, mert a kép készítésekor a kijelölés első cellájában szerepelt az 5 szám. Számtani sorozatnál a kezdőérték és a növekmény megadása szükséges. Mértani sorozat esetén a „növekmény” valójában a két szomszédos elem hányadosaként értelmezendő. Dátumsorozat esetén a jobboldali rész válik elérhetővé és azt lehet beállítani, hogy a növekmény napban, hétköznapban, hónapban vagy évben értendő.



5. ábra: A Kitöltés sorozattal párbeszédablak

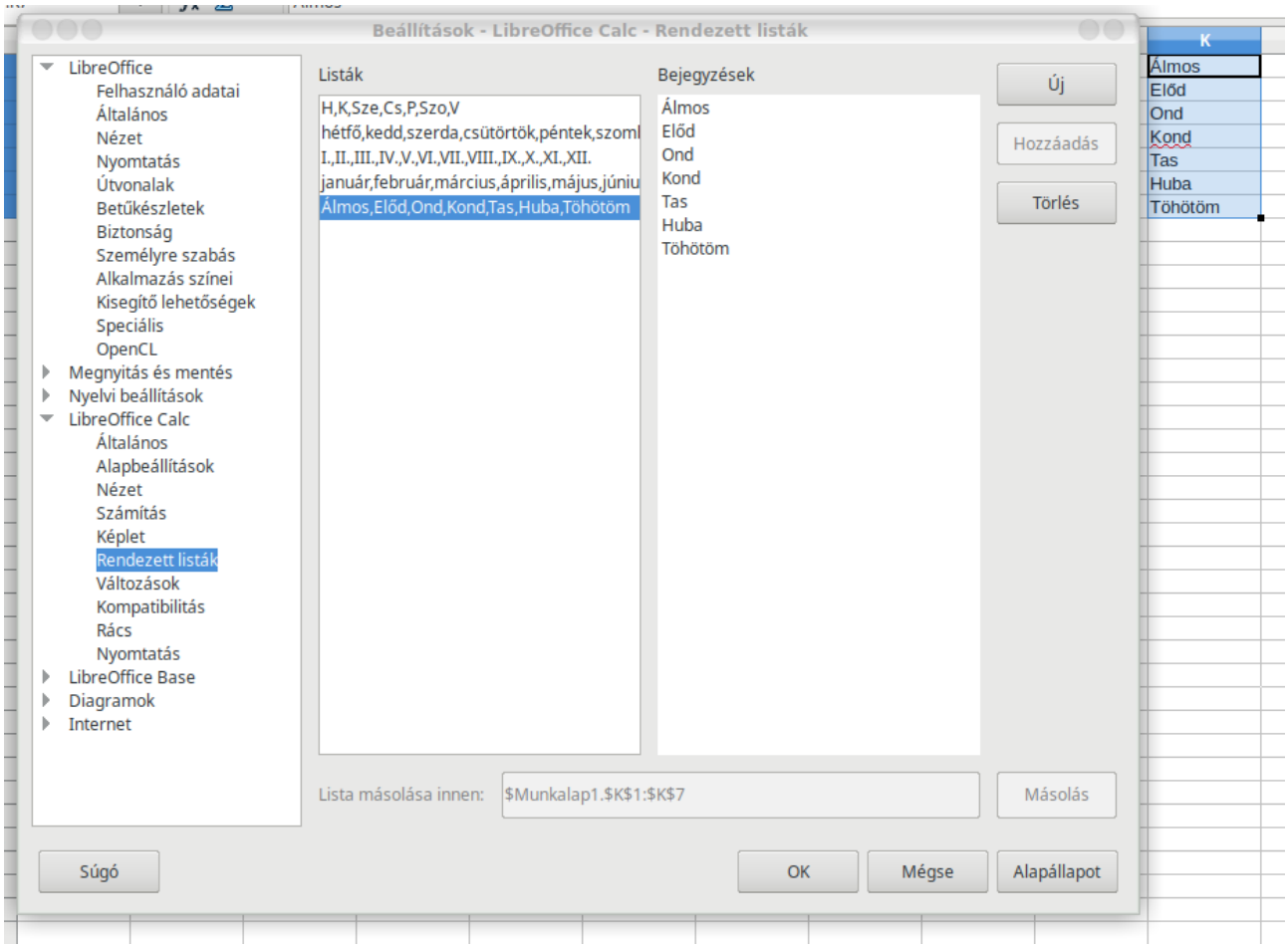
A baloldalon az irányt beállítva akár csökkenő listát is létre lehet hozni – persze számtani sorozatnál a párbeszédablak nélkül is lehetséges ez a megfelelő kezdőértékek megadásával.

Hasznos lehet ez a funkció például olyankor is, amikor dátumokat kell elhelyezni az első oszlopban, de úgy, hogy csak a munkanapok (azaz a hétköznapok) szerepeljenek benne. Erre megoldás például a dátumsorozat választása a kezdődátummal a kezdőértéknél, 1 növekménnyel, időegységnek a „*Hétköznap*” lehetőséget választva.

1.3.3. Rendezett listák

Tegyük fel, hogy a hét napjainak listájára van szükségem egy oszlopban. Az automatikus kitöltés erre is használható: első cellába beírom a „Hétfő” szót és lemásolom. Ugyanezt megtehetem a hónapok neveivel is. Sőt ugyanezt megtehetném a Hét Vezér vagy a Hét Törpe nevével, de ezek alaphelyzetben nem működnek.

A fentiek mind a *rendezett listák* használatára példák. Az első kettő azért működik, mert azok alapból definiált listák. A másik két példát is működőképessé lehet tenni – ahogy bármely saját listát –, a lista definiálásával.



6. ábra: A Hét Vezért tartalmazó rendezett lista definiálása

Ehhez nem kell mást tenni, mint a lista leendő elemeit begépelni egymás alatti cellákba, majd megkeresni a „*Rendezett listák*” részt az *Eszközök→Beállítások* menüpont párbeszédablakában, ahogyan azt a 6. ábra mutatja. A párbeszédablak alján a „*Lista másolása innen:*” beviteli mezője a kijelölt tartomány címét tartalmazza. A mellette levő „*Másolás*” gombra kattintva a kijelölés által megadott lista felvételre kerül.

A párbeszédablakban a „*Listák*” oszlop mutatja a programban pillanatnyilag definiált listákat, amelyek közül bármelyik kiválasztható és szerkeszthető. A kiválasztott lista elemei a jobboldali, „*Bejegyzések*” nevű téglalapban jelennek meg egymás alatt.

Új lista létrehozható az „*Új*” gombra kattintva is. Ekkor egy üres lista jön létre, amelynek elemeit a „*Bejegyzések*” részben kell begépelni, minden sorba a lista egy elemét írva. Bármely kiválasztott lista törölhető is.

Fontos megjegyezni, hogy az itt látható rendezett listák nem az adott dokumentumban vannak definiálva, hanem a LibreOffice adott példányában. Ez azt jelenti, hogy definiálás után bármely munkafüzetet megnyitva a definiált listák használhatók.

1.3.4. Tartomány kitöltése

Egy adott tartományt kijelölve, látható, hogy az egyik cella másképp néz ki, mint a többi. Ez a cella az aktív cella, azzal a különbséggel, hogy nem minden billentyű viselkedik úgy, mintha nem lenne kijelölés. A kijelölésen belül az aktív cella mozgatható, de nem a kurzorgombokkal vagy egérkattintással, mivel ezek megszüntetik a kijelölést.

De a lefelé haladásra az **Enter**, felfelé haladásra a **Shift+Enter**, jobbra haladásra a **Tabulátor**, balra haladásra a **Shift+Tabulátor** billentyűkkel továbbra is lehetőség van. Ezzel együtt az aktív cellába új tartalom is gépelhető, amely felülírja a régit – ha volt. A fenti billentyűk valamelyikével másik cellára átlépve lehet a bevitelt véglegesíteni, az **ESC** billentyű visszavonja az adatbevitelt.

Hogy miért jó ez a lehetőség? Ennek szemléltetésére egy több soros és több oszlopos kijelölés szükséges. Lépegessünk ebben a kijelölésben először lefele az Enter billentyűvel egészen a legalsó sorig. Az utolsó sorban lenyomva az **Enter** billentyűt, láthatjuk, hogy az aktív cella a következő oszlop tetejére kerül. Hasonlóan a **Tabulátor** billentyűvel sorról sorra lehet haladni. A kijelölés valamelyik sarkából pedig a túlsó sarokba lehet gyorsan átjutni.

Ez az adatbeviteli mód akkor hasznos ha egy téglalap alakú tartomány minden cellájába kell sorról sorra vagy oszlopról oszlopra haladni.

1.3.5. Sorok intelligens kitöltése

A sorról sorra történő adatbevitelre kínál egy másik lehetőséget a táblázatkezelő azon tulajdonsága, hogy ha egy soron belül egyik celláról a másik cellára a **(Shift+)Tabulátor** billentyű segítségével lépünk át, akkor amikor az **Enter** billentyűt lenyomjuk, a következő sor azon cellája lesz az aktív cella, amely oszlopából a legutolsó **Tabulátorral** lépegető sorozat elindult.

Például ha a **B2** cellába történő adatbevitel után a **Tabulátorral** lépünk a **C2**-re, onnan újabb adatbevitel után a **D2**-re, majd ott lenyomjuk az **Enter** billentyűt, akkor az aktív cella a **B3** cella lesz. Ellenben, ha a **B2**-ről a **C2**-re a kurzor jobbra gombbal vagy kattintással lépünk, és csak onnan a **D2**-re lépünk a **Tabulátorral**, akkor az **Enter** a **C3** cellára viszi az aktív cellát.

1.4. Adatformátumok

A táblázatkezelő többféle adatformátumot képes kezelni. A legtöbb esetben a bevitel után felismeri annak formátumát és ennek megfelelően jeleníti meg a tartalmat a cellában.

Figyeljük meg, hogy hogyan jelenik meg a cella tartalma az alábbi esetekben:

1. Szöveg;
2. Egész szám;
3. Tizedes vesszőt tartalmazó szám;
4. Két szám ponttal elválasztva;
5. Két szám kettősponttal elválasztva.

Az első elég egyértelmű: szöveg típusú adat, amelyet általában a cellában balra szokás igazítani. A második szintén egyértelmű: szám típusú adat, amelyet jobbra szokás igazítani.[2] A továbbiakhoz azonban már kell némi magyarázat.

Tört számot úgy írunk be, hogy az egész rész és a tizedes rész közé vesszőt írunk. Ez nekünk magyaroknak természetes, de az angolok tizedes pontot használnak, így sokan alapból úgy gondolhatják, hogy a számítógép mindenhol tizedes pontot használ. **A táblázatkezelésben a tizedes vesszőt használjuk.**

Tehát a harmadik példa is szám, de ennek van tört része is.

A dátum és idő adatokat is számként kezeli a program, de dátum vagy idő formátumban jeleníti meg. Erre példa a negyedik, illetve az ötödik pont. Dátumot a legegyszerűbben az *év.hónap.nap*

2 *Sajnos a Microsoft Excelhez készült segédanyagokban szereplő mintapéldák nyomán nagyon mélyen beleivódott sokak elméjébe néhány tipográfiai szempontból egyenesen rémes formázási szokás, amelytől nehéz megszabadulni. Nagyon sok táblázatkezelést tanító tanár, tankönyv – és sajnos emiatt az informatika érettségi feladatai is – hibás formázási gyakorlatot alkalmaz. Ne kövessük a rossz példát! Szokjuk meg: szöveget balra, számértéket jobbra igazítunk a táblázatokban!*

alakban lehet megadni. Ekkor mind az *év*, mint a *hónap*, mind a *nap* egy-egy szám, amelyek közé pontot kell begépelni. **A végére azonban tilos ilyenkor begépelni a pontot, ahogy szóközt sem szabad közben begépelni.** Formázással később el lehet érni, hogy a cella értéke a magyar helyesírás szabályai szerint jelenjen meg. A dátumok belső tárolására később majd még visszatérünk, egyelőre legyen elég annyi, hogy számként tárolódnak a dátumok és az idők is. A bevitelkor a pontok helyett lehet `/` karaktert is használni, akkor azonban elképzelhető, hogy a program nem a magyar, hanem például az angol vagy az amerikai írásmód szerinti sorrendben értelmezi a megadott értékeket. Természetesen ha csak két számjegyet adunk meg ponttal (vagy `/` jellel) elválasztva, akkor alapértelmezés szerint az aktuális évben megadott hónapnak és napnak tekinti azokat a program.

Idő adat hasonlóan adható meg, de az elválasztójel a kettőspont (`:`) karakter lesz: Amennyiben két számadat szerepel, akkor *óra*:*perc* értelmezésben, teljes idő megadása esetén *óra*:*perc*:*másodperc* alakban kell megadni.

Megadható együtt dátum és idő is, mivel a belső ábrázolás szerint a dátum egy egész szám, az idő pedig az *egy* törtrészeként kerül tárolásra. Ekkor az (*év*.)*hónap*.*nap óra*:*perc*(:*másodperc*) alakot kell használni (a dátum és az idő között egy szóköz szerepel.

Szintén a szám adattípus lehetséges megjelenési formája a *százalék* és a *pénznem* is. A százalék a számérték százszorosát mutatja, be is írható bárhova, ahol szám lehet a `szám%` formában. Ekkor a megadott szám századrészét fogja eredményezni a százalékjel.

Ha beírjuk egy cellába egy számot és szóközt követően a „*Ft*” karaktereket, akkor a számot tartalmazó cella alapból pénznem formátumban fog megjelenni.[3] A szóközt nem szabad kihagyni!

További adattípus még a szöveg és a szám mellett a logikai típus is. Ennek két lehetséges értéke van: `Igaz` és `Hamis`. Ha a program angol függvénynevek használatára van állítva, akkor is magyarul kell beírni! A hamis értéknek megfelel a 0, az igaz értéknek bármely nem nulla érték. Ellenkező irányban: ha olyan helyen szerepel, ahol számnak kellene lennie, akkor a hamis érték 0, az igaz érték 1 értéknek fog számítani.

3 *Ez természetesen akkor igaz, ha pénznemként a Forint van beállítva.*

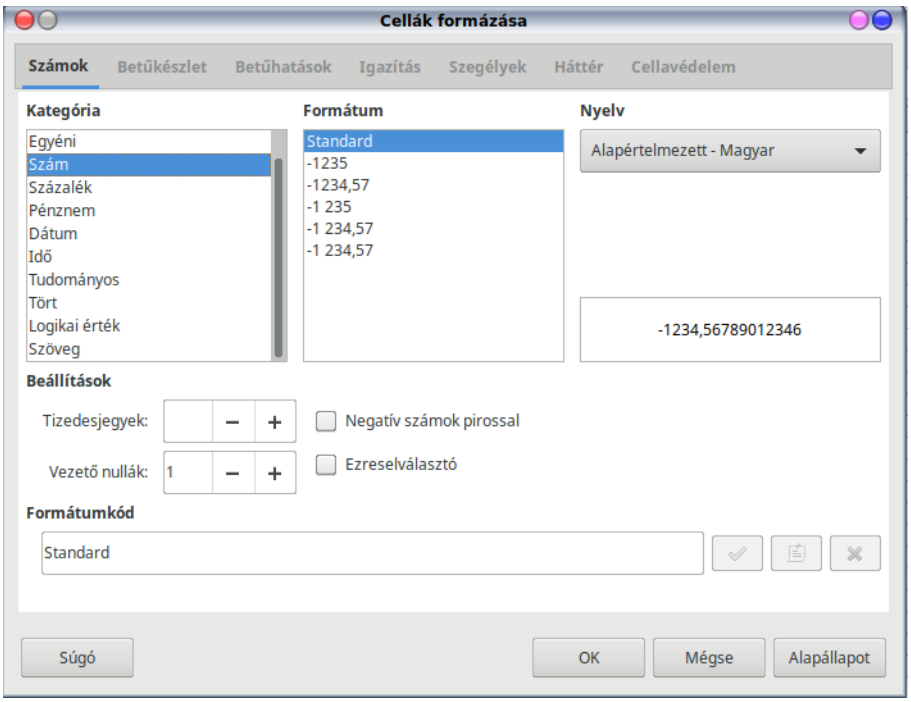
2. Cellaformázás

A táblázat celláit a tartalmuk megjelenésében és a cella díszítésében is testre lehet szabni.

2.1. Cellaformátumok

Amennyiben nem szöveget tartalmaz a cella, akkor a tartalom megjelenítéséhez nagyon sokféle lehetőséget kínál a táblázatkezelő program.

A *Formátum → Cellák* menüpont által előhívott „*Cellák formázása*” párbeszédablak „*Számok*” lapján (7. ábra) lehet meghatározni, hogyan jelenjen meg a cella tartalma. Természetesen további beállítások is lehetségesek (Betűkészlet, Betűhatások, Igazítás), amelyek a megjelenést befolyásolják, de ezúttal a tartalom formai megjelenésére koncentrálunk.



7. ábra: A cellatartalom megjelenését szabályozó párbeszédablak

A „*Kategória*” részen a cellában levő adat típusát adhatjuk meg. De itt nem csak az alaptípusok választhatóak, hanem az egyes alaptípusok altípusokba vannak csoportosítva:

- Szám
  - Szám: Normál számok megjelenítésére való.
  - Százalék: A számot, mint százalékértéket jeleníti meg, ami magában foglalja a százzal szorzást is.
  - Pénznem: A számot pénzértékként formázva lehet megjeleníteni. Két dologban különbözik a Szám kategóriától: egyrészt beállítható a megjelenítendő pénznem, amely egyes pénzeknél a szám előtt, más esetben a szám után jelenik meg. Másrészt lehetőség van a negatív értékek piros színnel megjelenítésére is.
  - Tudományos: A normál alak szokásos számítógépes jelölését lehet vele megvalósítani.
  - Tört: Egész és törtérész megjelenítésére alkalmas.
- Dátum/Idő (belül a program ezt is számként tárolja)
  - Dátum: A dátum különböző megjelenési módjait lehet beállítani.
  - Idő: Ugyanez vonatkozik az egy napon belüli időpont vagy az egy napnál hosszabb időtartamok megadására. Van a formátumok között olyan változatai is, amellyel a dátumot és az időt is meg lehet egyszerre jeleníteni.

Amennyiben a fenti kategóriákban található lehetőségek egyike sem felel meg a célnak, akkor természetesen lehetőség van további formátumok definiálására is. Például ha valamilyen mértékegységet szeretnénk megjeleníteni a számérték mögött – ilyet a pénznem tud, de csak pénzértékekre –, akkor egy egyéni formátum beállítására van szükségünk.

Az egyéni formátumkód kialakításában sokat segít, hogy bármely előre beállított formátum kiválasztása esetén a „*Formátumkód*” mezőben megjelenik az adott formátumot definiáló formátumkód. Emellett természetesen a Súgó gombra kattintva elolvashatjuk a formátumkódok kialakítására vonatkozó szabályokat is, amelyből a fontosabb lehetőségek az alábbiakban olvashatóak.

További segítség még, ha a kívánt formátum kialakítását a „*Formátumkód*” mező feletti beállítási lehetőségekkel kezdjük. Ezek a beállítások automatikusan megváltoztatják az aktuális formátumkódot, amelyet ezután tetszés szerint tovább lehet alakítani.

Ezek a beállítási lehetőségek a következők:

- **Tizedesjegyek**: Megadja, hogy a számnak mennyi tizedesjegye jelenjen meg. Ha a mező üres, akkor kerekítés nélkül a pontos számérték jelenik meg. Ha értéke nulla, akkor a cella értéke egészre kerekítve fog megjelenni, pozitív érték esetén a beállított számú tizedesjegyre lesz kerekítve a cella értéke. Figyelem: Ez csak a cella megjelenését szabályozza, de a cellában továbbra is a pontos érték található. A cella értékének kerekítését a megfelelő függvénnyel kell elvégezni.
- **Vezető nullák**: A tizedesjegyek ellenkezőjeként, az egészrészre határozható meg, hogy mennyi számjegy jelenjen meg kötelezően. Természetesen ha a szám az itt beállított értéknél többet igényel, akkor több jegy fog megjelenni. Ám ha kevesebbet, akkor is lesz legalább ennyi megjelenítve – csak éppen ebben az esetben vezető nullákat használva.
- **Negatív számok pirossal**: Elsősorban pénznemnél szokás a színnel jelölni a negatív értékeket, de bármely szám esetén bekapcsolható ez a lehetőség.
- **Ezreselválasztó**: A szám számjegyeit hármas csoportokba rendezhetjük az egyes helyiértéktől kezdve ezzel a kapcsolóval. Minden csoport között egy-egy szóköz jelenik meg a számban. Nagyobb számok esetében segíti ez a megjelenítés a szám leolvasását, hiszen a nagyságrend könnyebben megállapítható.

2.1.1. Formátumkódok

A számformátumkódok legfeljebb négy, egymástól pontosvesszővel (`;`) elválasztott részből állhatnak.

1. Egy két részből álló számformátumkódban az első rész pozitív értékekre (beleértve a nullát is), míg a második rész negatív értékekre vonatkozik.
2. Egy három részből álló számformátumkódban az első rész pozitív értékekre, a második részt negatív értékekre, még a harmadik rész nullára vonatkozik.
3. Akár feltételeket is lehet a három részhez adni, ilyenkor a formátumot csak a feltételek megléte esetén lehet használni.
4. A negyedik szakasz akkor lép életbe, ha a tartalom nem érték, hanem szöveg. A tartalmat ilyenkor egy `@` jel képviseli.

A fenti öt bekezdés a LibreOffice súgójából van szó szerint kimásolva...

A számunkra lényeges rész, hogy mit lehet a formátumkódba írni, és annak mi lesz a jelentése.

Helykitöltők

A formátumkódban szerepelő nulla (`0`), számjel (#) vagy kérdőjel (`?`) jelzi egy számjegy helyét, ezért ezeket **helykitöltő**nek lehet nevezni. A # csak a lényeges számjegyeket jeleníti meg, azaz ha a számnak oda már nem jut értékes helyiértéke, akkor nem jelenik meg a nulla érték, ám ha esik ide értékes jegy, az megjelenik. Egészrésznél az ezres csoportosításhoz használatos, törtrésznél az adható meg, hogy ez a tizedesjegy megjelenhet még, de nem kötelező megjeleníteni, ha már nulla lenne.

Hasonló hatása van a `?` karakternek, ám ebben az esetben a nulla érték miatt nem megjelenő számjegy helyett szóköz kerül megjelenítésre. Ez akkor hasznos, ha változó tizedesjeggyel úgy akarunk egymás alatti cellákban számokat megjeleníteni, hogy a tizedesvessző mindig ugyanarra a pozícióra kerüljön. Ehhez természetesen a cellákat jobbra is kell igazítani és olyan betűtípust kell alkalmazni, amelyben állandó a betűszélesség (monotype). A 8. ábra a `0,????` formátumkódot mutatja jobbra igazított *Courier New* betűtípus alkalmazása mellett.

	A
1	1,
2	1,23
3	2,356
4	3,54
5	1,02
6	3,
7	56,5
8	13,6
9	3213,321
10	2,1

8. ábra

Amennyiben a formátumkódban valahol a `0` karakter szerepel, az azt jelenti, hogy azon a helyen mindenképpen meg kell jelennie egy számjegynek. Ez azt jelenti, hogy ha a számnak oda már nem esne értékes helyiértéke, akkor is megjelenik a nulla érték ezen a helyiértéken. Ezzel lehet adott tizedesjegyre kerekítetten megjeleníteni a számokat: a tizedesvessző jobboldalára annyi `0` karaktert kell írni a formátumkódban, ahány tizedesjegyre kerekítve a számot meg kell jeleníteni. Fontos megjegyezni, hogy ez a kerekítés csak a szám megjelenítésére fog vonatkozni, míg a cella értéke továbbra is az eredeti érték marad.

A # karakter használható az ezres csoportosítás megadására is: a „`#  ###`” karaktersorozat (benne egy szóközzel azt eredményezi, hogy a tizedesvessző baloldalán minden harmadik számjegy után egy szóköz következik. Mint látható, ebben az esetben nem kell a szám várható teljes hosszára formátumkódot írni, mert a program automatikusan kiterjeszti a kódot a nagyobb nagyságrendű számokra. Amennyiben az angoloknál szokásos módon, az ezres csoportosításban a szóköz helyett vesszőt szeretnénk megjeleníteni – vigyázat: ekkor zavaró lesz a tizedesvessző –, akkor a formátumkód „`#,###`” alakra változik. A formátumkódokban a jobboldali karakter a `0` karakterre is cserélhető.

Szöveg a formátumkódban

A formátumkódba – és vele a cellában megjelenő tartalomba – tetszőleges szöveget is be lehet szúrni. Ez teszi lehetővé a mértékegység megjelenítését a cellában, hiszen ha a cellába begépelnénk, akkor a cella szöveg típusú lenne. Másrészt akár olyan trükköket is meg lehet oldani vele, hogy a cellában egy szöveg jelenjen meg, amelyen belül valahol van egy szám – ez a szám a cella valódi tartalma.

A szöveg beillesztéséhez a formátumkódba a szöveget **dupla idézőjelek** – a köznapi életben „*macskakörömnek*” csúfolt karakterek – közé kell írni. Az ilyen idézőjeles szöveg a formátumkódban gyakorlatilag bárhol elhelyezhető, akár ugyanabban a formátumkódban több helyen is.

Például ha méterben megadott adatok vannak a cellában, akkor a mértékegység megjeleníthető a „`0" méter"`” vagy a „`0,00" m"`” formátumkóddal. (Előbbi egészre kerekítve, utóbbi két tizedesjegyre kerekítve írja ki a számértéket, előbbinél kiírva a „*méter*” szót, utóbbinál a mértékegység hivatalos rövidítését megjelenítve.) Ha nem akarunk kerekíteni, akkor a „`Standard" m"`” formátumkódot lehet használni.

Dátum- és időformátumok

A dátumot a program az 1989. december 3. óta eltelt napokban, az időadatokat a nap törtrészében tárolja (tehát a 0,25 reggel hat órát jelent például). Így tehát a cellában egy számérték van, amelynek a megjelenítésére lehet valamilyen dátum- vagy időformátumot alkalmazni. Ehhez a formátumkódban a következő karakterek használhatóak:

- Y: év
- M: hónap
- D: nap
- H: óra
- M: perc (igen, újra az M)
- S: másodperc

A fentiek nem következhetnek tetszőleges sorrendben, így a hónap és a perc nem fog összekeveredni.

Az egyes karakterek száma is lényeges, mivel különböző formájú megjelenéseket jelentenek:

Formátum	**Formátumkód**
Hónap „3” formátumban	M
Hónap „03” formátumban	MM
A hónap neve hárombetűs rövidítésben („Jan” – „Dec”)	MMM
A hónap neve teljesen kiírva	MMMM
A hónap nevének első betűje (nem minden hónapnak van egyedi kezdőbetűje!)	MMMMM
Nap sorszáma „2” formában	D
Nap kétjegyű sorszáma „02” formában	DD
A nap nevének három betűs rövidítése („Hét” – „Vas”)	NN vagy DDD vagy AAA
A nap neve teljesen kiírva	NNN vagy DDDD vagy AAAA
Nap neve és utána vessző („Vasárnap,”)	NNNN
Év kétjegyű változatban (00-99)	YY
Év négyjegyű változatban (1900-2078)	YYYY
Naptári hét sorszáma	WW

Időadat formátumai:

Formátum	**Formátumkód**
Órák 0-23 formátumban	H
Órák 00-23 formátumban	HH
Órák ha nem akarunk 24:00-nál megállni	[HH]
Percek 0-59 formátumban	M
Percek 00-59 formátumban	MM

Formátum	Formátumkód
Percek ha nem akarunk 60 percnél megállni	[MM]
Másodpercek 0-59 formátumban	S
Másodpercek 00-59 formátumban	SS
Másodpercek ha nem akarunk 60 másodpercnél megállni	[SS]

Amennyiben századmásodperceket is szeretnénk megjeleníteni, akkor például a „`HH:MM:SS.00`” formátumkódot lehet használni. Ekkor az időérték például ilyen lesz: „`01:03:55.19`”.

A szögletes zárójeles változat azt eredményezi, hogy az arra a részre eső időrészt már nem váltja tovább a következő időegysére. Így ennek használata egyedül a legnagyobb megjelenítendő időegységre lehetséges. Például a „`[HH]:MM`” formátumkódban „`26:30`” formában megjelenő időadat – amely több, mint egy nap, tehát időtartamként értelmezendő – a „`HH:MM`” formátumkódot használva „`02:30`” formában fog megjelenni.

2.2. Szegélyezés

A szövegszerkesztésben megismert szegélyezési módszerek a táblázatkezelőben is elérhetőek, azzal a különbsséggel, hogy itt a használatuknak nagyobb jelentősége van, hiszen ha a táblázatunkat – vagy annak egy megfelelő részét – szeretnénk kinyomtatni, akkor ott a szegélyek is fontos szerepet kapnak.

A táblázatkezelőben alahelyzetben a cellák között szürke vonalak láthatóak. Ezeket ne tévesszük össze a szegélyekkel! Ezek csak a cellák határát jelzik. Akit ez zavar, a 9. ábrán bekapcsolva látható gomb kikapcsolásával el lehet ezeket a vonalakat tüntetni. Ám ekkor kifejezetten nehéz megtalálni az egyes cellájat.

9. ábra

A szegélyezés a *Formátum* → *Cellák* párbeszédablak „*Szegélyek*” lapján állítható be (10. ábra). Ahogy az ábrán látható, ez a párbeszédablak szinte teljesen megegyezik a Writerben a táblázatok szegélyezésére alkalmas párbeszédablakkal. Az ott tanultak itt is érvényesek. Az „*Előbeállítások*” résznél levő gombok közül az utolsó azonban olyan új lehetőséget tartalmaz, amely a szövegszerkesztőnél nem létezett. Alatta, az „*Egyéni*” nevű mintán ugyanez a többlet lehetőség látható: a kiválasztott cella átlós vonallal áthúzható akár egyik, akár mindkét irányból. Ezzel lehet jelölni például ha valamely cellában nincs vagy nem is lehet adat.

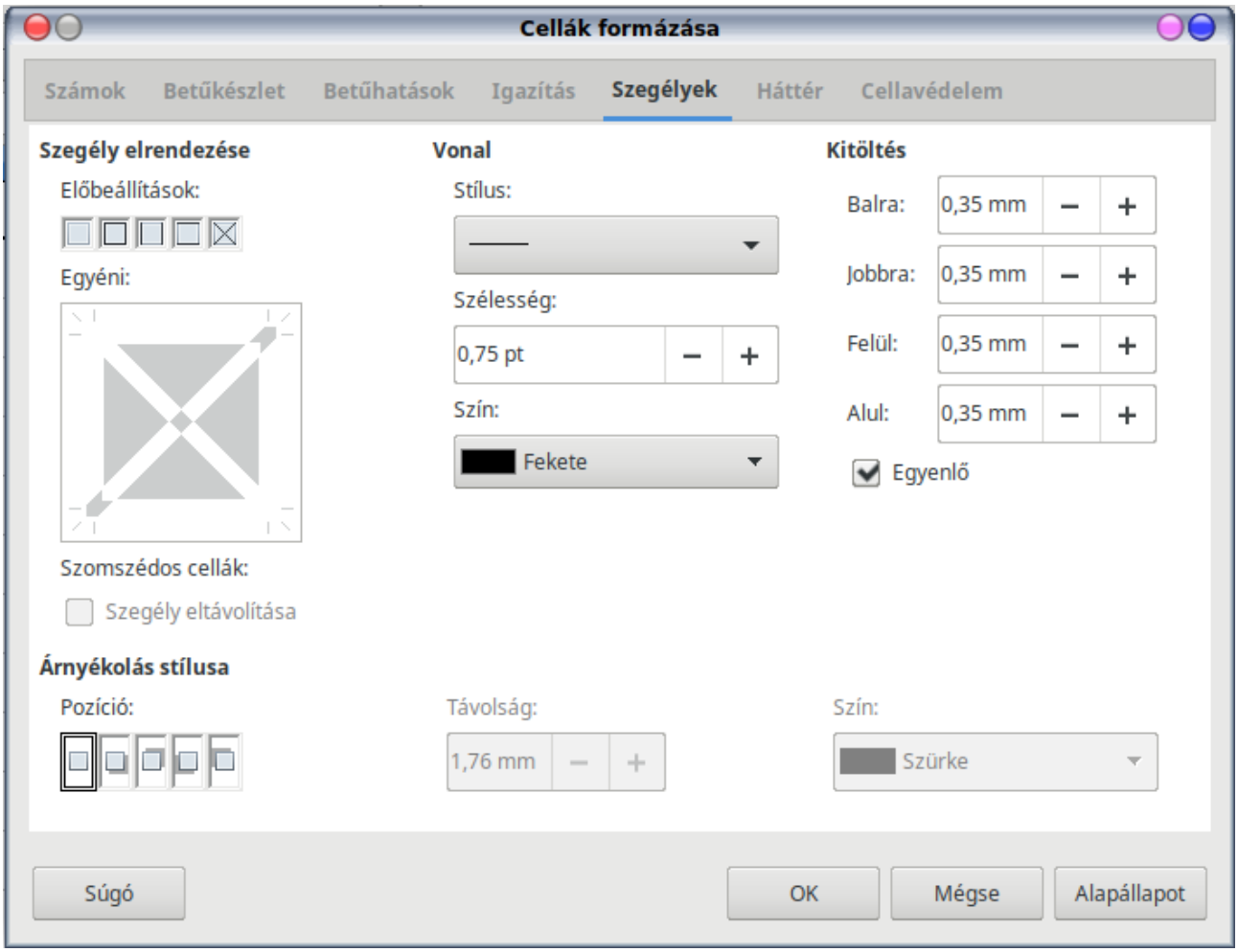
Az ábra azt az esetet mutatja, amikor egyetlen cella szegélyezése állítható be, vagyis amikor nincs kijelölve semmi és ekkor az aktív cellára fog vonatkozi a formázás.

Egy tartomány kijelölése esetén az „*Előbeállítások*” gombjai a már ismert lehetőségeket kínálják, így a jobboldali a belső szegélyek változatlanul hagyása mellett a külső szegélyek kiválasztását eredményezi.

Ilyenkor is lehet azonban átlós szegélyt beállítani az „*Egyéni*” beállításnál.

A szegély vastagságánál ezúttal is figyelni kell arra, hogy ha a „Szélesség” értéke kisebb 1 pt-nél, akkor a dupla szegélyek a „*Stílus*” listában nem jelennek meg. Másrészt a LibreOffice a dupla szegélyeket nem szívesen jeleníti meg, viszont nyomtatásban látszik a dupla szegély.

A 6.0-ás verzióban a belső margó neve „*Kitöltés*” lett a magyar nyelv esetén – rendszeresen változnak a magyarítás során alkalmazott elnevezések, ami nem könnyíti meg a program használatát. A belső margó értéke meghatározza, hogy a cellában levő tartalom és a szegély között mekkora távolság legyen. Arra azonban figyelni kell, hogy akármilyen vastagra van állítva a szegély, a belső



10. ábra: A szegélyek beállítására alkalmas párbeszédablak

margó szempontjából nulla szélességűnek számít. Azaz ha vastagabb szegélyt alkalmazunk, akkor a belső margó értékét nagyobbra kell állítani.

A későbbiekben lesz szó arról, hogyan lehet egy sor magasságát vagy egy oszlop szélességét úgy beállítani, hogy minden tartalom megjelenjen. Ekkor a beállított belső margók is a cella szükséges méreteihez adódnak, így az optimális méretezésnél figyelembe veszi ezeket is a program.

A szegélyezéssel együtt található meg a cellák árnyékolásának beállítása is. Ebben az esetben mindig figyelni kell arra, hogy az árnyékolás összképe megfelelő legyen. Soha ne alkalmazzunk olyan árnyékolást, amely esetben az árnyék rálóg egy értéket tartalmazó cellára!

2.3. Cellák méretezése

A táblázatkezelő celláinak mérete ugyan egyforma, de ez nem jelenti azt, hogy nem lehet más méretű cellát létrehozni. Egész egyszerűen akár a sorok magasságát, akár az oszlopok szélességét módosítani lehet – több módon is.

Az egyik legegyszerűbb megoldás, ha az egérrel módosítjuk a szélességet, illetve a magasságot. Ha az oszlopszélességen akarunk változtatni, akkor a módosítani kívánt oszlop jobb oldalán, az oszlopok nevét tartalmazó gombok közti vonalra kell vinni az egeret, megfogni a vonalat és arrébb húzni. Ugyanígy, a sor magasságát a sor alatti vonalat megfogva lehetséges módosítani. Természetesen csak a sorok számát mutató szürke gombok között lehet a vonalat megfogni.

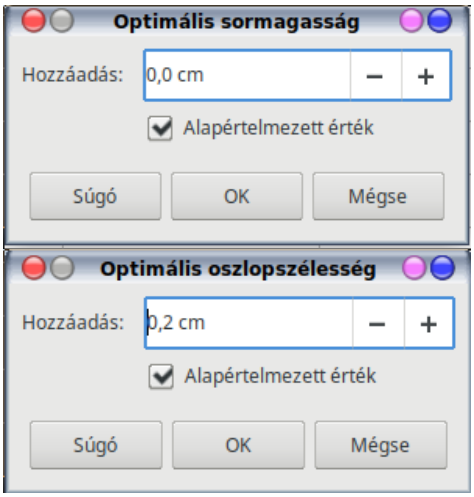
Amennyiben nem tudjuk mekkora magasságot/szélességet akarunk kialakítani, csak az a célunk, hogy minden cella tartalma elférjen, akkor elég a fent említett vonalon duplán kattintani és a program az **optimális oszlopszélesség**et, illetve **optimális sormagasság**ot állítja be. Arra kell csak vigyázni, hogy ez a funkció egy kijelölés esetén a kijelölt cellák tartalmát veszi csak figyelembe.

Másik lehetőség a szélesség, illetve a magasság pontos beállítására az oszlop vagy a sor gombjára az egér jobb gombjával kattintva elérhető menüben található. Az „Oszlopszélesség”, illetve a „Sormagasság” menüpontot választva egy kis párbeszédablakban pontos értéket adhatunk meg centiméterben. Sőt: még a centiméter mértékegység sem kötelező, hanem a LibreOffice-től már megszokott módon bármely mértékegységet – így akár pt-t a nyomdai pont számára – be lehet gépelni, azt a program át fogja számolni.

Itt tudni kell, hogy a MS Excel egészen másképpen számolja az oszlopszélességet és a sormagasságot. Ott csak egy számot lehet megadni, ami valami egészen exotikus mértékegységben értendő: egy adott betűtípus és betűméret esetében mennyi karakter fér el a kívánt távolságon. A szörnyű ebben nem csak az, hogy lehetetlen a hagyományos mértékegységekkel összefüggésbe hozni ezt az értéket, még azt sem engedi az Excel, hogy beírjuk a kívánt mértékegységet.

A helyi menü használatának van egy előnye az egérrel mozgatáshoz vagy dupla kattintáshoz képest: egyszerre több oszlopra vagy sorra is alkalmazható, csak azokat előbb részben vagy egészben ki kell jelölni.

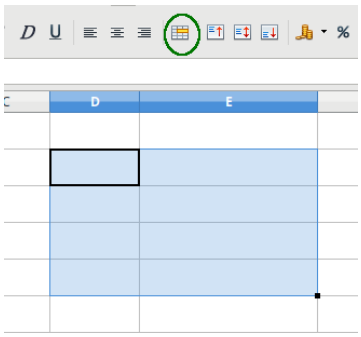
A helyi menüből az optimális oszlopszélesség és az optimális sormagasság is elérhető. Ebben az esetben egy párbeszédabalkot kapunk (11. ábra), amely az oszlopszélesség és a sormagasság beállítására szolgáló ablakkal azonos felépítésű, csak éppen nem a beállítani kívánt hosszértéket kell megadni, hanem azt, hogy mekkora ráhagyással dolgozzon a program. Így a legtöbb esetben elég egyszerűen az OK gombra kattintani és már kész is van.



11. ábra: Az optimális sormagasság és oszlopszélesség beállítása

2.4. Cellák egyesítése

A táblázat elrendezése igényelheti azt is, hogy több cellát egy cellaként kezeljünk. Ezt nevezzük cellaegyesítésnek. A 12. ábra mutatja zölddel bekarikázva az eszköztáron a cellaegyesítrés gombját. Rákattintva a kijelölt cellák helyén egyetlen cella keletkezik, amely a kijelölt tartomány bal felső sarkában levő cellával egyezik meg, míg a többi cella elérhetetlenné válik. Valójában a kijelölés bal felső sarkában levő cella tölti ki a kijelölés helyét. Emiatt erre a cellára történő hivatkozással érhető el az egyesített cella.



12. ábra: A cellaegyesítés gombja bekeretezve

A cellaegyesítéskor előfordulhat, hogy a program megkérdezi, hogy a rejtett cellák tartalmát áthelyezze-e. Ez is amiatt van, hogy a kijelölés bal felső sarkában levő cella fogja a kijelölés területét kitölteni és a többi cella rejtetté válik. Az áthelyezés azt jelenti, hogy minden, az egyesítésben részt vevő – és emiatt rejtetté váló cella tartalma átmásolódik a megmaradó cellába.

A cellaegyesítést ugyanazzal a gombbal lehet megszüntetni, amellyel létrehoztuk. A rejtett cellákba nem másolódik vissza automatikusan azok tartalma.

A *Formátum* menü is használható a cellaegyesítéshez. A „*Cellák egyesítése*” menüpontnak három alpontja is van:

1. *Cellák egyesítése és középre igazítása*: Valójában ez a funkció található meg az eszköztáron. Nem csak egyesíti a cellákat, de a tartalmat függőlegesen és vízszintesen is középre igazítja.
2. *Cellák egyesítése*: A cellaegyesítés úgy történik, hogy a bal felső cellára aktuálisan érvényes igazítás érvényben marad.
3. *Cellák felosztása*: Ha egyesített cella (vagy több egyesített cella) van kijelölve, akkor ez a funkció érhető el az eszköztáron is. Egyszerűen megszüntet minden egyesítést, vagyis több egyesített cella kijelölése esetén egyszerre megszüntethető azok egyesítése.

2.5. Feltételes formázás

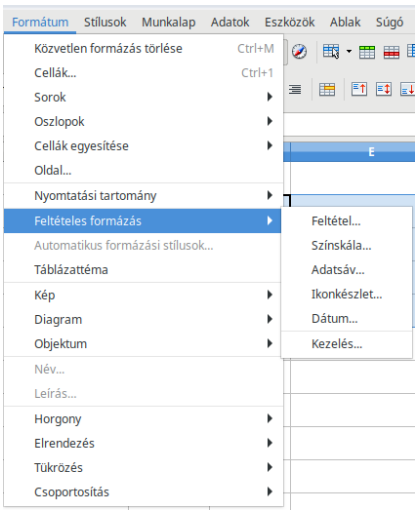
Sok egyéb formázási lehetőség létezik még, amelyeket kísérletezéssel is fel lehet deríteni. Van azonban egy olyan funkció, amelyet kísérletezéssel nem valószínű, hogy fel lehet fedezni. Ez a **feltételes formázás** lehetősége.

Ennek használatához ismerni kell a stílusokat is, amelyek a szövegszerkesztőben megismert stílusok rokonai. Egy cellastílusra minden olyan tulajdonság beállítható, amelyet egy cellára be lehet állítani. Nagyon jól használhatók ezek a *cellastílusok* a táblázat egységes formázásának kialakítására. Emellett a feltételes formázás használatához is szükségesek.

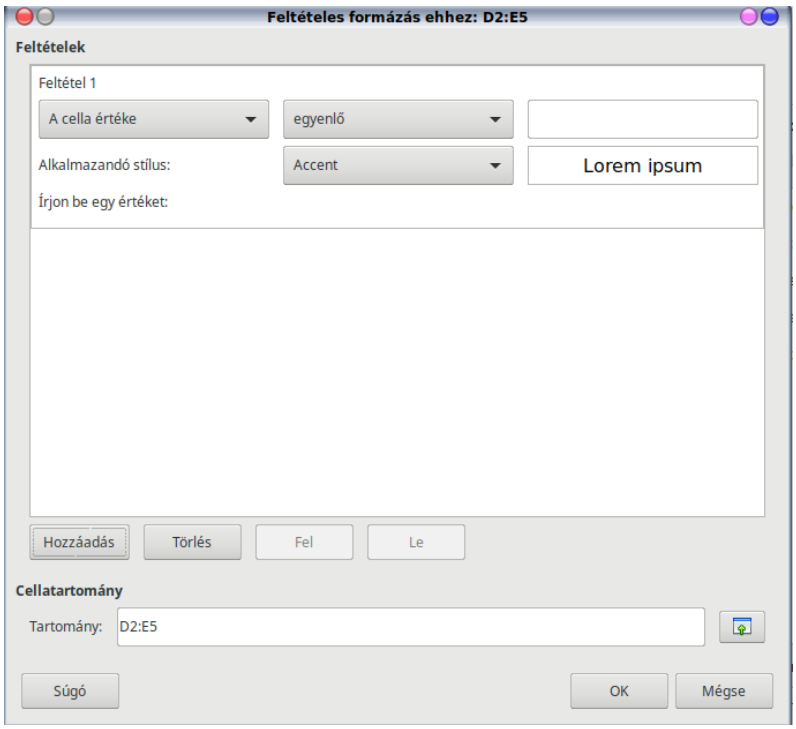
A feltételes formázással egy cellának a megjelenését annak értékétől függően határozhatjuk meg.

A *Formátum* menüben található a „*Feltételes formázás*” almenü és annak menüpontjaival lehet a feltételes formázást definiálni, módosítani vagy törölni (13. ábra).

A Feltétel menüpontnál lehet definiálni a feltételt. Ám mielőtt definiálnánk a feltételt, már léteznie kell a felhasználni kívánt cellaformátumnak!



13. ábra: A feltételes formázás menüje



14. ábra: Feltételes formázás feltételének megadása

A párbeszédablak a 14. ábrán látható. A feltétel a cella értéke és egy megadott érték közötti relációra állítható be elsősorban, de a reláció helyett további lehetőségek is rendelkezésre állnak a legördülő listában. Ráadásul több feltétel is megad-

ható (az alul található „*Hozzáadás*” gomb segítségével), amelyek révén egész feltétel-láncot is ki lehet alakítani.

A feltétel második sorában kell megadni a listában a használni kívánt stílust, mellette pedig egy minta mutatja, hogyan fog kinézni a cella tartalma az adott feltétel teljesülése esetén.

Amennyiben több feltételt is megadunk, akkor azokat a táblázatkezelő felülről lefelé fogja vizsgálni és az első teljesülő feltételnél megadott stílust fogja alkalmazni a cellára. A feltételek sorrendjét az alul levő „*Fel*” és „*Le*” gombokkal lehet módosítani. Egy feltétel törölhető is a „*Törlés*” gombbal.

További lehetőségeket kínálnak a további menüpontok.

A *Színskála* menüpont segítségével például egy értékskálát lehet színkóddal jelezni. A két szélső színt kell kiválasztani és megadni a számítási módot (Minimum, Maximum, Százalékosztály, Érték, Százalék, Képlet).

Az *Ikonkészlet* hasonló lehetőséget kínál, de nem a cellák színezésével, hanem ikonok megjelenítésével. Mindkét lehetőség jól használható például értéksávok megadására (például osztályzatok szemléltetésére).

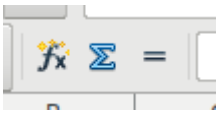
A *Kezelés* menüpont a már beállított feltételek áttekintését és kezelését teszi lehetővé.

A feltételes formázás elsősorban kimutatási táblázatok számára lehet hasznos, de jól használható olyan esetben is, ha bizonyos celláknál figyelmeztetést szeretnénk megjeleníteni problémás értékek megjelenése során, vagy ha szeretnénk egy tartomány celláiból automatikusan kiemeltetni egy-egy cellát.

3. Képletek készítése

A táblázatkezelő program fő funkciója a számítások végzése. Emiatt egy cellába nem csak egy adott értéket lehet elhelyezni, hanem olyan képleteket is, amelyek más cellák értékének felhasználásával valamilyen új értéket számolnak ki. (Itt most az érték természetesen nem csak számérték lehet.)

A táblázatkezelő celláiba írt képletek mindig = karakterrel kezdődnek és a legtöbb esetben más cellákra is hivatkoznak. Képletek bevitelekor a *Képlet eszköztár*on a *Szerkesztőléc* és a tőle balra levő három gomb (15. ábra) fontos szerepet játszik.



15. ábra

Már volt róla szó, hogy a *Szerkesztőléc* a cella tartalmát mutatja, de nem mindig úgy, ahogy a cellában megjelenik. Aki a különböző cellaformázásokat kipróbálta, az már tapasztalhatta, hogy a *Szerkesztőléc* a cella pontos értékét mutatja még akkor is, ha a cella formázása rövidebb formát írt elő. Még nagyobb különbség van akkor, ha a cella egy képletet tartalmaz.

Ha egy cella képletet tartalmaz, akkor – a program alapbeállítása esetén – a cellában a képlet eredményeként keletkező értéket mutatja, míg a *Szerkesztőléc* magát a képletet fogja mutatni. Hasonlóan, ha a cellára duplán kattintunk, akkor a cellában található képletet lehet szerkeszteni. Ennek egyszerűen az a magyarázata, hogy a cella valódi tartalma a képlet, míg a cellában a képlet értéke jelenik meg.

3.1. A képletekben használható operátorok és operandusok

*Operátor*nak nevezzük az egyes képletekben megjelenő műveleti jeleket, míg azokat az értékeket, változókat és egyéb objektumokat, amelyekkel a műveletek dolgoznak, *operandus*nak.

A táblázatkezelőben operandus lehet egy konstans, egy cellahivatkozás, bizonyos esetekben egy tartományra történő hivatkozás. Konstans lehet:

- Számérték, amelyet számtípusként kezel a program.
- Macskakörmök közé ért szöveg, amelyet szövegként fog kezelni a program.
- Logikai értékként: `Hamis` vagy `Igaz` szöveg.

Nem lehet operandus egy dátum vagy időérték! Ezeket önálló cellába be lehet írni, de képleten belül nem szerepelhet, helyette a megfelelő értéket előállító függvényt kell használni.

Érdemes még tudni, hogy a képleteken belül a kis- és nagybetűk azonosnak számítanak, a szövegkonstansokat kivéve.

3.1.1. Cellahivatkozások

Egy képleten belül egy másik cellára lehet hivatkozni annak nevének megadásával. Ennek módja többféle lehet:

- *Munkalapon belüli hivatkozás* esetén előbb az oszlop nevét, utána a sor számát kell megadni, amelyek együtt azonosítják a cellát. Például az **E** oszlop **18**. sorában levő cellára történő hivatkozás így adandó meg: `E18`.
- A munkalap nevét is megadhatjuk a cellahivatkozásban a fenti megoldás elé írva a munkalap nevét, majd egy pontot (`.`) írva. Például a „*Bevételek*” nevű munkalap **G** oszlopának **12**. sorát a következő cellahivatkozás adja meg: `Bevételek.G12`.

Ebben a dokumentumban a továbbiakban az oszlopok nevének, a sorok számának, valamint egy adott cellának a nevének a megadása mindig **erős hangsúlyozás**sal történik.

A fentiek azonban még nem minden lehetőséget írnak le. Próbáld ki, mi történik, ha például a **B1** cellába beírt `=A1` képletet lemásolod az automatikus kitöltést vagy a vágólapot használva a **B2** cellába! Mi lesz a **B2** cellában levő képlet?

A fenti alakú cellahivatkozásokat **relatív cellahivatkozás**oknak nevezzük. Ezek a cellahivatkozások valójában nem a megadott cellát azonosítják, hanem azt adják meg, hogy a képletet tartalmazó cellától vízszintesen és függőlegesen milyen távolságra levő cella tartalmát kell a képletben figyelembe venni. A képlet átmásolása esetén ez a távolság nem változik, így az új képlet a tőle ugyanabban a távolságban levő cellát fogja használni.

Előfordul azonban nagyon sokszor, hogy olyan cellahivatkozást tartalmazó képletet akarunk másolni, amelynek mindig pontosan ugyanarra a cellára kell mutatnia. Az ilyen hivatkozásokat hívjuk **abszolút cellahivatkozás**oknak.[4] Az abszolút hivatkozás használata esetén megmondhatjuk a táblázatkezelőnek, hogy egy adott eleme a cellahivatkozásnak akkor sem változhat, ha a képletet másik cellába másoljuk. Ebben az esetben a cellahivatkozás már nem a képletet tartalmazó cellától való távolságot, hanem a konkrét oszlopot vagy sort adja meg, így a képlet másolásakor sem fog változni.

Az abszolút hivatkozás megadásához a rögzítendő komponens elé egy `$` jelet kell tenni. A cellahivatkozás komponensei egymástól függetlenül tehetőek abszolút hivatkozássá – erre utal a máshol használt „*vegyes*” hivatkozás elnevezés is.

Például ha a képletben szereplő, az **A5** cellára történő hivatkozást szeretnénk védeni a jobbra másolással szemben, akkor az oszlop nevét abszolút hivatkozássá kell tenni: `$A5`. Ellenben ha ugyanezt a képlet függőleges másolása esetén szeretnénk rögzíteni, akkor a sorszámot kell abszolút hivatkozássá tenni: `A$5`. Amennyiben mind a vízszintes, mind a függőleges másolással szemben védeni akarjuk a cellahivatkozást, akkor mind az oszlop nevét, mind a sor számát abszolút hivatkozássá kell tenni: `$A$5`.

Amennyiben a cellahivatkozás egy munkalap nevét is tartalmazza, akkor a képletet átmásolva egy másik munkalapra, alaphelyzetben szintén relatív lesz a hivatkozás. Ha mindig ugyanarra a munkalapra akarunk hivatkozni, akkor a munkalap neve elé is írni kell egy `$` jelet: `$Munkalap1.A2` minden munkalapra másolva a „*Munkalap1*” nevű munkalapra fog hivatkozni, ellenben a `Munkalap1.A2` egy munkalappal jobbra másolva már arra a munkalapra fog hivatkozni, amely a „*Munkalap1*” nevű jobboldalán található. Ez a viselkedés eltér az Excel dokumentációja szerinti viselkedéstől, ahol a munkalap mindig abszolút hivatkozásként viselkedik.[5]

3.1.2. Tartomány-hivatkozás

Tartományra kétféleképpen lehet egy képletben hivatkozni.

Az egyszerűbb megoldás, ha a tartomány bal felső sarkát és a jobb alsó sarkát megadó cella nevét egy kettőspont (:) karakterrel elválasztva megadjuk. Például a `B3:F14` a **B3** és az **F14** cellák által határolt téglalap alakú tartományt jelenti. Természetesen a $ jel ilyenkor is használható az egyes komponensek abszolút hivatkozássá alakítására. Akár olyan megoldás is lehetséges, hogy a bal felső sarok rögzítve van, míg a jobb alsó nem, aminek hatására a képlet másolására egy folyamatosan bővülő tartomány jön létre: `$A$1:A1` ugyan egy cellát ad meg, de a képletet egy sorral lejjebb másolva már az **A1:A2** tartományt fogja megadni.

4 *A legtöbb táblázatkezeléssel foglalkozó könyv megkülönböztet relatív, abszolút és vegyes hivatkozást. Az utóbbi azokra a hivatkozásokra vonatkozik, amelyek egyik irányban relatív, a másikban abszolút hivatkozásnak számítanak.*

5 *De a dokumentációnak nem feltétlenül kell hinni, így ha valaki Excelben akar munkalap-hivatkozásokat használni, akkor próbálja ki, hogy viselkedik a valóságban.*

A tartományra hivatkozás másik módja, ha a tartományt előzőleg elneveztük a *névdoboz* használatával. Ebben az esetben a nevet is beírhatjuk közvetlenül a képletbe. Az ilyen tartománymegadás mindig abszolút hivatkozásnak számít.

Ritka az az eset, amikor a képletben operandusként tartomány szerepel, inkább függvények paramétere szokott lenni.

3.1.3. Operátorok

A képletben a következő operátorok használhatóak:

- Összeadás: `+`
- Kivonás: `-`
- Szorzás: `*`
- Osztás: `/` (Mivel a : a tartomány megadására már el lett használva, így más jelet kellett az osztásra találni.
- Századrész (százalékérték): `%`
- Hatványozás: ^ (magyar billentyűzeten az ALT-Gr+3 billentyűkombinációval érhető el)
- Szövegrészek összefűzése: `&`

3.2. Függvények

A táblázatkezelő program rengeteg függvénnyel rendelkezik, amelyek mindegyike valamilyen számítási műveletet[6] elvégzésére alkalmas. Ezek egy részével a továbbiakban majd részletesebben is meg fogunk ismerkedni és az utolsó fejezet az összes tanult függvényt összefoglalja kategóriánként ábécé sorrendben. Most egyelőre elég annyit tudni, hogyan kell a képleten belül a függvényeket megadni.

A függvénynek mindig van egy neve és egy paraméterlistája. Utóbbi megadása akkor is kötelező, ha üres, hiszen egy paraméterlista nélküli nevet a program nevesített tartománynak fog tekinteni még akkor is, ha az adott név nincs definiálva (ilyenkor hibaüzenet keletkezik).

A paraméterlistát mindig közvetlenül a függvény neve után kell írni, nem előzheti meg szóköz. A paraméterlista mindig egy kezdő zárójellel „`(`„ kezdődik és egy záró zárójellel „`)`” fejeződik be. A LibreOffice Calc magyar változatában a függvények nevének nyelve átállítható az eredeti angol és annak Excelbeli magyarított neve között. Ez az *Eszközök → Beállítások* menüvel elérhető párbeszédablak *LibreOffice Calc/Képlet* részénél tehető meg (16. ábra). Az „*Angol függvénynevek használata*” kapcsolóval tehető ez meg. Mivel a függvények nevét több esetben hibásan magyarította, aki annak idején ezt megtette, így hasznosabb lehet a függvények angol nevét is megtanulni, ezért ebben a dokumentumban ezek is szerepelni fognak.

Az alábbiakban néhány példa látható olyan képletekre, amelyekben függvények is szerepelnek:

- `=ma()` vagy angol változatban `=today()` az éppen aktuális dátumot fogja a cella megjeleníteni. A függvénynek nincs paramétere ezért a paraméterlista üres.
- `=szum(A1:C5)` vagy angol változatban `=sum(A1:C5)` az **A1:C5** tartomány számot tartalmazó celláinak összegét fogja a cellában megjeleníteni.
- `=fkeres(a5;c8:f12;2;0)` vagy angolul `=vlookup(a5;c8:f12;2;0)` képletek az **A5** cellában szereplő értéket keresik a **C8:F12** tartomány első oszlopában és a második oszlopában levő cella értékét adják eredményként. Ebben a függvényben az első paraméter egy cella, a második egy tartomány, a harmadik egy egész szám, a negyedik pedig egy logi-

6 *Most számítási műveletnek tekintjük a szövegen történő manipulációt is.*

kai érték, amely ebben az esetben a `Hamis` konstans helyett a vele megegyező értéket eredményező `0` számmal van megadva.

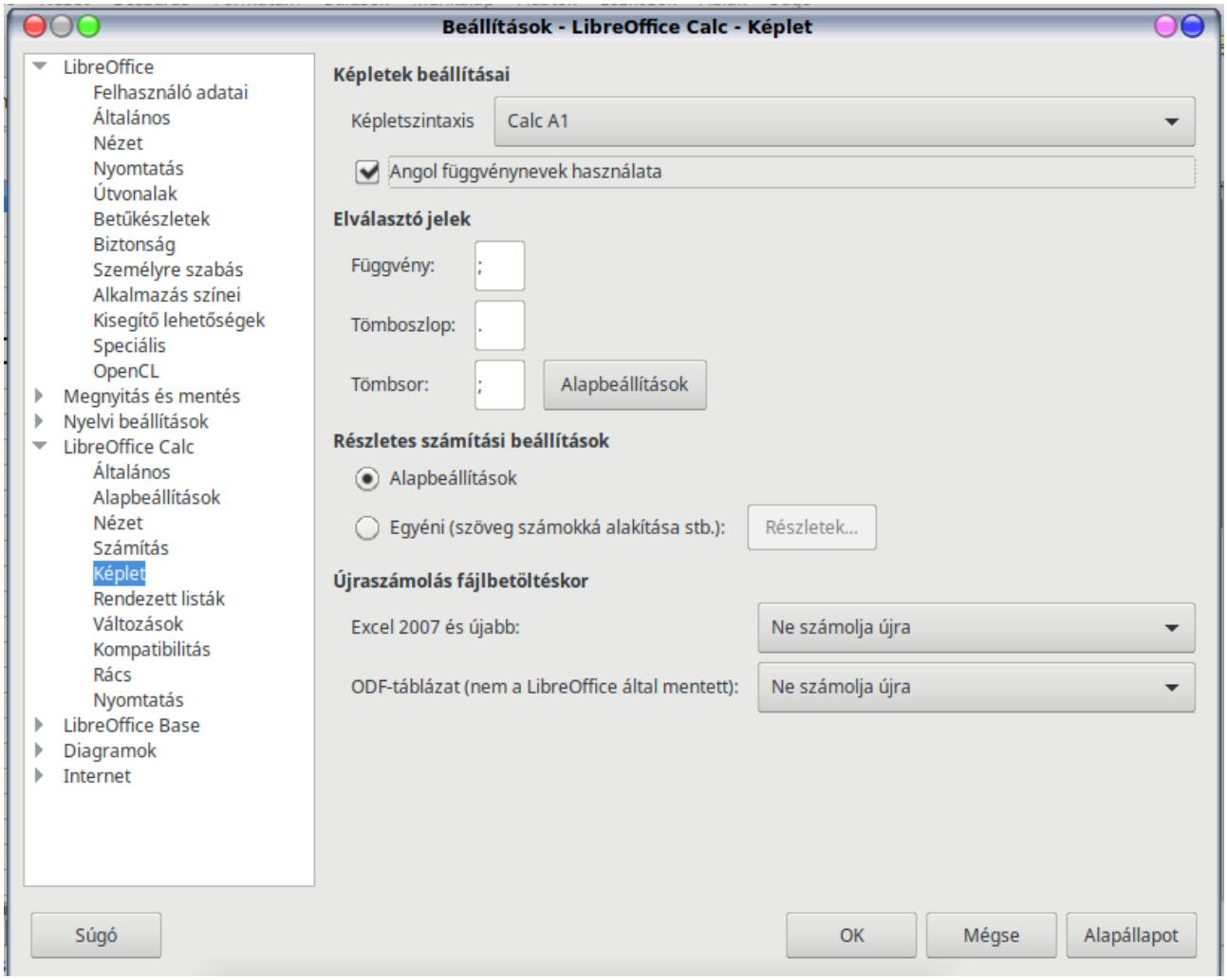
3.3. Képletek bevitelét segítő eszközök

A Képlet eszköztáron a Szerkesztőléc baloldalán levő három gomb (15. ábra) a képletek bevitelének segítésére szolgál. Mint korábban már volt szó róla, ha a cellába valamilyen tartalmat gépelünk, akkor ezen a helyen egy zöld pipa és egy piros x jelenik meg, amelyekkel a bevitelt lehet jóváhagyni vagy visszavonni. Ha viszont nem cellabevitel közben vagyunk, akkor a jobboldalon levő egyenlőségjelet mutató gombbal egy képlet bevitelét lehet elkezdeni.

Ennek a lehetőségnek a használata nem tűnik túl sok segítségnek, mivel mindössze annyit tesz, hogy az aktuális cellában megjelenik a képletek elején kötelező egyenlőségjel és folytatható a bevitel.

Ám a baloldalán levő *szumma* jel már többet tud segíteni, az „f_x” feliratú gomb pedig még többet.

A *szumma* gombra kattintva az egyik leggyakoribb műveletet lehet kezdeményezni, amely nem más, mint az összegzés. Ez azt jelenti, hogy az aktuális cellában az összegzést végző függvény jelenik meg, amely angolul Sum, magyarul Szum. Ez a gomb nagyon jól használható arra is, hogy gyorsan kideritsük, angol vagy magyar függvénynevekre van-e éppen állítva a program (lásd

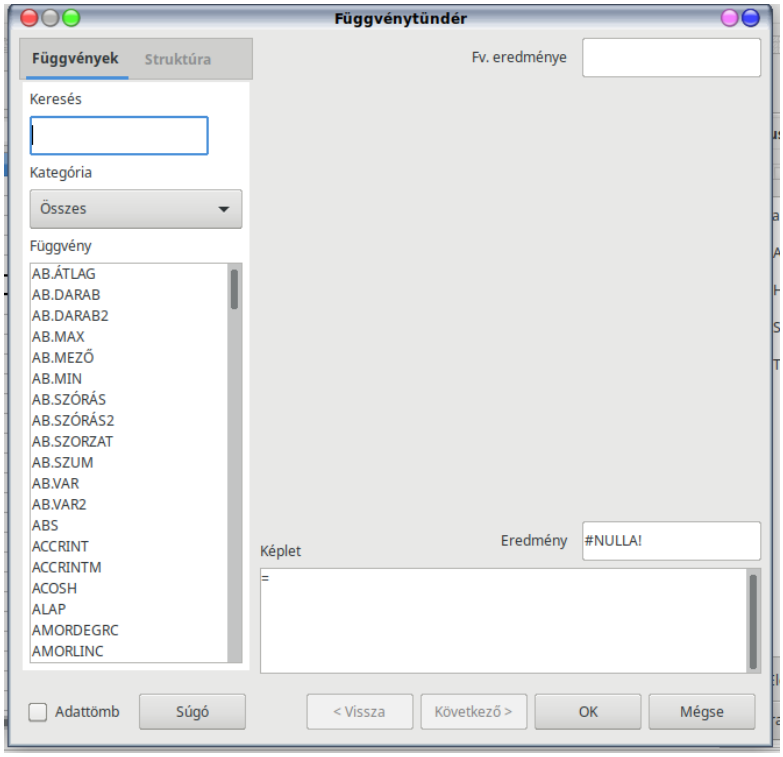


16. ábra: A képletre vonatkozó beállítások. A függvénynevek nyelvének beállítása felül, szegéllyel ellátva látható.

16. ábra). Amennyiben az aktív cella felett vagy a bal oldalán olyan értékeket tartalmazó cellák találhatóak, amelyeket van értelme összeadni, akkor a függvény paraméterének a program automatikusan felkínálja ezt a tartományt. Ha nem megfelelő a felkínált tartomány, akkor az egérrel nyugodtan ki lehet helyette jelölni a megfelelőt.

3.3.1. A függvénytündér

A többi függvényt is könnyen el lehet érni egy képlet beírásához – akár a képlet szerkesztése közben, akár a képlet írásának elkezdése előtt – az „f_x” feliratot mutató gombra kattintva. Ezzel a gombbal a *függvénytündért* lehet előhívni (17. ábra).[7]



17. ábra: A függvénytündér ablaka

A függvénytündér a függvényeket kategóriákba sorolja, és azon belül az éppen beállított nyelv szerint betűrendben jeleníti meg. A kategóriák segítik a függvények rendszerezését, de egyben azt is segítik, hogy a felhasználó, ha nem tudja a használni kívánt függvény pontos nevét, akkor is az elvégzendő művelet jellege alapján könnyebben megtalálja a megfelelő függvényt.

A függvénytündér két esetben nagyon hasznos: ha egy adott függvény pontos használati módját szeretnénk megismerni és ha tudjuk, mire van szükségünk, de a függvény neve nem jut eszünkbe.

Bármelyik függvényt kiválasztva a listából, egy rövid leírást kapunk róla a jobboldali területen. Ez valóban csak rövid leírás, de szerencsés esetben ez is elég lehet arra, hogy megtaláljuk a keresett függvényt – normális ember nem magolja be az összes létező függvényt, csak amíg például az érettségi vizsgán erre kényszerül.

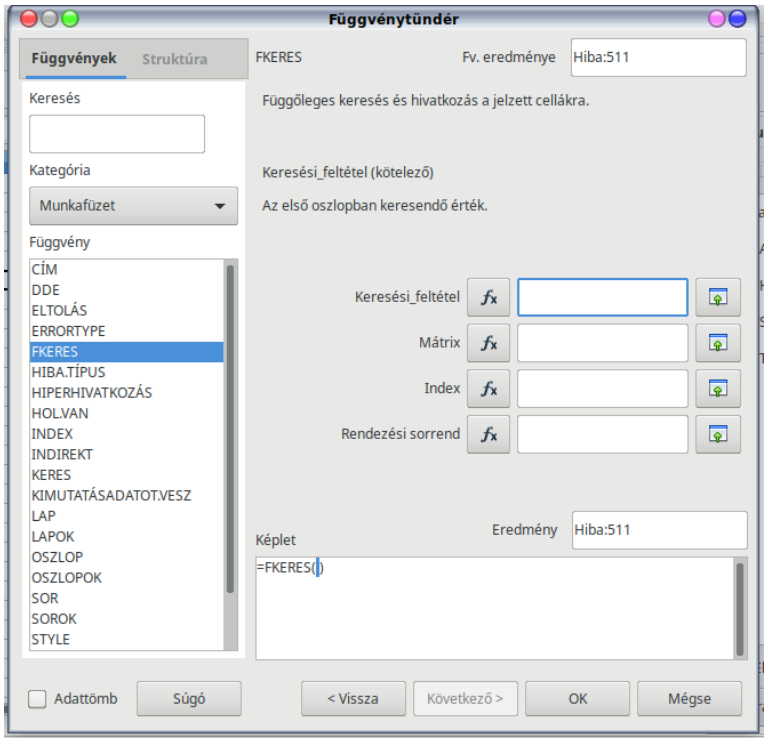
Ha megvan a használni kívánt függvény, akkor az alsó gombok közül a „*Következő*” feliratúra kell kattintani. Ekkor a 18. ábrán láthatóan megjelennek a függvény lehetséges paraméterei, valamint alul a szerkesztés alatt álló képlet jelenlegi szövege. Amennyiben valamelyik paraméter mezőjébe kattintunk az egérrel, vagy a **Tabulátor** billentyűvel átlépünk a következő mezőre, akkor a mezők fölött az adott mező funkciójáról is kapunk egy rövid ismertetőt. Alul a „*Súgó*” gombra kattintva további tájékoztatást kaphatunk az adott függvényről a program súgóját felhasználva.

A mező kitöltésére ugyanazok a lehetőségek állnak rendelkezésre, mint ha csak simán elkezdtük volna begépelni a mezőbe a képletet (lásd lentebb). Ha a függvénytündérrel akarunk újabb függvényt keresni az adott mezőbe, akkor az „f_x” gombra kell a mező mellett kattintani.

7 *Míg a MS Office varázslókat használ az összetettebb műveletek elvégzésének segítésére, a LibreOffice az ilyen segédeszközöket „tündér” névvel illeti. Ennek nyilván a fejlesztők azon szándéka oka, hogy minél egyértelműbbé tegyék a különbséget a két programcsomag között. Másrészt azt sem szabad elfelejteni, hogy a mesékben sokkal gyakrabban fordul elő gonosz varázsló, mint gonosz tündér...*

Az alsó mezőben mindig a képlet aktuális alakja látszik. Ezt a mezőt arra is lehet használni, hogy a megfelelő részére kattintva egy több függvényt is tartalmazó képletnek a szerkeszteni kívánt függvényt hívjuk elő az ablakban. Már korábban létrehozott képleteket is lehet a függvénytündér segítségével formázni.

A függvénytündér az olyan függvényeket is jól kezeli, amelyeknek a paraméterlistája változó számú paramétert tartalmaz, mint az ábrán is látható Fkeres(), de akár a Szum() függvény is. Az egyes mezőkről megmondja, hogy kötelező vagy opcionális.



18. ábra: A függvénytündér az Fkeres() függvény kiválasztása után

Amennyiben tudjuk a függvény nevét, de nem tudjuk, hogyan kell használni – vagy tudjuk hogyan kell használni, de kényelmesebbnek találjuk a függvénytündér használatát –, még a kategóriáját sem kell a 6.x LibreOffice verziónál kiválasztani, mert a kategória kiválasztására szolgáló lista feletti mezőbe beírva a függvény nevét, azt gyorsan ki lehet választani.

A megszerkesztett képlet akkor kerül a cellába, ha az „*OK*” gombra kattintunk. Viszont az „*Eredmény*” mezőben már ezt megelőzően látható a képlet (vagy az éppen szerkesztett függvény) várható eredménye.

3.3.2. Cella vagy tartomány kiválasztása egérrel

Nem kell a képletekbe a cellákra vagy tartományokra hivatkozást begépelni. Helyette használható az egér is: képleten belül egy cellára kattintva – vagy a kurzor gombokkal arra a cellára állva – elérhető a beviteli kurzor helyén a cellahivatkozás megjelenítése. Éppen ez az, amiért nem szabad a cellába történő adatbevitelt más cellára kattintással vagy a kurzorbillentyűkkel befejezni.

Hasonló módon, ha képlet bevitele közben az egérrel kijelölünk egy tartományt, akkor annak a tartománynak a neve kerül bele a képletbe. Ez általában egy függvény paraméterében szokott elhelyezkedni.

Mind az egyetlen cella, mind a tartomány kijelöléssel történő bevitele működik közvetlen bevitel és a függvénytündér használata esetén is.

3.3.3. Váltás relatív és abszolút cellahivatkozások között.

Közvetlenül a cella- vagy tartomány hivatkozásának bevitele után vagy később a megfelelő cellahivatkozásba kattintva, illetve a tartományhivatkozás kijelölése után lehetséges az **F4** billentyűt használva váltani a hivatkozás fajtáját. Régebbi LibreOffice verziókban a **Shift+F4** billentyűkombinációval volt ez lehetséges, de az Excel-lel való kompatibilitás érdekében ez megváltozott.

A billentyű a négy lehetőség között körkörösen vált, azaz a negyedik lehetőség után ismét az első lehetőség következik. A sorrend: teljesen relatív → teljesen abszolút → relatív oszlop és abszo-

lút sor → abszolút oszlop és relatív sor → ismét teljesen relatív. A munkalap relatív/abszolút állapotát is lehet így változtatni. Ekkor a ciklus nyolc eleméből az első négy a fenti lesz, majd ezután következik a váltás a munkalapra relatívról abszolútra (vagy vissza) ugyanezzel a négy lehetőséggel.

Ez a lehetőség nem működik a függvénytündéren belül, így ott a szükséges helyekre a $ jeleket kézzel kell begépelni.

4. Diagramok

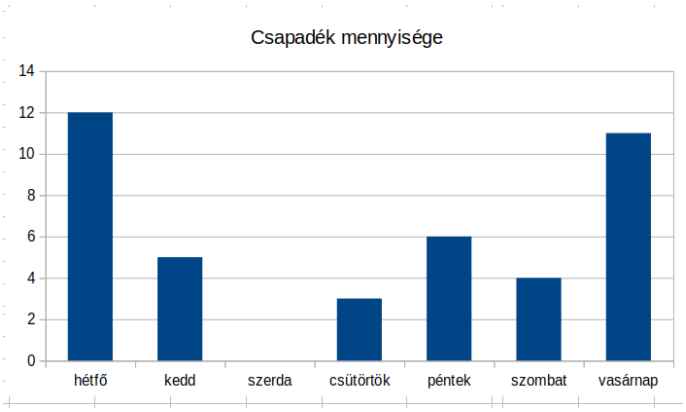
A táblázatkezelő programmal létrehozott táblázatok adatait grafikus formában szemléletessé lehet tenni diagramok segítségével. Attól függően, hogy milyen jellegű adatokból mit szeretnénk megmutatni, különböző típusú diagramokat használhatunk.

4.1. Diagramok fajtái

Rengeteg formában lehet egy diagramon ugyanazokat az adatokat megjeleníteni. Azonban ezek közül nem mindegyik alkalmas minden adatfajta ábrázolására. Éppen ezért ismerni kell, hogy melyik diagramfajta milyen célt szolgál. Az alábbiakban először a leggyakrabban használt három diagram típus bővebb ismertetése következik, majd a ritkábban használt további típusok bemutatására is sor kerül.

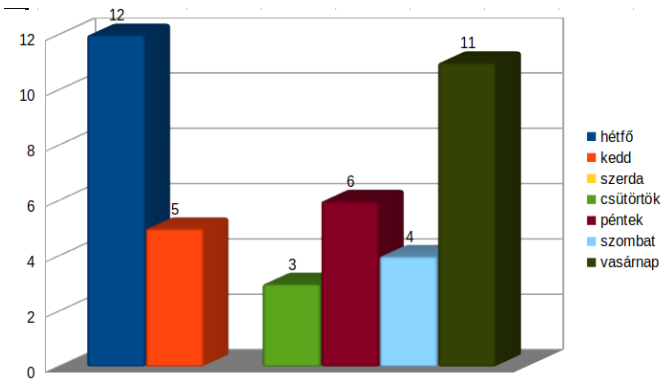
4.1.1. Oszlopdiagram

A leggyakrabban mennyiségek egymáshoz való viszonyát akarjuk bemutatni, hogy azokat össze lehessen hasonlítani. Erre alkalmas az **oszlopdiagram** (19. ábra). Az egymás mellett elhelyezkedő oszlopok azonnal mutatják az ábrázolt mennyiségek nagyságát.



19. ábra: Oszlopdiagram

Az oszlopok alatt szokás megjeleníteni az egyes mennyiségek megnevezését is, mint az ábrán az adott nap nevét. Ha túl sok oszlop van, akkor az *y-tengely*ről már esetleg nehéz leolvasni az értékeket, amit meg lehet oldani azzal, ha az oszlop tetején is megjelenítjük az adott oszlop magasságát, azaz az általa reprezentált mennyiség értékét.



20. Ábra: Térbeli oszlopdiagram, más néven hasábdiagram

Természetesen lehetséges térbeli oszlopokat is használni – ilyenkor szokás hasábdiagramról beszélni. A 20. ábra egy ilyen *térbeli oszlopdiagram*ot mutat, ahol az oszlopok hasáb alakúak és az oszlopokhoz tartozó értékek is megjelennek az oszlop tetején.

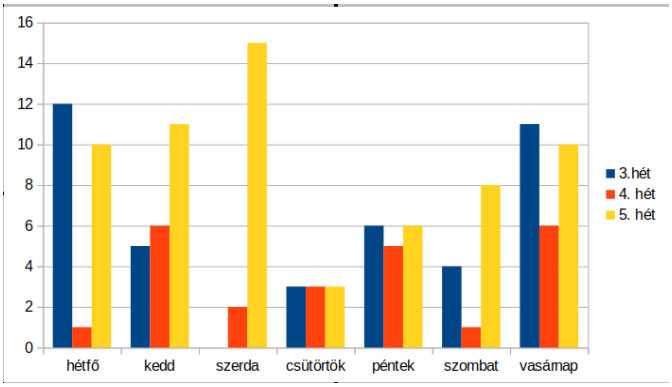
Ebben az esetben az ábrán[8] ráadásul az egyes oszlopok színe eltérő és nem alul látható az oszlopok jelentése, hanem a jobboldalt található jelmagyarázat adja meg, hogy melyik szín mit jelent. Ezen és az előző ábrán a szerda oszlopa azért nem jelenik meg, mert a hozzá tartozó érték nulla, így az oszlop magassága is nulla.

8 *Azt, hogy az oszlopok egy vagy több színűek, nem az dönti el, hogy két vagy háromdimenziós a diagram, hanem hogy egy vagy több adatsort ábrázol-e!*

Térbeli oszlopok esetén az oszlopok alakja is megváltoztatható, így nem csak *hasábdiagram* lehet a térbeli oszlopdiagram: a LibreOffice Calc esetén beállítható még *henger*, *kúp* vagy *gúla* alakú oszlopokat tartalmazó diagram is. Csapadékmennyiség megadására például szemléletesebb lehet henger alakú oszlopokat használni, amelyek – főleg kék színű oszlopok használata esetén – a vízoszlopot is szemléltethetik.

Az eddigi két ábra alapján úgy tűnik, hogy az oszlopdiagram csak egy adatsort képes ábrázolni, de ez nem igaz. Ha egy oszlopdiagramon több adatsort ábrázolunk, akkor több lehetőség is rendelkezésre áll.

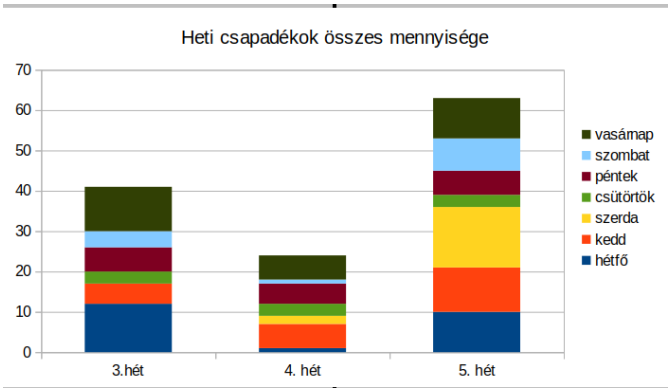
Alapesetben az egyes adatsorok eltérő színű oszlopokkal jelennek meg egymás mellett, az azonos kategóriába tartozó oszlopokat egymás mellés sorolva. A kategória neve alul jelenik meg, míg az egyes színek kódolását a *jelmagyarázat*ból lehet leolvasni. Valójában a térbeli oszlopdiagramra példát mutató 20. ábra egy több adatsoros diagram, ezért színesek az oszlopok. Ezen a diagramon egyetlen *kategória* – a csapadék mennyisége – látható, ezért nincs alul semmilyen szöveg.



21. ábra: Több adatsoros oszlopdiagram

Ilyen több adatsoros diagramra látható példa a 21. ábrán, ahol három héten a napi csapadékmennyiségek láthatóak úgy, hogy az egyes napok csapadékmennyiség lehet egymással összehasonlítani a hét minden napjára. Egyben az egyes hetek csapadékmennyiségének alakulása is látható, ha egy adott színű oszlopokat nézzük.

Arra is van lehetőség, ha úgy akarunk mennyiségeket szemléltetni, hogy azok összege is könnyen leolvasható legyen. Az ilyen diagramokat *halmozott oszlopdiagram*nak hívjuk (22. ábra), mert az egyes kategóriákban – amelyek neve továbbra is alul jelenik meg – az egyes adatsorokból származó oszlopok egymára rakva jelennek meg. Az egyes színek az adott adatsort jelentik ekkor is, az oszlop teljes magassága pedig ezek összegét mutatja.



22. ábra: Halmozott oszlopdiagram

További lehetőség a *százalékos halmozott oszlopdiagram*, amely esetben az összeg oszlopok magassága megegyezik – így a mennyiségek összehasonlítása itt már nem lehetséges –, az egyes színek ezen belül az adott adatsor értékének a kategória összértékéből kivett arányát mutatja. Ez a diagramfajta valójában nem arra alkalmas, amire általában az oszlopdiagramok, hanem *százalékos eloszlás* ábrázolására, ahogyan a kördiagram is.

Mint már fentebb volt róla szó, az oszlopdiagram a leggyakrabban használt diagramtípus, hiszen a legtöbbször értékek összehasonlítására alkalmazunk diagramot.

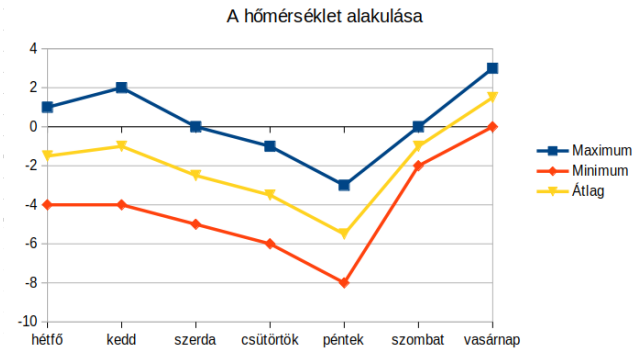
Meg kell még említeni egy lehetőséget arra az esetre, ha az összehasonlítani kívánt értékek között nagyságrendbeli eltérések vannak. Ebben az esetben a hagyományos skálájú diagramon a legnagyobb érték a többit teljesen elnyomja, így lehetetlenné válik az adatok összehasonlítása.

Ilyenkor jön jól az a lehetőség, hogy az *y-tengely* skáláját l*ogaritmikus skálá*ra lehet átállítani, ami azt jelenti, hogy a 10 hatványai lesznek azonos távolságra elhelyezve egymástól. Természetesen ebben az esetben nem a tényleges értékek összehasonlítására lesz lehetőség a diagramon, hanem

azok nagyságrendjének összehasonlítására, hiszen például a 85 és a 100 értékek gyakorlatilag azonos magasságú oszlopot eredményeznek.

4.1.2. Vonaldiagram

Amennyiben időben *folyamatosan változó adatok*at akarunk ábrázolni, arra a vonaldiagram alkalmas (23. ábra). A **vonaldiagram** olyan, mint egy grafikon, de a *vízszintes tengely*en mindig az *időt ábrázolja*, a vonal pedig a *függőleges tengely*en reprezentált egységekben az adatok adott időpontban felvett értékét.



23. ábra: Vonaldiagram

Legtöbbször valóban folyamatosan változó adatokat ábrázolunk vonaldiagramon, de néha előfordul az is, hogy bizonyos időpontokban ismert adatok közötti változást akarunk jelezni a mért értékek folyamatos vonallal történő összekötésével. Ilyenre példa a 23. ábra is, ahol a nyilvánvalóan naponta csupán egyszer felvett maximum vagy minimum hőmérsékletek összekötésével az időjárás napi változását lehet hangsúlyozni.

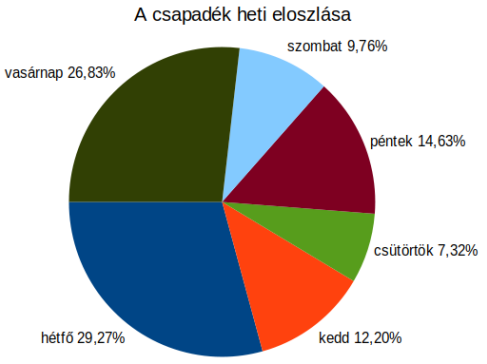
Mint az ábrán látható, a vonaldiagram is képes több adatsor együttes ábrázolására. Az ábrán a ténylegesen mért adatokat külön jelek is jelzik, ami szintén utal arra, hogy a köztük levő vonalak ez esetben csupán képzeletbeli összeköttetést jelentenek.

Az adatsoroknak és egy adatsoron belül az adatoknak az oszlopdiagramhoz hasonlóan itt is azonos nagyságrendbe kell esniük – bár akár a vízszintes, akár a függőleges tengelyen lehetőség van a logaritmikus skála használatára ezúttal is.

A vonaldiagramnak is van térbeli változata, illetve létezik *terület diagram* is. Ezeknek az a hátránya, hogy ezek kitakarhatják az adatok egy részét, így látványos lehet ugyan, de értelmesen szinte egyáltalán nem lehet használni.

4.1.3. Kördiagram

Amennyiben egy adatsor nem egymással összehasonlítandó adatokat tartalmaz, hanem egy nagyobb egységen belüli felosztás értékeit, akkor a **kördiagram** (24. ábra) segítségével az adatok egymás közti százalékos megoszlását lehet ábrázolni. Ilyen lehet például ha a héten lehullott csapadék mennyiségéről szeretnénk megmutatni, hogy annak az egyes napokra mekkora hányada jutott. Ekkor a heti teljes csapadékmennyiség adja a teljes kört. Az egyes körcikkek pedig azt mutatják, hogy ennek mennyi százaléka esett az egyes napokra. A diagram a heti teljes csapadékmennyiséget nem tudja megmutatni.

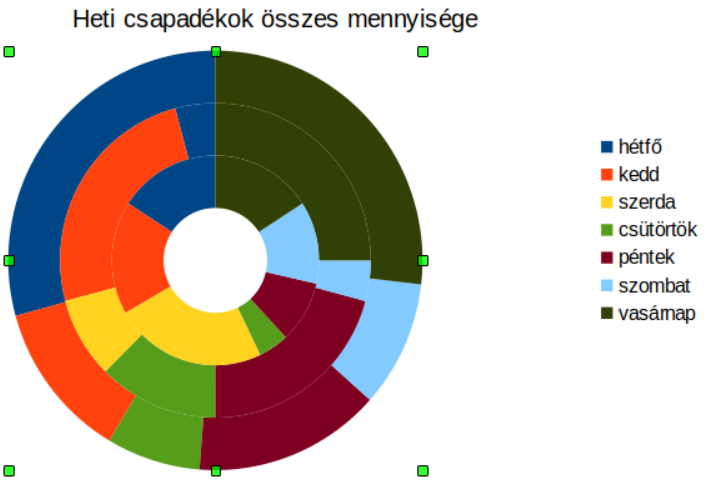


24. ábra: Kördiagram

A heti teljes csapadékmennyiséget egy halmozott oszlopdiagramról lehetne leolvasni. A 24. ábrán látható kördiagram látható a 22. ábra oszlopdiagramjának baloldali oszlopán, ahol leolvasható, hogy összesen valamivel több, mint 40 mm csapadék eloszlásáról van szó.

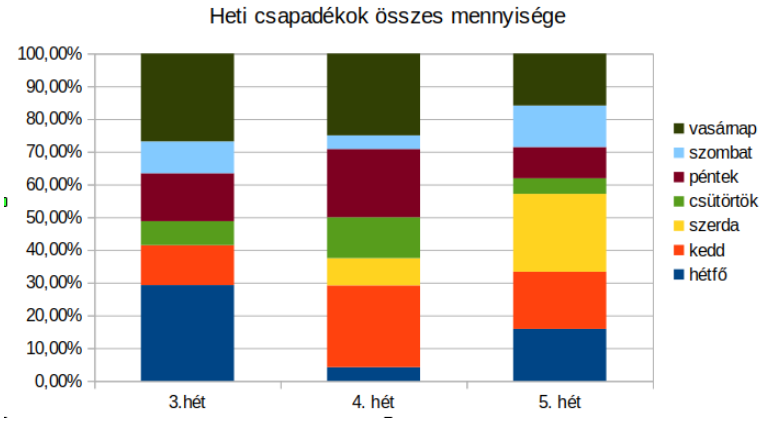
Amennyiben a példában szereplő mindhárom hét esetében szeretnénk megmutatni a napi csapadék eloszlását, akkor három kördiagramot kellene elkészítenünk, mivel a kördiagram csak egy

adatsort képes egyszerre ábrázolni. Igaz, létezik a kördiagramnak olyan változata is (lásd 25. ábra), amelyen egy-egy körsávban lehetséges több adatsort is megjeleníteni, de ekkor a körcikkek mellé nagyon nehéz elhelyezni már a százalékértékeket mutató adatfeliratokat, így ez a fajta diagram nagyon nehezen értelmezhető.



25. ábra: Több adatsoros kördiagram

Ha több adatsor esetén szeretnénk egy diagramon szemléltetni a százalékos megoszlást, akkor inkább az oszlopdiagramnak egy speciális változatát célszerű alkalmazni. Ez egy olyan halmozott oszlopdiagram, amelynél az oszlopok magassága egyforma, így az oszlopon belül a színek az adott kategória százalékos felosztását mutatják az egyes értékek között. Ezen jobban láthatók az egyes adatsorok adatai.



26. ábra: Halmozott, százalékos oszlopdiagram több adatsor százalékos megoszlásának szemléltetésére

Az ilyen oszlopdiagramot hivatalosan **százalékos halmozott oszlopdiagram**nak (26. ábra) hívják. Ez a többi oszlopdiagramtól eltérően nem adatok egymással összehasonlítására, hanem több adatsor százalékos eloszlásának szemléltetésére alkalmas.

Mint az ábra mutatja, a függőleges tengely ilyenkor a százalékértékeket mutatja, alul az egyes száz százaléknak számító kategóriák neve látható és a színek jelentését a jelmagyarázat adja meg.

Természetesen ennek is, mint a többi oszlopdiagramnak, van térbeli változata is, ahol a henger és a hasáb alak a jól használható, míg a másik kettő, amely felül elvékonyodik, nem feltétlenül alkalmas az adatok szemléltetésére.

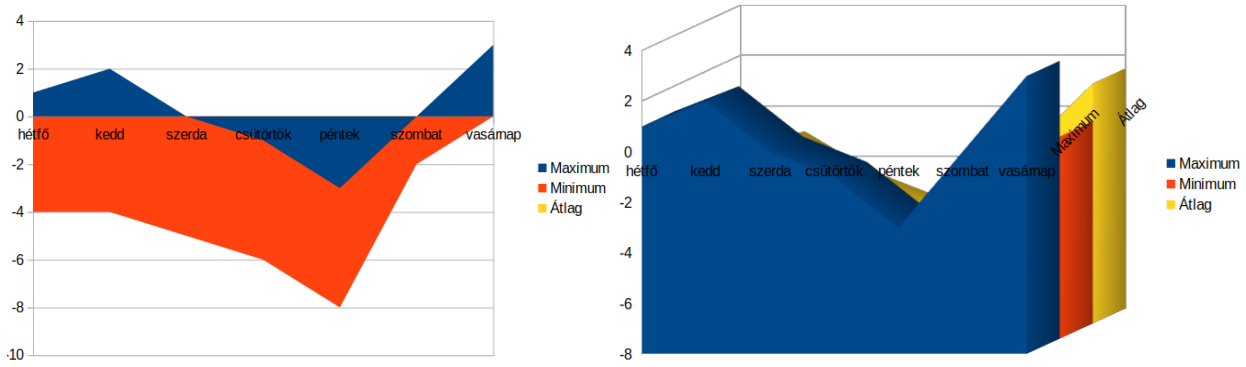
Visszatérve a kördiagramra, természetesen annak is létezik térbeli változata, amelyet **tortadiagram**nak nevezünk. Ennek használata során figyelni kell arra, hogy a körcikkek arányosan látszódjanak.

4.1.4. További diagramtípusok

A LibreOffice további diagramtípusokat is ismer, amelyekről az alábbiakban lesz szó. Ezek ritkábban használt, speciálisabb esetekben alkalmazható típusok.

Az első ezek közül a *sávdiagram*, amely gyakorlatilag megegyezik az oszlopdiagrammal, mindössze egyetlen különbséggel: a vízszintes és a függőleges tengely felcserélődik. Ennek eredményeként az oszlopok nem függőlegesen, hanem vízszintesen helyezkednek el. Jól használható például *korfa* ábrázolására.

A *területdiagram* (27. ábra) az értékeket pontonként jeleníti meg az *y-tengely*en. A pontokat azonban a vonaldiagrammal ellentétben nem egyszerűen vonalak kötik össze, hanem a vonal és a vízszintes tengely közti terület az adott kategóriához tartozó színnel ki lesz töltve. A térbeli változa-



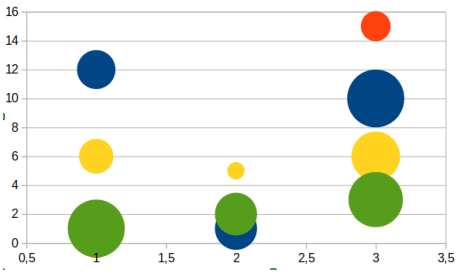
27. ábra: Terület diagram (balra a sík, jobbra a térbeli változata). A baloldali diagramon az Átlag adatsor nem látszik, mert a másik kettő valamelyike mindig elfedi.

ta még furcsább eredményt ad: amennyiben több adatsor ábrázolására kerül sor, akkor azok egymás mögött jelennek meg, nagy valószínűséggel az egyik kitakarja a másikat.

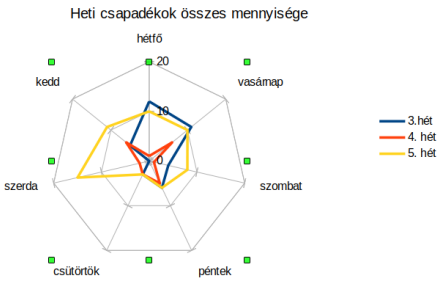
A vonaldiagramról ugyan már volt szó, de érdemes megemlíteni, hogy vannak különböző változatai, amelyek között van *térbeli*, csak a vonalakat megjelenítő, csak a tényleges értékeknél pontokat megjelenítő és a 23. ábrán már bemutatott „*pontok és vonal*” altípusa.

Látszólag a vonaldiagrammal azonosan néznek ki a „*Pont (XY)*” típus altípusai. Azonban ebben az esetben a vízszintes tengely nem az időt tartalmazza. Ez a diagram alkalmas grafikonok készítésére, mivel az egyes pontok egy adatsor $x \rightarrow y$ érték-hozzárendeléseit jelenítik meg. Vagyis ha egy függvény grafikonját szeretnénk ábrázolni – vagy koordináta-geometriai feladat megoldásához akarunk segítséget –, akkor ezt a típust kell használnunk.

Statisztikusoknak való a *buborékdiagram*, amely három változó egymáshoz képesti viszonyát képes ábrázolni: az *x* és *y* tengelyek menti elhelyezkedés az első két változó értéke, a harmadik érték pedig a buborék átmérője.



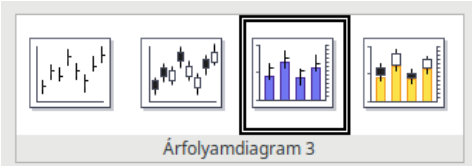
28. ábra: Buborékdiagram



29. Ábra: Háló diagram

Szintén statisztikusok használják előszeretettel a *háló* típusú diagramokat. Ez a diagramfajta nem derékszögű koordináta rendszert használ, hanem egyfajta polárkoordináta rendszert. Az egyes tengelyek mentén egy-egy kategória jelenik meg, az adatsor értéke pedig ezen bentről kifelé haladva a megfelelő távolságra metszi a tengelyt. Ilyen módon minden egyes adatsorhoz kialakul egy poligon – amelynek pontosan annyi csúcsa van, ahány kategória-tengely van a diagramban.

Az *árfolyamdiagram* a neve alapján pénzügyi adatok ábrázolására alkalmas, de ahogy a 30. ábrán látható, a jellemzője elsősorban az, hogy nem egy konkrét érték, hanem egy érték-intervallum jelenik meg minden adat esetén. Ezáltal olyan helyeken használható, ahol nem csak a pillanatnyi értéket, hanem annak hibahatárát is ábrázolni szeretnénk. Ilyen természetesen az árfolyam, de hasznos lehet ez a diagramfajta akár a csillagászoknak is, akik például egy csillag fényességét és annak változása során előforduló eltérését is szemléltetni szeretnék.



30. ábra: Árfolyamdiagram altípusai

Az utolsó típus valójában egy összetett diagram, ahol az adatsorok egy része vonaldiagramként, más része oszlopdiagramként jelenik meg. Hogy mennyi adatsor kapjon vonalat, azt be lehet állítani.

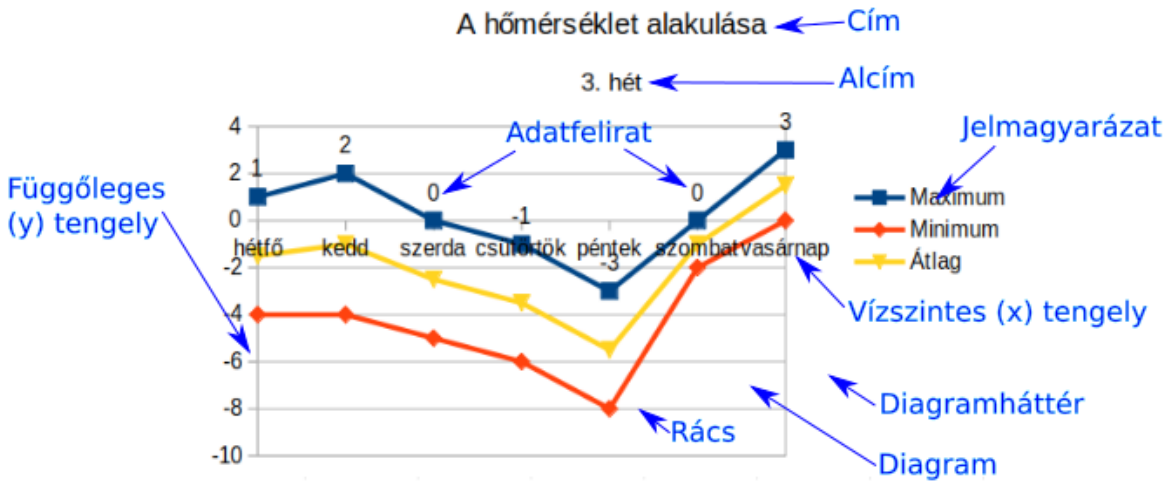
4.2. A diagram alkotóelemei

A diagram elemeit – lévén adatok grafikus ábrázolására alkalmas eszközről szó – elsősorban ábrán lehet bemutatni, nem írott szövegben. Az alábbi ábrákon bemutatott elemek mindegyike esetében lehet formázni a megjelenést, egyes elemek esetében még azt is, hol jelenjen meg.

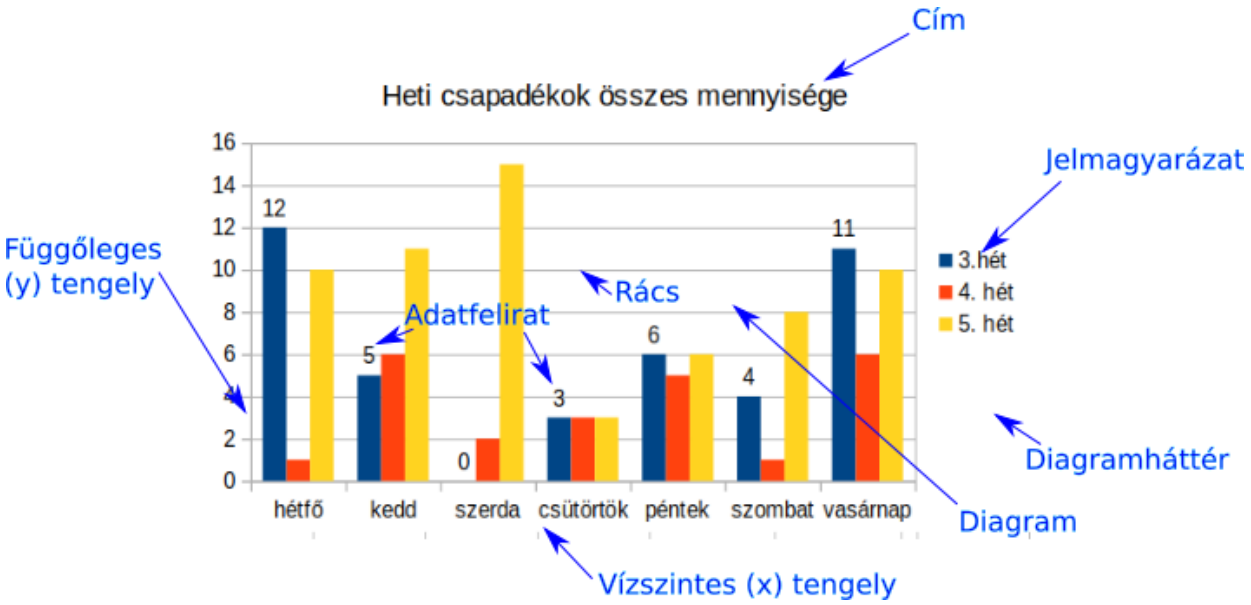
Az ábrákon persze nem minden formázható elem szerepel részben azért, mert nem lehetett volna egyértelműen bejelölni, részben mert nem minden elem jelenik meg minden esetben.

A diagramelemek röviden felsorolva a következők:

- A *diagramháttér* vagy *diagramterület* a teljes, a diagramot tartalmazó vászon. A diagram minden eleme erre kerül rá. A kitöltési módját – másképp fogalmazva: a hátterét – lehet beállítani. Mint általában a hátterek esetében, itt is figyelni kell arra, hogy ne tegye olvashatatlanná az előtte levő tartalmat.
- Maga a diagram is rendelkezik háttérrel! Az ábrákon ezt jelzi a „*Diagram*” szöveg. A diagram fő része ezen helyezkedik el.
- Vonaldiagram és oszlopdiagram esetén lehet *vízszintes* és *függőleges tengely*ről beszélni. Vonaldiagram esetén ezeket egyértelműen lehet *x*, illetve *y tengely*nek is nevezni. Oszlopdiagram esetén a függőleges tengelyt továbbra is lehet *y tengely*nek nevezni, de a vízszintes tengelyen az adatok kategóriái jelennek meg.
- Szintén vonaldiagram és oszlopdiagram esetén van rács és alrács (utóbbi nincs az ábrákon), amelyek akár vízszintes, akár függőleges (ez sem látható az ábrákon) irányban elhelyezkedve az egyes értékek leolvasását segíthetik. Beállításuk a megfelelő tengely beállításánál kereshető. Térbeli diagram esetén még lehet *z tengely* is.

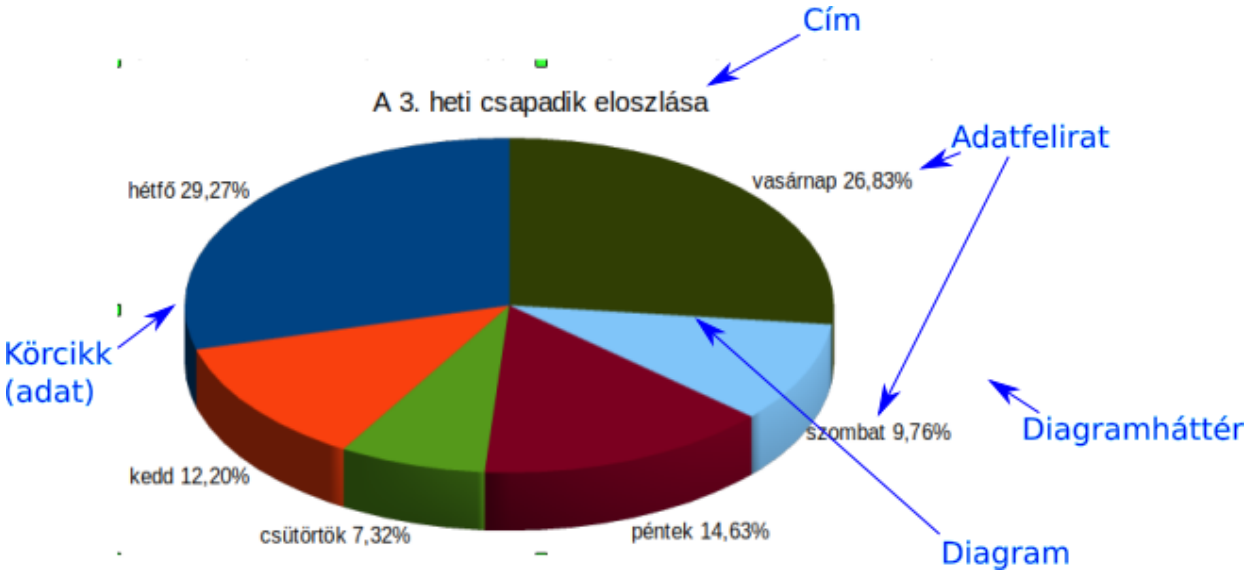


31. ábra: A vonaldiagram elemei



32. ábra: Az oszlopdiagram elemei

- *Adatfelirat* megjelenítésével szintén segíthetünk a pontos értékek leolvasásában. Kördiagram esetén általában ezt használjuk a jelmagyarázat helyett is.
- *Jelmagyarázat* bármelyik típusú diagramhoz alkalmazható (az ábrákon a kördiagram esetén nem szerepel, mert ott a funkcióját az adatfeliratokkal meg lehet oldani). A jelmagyarázat azonosítja be az egyes adatsorokat azok színeinek megadásával. Vonaldiagramnál még az *adatpontok* formája is megjelenik a jelmagyarázaton. A jelmagyarázat pozíciója változtatható: megjelenhet baloldalt – mint az ábrákon –, jobboldalt, alul vagy felül. Felülre persze nem jó ötlet tenni, hiszen ott van a cím is!
- Vonaldiagramon a pontos adat helyét lehet jelezni az *adatpont* megjelenítésével is.
- A táblázatkezelő *adatpont*nak hívja az egyes adatokat megjelenítő oszlopot/vonalat/körcikket is. Ezzel szemben egy *adatsor* az azonos színű diagramelemek halmaza, ezt mutatja meg a jelmagyarázat.
- A *cím* és az *alcím* megadja, hogy mit ábrázol a diagram.



33. ábra: A kördiagram elemei

4.3. A diagramkészítés szabályai

Amennyiben valamilyen adat szemléltetésére diagramot akarunk készíteni, mindig tartsuk be az alábbi szabályokat!

- Egy diagramon mindig egymással összetartozó adatokat szabad csak ábrázolni. Nem való ugyanarra a diagramra például az emberek magassága és a tulajdonukban levő autók életkora, mivel ezek az adatok nem kapcsolódnak egymáshoz. Lehet mondani persze, hogy ugyanazokról a személyekről van szó, de ettől még a két adatsornak nincs köze egymáshoz.
- Az ábrázolni kívánt adatok nagyságrendje egyezzen meg! Amennyiben az adatok nem esnek egy nagyságrendbe, akkor a többinél nagyobb nagyságrendű adat elnyomja a társait (lásd 34. ábra), a többieknél kisebb nagyságrendű adat pedig nem fog látszani.
- Ha mégis eltérő nagyságrendű adatokat kell ábrázolni, ahol éppen a nagyságrendek szemléltetése a cél, akkor használjunk logaritmikus skálát.

34. ábra: Amikor egy adat nagyságrendje eltér a többitől...

4.4. A diagramkészítés folyamata

A LibreOffice Calc és az MS Office Excel diagramkészítési módszere nagyon eltér egymástól, így az alábbiak csak az előbbire érvényesek, ám az Excel esetén szükséges lépéseket könnyen meg lehet állapítani a gyakorlatban a Calc-ban alkalmazandóak ismeretében.

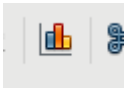
1. Első lépésként meg kell határozni a diagramon ábrázolandó adatokat. A diagram elkészítése előtt ki kell jelölni ezeket az adatokat. Fontos, hogy a kijelölésnek téglalap alakúnak kell lennie. Ez persze nem jelenti azt, hogy egyetlen összefüggő tartományt lehet csak kijelölni. A 35. ábrán a fejezet eddigi példáihoz használt táblázat látható egy csapadékmennyiségeket ábrázoló diagram számára történt kijelöléssel (a világoskék hátterű cellák vannak kijelölve). Mint az ábrán látható, a **B5:B11** tartományban vannak a hét napjai, ennek ellenére a kijelölés a **B4:B11** tartományt tartalmazza. Ennek oka, hogy ha a **G** oszlopban az adatsor nevét is kijelöljük, akkor így áll elő téglalap alakú kijelölés. Ha a **C:F** oszlopokat képzeletben eltüntetjük, akkor a kijelölés egy teljes téglalappá áll össze – ezt jelenti a téglalap alakú kijelölés.

	A	B	C	D	E	F	G	H
1								
2		Hőmérséklet és csapadék adatok Celsius fokban						
3		3.hét					3.hét	
4			Maximum	Minimum	Átlag	Ingadozás	Csapadék	
5		hétfő	1	-4	-1,5	5	12	
6		kedd	2	-4	-1,0	6	5	
7		szerda	0	-5	-2,5	5	0	
8		csütörtök	-1	-6	-3,5	5	3	
9		péntek	-3	-8	-5,5	5	6	
10		szombat	0	-2	-1,0	2	4	
11		vasárnap	3	0	1,5	3	11	
12		átlag						

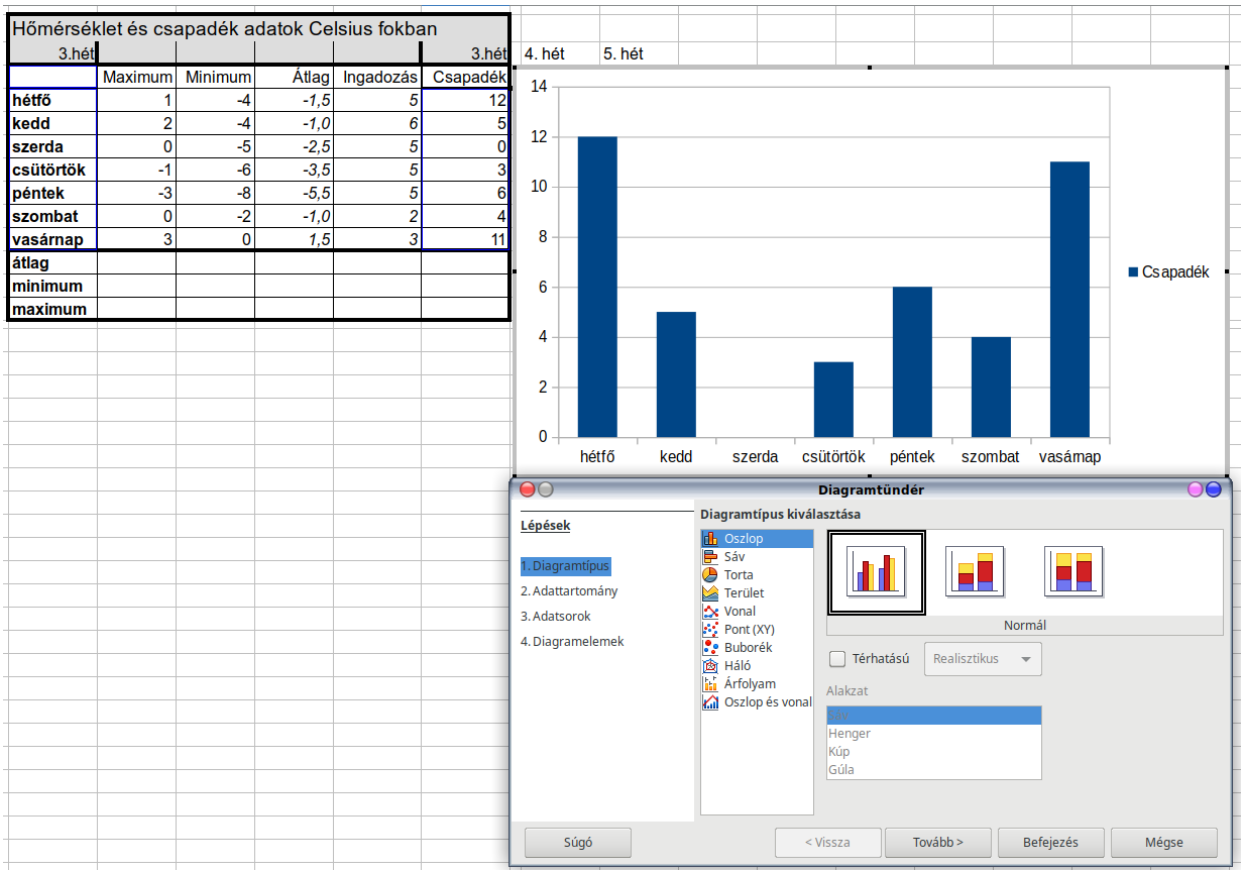
35. ábra: Téglalap alakú, de nem összefüggő kijelölés

   A több külön álló elemből álló kijelölés létrehozásához először ki kell az egyik komponenst jelölni, majd ehhez a **CTRL** billentyű lenyomva tartása mellett lehet új komponenseket hozzá venni.

2. Amennyiben megtörtént a megfelelő cellák kijelölése, elkezdhető a diagram beszúrása. Erre a *Beszúrás → Diagram* menü vagy „*Szokásos*” eszköztáron található gomb (36. ábra, *Galaxy gombstílus* használata esetén) megnyomása ad lehetőséget. Bármelyiket választjuk is, elindul a „*Diagramtündér*” nevű párbeszédablak és egyben valahol megjelenik a leendő diagram is úgy, ahogyan az aktuális beállítások alkalmazása mellett kinézne. Ha a dokumentum olyan nagyítási beállítással rendelkezett, hogy az eredeti táblázatot sem a párbeszédablak, sem a diagram nem takarják el, akkor a kijelölés celláit kékkel bekeretezve lehet látni (lásd 37. ábra).

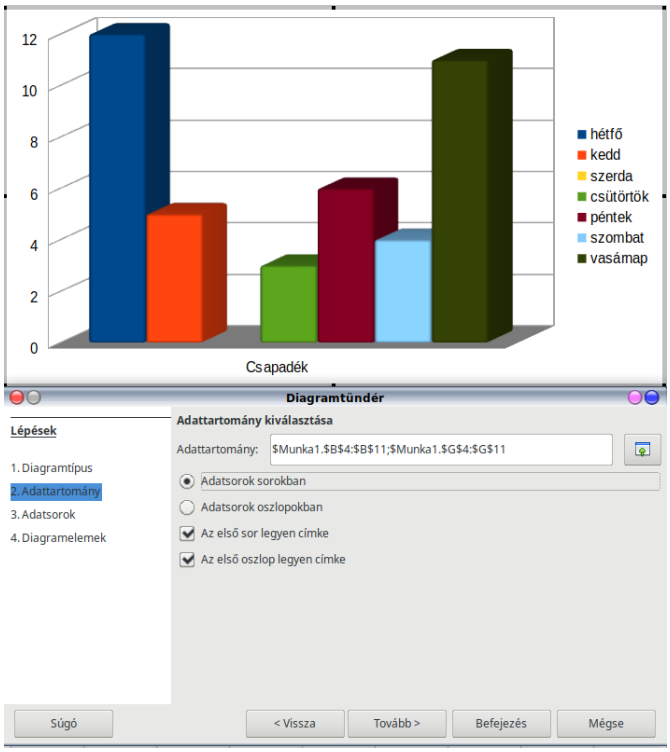


36. ábra



37. Ábra: A Diagramtündér a leendő diagram mintájával és az eredeti kijelöléssel

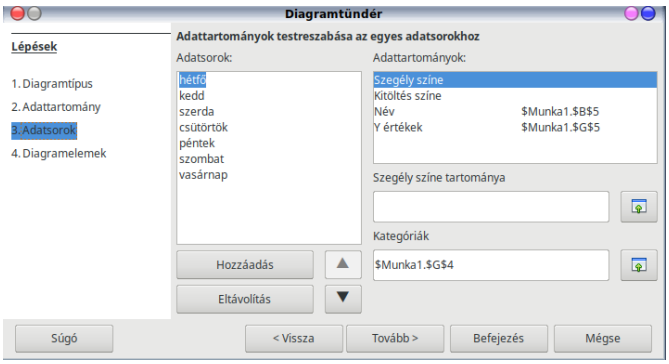
3. A diagramtündér négy lépésben segít a diagram elkészítésében, de a „*Befejezés*” gombra kattintva is lesznek még beállítási lehetőségek. A tündér első lépésében a diagram típusát lehet kiválasztani. A fejezet korábbi részében ismertetett valamennyi típust és azok altípusait ki lehet itt választani. A diagram kis késéssel automatikusan átalakul a kiválasztott típusra, tehát gyakorlatilag élőben meg lehet nézni, hogyan fog kinézni a diagram.
4. A tündér következő lépése az adattartomány megadása. Amennyiben esetleg úgy jutottunk el idáig, hogy nem volt semmi sem kijelölve, akkor ez az „*Adattartomány*” résznél most pótolható – és pótlandó. Az alatta levő választási lehetőség azt határozza meg, hogy a kijelölt tartomány melyik része számít egy adatsornak. Egy adatsor azonos színű oszlopokat hoz létre oszlopdiagram esetén, így a 38. ábrán látható színes oszlopok is előállíthatóak eredetileg egy adatsorra is a megfelelő beállítással. Kördiagram esetén nem



38. Ábra: Az adattartomány beállítása

mindegy, mi van itt beállítva, mivel a kördiagram csak egy adatsort tud ábrázolni, így ha ezt rosszul állítjuk be, akkor egyetlen száz százalékot érő, egyszínű kör lesz az eredmény! Szintén az adatsorok meghatározását segíti a két kapcsoló, amellyel a kijelölés első sorát és oszlopát lehet címkeként definiálni. Így az itt található értékek jelennek meg a kategória és az adatsor neveként – ezért kellett a **G4** cellát, és vele a **B4** cellát is kijelölni: a **B4** cella tartalma jelenik meg a vízszintes tengelyen kategórianévként.

5. Vannak esetek, amikor nem lehetséges téglalap alakú területet kijelölni. Ilyenkor használjuk az „*3. Adatsorok*” lépést (39. ábra) a kijelölt adatok módosítására. Normális esetben ehhez nem kell hozzányúlni. Ám ha nem sikerült úgy kijelölni az adatokat, hogy a jelmagyarázat illetve a kategórianevek megfelelően látszódjanak, akkor ezeket meg lehet adni akár a megfelelő cellák megadásával, akár (a kategória neve esetében) közvetlenül beírva.



39. ábra: Az adatsorok megadására szolgáló lépés

6. A diagramtündér utolsó lépésében (40. ábra) lehet megadni a diagram címét, alcímét, valamint – a korábban, a diagramelemeken nem szereplő – tengelycímkéket, amelyek az adott tengely mellett megjelennek. A *z tengely* csak egyes diagramok esetében jelenik meg, a másik kettő mindig szerepel valamilyen formában. A párbeszédablak jobb oldalán lehet megadni, hogy látszódjon-e a jelmagyarázat és ha igen, akkor hol helyezkedjen el.

40. ábra: A cím és a tengelyek nevének megadása

7. A „*Befejezés*” gombra kattintást követően a párbeszédablak bezárul, a diagram pedig megmarad, de ez még mindig módosítható. A diagram határát egy szürke keret jelzi. Amíg ez a keret látszik, addig a diagram formázható, szerkeszthető. A diagramon kívülre kattintás után a szegély eltűnik, a diagram egyetlen rajzobjektumként viselkedik. A szerkesztő módba a diagramra történő dupla egérkattintással vagy a helyi menü legutolsó, „*Szerkesztés*” menüpontjával bármikor vissza lehet menni. Az egyes diagramelemekre kattintva az egér jobb gombjával, különböző szerkesztési, formázási lehetőségek érhetőek el. Többek között itt lehet beszúrni az egyes adatsorokhoz az adatfeliratokat – majd külön lépésben ezek formázását kezdeményezni.

A diagram formázási lehetőségeit nem érdemes itt felsorolni – annyi lehetőség van. Ezeket néhány kísérletképpen létrehozott diagramon érdemes inkább felfedezni.

Egy fontos dolgot kell még megjegyezni: Amíg a diagram szerkesztő módban van, a program mentés funkciója nem a munkafüzetet, hanem csak a diagramot fogja menteni. Ez hasznos lehet, amennyiben tényleg ez a cél, hiszen így az önálló `.odc` állományban elhelyezkedő diagramot például egy szöveges dokumentumba is be lehet illeszteni – az egyik funkciója a diagramoknak éppen az lehet, hogy egy kutatás eredményét mutassuk be vele egy cikkben.

Ha viszont az egész munkalap mentése helyett sikerül csak a diagramot menteni, annak nagyon kellemetlen következményei lehetnek!

5. Egyszerű statisztikai függvények

A táblázatkezelőben a képletekben használhatunk függvényeket is, amelyek előre megírt műveletsorokat hajtanak végre a paraméterül kapott adatokon.[9] Az alábbiakban a leggyakrabban használt, elsősorban statisztikai számításokhoz használható függvények szerepelnek.

5.1. Összegzés: Sum

A leggyakrabban előforduló művelet az egyes cellákban szereplő értékek összegzése. Erre alkalmas a **Sum()** vagy magyar változatban **Szum()** függvény. Ez a függvény bármely számként értelmezhető adaton működik. A függvénynek legfeljebb 30 paramétere lehet. Minden paraméter lehet egyetlen cella vagy egy tartomány is. Utóbbi esetben a tartomány valamennyi számként értelmezhető celláját külön értékként kezeli.

5.2. Átlagolás: Average

Sokszor előfordul, hogy értékek átlagát kell kiszámolni. Erre alkalmas a legegyszerűbb esetben az **Average()**, magyarul: **Átlag()** függvény. A **Sum()** függvényhez hasonlóan működik, csak éppen a megadott értékeknek nem az összegét, hanem az átlagát számolja ki.

Azaz paraméterként számként értelmezhető értékeket tartalmazó cellákat, tartományokat vagy konkrét számértékeket lehet megadni – ebben az esetben is legfeljebb 30 paraméterig.

Amennyiben a paraméterben szereplő cellák egyikében sincs számként értelmezhető adat, akkor a függvény hibaüzenetet eredményez, hiszen nulla darab számérték átlagolása nem lehetséges (egészen pontosan a nullával osztás miatt keletkezik hiba).

Amennyiben a függvény nevét kiegészítjük egy A betűvel, akkor olyan függvényt kapunk (**AverageA()** illetve **ÁtlagA()** néven), amely a szöveget tartalmazó cellákat nulla értékkel veszi figyelembe.

5.3. Legkisebb/legnagyobb érték meghatározása

Időnként szükség van arra is, hogy egy értékhalmazból a legkisebb, a legnagyobb, esetleg a valahanyadik legkisebb/legnagyobb értéket meg lehessen határozni. Erre alkalmasak a **Min()**, **Max()** **Small()** és **Large()** függvények. Előbbi kettő magyar neve megegyezik az angollal, utóbbi kettő magyarul rendre **Kicsi()** és **Nagy()** néven létezik.

A **Min()** és a **Max()** függvények legfeljebb 30 paramétert kaphatnak a **Sum()** és **Average()** függvényeknél megismert módon – ezúttal a legkisebb, illetve a legnagyobb értéket adva meg a paraméterben szereplők közül. Mindkét függvény kiegészítve egy A betűvel (**MinA()** és **MaxA()**) a szöveget tartalmazó cellák is figyelembe lesznek véve 0 értékkel.

A **Small()** függvény a **Min()** függvénynek, a **Large()** pedig a **Max()** függvénynek olyan általánosítása, amelyben második paraméterben egy sorszámot kell megadni. Ez a sorszám meghatározza, hogy hanyadik legkisebb/legnagyobb értéket kell a függvénynek eredményül adnia. Ebben a változatban csak egyetlen paraméterrel adhatóak meg a vizsálni kívánt cellák.

9 *Léteznek olyan függvények is, amelyek nem kapnak semmilyen adatot paraméterként. Ezeknek üres a paraméterlistájuk és nyilvánvalóan az általuk végzett művelet eredménye nem függ semmilyen paraméter értékétől sem.*

Például a `Large(A2:A20; 3)` függvény az **A2:A20** tartomány harmadik legnagyobb számértékét adja eredményül.

5.4. Megszámlálások

Amennyiben valamilyen tulajdonságnak megfelelő cellákat szeretnénk megszámolni, akkor a következő függvényeket hívhatjuk segítségül: **Count()** vagy **Darab()**, **CountA()** vagy **Darab2()**, **CountBlank()** vagy **DarabÜres()** és **CountIf()** vagy **Darabteli()**.

A **Count()** függvény a legfeljebb 30 paramétere által megadott cellák közül megadja, mennyi tartalmaz számként értelmezhető értéket.

A **CountA()** nem csak a számokat, hanem a szöveget tartalmazó cellákat is figyelembe veszi, azaz azt adja meg, hogy a paramétereiben összesen hány nem üres cella található.

A **CountBlank()** éppen ellenkezőleg, azt adja meg, hogy a paraméterei összesen hány üres cellát tartalmaznak (vigyázat: az üres szöveget tartalmazó cella nem üres)[10].

A fenti függvények legfeljebb 30 paramétert kaphatnak a korábban megismert módon. Ezzel szemben az utolsó függvény összesen két paramétert kaphat:

```
CountIf(tartomány; feltétel)
```

Aki annak idején először magyarra fordította az Excel függvényneveit, az nyilvánvalóan nem értette, mit csinál ez a függvény, hiszen a függvény nevének helyes fordítása *DarabHa* lett volna, nem pedig **DarabTeli()**. Mindenesetre a hiba azóta sem lett kijavítva – nyilván a visszafelé kompatibilitás miatt.

A függvény a `tartomány` celláiban megszámolja azokat a cellákat, amelyek megfelelnek a `feltétel`nek. A feltétel vagy egy konkrét érték, vagy idézőjelek között egy reláció. Ebben az esetben is mindig először a relációs jel, utána pedig az érték következik, amellyel a vizsgált cella értékét össze kell hasonlítani. Adott érték esetén a függvény akár szövegek előfordulásának megszámlálására is alkalmas.

5.5. Összegek szorzata

A **Sum()** függvényhez hasonlóan egy vagy több tartomány celláit össze is lehet szorozni a **Product()** azaz a **Szorzat()** függvénnyel.

A **SumProduct()** vagy magyar nevén **Szorzatösszeg()** függvénnyel az egyébként két lépéses műveletet, amikor több oszlop vagy sor celláinak szorzatát kell összegezni, egyetlen lépésként végre lehet hajtani.

Vektoraritmetikában használatos kifejezésekkel élve a paraméterében megadott egydimenziós tartományoknak mint vektoroknak a skaláris szorzatát adja eredményül:

A `Sumproduct(A2:A10; C2:C10; E3:E11)` függvény valójában a következő képlet végrehajtását jelenti:

```
=A2*C2*E3+A3*C3*E4 + … + A10*C10*E11
```

Legfeljebb 30 paramétere lehet a függvénynek, amelyek mindegyike azonos számú cellát megadó tartomány kell legyen. Hogy az egyes tartományok sorokat vagy oszlopokat adnak meg, az a függvényen belül keverhető, de mindegyik tartománynak vagy sort, vagy oszlopot kell megadnia. A függvény az egyes tartományok azonos sorszámú elemeit összeszorozza, majd az így kapott szorzatokat összegzi.

10 *Ennek a mondatnak majd a feltételes függvények használatakor lesz jelentősége.*

5.6. Kerekítések

Egy számértéket ha formázással kerekítünk, akkor a cella megtartja az értékét. Viszont vannak esetek, amikor ténylegesen a kerekített értékre van szükség a további számításoknál. Erre alkalmasak az alábbi kerekítő függvények:

`Round(szám; pontosság)` magyarul `Kerekítés(szám; pontosság)`: a `szám` értékét a `pontosság` által meghatározott pontosságúra kerekíti a matematikai kerekítés szabályai szerint. A `pontosság` a megtartandó helyiértékek számát adja meg, értéke lehet negatív is.

Például: `Round(12,34; 1)` értéke *12,3* lesz, `Round(1,6; 0)` értéke *2*, `Round(123,2; -1)` értéke *120*.

`RoundUp(szám; pontosság)` magyarul `Kerek.Fel()`: a `szám` értéket mindenképpen felfelé kerekíti a megadott `pontosság`nak megfelelően.

`RoundDown(szám; pontosság)` magyarul `Kerek.Le()`: a `szám` értéket mindenképpen lefelé kerekíti a megadott `pontosság`nak megfelelően.

A `pontosság` paraméter mindkét esetben a **Round()** függvénnyel azonos módon használható.

6. Rendezés

A táblázatban levő adatokat könnyebb átlátni, ha azok valamely oszlopuk alapján sorba vannak rendezve. Ezen felül további lehetőségek is vannak az adatok rendezésére. Ilyen lehetőség például a sorok helyett az oszlopok rendezése, vagyis amikor nem egy oszlop, hanem egy sor szerint történik a rendezés.

A másik lehetőség a több szempont szerinti rendezés sűrűbben fordul elő. Ennek abban az esetben van jelentősége, ha abban az oszlopban, amely szerint a rendezés történik, ugyanaz az érték többször is megismétlődik. Ebben az esetben egy második – szükség esetén harmadik, negyedik stb. – oszlop alapján az elsődleges szempont szerint azonosnak tekintett sorok sorrendjét lehet eldönteni.

6.1. A rendezés menete

Mielőtt a táblázatot rendeznénk, néhány fontos szabályt még meg kell ismerni!

Először is a rendezés mindig az **egész táblázatra vonatkozik**, nem pedig arra az oszlopra, amely a rendezés szempontját tartalmazza. Gyakori hiba szokott lenni, amikor valaki csak azt az oszlopot jelöli ki, amely alapján rendezni kell. A rendezés a táblázat sorainak felcserélését jelenti olyan módon, hogy a rendezés szempontjának választott oszlopban az előírt sorrend (növekvő vagy csökkenő) alakuljon ki.

A fentivel kicsit ellentmondónak tűnik a második szabály: a táblázatnak csak a rendezésbe beletartozó részeit jelöljük ki, azokat a sorokat már ne, amelyek a fő adatrészből számolt összesítő adatokat tartalmazzák. Ugyanis a rendezés – az első sor kivételével – minden kijelölt sorra fog történni, az összesítő sorokat viszont mindenképpen a helyükön kell hagyni.

Példaként a 41. ábra egy táblázatot mutat, amelyben az alapadatok alatt még három sor látható.

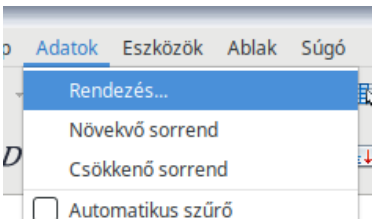
	A	B	C	D	E	F	G	H	I
1									
2		**Név**	**Helyi**	**Vidéki**	**GSM**	**Külföld**	**Összesen**	**Érték**	
3		Havasi E.	187 perc	77 perc	0 perc	75 perc	*339 perc*	*14 324 Ft*	
4		Kristóf I.	179 perc	40 perc	12 perc	44 perc	*275 perc*	*9 688 Ft*	
5		Antók E.	254 perc	0 perc	54 perc	0 perc	*308 perc*	*6 018 Ft*	
6		Lutz G.	24 perc	12 perc	87 perc	12 perc	*135 perc*	*6 993 Ft*	
7		Medve L.	96 perc	150 perc	0 perc	0 perc	*246 perc*	*7 152 Ft*	
8		Kovács J.	34 perc	42 perc	365 perc	36 perc	*477 perc*	*26 483 Ft*	
9		Három A.	11 perc	35 perc	15 perc	0 perc	*61 perc*	*2 357 Ft*	
10		Katrona E.	120 perc	36 perc	120 perc	58 perc	*334 perc*	*16 440 Ft*	
11		Iskola B.	198 perc	92 perc	0 perc	50 perc	*340 perc*	*12 056 Ft*	
12		**Összesen:**	*1 103 perc*	*484 perc*	*653 perc*	*275 perc*	***2 515 perc***	***101 511 Ft***	
13		**Érték**	*13 236 Ft*	*19 360 Ft*	*35 915 Ft*	*33 000 Ft*			
14									
15		Egységárak	12 Ft	40 Ft	55 Ft	120 Ft			
16									

41. ábra: A helyes kijelölés rendezéshez

Ebben az esetben az „*Összesen*” és az „*Érték*” sorok már összegző adatokat tartalmazó sorok, amelyeknek mindenképpen alul kell maradniuk, ezért azok nem kerültek kijelölésre.

Ugyanígy a két jobboldali oszlop is összegző adatokat tartalmaz, így logikusnak tűnik azokat is kihagyni a kijelölésből. Mivel sorokat fogunk rendezni, így ezek együtt fognak mozogni a rendezendő adatokkal akkor is, ha belevesszük a kijelölésbe és akkor is, ha nem vesszük bele – feltéve, hogy helyesen, képlettel lettek az értékek meghatározva ezekben a cellákban –, így a rendezés eredményét nem fogja befolyásolni ezen oszlopok kijelölése. Ám ha például éppen az összes beszélgetési idő alapján akarjuk rendezni a táblázatot, akkor feltétlenül szükséges az „*Összesen*” oszlop belevétele a kijelölésbe, hiszen ez lesz a rendezési szempont.

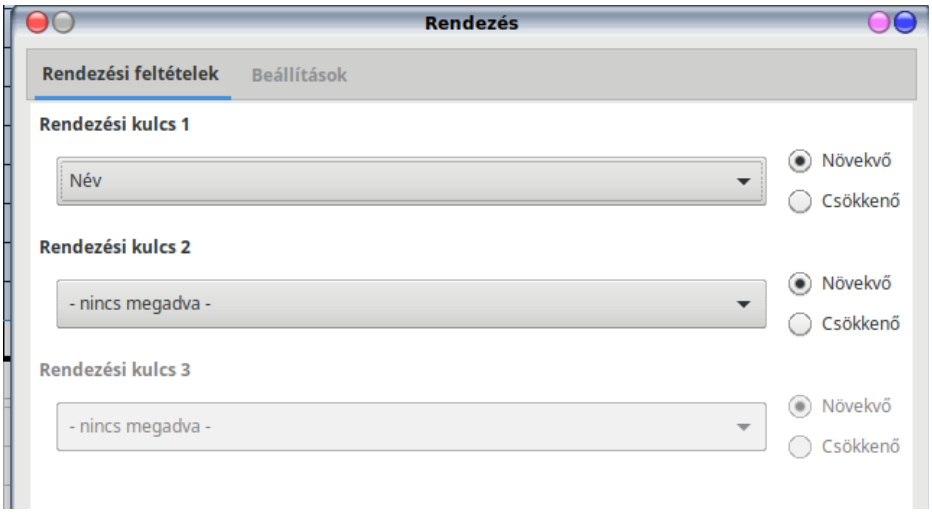
Az ábrán a táblázat első, fejléc sora is ki van jelölve. Amennyiben lehetséges, akkor a fejlécet tartalmazó sort mindenképpen ki kell jelölni, mert ekkor könnyebb lesz megadni a rendezés szempontját.



42. ábra

Mikor lehetséges ez a kijelölés? Amikor az ábrán látható módon, közvetlenül a rendezendő adatok felett elhelyezkedő egyetlen sorként szerepel. Amennyiben van a fejléc és a rendezendő adatok között ki nem jelölendő sor vagy a fejléc több sorban helyezkedik el, illetve van benne egyesített cella, akkor nem lehet hozzávenni a kijelöléshez.

Miután a megfelelő kijelölés elkészült, az *Adatok* → *Rendezés* menüpontot (42. ábra) kell kiválasztani a „*Rendezés*” párbeszédablak megnyitásához.



43. ábra: A rendezés párbeszédablaka

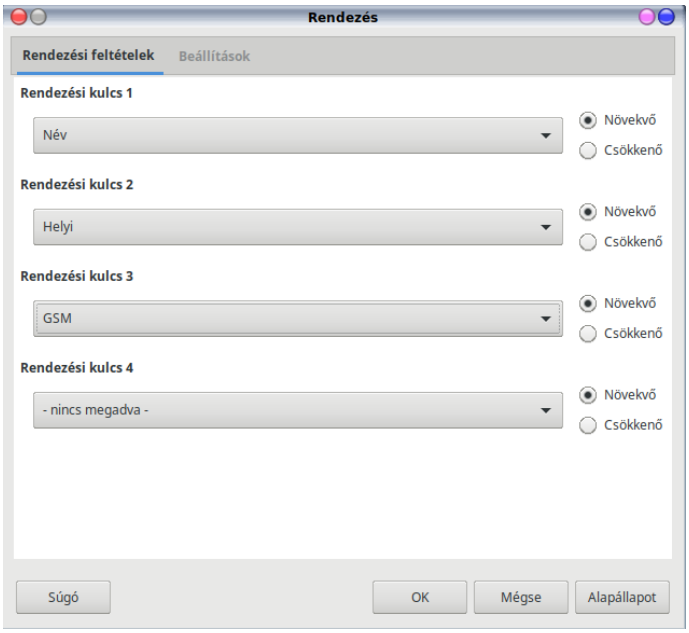
A párbeszédablaknak két lapja van. Ezek közül az első, a „*Rendezési feltételek*” teszi lehetővé a rendezés szempontjának vagy szempontjainak megadását. Ahogy az az ábrán látható, elsődleges szempontként egyből az az oszlop van felkínálva, amelyben az aktuális cella található volt a párbeszédablak megnyitásakor – ebben az esetben a „*Név*” oszlop. Természetesen a legördülő listából bármelyik oszlop kiválasztható.

Mint látható, a program a kijelölés első sorát fejlécként értelmezi és a legördülő listában az első sorban szereplő értékeket jeleníti meg. Így könnyen meg lehet találni azt az oszlopot, amely szerint a rendezést kell végezni.

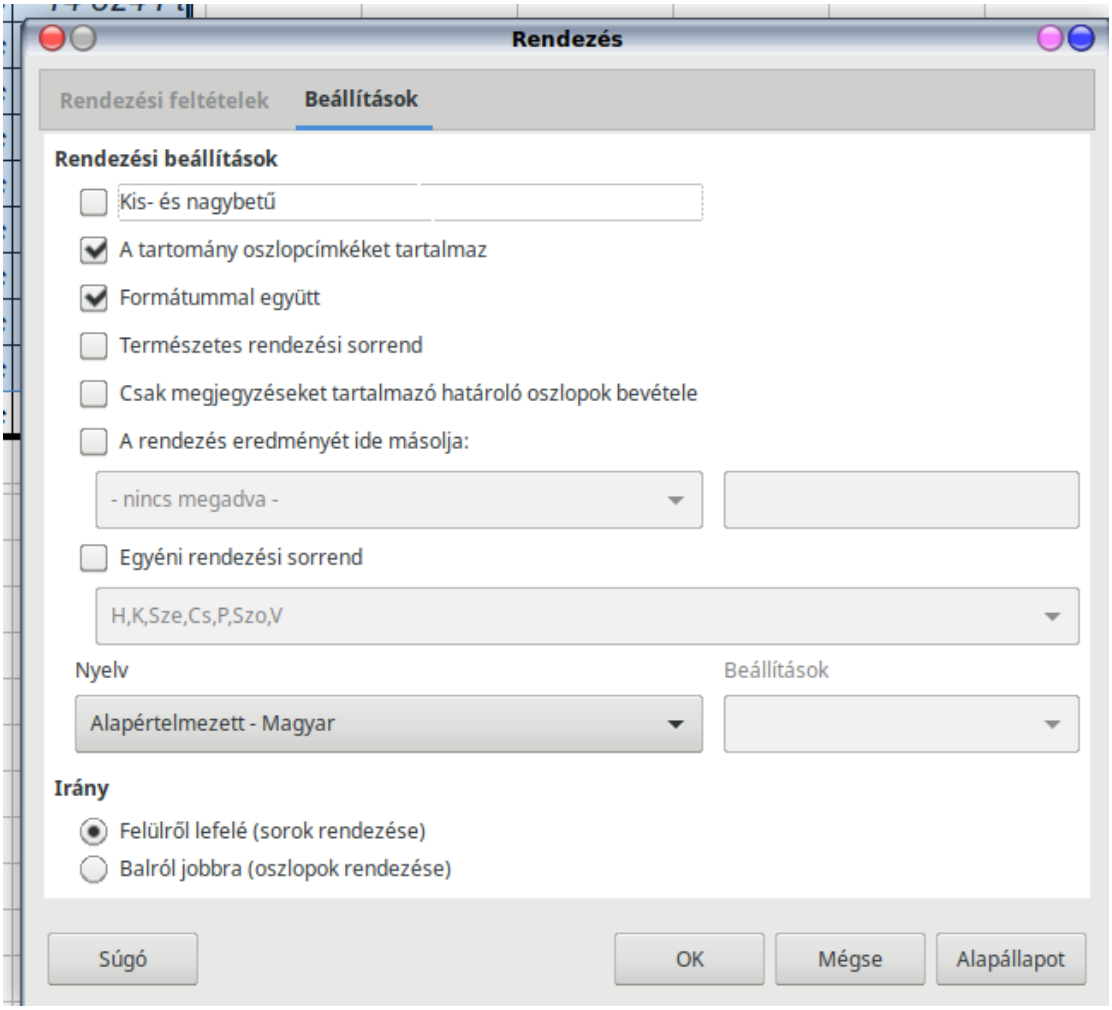
A legördülő lista jobb oldalán levő kapcsolóval könnyen beállítható a rendezés iránya is az adott szempontra.

Amennyiben beállítottunk egy rendezési szempontot, akkor elérhetővé válik a következő szempont is. Nem kell megijedni attól sem, hogy csak három szempont jelenik meg. Egyrészt ez azért van, mert az elsődleges szempont már be van állítva, így elérhető a második és még nem elérhetően a harmadik is látszik, de a harmadikat beállítva már a negyedik szempont is látszani fog. Másrészt nem valószínű, hogy két-három szempontnál többre van szükség.

A párbeszédablak másik lapjára, amely a „*Beállítások*” füllel érhető el (45. ábra) alaphelyzetben nem szokott szükség lenni. Például akkor van rá szükség, ha a kijelölés első sorát sem szabad fejlécként kezelni, a program mégis úgy akarja kezelni. Lássuk sorban, milyen beállítási lehetőségeket kínál ez a lap!



44. ábra: Újabb szempontok várnak elérhetővé, ha a már elérhetőeket beállítjuk.



45. Ábra: A Beállítások fül tartalma

- **Kis- és nagybetű**: bekapcsolt állapotában különbözőnek tekinti a rendezési szempont oszlopában a kis- és a nagybetűket a rendezés során és előre veszi a nagybetűket, kikapcsolt állapotban azonosnak számítanak.
- **A tartomány oszlopcímkéket tartalmaz**: Általában a program automatikusan megpróbálja megállapítani ennek a kapcsolónak a helyes állapotát. Bekapcsolt állapotban a kijelölés első sora – egy később ismertetésre kerülő beállítástól függően esetleg első oszlopa – jelenik meg a rendezési szempontok legördülő listáiban és ez a sor – oszlop – nem kerül rendezésre. Kikapcsolt állapotban a kijelölés egésze részt vesz a rendezésben és a rendezési szempontokban csak a – nagyon magyaros – „*Oszlop A*” … „*Oszlop E*” stb. szövegek jelennek meg. Ilyenkor nyilván nehezebb megkeresni a megfelelő rendezési szempontot, így ha csak lehet használjuk ki az oszlopcímkék használatának lehetőséget.
- **Formátummal együtt**: Alapvetően nem célszerű a szegélyezés után rendezni a táblázatot, mert ha ez a kapcsoló be van kapcsolva, akkor minden cella viszi magával a rá beállított összes formázást is. Így ha a táblázatnak kívül vastagabb szegélye volt és az utolsó sor valahova máshova kerül, akkor elromlik a szegélyezés. Ha a kapcsoló ki van kapcsolva, akkor a rendezés csak a cellák értékét – beleértve a képleteket is – fogja mozgatni, de a formázás helyben marad.
- **Természetes rendezési sorrend**: Ezzel a kapcsolóval a szöveggel kezdődő, de számmal végződő értékeket lehet olyan módon rendezni, mintha csak a számérték szerepelne. Ha a kapcsoló nincs bekapcsolva, akkor ezek az értékek a szövegeknél szokásos *lexikografikus rendezéssel* lennének rendezve. Például a természetes rendezési sorrendet alkalmazva az A1, A2, A3,… , A9, A10, A11, A12, … sorrend keletkezik, míg ugyanez lexikografikusan így nézne ki: A1, A11, A12, A13, … , A19, A2, A20, A201, A202, A21, A210, …
- **A rendezés eredményét ide másolja**: Amennyiben az eredeti táblázatot békén hagyva az eredményt egy másik helyen szeretnénk elhelyezni, akkor itt lehet azt megadni. A kapcsoló bekapcsolása után két lehetőség van:
  - Egy névvel rendelkező tartomány nevének kiválasztása a legördülő listában az eredménynek az adott tartományba helyezésére.
  - Egy cella vagy tartomány megadása a jobboldali mezőben (A 6.0 verzióban nem lehet egérrel kijelölni). Amennyiben egyetlen cella van megadva, akkor az lesz az eredményt tartalmazó cella bal-felső sarka. Tartomány megadása esetén az eredmény kilóghat a tartományból, tehát valójában ekkor is a bal-felső sarok számít csak!
- **Egyéni rendezési sorrend**: Az előre definiált listák használatával lehetőség van arra is, hogy a rendezési szempontot egy ilyen lista alapján rendezzük (például a hét napjai kerüljenek növekvő sorrendbe). Bármely már definiált egyéni listával lehetőség van rendezni, ha azt az itt található listából kiválasztjuk. Nyilván ilyen esetben nincs értelme a több szempont szerinti rendezésnek.
- **Nyelv**: Megadható, hogy a szöveg pontosan milyen ábécé szerint legyen rendezve. Nem csak a nyelvnek van jelentősége, hanem azon belül is lehet több lehetőség. Például német nyelv esetén a „telefonkönyv” lehetőségét választva az umlautos karakterek is részt vesznek a rendezésben. Ahol csak egyféle rendezési lehetőség van – mint a magyarban –, ott a második legördülő lista nem elérhető.
- **Irány**: Amennyiben nem a sorokat, hanem az oszlopokat mozgató rendezésre van szükség, akkor itt lehet az alapértelmezett „Felülről lefelé” helyett a „Balról jobbra” történő rendezést kiválasztani. Ebben az esetben a sorok és az oszlopok funkciója értelemszerűen felcserélődik, amit a kijelölésnél is figyelembe kell venni.

7. Dátumkezelő függvények

A táblázatkezelőben időadatokkal is lehet dolgozni. Ehhez nagyon sok függvényt kínál a program, amelyek a „Dátum és idő” kategóriában találhatóak. Ezek a fontosabbakról lesz szó ebben a fejezetben.

7.1. A dátum és az idő formátum

Előbb azonban a dátumok és időadatok kezelésére alkalmas számformátummal célszerű egy kicsit közelebbről megismerkedni.

A cellában az adat belső tárolása számként történik, azaz minden dátumnak és minden időnek egy-egy számérték felel meg. Ahhoz, hogy ez a számérték dátumként, időként vagy dátum- és időadatként jelenjen meg, csupán a megfelelő megjelenítési formátumra van szükség.

Egy adott dátum esetén a mögötte húzódó számérték nem más, mint a táblázatkezelő kezdődátumától eltelt napok száma – egész értékként. Ez a belső ábrázolási mód azt is lehetővé teszi, hogy két dátum között eltelt napok számát egyszerűen a két dátumot tartalmazó cella kivonásával meg lehessen kapni.

A kezdődátum programonként változik. Az Excel egy nem létező dátumot használ erre: 1990. január 0-át. Ennek magyarázata, hogy így 1900. január 1-hez már az 1 érték fog tartozni. LibreOffice Calc esetén alapértelmezésben a kezdődátum 1899. december 30., de átállítható a StarOffice-ból örökölt 1900. január 1. értékre is – amely az Excel kezdődátumához közeli valós dátum használatát jelenti –, vagy 1904. január 1. értékre is. Ez utóbbi értéket bizonyos külső formátumokból történő adatimportálás lehetővé tétele érdekében tették bele a fejlesztők.

Az időadatok belső ábrázolása a dátumokkal való kompatibilitást is lehetővé teszi. Egy napot a program ugyanis valós számként kezelve tud részekre bontani. Ezt a való számot kell a 24 óra, azon belül pedig a 60 perc, 60 másodperc mértékegységekre átkonvertálni – a kettes számrendszerben való ábrázolás miatt ebből lehetnek problémák. Így tehát például a 0,5 felel meg délnek, azaz 12:00:00 időértéknek, a 0,25 reggel 6 órának (6:00:00).

A fentiekből következően egy adott cella értéke egyszerre tud dátumot és az azzal megadott napon belüli időadatot is tárolni: a tárolt szám egész része adja meg a dátumot, míg a tört része az adott napon belüli időpontot.

7.2. Dátum- és időkezelő függvények

A két legegyszerűbb, de hasznos függvénnyel kezdjük: `Today()` magyarul: `Ma()` az éppen aktuális dátumot adja. A táblázat minden megnyitásakor az aktuális dátumot adja, paramétere nincs. `Now()` magyarul: `Most()` pedig képlet beírásakor éppen érvényes időpontot adja (a képletértékek frissítésekor változik az érték).

Amennyiben szeretnénk egy dátum vagy időérték valamelyik komponensét külön kinyerni, arra rendelkezésre állnak a Year(Év), Month(Hónap), Day(Nap), Hour(Óra), Minute(Perc) és Second(Másodperc) függvények. Mindegyik függvény egyetlen paramétert vár, amely vagy egy számérték, vagy egy cellahivatkozás lehet. A paraméterben levő dátum/idő értéknek a megfelelő komponensét adja eredményül.

Mikor lehet ezekre szükség? Például ha egy csoport tagjainak a születési dátuma alapján meg akarjuk határozni, egy adott évben milyen sorrendben következik a születésnapjuk. Ekkor a dátum-

ból a hónap és a nap komponensekre van szükségünk, az év komponensre nem. Külön oszlopba kiszámolva a hónap és nap komponenseket már lehet rendezni a névsort első szempontban a hónapot, második szempontban a napot megadva.

Más olyan esetek is lehetnek, amikor valamilyen képletben a dátumoknak vagy idő adatoknak csak egy-egy elemét kell felhasználni. Például ilyen lehet, amikor az időadatok közül a 10 és 11 óra közötti értékekkel kell csak valamit kezdeni. Ekkor nyilván az óra komponens értéke 10 lesz – feltételezve, hogy a 11 óra már nem szükséges.

A fordított irányra is szükség van, ugyanis képletbe csak számértékek írhatóak be, a dátum vagy az időadat közvetlenül nem. Ha nem akarjuk a kérdéses dátumot vagy időadatot valamelyik cellába beírni, hogy arra lehessen a képletben hivatkozni, akkor használhatjuk a Date(dátum), illetve a Time(idő) függvényt. Ezek a függvények három-három paramétert várnak:

- `date(év; hónap; nap)` Az év értéke 1583 és 9956 közötti érték lehet, mert ilyen dátumokat tud a program kezelni.
- `time(óra; perc; másodperc)`

Ha egy adott dátumon belül egy adott időpontot kell megadni, akkor a két függvénnyel előállított értékek összegét kell alkalmazni.

Amennyiben két dátum között eltelt napok, hónapok, évek számára vagyunk kíváncsiak, akkor lehetséges a két dátum különbségéből „*kézzel*” kiszámolni, de ez igazából csak napok esetén működik helyesen. Szerencsére létezik függvény, amellyel ez is elvégezhető anélkül, hogy foglalkozni kellene az egyes hónapok hosszával.

Ez a függvény a `DateDif(kezdő dátum; befejező dátum; intervallum)`. Magyar neve a függvénynek: `Dátumtólig()`

Az első két paraméter egy-egy dátumot reprezentáló szám (vagy azt tartalmazó cella). A harmadik cellába egy szöveget kell írni annak megadására, mit szeretnénk eredményül kapni:

- `"d"`: napokban adja meg a különbséget;
- `"m"`: a két dátum közti teljes hónapok számát adja;
- `"y"`: a két dátum közti teljes évek számát adja;
- `"ym"`: teljes hónapok száma a kezdő dátum és avégdátum különbségéből az éveket kivonva (azaz figyelmen kívül hagyva);
- `"md"`: teljes napok száma a két dátum különbségéből az éveket és hónapokat kivonva (azaz figyelmen kívül hagyva);
- `"yd"`: teljes napok száma a két dátum különbségéből az éveket figyelmen kívül hagyva.

Van olyan függvény, amely egy adott hónap napjainak a számát adja meg (**DaysInMonth()**) illetve egy adott évben (**DaysInYear()**). Ezek paraméterként egy dátumot várnak, amelyből az adott hónap/év napjainak számát adják meg. Utóbbi arra alkalmas, hogy egy dátumról eldönthető legyen, szökőév-e.

A *Húsvét* időpontja a telihold időpontjától függően évről évre máskor van – és ettől függően a *Pünkösd* is mozog. Egy adott évbeli időpontját adja meg az `EasterSunday(évszám)` függvény, amelyből a többi tőle függő dátumot már a megfelelő egész szám hozzáadásával lehet kiszámolni. A függvénynek nincs magyar neve.

8. Százalékszámítás

Sokszor fordul elő olyan feladat, amikor egyes mennyiségeknek az egészhez viszonyított százalékos arányát kell meghatározni. Ugyancsak gyakori – főleg árakkal kapcsolatos táblázatoknál – hogy két mennyiség egymáshoz való viszonya százalékban szerepel. Emiatt a táblázatkezelő programok kiemelt lehetőséget adnak a százalékokkal való számolásra.

A matematikából ismert százalékszámítási módszerek a táblázatkezelés során egy kicsit módosítva használatosak. A százalék itt valójában egy műveleti jel, amely az előtte szereplő számérték századrészét eredményezi (azaz százzal történő osztás történik). Hasonló módon, van lehetőség egy számérték százalékként való megjelenítésére, amely nem más, mint a számérték százszorosának és utána a százalékjelnek a megjelenítés.

Fontos hangsúlyozni, hogy a százalékjelet nem mértékegységként használjuk. Erre utal az is, hogy a magyar nyelvben a mérőszámok és a mértékegységük között mindig kell egy – nem törhető – szóközt hagyni, míg a százalékjelet egybe írjuk a számmal.

A fentiek miatt a táblázatkezelőben nem kell a képletekben százzal osztani vagy szorozni, hanem egyszerűen a százalékjel karaktert (%) kell a megfelelő helyen elhelyezni, illetve a cella formátumát kell százalékértékre állítani. Egy cellába történő adatbevitel során – megint a mértékegységekkel ellentétben – begépelhető a szám után a százalékjel. Ebben az esetben a program automatikusan százalékérték típusúra állíthatja[11] a cella formátumát.

8.1. Konkrét példák százalékszámításra

Vegyik első példának az Élelmiszerpiac feladatot, amelyben különböző élelmiszerek mennyisége, egységára, valamint az azok eladására érvényes ÁFA értékek vannak megadva és ki kell számolni a fizetendő bruttó értéket. (lásd: 46. ábra)

	A	B	C	D	E
1	Név	Mennyiség	Egységár	ÁFA	Bruttó
2	Alma	25 kg	95 Ft/kg	25%	
3	Körte	12 kg	107 Ft/kg	25%	
4	Szőlő	16 kg	88 Ft/kg	25%	
5	Barack	8 kg	158 Ft/kg	25%	
6	Krumpli	35 kg	28 Ft/kg	12%	
7					
8	Összesen				
9					

46. ábra: Az Élelmiszerpiac táblázatának kezdeti tartalma

1. megjegyzés: A táblázat cellában a **B** és **C** oszlopban csak számértékek találhatóak, míg a **D** oszlopba a százalékjellel együtt lett a szám begépelve. Emiatt a **D** oszlop értékei 0,25 és 0,12 lett.

2. megjegyzés: Az árak és a jövedelmek esetében is beszélni szoktunk *bruttó* és *nettó* értékről. A *nettó* mindig az az érték, amit megkapunk (akár a jövedelemből, akár az árból eladóként), míg a *bruttó* érték tartalmazza az adót is. Ebből következően a *bruttó érték* mindig magasabb a *nettó értéknél*.

11 *Amennyiben a cella még alapértelmezett formátumú, akkor mindig meg is teszi. De ha már kapott valamilyen formátumot, akkor nagy eséllyel marad a már beállított formátum.*

3. megjegyzés: a példában szereplő ÁFA értékek nem feltétlenül felelnek meg a valóságban aktuálisan alkalmazott adótartalomnak!

A **B** és **C** oszlop értékeinek szorzata nyilván a nettó árat adja meg, amelyhez még hozzá jön az ÁFA, amely a nettó ár százalékában van jogszabályban meghatározva. Ezzel tehát meg kell növelni a nettó árat.

A szorzás egyszerűen elvégezhető a `=B2*C2` képlettel – a második sor esetén, amely gond nélkül másolható lenne. Ezt az értéket azonban még meg kell növelni az ÁFA-val. Vigyázat: nem az ÁFA százalékértékével, hanem az ÁFA összegével.

Az ÁFA értéke viszont a második sor esetén a `=B2*C2*D2` képlettel lenne kiszámolható, hiszen a **D2** cella a százalékot tartalmazza. Ezt kell az eredeti nettó értékhez hozzáadni.

Azaz a képletünk általánosságban így nézne ki: *bruttó érték* = *nettó érték* + *nettó érték* * *adó százalékértéke*. Ugyanez rövidebben a matematikai szabályok alkalmazásával így is felírható: *bruttó érték* = *nettó érték* * (1+*adó százalékértéke*). Azaz a mi esetünkben az E2 cellába írandó képlet: `=B2*C2*(1+D2)`.

Hasonlóan, ha százalékos kedvezményt adunk az árból, akkor a zárójelben az összeadás helyett kivonást alkalmazva történhet meg a százalékkal csökkentése az eredeti értéknek.

Másik eset, amikor azt kell meghatározni, hogy egy adott érték hány százalékát teszi ki az egésznek. Ebben az esetben el kell osztani a kérdéses értéke az egész értékével. Az így kapott értéknek a $0 \leqslant x \leqslant 1$ intervallumba kell esnie. Ha nem így történik, akkor valamit rosszul csináltunk! A kapott értéket tartalmazó cellát százalék formátumúvá kell formázni. Erre a formázó eszköztáron a 47. ábrán látható gombbal, vagy a Formátum → Cellák párbeszédablak segítségével lehet megtenni.

A normál számformátumhoz és a pénznem formátumhoz hasonló beállításokra a százalékok esetén is lehetőség van: a megjelenő tizedesjegyek száma megadható, illetve ezres csoportosítás is lehetséges lenne, de ez utóbbinak száznál nem nagyobb értékek esetén nincs jelentősége.

47. ábra

9. Logikai kifejezések

Vannak esetek, amikor valamilyen feltétel teljesülése, illetve nem teljesülése esetén más-más módon kell valamit kiszámolni. Ilyen esetekben van szükség a *logikai függvények*re, amelyekben logikai eredményt adó kifejezések alapján történik döntés a további teendőkről. Ezek a logikai eredményt adó kifejezések vagy röviden *logikai kifejezések* kétféle eredményt adhatnak: *igaz*, *hamis*.

Bármilyen *összehasonlítás*t tartalmazó kifejezés logikai kifejezés. Például ha egy táblázatban testmagasságok szerepelnek (lásd 48. ábra) a **B** oszlopban, és a legalább 180 cm magasságú személyek esetén kell valamit tenni, míg az ennél alacsonyabbaknál valami mást, akkor a két eset szétválasztásához a **B2** cella esetében a `B2<180` kifejezést vagy a `B2>=180` kifejezést lehet használni (feltételezzük, hogy a testmagasságh cm-ben van megadva a **B2** cellában).

	A	B	C
1	**Név**	**Testmagasság**	**Kosárlabdához**
2	Alacsony Géza	120 cm	
3	Horváth Gábor	175 cm	
4	Almásy József	205 cm	
5	Barna Olivér	181 cm	
6	Joó Péter	192 cm	
7			
8			

48. ábra

Az előbbi kifejezés akkor lesz igaz, ha alacsonyabb az illető 180 cm-nél, míg a másik az ellenkező esetben.

A C oszlopban kellene megjeleníteni az „alkalmas” vagy a „nem alkalmas” szöveget attól függően, hogy az illető eléri-e a 180 cm magasságot.

Ennek megoldásához tehát a két fent említett logikai kifejezés valamelyikét kell alkalmaznunk. Ha például az elsőt választjuk, akkor a C2 cellában a következő képletet kell elhelyezni:

```
=IF(B2<180;"nem alkalmas";"alkalmas")
```

Az If() az egyik logikai függvény.[12] Ezt a logikai függvényt használjuk a leggyakrabban, mivel a többi logikai függvénynek is csak ezzel együtt van értelme. A függvény három paraméterrel rendelkezik és mindhárom paraméter használata kötelező. Látszólag ugyan működik két paraméterrel is, de ebben az esetben gyakorlatilag hibás működést produkál.

Az első paraméter a logikai kifejezés helye, amelynek értéke alapján dől el a további két paraméter sorsa. Ezt a feltételt *feltétel*nek hívjuk.

	A	B	C
1	**Név**	**Testmagasság**	**Kosárlabdához**
2	Alacsony Géza	120 cm	nem alkalmas
3	Horváth Gábor	175 cm	nem alkalmas
4	Almásy József	205 cm	alkalmas
5	Barna Olivér	181 cm	alkalmas
6	Joó Péter	192 cm	alkalmas

49. ábra

A második paraméterben és a harmadik paraméterben is egy tetszőleges kifejezés szerepelhet – még azonos típusú értéket sem kell adniuk. A mi esetünkben egy-egy szöveg szerepel bennük, ezért vannak idézőjelbe téve: pontosan az idézőjelben levő szöveget adja a kifejezés.

A második paraméterben szereplő kifejezés értéke lesz a függvény értéke, ha a feltétel, azaz az első paraméterben megadott logikai kifejezés értéke *igaz*. A harmadik paraméterben szereplő kifejezés értéke lesz a függvény értéke abban az esetben, ha a feltétel az első paraméterben *hamis*.

12 *A függvény magyar neve: Ha().*

Ez az egyszerű függvény ilyen módon bármilyen feltétel alapján választási lehetőséget biztosít két lehetőség között. Azonban vajon mi a teendő akkor, ha több feltétel is van?

Az egyik lehetőség az If() függvények egymásba ágyazása, amelyre a későbbiekben fogunk példát nézni.

A másik lehetőség akkor hasznos, ha a feltételeknek egyszerre kell teljesülniük, illetve akkor ha a feltételek közül legalább az egyiknek kell teljesülnie. Mindkét esetben a feltételeket valahogy össze kell kapcsolni egyetlen feltétellé. Erre használhatóak a további logikai függvények, amelyeket az alábbiakban foglalunk össze.

9.1. Bool-i logika használata táblázatkezelőben

Van két olyan logikai függvény, amelyek valójában konstans értéket adnak: `False()` = `Hamis` és `True()` = `Igaz`. Ezeket akár függvényként (ekkor képletben kell megadni), akár az egyenlőségek baloldalán látható szövegként – idézőjelek nélkül – megadhatjuk.

Egy logikai értéket tagadni lehet a **Not()** függvénnyel – magyarul: **Nem()** –, amelynek egyetlen paramétere az a kifejezés, amelyet tagadni kell.

Amennyiben két vagy több logikai értéknek egyszerre kell igaznak lennie, akkor azokat az **And()** függvény paramétereiként kell felsorolni (természetesen pontosvesszővel elválasztva). A függvény magyar neve: **És()**. Ez a függvény kizárólag akkor ad *Igaz* értéket, ha minden paraméterében *Igaz* értékű kifejezés szerepel.

Amennyiben a több feltétel valamelyikének elég teljesülnie, akkor használható az **Or()** vagy magyarul **Vagy()** függvény, amely minden olyan esetben *Igaz* értéket ad, ha legalább az egyik paraméterében szereplő logikai kifejezés értéke *Igaz*.

A magyar nyelvben a „vagy” szót általában nem a fenti értelemben, hanem „kizáró vagy” értelemben használjuk, azaz amikor pontosan egy feltétel teljesül. Erre szolgált eredetileg a **Xor()** függvény, ám mivel ennek is lehet kettőnél több paramétere, így a definíciója kibővült: akkor ad igaz értéket, ha a paramétereiben levő kifejezések közül páratlan számú ad vissza *igaz* értéket. Ennek a függvénynek nincs magyar neve.

Az alábbi táblázat mutatja a logikai műveletek igazságtábláját. Az első két oszlop a két paraméter (a **Not()** csak az elsőt használja).

A	B	Not(A)	Or(A;B)	And(A;B)	Xor(A;B)
HAMIS	HAMIS	IGAZ	HAMIS	HAMIS	HAMIS
HAMIS	IGAZ	IGAZ	IGAZ	HAMIS	IGAZ
IGAZ	HAMIS	HAMIS	IGAZ	HAMIS	IGAZ
IGAZ	IGAZ	HAMIS	IGAZ	IGAZ	HAMIS

50. ábra: A logikai függvények igazságtáblázata

9.2. Relációk

A logikai kifejezések bármelyike, amely az **If()** első paraméterébe kerülhet, állhat a fenti függvények megfelelő kombinálásából is. Ám ahhoz, hogy valódi logikai kifejezést kapjunk, ahhoz két értéket össze kell hasonlítani. Ehhez kellenek relációs jelek:

- =: a két oldalán levő kifejezés értéke egyenlő. Valós számokra használni veszélyes, mert a számábrázolási pontatlanság miatt nem garantálható mindig az egyenlőség!

- <: A baloldali kifezés kisebb értékű, mint a baloldali. Számoknál ez nem okoz gondot, szöveg esetén a „kisebb” azt jelenti, hogy korábban szerepel az abc-rendben; dátum és időpont esetén pedig a régebbi a kisebb.
- >: Az előbbi fordítottja: a jobboldali kifejezés értéke kisebb.
- <=: Az a reláció tulajdonképpen annak a rövidítése, mintha az `A<=B` helyett azt írnánk, hogy `OR(A<B; A=B)`, vagyis a baloldali kifejezés kisebb a jobboldalinál, vagy a két kifejezés értéke egyenlő. Fontos, hogy ez egy szimbólum, azaz a két karakter nem felcserélhető!
- >=: A jobboldali kifejezés késebb vagy egyenlő, mint a baloldali. Az `A>=B` reláció megfelel az `OR(A>B; A=B)` kifejezésnek. A két karakter ebben az esetben sem felcserélhető!
- <>: Nem egyenlő a két oldalon szereplő kifejezés. A két karakter ebben az esetben sem cserélhető fel!

Minden reláció esetén igaz, hogy az összehasonlítás csak akkor lehetséges, ha a két kifejezés típusa összehasonlítható. Több reláció nem láncolható össze, tehát a `B1 < C1 < D1` nem szabályos kifejezés!

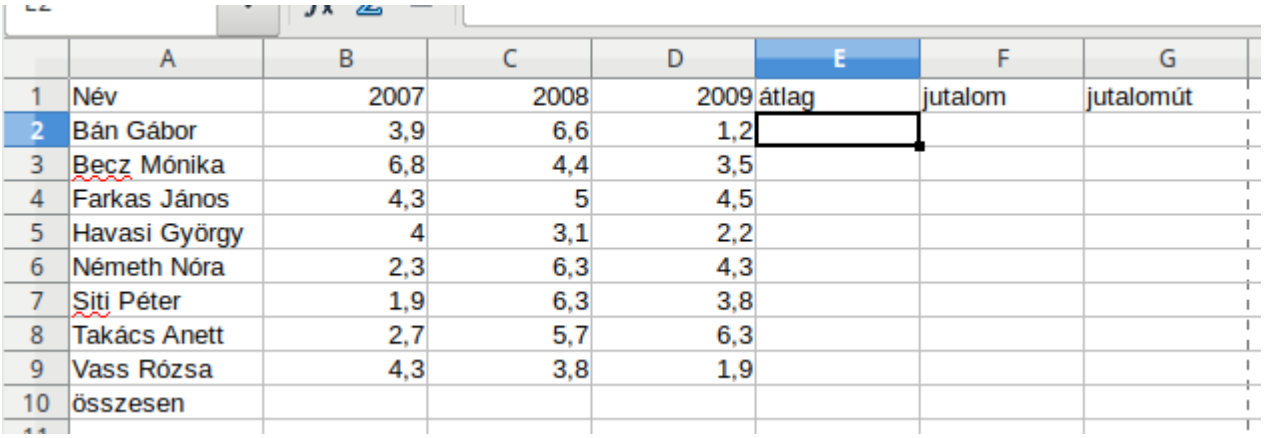
9.3. Példá az If() függvény használatára

A következőkben egy összetett feladat megoldása következik, amely az **If()** függvény használatát igényli. A feladat órán megoldandó, amihez a `public/tablazatkezeles/logikai_kifejezesek/teljesitmeny/teljesitmeny.ods` forrásállomány használható.

Név	2007	2008	2009	átlag	jutalom	jutalomút
Bán Gábor	3,90	6,60	1,20	3,90	nem jár	nem kap
Becz Mónika	6,80	4,40	3,50	4,90	jár	kap
Farkas János	4,30	5,00	4,50	4,60	jár	kap
Havasi György	4,00	3,10	2,20	3,10	nem jár	nem kap
Németh Nóra	2,30	6,30	4,30	4,30	jár	nem kap
Siti Péter	1,90	6,30	3,80	4,00	jár	nem kap
Takács Anett	2,70	5,70	6,30	4,90	jár	nem kap
Vass Rózsa	4,30	3,80	1,90	3,33	nem jár	nem kap
összesen	30,20	41,20	27,70			

jutalmat kap	5 fő
jutalomutat kap	2 fő

51. ábra: Az elkészítendő táblázat (az utolsó oszlop nélkül)



	A	B	C	D	E	F	G
1	Név	2007	2008	2009	átlag	jutalom	jutalomút
2	Bán Gábor	3,9	6,6	1,2			
3	Becz Mónika	6,8	4,4	3,5			
4	Farkas János	4,3	5	4,5			
5	Havasi György	4	3,1	2,2			
6	Németh Nóra	2,3	6,3	4,3			
7	Siti Péter	1,9	6,3	3,8			
8	Takács Anett	2,7	5,7	6,3			
9	Vass Rózsa	4,3	3,8	1,9			
10	összesen						

52. ábra: Az állomány eredeti tartalma

Az állomány megnyitás utáni állapota a 52. ábrán látható. A táblázatban egy cég dolgozóinak éves eladásai láthatóak millió forintban megadva. A megoldáshoz a következő információk állnak rendelkezésre:

1. Névsorba kell rendezni a táblázatot.
2. Jutalom jár, ha az átlagos évi eladás legalább négymillió forint.
3. Jutalomút jár, ha minden év forgalma hárommillió forint felett van.
4. A jutalomút összege a milliókban mért átlagnak az 50 000-szerese.
5. A táblázat alatt a jutalmat és a jutalomutat kapó személyek számát is képlettel kell meghatározni!

Első lépésként célszerű a cellák megjelenését, igazítását beállítani, majd elvégezni a rendezést most, amikor még kevés oszlopban vannak adatok.

Ezek elvégzése a korábban tanultak alapján nem okozhat gondot... Ezt követően az átlag oszlop és az összesen sor kiszámolása következhet, amely után a 53. ábrán látható táblázatot kell kapni.

A jutalom és a jutalomút oszlopok meghatározása egyaránt egy-egy feltételes képletet igényel. A jutalom akkor jár a feladat szövege szerint, ha az átlagos eladás (**E3** cella) legalább négymillió forint. Mivel a cellákban az értékek millió forintban vannak, a szükséges képlet így néz ki:

```
=If(E3>=4;"jár";"nem jár")
```

Természetesen azért kell az idézőjel, mert szöveget akarunk megjeleníteni.

Hasonlóan fog kinézni a jutalomút meghatározásához tartozó képlet is. Azonban ebben az esetben a feltételt kicsit összetettebben kell felépíteni, mivel azt kell vizsgálni, hogy vajon volt-e olyan

	A	B	C	D	E	F	G
1	Név	2007	2008	2009	átlag	jutalom	jutalomút
2	Bán Gábor	3,90	6,60	1,20	3,90		
3	Becz Mónika	6,80	4,40	3,50	4,90		
4	Farkas János	4,30	5,00	4,50	4,60		
5	Havasi György	4,00	3,10	2,20	3,10		
6	Németh Nóra	2,30	6,30	4,30	4,30		
7	Siti Péter	1,90	6,30	3,80	4,00		
8	Takács Anett	2,70	5,70	6,30	4,90		
9	Vass Rózsa	4,30	3,80	1,90	3,33		
10	összesen	30,20	41,20	27,70			
11							

53. ábra

	A	B	C	D	E	F	G	H
1	Név	2007	2008	2009	átlag	jutalom	jutalomút	jutalomút összege
2	Bán Gábor	3,90	6,60	1,20	3,90	nem jár	nem kap	
3	Becz Mónika	6,80	4,40	3,50	4,90	jár	kap	
4	Farkas János	4,30	5,00	4,50	4,60	jár	kap	
5	Havasi György	4,00	3,10	2,20	3,10	nem jár	nem kap	
6	Németh Nóra	2,30	6,30	4,30	4,30	jár	nem kap	
7	Siti Péter	1,90	6,30	3,80	4,00	jár	nem kap	
8	Takács Anett	2,70	5,70	6,30	4,90	jár	nem kap	
9	Vass Rózsa	4,30	3,80	1,90	3,33	nem jár	nem kap	
10	összesen	30,20	41,20	27,70				
11								
12				jutalmat kap				
13			jutalomutat kap					
14								

54. ábra

év, amikor az eladás mennyisége nem érte el a hárommilliót. Ha ilyen nem volt, akkor jár a jutalomút. Ezt meg lehetne úgy is oldani, hogy mindhárom év értékét összehasonlítjuk a 3 értékkel, és ha mindháromra igaz, hogy nagyobb, akkor jó: `And(B3>3;C3>3;D3>3)`. Azonban ennél van egy egyszerűbb megoldás is: ha a legkisebb érték nagyobb hárommilió forintnál, akkor nyilván a nála nagyobbak is nagyobbak lesznek. Tehát meg kell határozni a legkisebb forgalom értékét dolgozónként a **Min()** függvény használatával, és ezt az értéket kell csak vizsgálni. Ennek megfelelően a képlet a **G3** cellában a következő lesz:

```
=If(Min(B3:D3)>3;"kap";"nem kap")
```

Jelenleg a táblázatunknak a 54. ábrának megfelelően kell kinéznie – természetesen néhány további cellával kiegészítve.

A jutalomút összege egyszerű lenne, ha mindenkinél ugyanazt kellene alkalmazni: meg kell szorozni 50 ezerrel az átlagot. Azonban üresnek kellene maradnia a cellának, ha az illető nem kap jutalomutat. Itt tehát ismét belép egy feltételviszgálat. Most látszólag elég lenne az If() függvény első két paramétere. Azonban ebben az esetben nem üres maradna a cella, ha nincs jutalomút, hanem a „HAMIS” felirat jelenne meg benne. Ezt elkerülendő tehát ilyen esetben is kell a harmadik paraméter, azonban ennek tartalma most egy üres szöveg lesz, amit két egymást követő idézőjellel lehet megadni.

A feltétel most egyszerűen a G3 cellában levő szöveget vizsgálhatja:

```
=If(G3="kap";E3;50000;"")
```

Végül még két kiszámolandó adatunk van, mielőtt a formázást be lehetne fejezni: meg kell számolni, hogy ki kapott jutalmat és ki jutalomutat. Mind a két esetben egy bizonyos feltételnek megfelelő cellákat kell összeszámolni, amire a CountIf() függvény alkalmas, amely az első paraméterben a megszámolandó tartományt, a másodikban pedig a feltételt várja. Tulajdonképpen itt nem az eddig értelemben vett feltételről van szó, mivel ha egy értéket írunk a második paraméterbe, akkor azt az értéket keresi a függvény.

A képlet, amely megszámolja a jutalmat kapókat:

```
=CountIf(F3:F10;"jár")
```

A jutalomutat kapókat megszámoló képlet:

```
=CountIf(G3:G10;"kap")
```

	A	B	C	D	E	F	G	H
1	Név	2007	2008	2009	átlag	jutalom	jutalomút	jutalomút összege
2	Bán Gábor	3,90	6,60	1,20	3,90	nem jár	nem kap	
3	Becz Mónika	6,80	4,40	3,50	4,90	jár	kap	245000
4	Farkas János	4,30	5,00	4,50	4,60	jár	kap	230000
5	Havasi György	4,00	3,10	2,20	3,10	nem jár	nem kap	
6	Németh Nóra	2,30	6,30	4,30	4,30	jár	nem kap	
7	Siti Péter	1,90	6,30	3,80	4,00	jár	nem kap	
8	Takács Anett	2,70	5,70	6,30	4,90	jár	nem kap	
9	Vass Rózsa	4,30	3,80	1,90	3,33	nem jár	nem kap	
10	összesen	30,20	41,20	27,70				
11								
12				jutalmat kap	5 fő			
13				jutalomutat kap	2 fő			
14								

55. ábra: A végeredmény

Természetesn a mértékegységet itt is formázással kell majd beállítani…

A formázás befejezése után megkapjuk a feladat leírásánál megadott mindának megfelelő képet, amely a 55. ábrán látható.

10. Keresőfüggvények

Amennyiben egy táblázat valamely oszlopában levő értéket megkeresve a táblázat ugyanazon sorának egy másik oszlopában levő értéket kell megadni, arra az eddig megismert If() függvény ugyan ad megoldást, ám ez annyi egymásba ágyazott If() függvényt igényel, amennyi sora a táblázatnak van. Pontosabban egyel kevesebbet ennél.

Például a Magyarországon használt ötfokozatú osztályozás esetén egy dolgozaton elért pontszámról megállapítani az azért járó osztályzatot, egy ötsoros táblázat megfelelő sorának kiválasztását igényli. Ennek első oszlopában lehet elhelyezni az adott osztályzathoz tartozó alsó ponthatárt, a második oszlopban a felső ponthatárt, a harmadikban pedig az osztályzat megnevezését (lásd 56. ábra).

0	65	elégtelen	50%
66	85	elégséges	65%
86	98	közepes	75%
99	117	jó	90%
118	130	jeles	100%
		maximum:	130

56. ábra: Egy tipikus ponthatár táblázat

Ebben az esetben a megfelelő osztályzatot úgy lehet megtalálni, hogy felülről lefelé haladva elkezdjük megvizsgálni a táblázat első oszlopát a második sortól kezdve. Ha az ott levő érték – a következő osztályzathoz szükséges minimális pontszám – nagyobb az elért pontszámnál (a keresett értéknél), akkor az előző sor vonatkozik arra a pontszámra.

Ez feltételezve, hogy az ábrán látható táblázat a **F4:I8** tartományban található, és az elért pontszám a **A6** cellában van, a következő algoritmust jelentené:

```
Ha A6<F5
akkor H4
különben ha A6<F6
         akkor H5
         különben ha A6<F7
                  akkor H6
                  különben ha A6<F8
                           akkor H7
                           különben H8
```

A fenti algoritmusnak megfelelő képlet a következőképpen néz ki táblázatkezelőben:

```
=If(A6<F5;H4;If(A6<F6;H5;If(A6<F7;H6;If(A6<F8;H7;H8))))
```

Amellett, hogy a fenti képlet borzasztóan hosszú és nehezen áttekinthető, a cellák másolás elleni védelme még nem is szerepel benne. Ilyen hosszú képletnél könnyű gépelési hibát is véteni, amit utána nagyon nehéz megtalálni. Ráadásul ugyanaz a cellahivatkozás (**A6**) többször is szerepel benne, hiszen minden alkalommal össze kell hasonlítani a következő sorbeli értékkel. És mindez csupán egy 5 soros táblázat, de hasonló műveletre akár több száz soros táblázatoknál is szükség lehet!

A programozásban ilyen bonyolult műveletekre szokás függvényeket definiálni, amelyek a szükséges adatokat csak egyszer kapják meg és annyiszor használják fel, amennyiszer szükséges. Az Excelben definiált, majd a LibreOffice Calcban is megvalósított függvények valójában ugyanilyen függvények. Így nem kell csodálkozni rajta, hogy a fenti feladatra specializálódott függvénnyel is rendelkeznek a táblázatkezelő programok. Ezeket nevezzük **keresőfüggvények**nek, hiszen egy adott érték megkeresését teszik lehetővé nagyobb táblázatrészen belül is egyszerű paraméterezéssel.

A függvény, amellyel a fenti keresés elvégezhető angolul a **Vlookup()**, magyarul pedig az **Fkeres()** nevet kapta. Három, vagy négy paraméteres változatban használható.

10.1. Az osztályzatkeresési feladat megoldása

A fenti példában szerepelő hosszú, négy If() függvényt igénylő megoldás helyett egyszerűen az alábbit lehet írni:

```
=Vlookup(A6;F4:I8;3)
```

A fenti függvény pontosan ugyanazt az algoritmust hajtja végre, mint az If() függvények. A különbség az, hogy minden paramétert egyszer kell csak megadni és a szükséges algoritmust ez alapján a függvény végrehajtja.

A fenti képlet paraméterei a következőt jelentik:

1. Az **A6** cella tartalmazza azt az értéket, amelyet meg kell keresni a táblázatban.
2. A táblázat az **F4:I8** tartományban található.
3. A tartomány 3. oszlopában levő értéket kell visszaadnia a függvénynek abból a sorból, amely az **A6** cellának megfelelő intervallumot adja meg.
4. Mivel a negyedik paraméter nem szerepel, így az **F4:I8** tartomány első oszlopa egy növekvő intervallumokból álló lista alsó értékeit tartalmazza.

10.2. A keresőfüggvények

A leggyakrabban használt keresőfüggvény szerepelt a fentiekben, de még van több is. Ezek a *Függvénytündér* „*Munkafüzet*” kategóriájában találhatóak meg. Íme a legfontosabbak:

- `VLookup`(*keresési érték*; *tábla*; *oszlopsorszám*; *tartományban keresés*): A függvény megkeresi az első paraméterében szereplő értéket a második paraméterben megadott *tábla* első oszlopában. A keresés eredménye kiválasztja valamelyik sort, majd ennek a sornak a harmadik paraméterben megadott sorszámú oszlopában levő cella értékét adja vissza a függvény. Amennyiben a negyedik paraméter nem szerepel – vagy szerepel, de *IGAZ* értékkel –, akkor a *tábla* első oszlopát mint növekvő intervallumok alsó határait kezeli a függvény. Amennyiben a negyedik paraméter *HAMIS* értékkel van megadva, akkor viszont a függvény csak a *tábla* első oszlopában ténylegesen szereplő értékeket találja meg, amelyek ebben az esetben bármilyen sorrendben jelen lehetnek. Ha az első paraméterben olyan érték szerepel, amely nem található meg a keresési tartomány (tábla paraméter) első oszlopában, akkor a függvény a „`#HIÁNYZIK`” hibajelzést adja.
- `HLookup`(*keresési érték*; *tábla*; *sorszám*; *tartományban keresés*): A VLookup() függvény megfelelője sorokban történő keresésre: egyszerűen meg kell cserélni a sorok és az oszlopok szerepét a fenti definícióban. Azaz a függvény első paraméterében megadott értéket a *tábla* első sorában keresi balról jobbra haladva. Amelyik oszlopban megtalálja, annak az oszlopnak a harmadik paraméterben megadott sorszámú sorában levő értéket adja eredményül. A negyedik paraméter jelentése azonos a VLookup() függvény negyedik paraméterével.
- `Lookup`(*keresési érték*; *keresési vektor*; *eredményvektor*): Az első paramétert megkeresi a második paraméterben megadott egy soros vagy egy oszlopos tartományban. A függvény értéke az így kapott sorszámú elem lesz a harmadik paraméterben szereplő, szintén egy soros vagy egy oszlopos tartományon belül.
- `Match`(*keresési érték*; *keresési tartomány*; *rendezettség típusa*): A függvény a VLookup() és a HLookup() függvény általánosítására alkalmas: Visszaadja a sorszámát a keresési tartomány azon cellájának, amely megfelel az első paraméterben szereplő értéknek. A harmadik paraméter meghatározza, hogy hogyan kell a második paraméterben szereplő egy soros vagy egy oszlopos tartomány rendezettségét figyelembe venni:

- Ha az érték 1, akkor a tartomány növekvő rendezésű intervallumokat ad meg. Ekkor a VLookup() és a HLookup() függvény háromparaméteres változatával azonos módon történik a keresés.
- Ha az érték 0, akkor a tartomány nincs rendezve, csak pontos egyezés lehetséges. Ekkor a keresés módja olyan, mint a VLookup() és HLookup() esetén a negyedik paraméter *HAMIS* értéke esetén.
- Ha az érték -1, akkor a tartomány csökkenő rendezésű intervallumokat reprezentál. Ekkor a keresés a növekvő rendezéshez hasonlóan viselkedik, de ellenkező irányba haladva.

- `Index`(*tartomány*; *sorszám*; *oszlopszám*, *tartományszám*): Az első paraméterében megadott tartománynak a második és a harmadik paraméterben megadott koordinátákon levő cellájának értékét adja eredményül. A negyedik (opcionális) paraméter akkor használható, ha az első paraméterben egy több részből álló tartomány *neve* szerepel – ilyen tartományt csak előre definiált névvel lehet megadni – használandó tartományként. Ebben az esetben ez a negyedik paraméter annak a résztartománynak a sorszámát adja, amelyben a cellát a második és a harmadik paraméter meghatározza. A függvényt a Match() függvénnyel együtt használva lehet olyan érték-keresést végezni, amelyre a VLookup() és a HLookup() függvények a korlátaik miatt nem alkalmasak.

10.2.1. Keresés intervallumokban

Mind a VLookup(), mind a HLookup() kétféle módon használható:[13] intervallumban keresésre és diszkrét értékek keresésére.

Amennyiben egy értékről el kell dönteni, hogy melyik intervallumba esik és egy ettől függő értéket kell a továbbiakban használni, akkor használható a két keresőfüggvény valamelyike három paraméteres változatban.

A VLookup() mindig egy téglalap alakú tartomány első oszlopában, a HLookup() egy téglalap alakú tartomány első sorában keresi a keresési értékét fentről lefelé, illetve balról jobbra haladva. Mindkét függvény úgy tekinti az itt található értékeket, mint a lehetséges értékeket különböző intervallumokba osztó értékeket, és ennek megfelelően a keresett értéknél még éppen nem nagyobb értéket keresik. Ez az érték határozza meg, hogy a keresett érték melyik intervallumba esik.

Ha a keresett intervallum megvan, az ezt reprezentáló sor/oszlop megfelelő celláját a függvény harmadik paraméterében levő sorszám határozza meg: ennek mindig egy pozitív egész számnak kell lennie, amely nem lehet nagyobb a második paraméterben megadott tartomány oszlopainak/sorainak számánál. A paramétert a visszaadandó érték oszlopának/sorának meghatározására használja a függvény.

A függvények használatára fentebb lehetett példát látni. Ebben az esetben ha például a 89 a keresett érték, akkor a VLookup() függvény megállapítja, hogy ez a középső intervallumba esik, hiszen az a [86...98] intervallum. Ezért a keresett értékhez tartozó sor: (86; 98; „közepes”; 75%). A harmadik paraméterben a 3 érték szerepel, vagyis a harmadik oszlop értékét kell a függvénynek eredményül adnia, amely ebben az esetben a „közepes” szöveg lesz.

13 *Ez a két függvény egymás testvére: pontosan ugyanazt csinálja mindkettő, csak egymáshoz képest a sorok és az oszlopok szerepe fel van cserélve. A nevük a keresés irányára utal: H=Horizontal (Vízszintes), V=Vertical (Függőleges)*

10.2.2. Keresés diszkrét értékekre

Amennyiben olyan táblázatban kell keresni, amelynek első oszlopa/sora nincs rendezve, akkor csak az ebben az oszlopban/sorban szereplő értékeket lehet a Vlookup()/HLookup() függvényekkel megkerestetni. Ekkor minden ugyanúgy működik, mint az intervallumban keresésnél, kivéve, hogy a függvény negyedik paraméterébe a *HAMIS* értéket vagy az ennek megfelelő `0` értéket is meg kell adni.

Ekkor a függvény fentről lefelé, illetve balról jobbra haladva megkeresi a keresett érték legelső előfordulását és az lesz a találati sor/oszlop, amelynek a harmadik paraméterben megadott sorszámú cellájának értékét fogja eredményül adni. Amennyiben az érték keresése során az oszlop/sor utolsó elemére ér és még mindig nincs meg a keresett érték, akkor a függvény a „*#HIÁNYZIK*” hibajelzést fogja eredményezni.

10.2.3. A VLookup() és HLookup() függvények hiányosságai

Mint a fenti leírásokból is látható, a VLookup() és a HLookup() függvények mindig a második paraméterben megadott tartomány első oszlopán/során végzik a keresést és vagy az itteni értéket (ha a harmadik paraméter értéke 1) vagy egy ettől jobbra/lefele levő értéket adnak eredményül. A VLookup() képtelen a kereséshez használt oszloptól balra levő oszlopban levő értéket visszaadni, ahogyan a HLookup() is képtelen a kereséshez használt sor feletti sorban levő értéket.

Az ilyen jellegű probléma aránylag egyszerűen megoldható: az eredményül várt oszlop/sor celláinak értékét másoljuk át a kereséshez használt oszlop/sor jobb oldalára, illetve alá és ezt a másolatot használjuk a keresés során. Ilyenkor célszerű a másolást nem a vágólapot használva, hanem képlettel megoldva elvégezni, hogy az értékek változását a másolat is követhesse!

A másik probléma abból adódik, hogy a negyedik paraméter csak annyit enged megadni, hogy növekvő rendezés vagy rendezetlen értékek között kell keresni. A csökkenő rendezés használatát gyakorlatilag lehetetlen az értékek másolásával megoldani.

Ezt a problémát már csak a Match() és Index() függvények használatával lehet megoldani:

1. Először keressük meg a Match() függvény segítségével, hogy a keresett érték a keresési vektor (egy soros vagy egy oszlopos tartomány) hányadik eleméhez tartozik. Ennek a függvénynek a harmadik paraméterébe -1 értéket írva csökkenő rendezésen is végezhetünk keresést.
2. Mivel a Match() függvény csupán a keresett értéknek megfelelő cella sorszámát adja, a másik helyen levő megfelelő érték meghatározására az Index() függvény használható. Az ilyen használat során az első paraméter a megfelelő egy soros vagy egy oszlopos tartomány lesz.
3. Ha egy soros tartomány, akkor a második paraméter mindig egy. Ha egy oszlopos tartomány, akkor a harmadik paraméter lesz mindig 1. Negyedik paraméter a függvény ilyen felhasználásakor nincs.
4. A Match() függvény által visszaadott értéket – vagy az egész Match() függvényt egy az egyben – a maradék paraméterbe kell helyezni: egy soros tartomány esetén a harmadik, egy oszlopos tartomány esetén a második paraméterbe.

Az Index-Match függvénypár nem csak a fenti két problémára ad kicsit bonyolult, de működőképes megoldást: ha a keresett értéket egy oszlopban kell megkeresni és az eredmény-értéket egy sorból kiválasztani (itt is felcserélhető a „*sor*” és az „*oszlop*”), akkor is használható.

10.3. Használati tanácsok

Mivel bár ezek a függvények nagyon hasznosak a táblázatkezelőben, elég nehéz megérteni a működésüket, íme néhány tanács a használatukhoz:

- Ha folyamatosan növekvő értékhatárokkal elválasztott tartományok közül kell választani, akkor a ?lookup() függvény használható, de a folyamatosan csökkenő értékhatárok esetén nem.
- Ha az értékhatárok oszlopban helyezkednek el, akkor a vertikális keresésre van szükség → VLookup().
- Ha az értékhatárok sorokban helyezkednek el, akkor a horizontális keresésre van szükség → HLookup().
- Ha a várt érték az értékhatárok oszlopától balra van, akkor egy üres jobboldali oszlopba képlettel átjuttatva az értékeket, a VLookup() használhatóvá válik. Átmásolás nélkül csak az Index-Match párossal oldható meg a feladat.
- Ugyanígy tehető használhatóvá hasonló helyzetben a HLookup() függvény is.
- Ha nem rendezett értékeket keresünk, akkor kell a negyedik paraméter a ?Lookup() függvényekben *HAMIS* értékkel. Ekkor a nem szereplő értékek keresése hibát eredményez!
- Bonyolultabb kereséseknél, ahol a ?Lookup() függvények nem használhatóak vagy csak trükközéssel, az Index-Match függvénypárral kell megoldani a feladatot, mivel bár a Match() függvény képes csökkenő rendezettségű értékhatárokkal is dolgozni, de csak a megtalált tartomány sorszámát adja vissza, ami alapján a keresést az Index() függvény tudja folytatni.

11. Szűrés

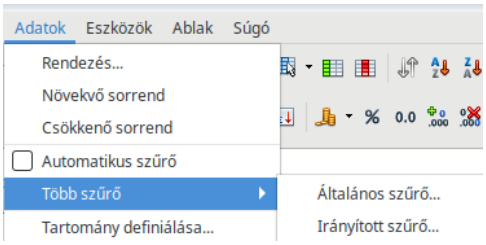
Táblázatkezelő használata során gyakran fordul elő, hogy sok sorban egymás után szerepelnek hasonló jellegű adatok, amelyeknek csak egy részére lenne szükség valamilyen információ számára. Ezeket időnként csak megjeleníteni szeretnénk úgy, hogy a többit ne lássuk addig; máskor ezekre a sorokra kellene valamilyen számításokat elvégezni. Az ilyen jellegű feladatok már átvezetnek ugyan az adatbázis-kezelés területére, azonban a legalapvetőbb műveletek a táblázatkezelőben is hasznosak lehetnek, hiszen így nem kell a táblázatot adatbázissá alakítani. Ebben és a következő leckében ezekről, a már adatbázis-kezelés felé hajló funkciókról lesz szó.

Adott egy táblázat, amelynek sorai közül szeretnénk azokat látni, amelyben egy adott oszlopban meghatározott feltételnek megfelelő értékek találhatóak. Ennek a feladatnak az elvégzésére szolgál a szűrés, amelynek két főbb változata van: az automatikus és az irányított szűrés.

Először nézzük meg az automatikus szűrés működését az Újságok feladat táblázatán (57. ábra)! A táblázat néhány – itt most betűjellel jelölt – újság olvasottsági statisztikáját mutatja ezer főben megadva.

11.1. Automatikus szűrő

Az automatikus szűrő használatához az *Adatok* menüt kell felkeresni, amelyben található egy „*Automatikus szűrő*” nevű menüpont, alatta pedig a „*Több szűrő*” almenü. Mint a 58. ábrán látható, az „*Automatikus szűrő*” menüpont kapcsolóként viselkedik, azaz ki/be lehet kapcsolni. Amennyiben az aktív cella egy olyan táblázatban van, amely pontosan téglalap alakú, nincs benne egyesített cella és minden oldalán (ahol van cella) üres cellák határolják; vagy ilyen alakú tartomány van kijelölve, akkor ezt a program egy *adattábla*ként értelmezi és erre kapcsolja be az **automatikus szűrő** funkciót.

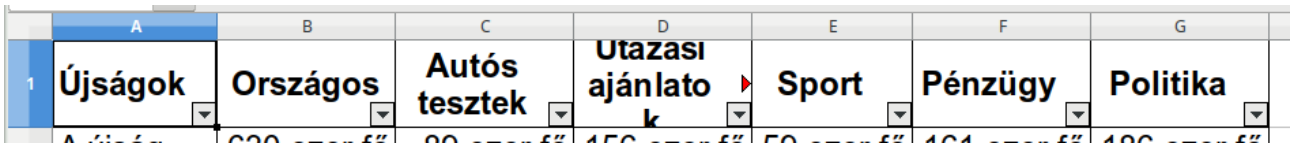


58. ábra: A szűréshez tartozó menüpontok

Az automatikus szűrő bekapcsolt állapotában a táblázat első sorát fejlécként kezeli a program és minden egyes ilyen fejléc-cellában egy nyíl látható, jelezve, hogy ott egy legördülő lista érhető el. Ebben a legördülő listában lehet beállítani az adott oszlopra vonatkozó néhány feltételt.

Újságok	Országos	Autós tesztek	Utazási ajánlatok	Sport	Pénzügy	Politika
A újság	630 ezer fő	89 ezer fő	156 ezer fő	59 ezer fő	161 ezer fő	186 ezer fő
B újság	178 ezer fő	32 ezer fő	69 ezer fő	60 ezer fő	37 ezer fő	55 ezer fő
C újság	270 ezer fő	56 ezer fő	90 ezer fő	98 ezer fő	73 ezer fő	50 ezer fő
D újság	270 ezer fő	40 ezer fő	74 ezer fő	30 ezer fő	36 ezer fő	50 ezer fő
E újság	144 ezer fő	21 ezer fő	29 ezer fő	13 ezer fő	43 ezer fő	31 ezer fő
F újság	49 ezer fő	10 ezer fő	8 ezer fő	16 ezer fő	18 ezer fő	12 ezer fő
G újság	241 ezer fő	115 ezer fő	66 ezer fő	48 ezer fő	49 ezer fő	79 ezer fő
H újság	47 ezer fő	20 ezer fő	27 ezer fő	35 ezer fő	5 ezer fő	15 ezer fő

57. ábra: A példákban használt kiinduló táblázat az Újságok feladatból



59. ábra

A listában megtalálható minden érték, amely az adott oszlopban előfordul, így lehetőség van konkrét értékek megjelenítésének előírására. Emellett a listát használva is lehetőség van az adott oszlop szerint a táblázatot akár növekvő, akár csökkenő sorrendbe rendezni. Lehetőség van arra is, hogy csak az érték szerint első 10 elemet tartalmazó sorok maradjanak meg. Az ebben az oszlopban üres cellákat tartalmazó sorok is kiválaszthatóak.

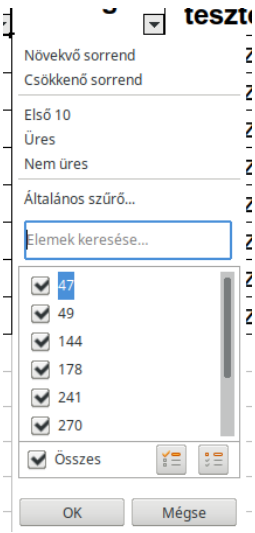
Akármelyik lehetőséget is választjuk, azok a sorok maradnak meg a táblázatból, amelyek a beállításoknak megfelelnek.

A szűrés végrehajtása után figyeljük meg, hogy abban az oszlopban, amelyben valamilyen feltételt állítottunk be, a nyíl kék színűre változik. Ezzel jelzi a program, hogy erre az oszlopra van szűrési feltétel beállítva.

Természetesen több oszlopra is beállítható szűrőfeltétel szükség esetén.

Amennyiben ki akarjuk a feltételt kapcsolni, akkor ismét a legördülő listára van szükség, ahol egyszerűen az „*Összes*” kapcsolót kell bekapcsolni.

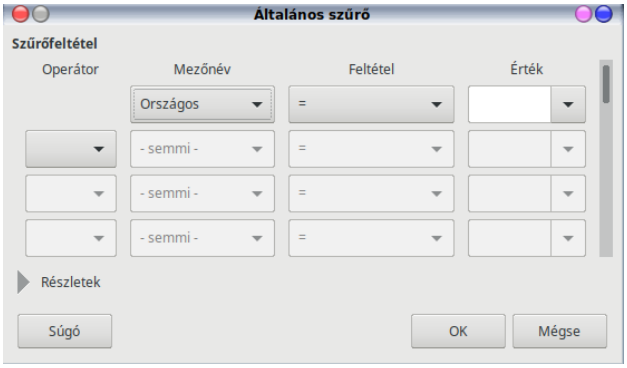
A szűrőfeltételeknek nem megfelelő sorok nem vesznek el, amit az is mutat, hogy a sorok megtartják eredeti sorszámukat: a feltételnek nem megfelelő sorok egyszerűen elrejtésre kerültek.



60. ábra

11.2. Általános szűrő

A legördülő listában van egy menüpont, amely egy kicsit összetettebb keresésre alkalmas: ez az „*Általános szűrő*”, amely közvetlenül a menüből is elérhető. Ez egy új párbeszédablakot eredményez (61. ábra), amelyben egyrészt több oszlopra lehet egyszerre feltételt megadni, másrészt nem csak konkrét értékeket lehet megadni, hanem más relációkat is.



61. ábra: Az általános szűrő párbeszédablaka

A „*Mezőnév*” listában a táblázat első sorában levő cellák értékei vannak felsorolva, így bármelyikre lehet feltételt megadni.

A „*Feltétel*” listában különböző relációk állíthatóak be: egyenlő, nem egyenlő, kisebb, nagyobb, kisebb-egyenlő, nagyobb-egyenlő, legkisebb, legnagyobb stb.

Az „*Érték*” oszlopban pedig akár az oszlopban levő értékek kiválaszthatók, akár be is lehet írni értéket.

Itt például lehet olyat is beállítani, hogy az adott oszlop három legnagyobb értéke legyen a feltétel. De egy szöveget tartalmazó oszlopnál lehet a szöveg egy részére is feltételt megadni („*Tartalmazza*”).

A második sortól kezdve van még egy lista, amelynek mindössze két eleme van, amellyel meg lehet adni, hogy az új sor az előzőekhez milyen logikai művelettel kapcsolandó:

- És: az új sor az előzőekkel együtt teljesítendő feltételt ad meg.
- Vagy: az új sor az előzőekkel alternatív feltételt ad meg.

Az általános szűrő segítségével már lehet olyan feltételt is beállítani, amellyel például egy oszlop értékeinek egy adott intervallumba kell esniük vagy azon kívül kell lenniük.

Például ha azokra az újságokra vagyunk kíváncsiak, amelyek olvasottsága országosan száz- és kétszázezer közé esik, akkor a 62. ábrán látható beállításra van szükség.

62. ábra

Amennyiben pont az ellenkező feltételre van szükség, akkor mindkét relációt meg kell fordítani és az *És* helyett a *Vagy* művelettel kell a két sort összekapcsolni: Országos < 100 vagy Országos > 200.

11.3. Irányított szűrés

Az automatikus szűrő és az általános szűrő alkalmazásának az a nagy hátránya, hogy a szűrés eredménye mindig az eredeti táblázatban, annak a szűrés során nem kívánt sorainak elrejtésével történik.[14] Ez azt jelenti, hogy a szűrés kikapcsolásakor visszaáll az eredeti táblázat.

Ha a szűrés eredményére tartósan szükség van, akkor vágólapon keresztül át kell máshova másolni a táblázatot.

Az irányított szűrő segítségével további lehetőségeket kapunk, amelyek egyike, hogy az eredményt máshova lehet másolni. A másik ok, ami miatt az irányított szűrővel feltétlenül meg kell ismerkedni, hogy a következő fejezetben bemutatásra kerülő adatbázis-függvények használatához is szükség lesz rá.

Az irányított szűrő használatához szükség van egy szűrőfeltételeket tartalmazó tartományra is. Ennek formája a következő:

- Az első sorban azon fejléc-cellák tartalma van felsorolva, amely oszlopokra feltételt akarunk megadni. Ezt célszerű az eredeti táblázat fejlécében szereplő megfelelő cella vágólapon keresztül történő másolásával előállítani, mivel abban az esetben, ha a cella tartalmát a program nem képes megtalálni az eredeti táblázat első sorában, az oszlopot nem fogja tudni figyelembe venni.
- Minden további sor egy-egy olyan feltételt ad meg, amelyek közül legalább az egyiknek teljesülnie kell a szűrendő tartomány egy sorára annak megjelenítéséhez. Ez magyarul azt jelenti, hogy a sorok között *VAGY* kapcsolat áll fenn.
- Az egyes sorokon belül a különböző oszlopokba írt feltételeknek egyszerre teljesülniük kell, azaz egy soron belül *ÉS* kapcsolat áll fenn.
- Amennyiben egy adott feltétel egy értékkel való pontos egyezést ír elő, akkor azt az értéket egyszerűen be kell írni a cellába. Ha olyan feltételt kell előírni, amely egy adott értéknél kisebb (nagyobb) értékekre teljesül, akkor a relációs jelet kell az értéknek követnie úgy, hogy abban az esetben teljesüljön a kívánt feltétel, ha a relációs jel baloldalára a vizsgált értéket képzeljük a szűrendő táblázatból.

14 *Az általános szűrőnél a „Részletek” részben már van ugyan lehetőség megadni, hogy hol szeretnénk a szűrés eredményét látni.*

J	K	L	M	N
Országos	Országos		Országos	Országos
<=200	>=100		<100	
				>200

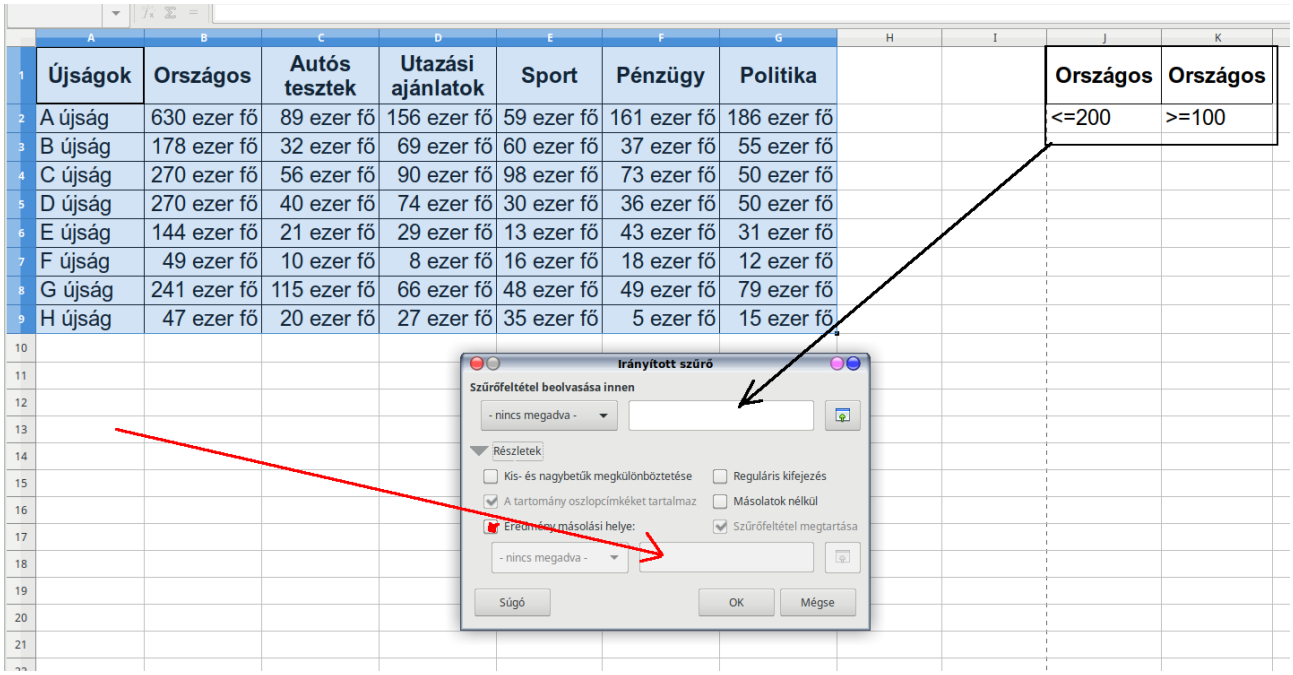
63. ábra: Baloldalt: az országos olvasottság 100 és 200 ezer között van. Jobbra: az országos olvasottság a [100;200] intervallumon kívül van.

A fentiekre példaként nézzük meg azt a példát, amikor az országos olvasottság 100 és 200 között van (ezer főre tekintve), illetve amikor ezen kívül: a <=200 és a >=100 feltételeknek egyszerre kell teljesülniük, tehát egy sorba írnadók (63. ábra, baloldal); ellenben ha az intervallumba nem eső újságokat keressük, akkor vagy 100 ezer alatt, vagy 200 ezer felett van az olvasottság, tehát a két feltételt két külön sorba kell írni (63. ábra, jobboldal).

Az első eset tartománya a **J1:K2** tartomány, a második eset tartománya az **M1:N3** tartomány lesz. A második esetet egyébként meg lehetett volna egy oszlopos tartománnyal is oldani, hiszen a két feltétel ugyanarra az oszlopra vonatkozik, így egymás alá lehetett volna őket tenni.

Amennyiben a szűrőfeltételeket tartalmazó tartomány – amelynek neve szűrőfeltétel a programban – készen van, akkor a szűrést magát az alábbi lépéseken keresztül lehet végrehajtani:

1. Jelöld ki az eredeti, szűrendő táblázatot!
2. Ezután az *Adatok → Több szűrő → Irányított szűrő* menüponton keresztül érhető el az *Irányított szűrő* párbeszédablaka (64. ábra).
3. A párbeszédablak alsó részét a „*Részletek*” kinyitásával lehet elérhetővé tenni. Ez később fontos lesz, mert ott van elrejtve több más beállítás mellett az is, hogy hova kell a szűrés eredményét megjeleníteni.
4. Felül a „*Szűrőfeltétel beolvasása innen*” részben ha a szűrőtartománynak van neve, akkor a legördülő listában az kiválasztható. Egyéb esetben a tartományra való hivatkozást kell ide



64. Ábra: Az irányított szűrő ablaka és a beállításokban megadandó hivatkozások

bejuttatni – a legegyszerűbb azt egérrel kiejlölni (előbb a mezőbe kell kattintani, utána lehet kijelölni). Vigyázni kell, hogy a szűrőfeltételhez a fejléc is hozzátartozik, tehát azt is ki kell jelölni!

5. Az alsó részen az „*Eredmény másolási helye:*” kapcsolót be kell kapcsolni, majd a szűrőfeltételhez hasonlóan vagy a nevesített tartományt kiválasztva, vagy a mezőbe beírva – kijelöléssel is lehet – az eredemény megjelenítésére szánt tartomány bal felső sarkát, meg kell adni, hol jelenjen meg az eredmény. Fontos, hogy itt elég egyetlen cellát megadni, amely az eredményt tartalmazó tartomány bal felső sarka lesz. Ha tartományt adunk meg, semmi garancia nincs arra, hogy abba belefér az eredmény! Az ábrán ez az **A13** cellától kezdve történő megjelenítést fogja jelenteni.

Az irányított szűrő – mint a fentiekből kiderül – minden esetben az eredeti adatok közül a feltételeknek megfelelő sorok másolásával egy új táblázatot hoz létre. Ezt akár egy külön munkalapra is lehet küldeni, ha akarjuk. Ennek az előnye, hogy egyszerre különböző feltételek szerinti szűrések eredményeit is lehet tárolni a dokumentumban.

12. Adatbázis-függvények

Tegyük fel, hogy nem egy adott feltételnek megfelelő sorokra vagyunk kíváncsiak a táblázatból, hanem egy ezek valamelyik oszlopában levő értékekből számolt adatra. Konkrét példán az előző fejezetbeli újságok közül szeretnénk tudni az általgos olvasószámát azoknak az újságoknak, amelyeknek a sportrovatát vagy a pénzügyi rovatát több, mint 50 ezer olvasó olvasta.

Ekkor a következőket kell tennünk:

1. Elkészítjük a szűrőtartományt, amely megfelel a feltételeknek.
2. Az irányított szűrő segítségével elkészítjük azt a táblázatot, amely a feltételnek megfelelő sorokat tartalmazza.
3. Az új táblázatban az Average() függvénnyel elvégezzük az átlagolást.

Tehát a példa esetében készíteni kell egy szűrőfeltételt, amellyel kigyűjthetők azok az újságok, amelyek sportrovatát vagy pénzügyi rovatát 50-nél több ezer olvasó olvasta (65. ábra). Az előző mondatban ugyan magyartalan a fogalmazás, de az „*50-nél több ezer olvasó*” kifejezés arra akar utalni, hogy a cellában levő értéknek ötvennél nagyobbnak kell lennie, mert az értékek mértékegysége „*ezer fő*”.

J	K
Sport	**Pénzügy**
>50	
	>50

65. ábra

Az így kapott szűrőtartománnyal elvégezve a szűrést, az eredmény „*Országos*” oszlopát kell átlagolni.

Ha sok ilyen műveletre van szükség, akkor az sok helyet igényel a szűrések eredményei miatt. Azonban az *adatbázisfüggvények* éppen arra alkalmasak, hogy ezt elkerüljük.

Az *adatbázisfüggvények* ugyanis a szükséges szűrést a *memóriában elvégzik*, nem igényelnek extra helyet a munkafüzetben.

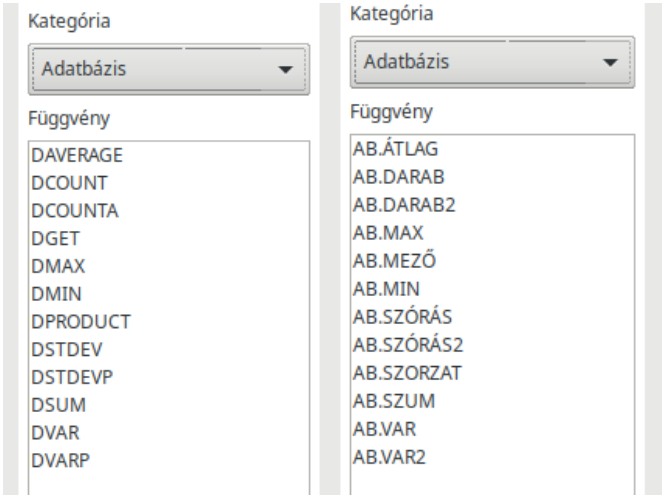
Ezen a módon így egyszerűsödik le a művelet:

1. Elkészítjük a szűrőtartományt, amely megfelel a feltételeknek.
2. A megfelelő adatbázisfüggvénnyel elvégezzük a műveletet. Jelen példánkban ez a DAverage() vagy magyarul AB.Átlag() függvény.

12.1. Az adatbázisfüggvények működése

Az adatbázisfüggvények külön kategóriát kaptak a függvénytündérben (66. ábra). Mindegyik ugyanazokkal a paraméterekkel rendelkezik, így nem érdemes őket egyenként ismertetni. A három paraméter a következő:

- **Adatbázis**: az a táblázat, amelyen dolgoznia kell a függvénynek. Szűréskor ebből a táblázatból akartuk a sorokat kikeresni, vagyis ez az eredeti táblázat.
- **Keresési feltétel**: *Ez a függvények harmadik paramétere!* A keresési



66. ábra: Az adatbázisfüggvények angol és magyar nyelven a függvénytündérben

feltétel az a tartomány, amely a szűrési feltételeket tartalmazza, vagyis a már megismert módon elkészítendő szűrőtartomány.

- A második paraméter: **Adatbázismező**: Kétféleképpen is megadható. Az egyik lehetőség, hogy megadom azt a fejléc-cellát, amelyiknek az oszlopán kell a függvény műveletét elvégezni; ekkor a szöveg lesz a lényeg, ami a paraméterben van, tehát elvileg maga az érték is megadható. A másik lehetőség, hogy megadjuk egész számmal az oszlop sorszámát – ahogyan a VLookup() függvénynél a harmadik paraméterben.

A függvények paraméterlistája tehát a következőképpen néz ki:

```
függvény(adatbázis; adatbázismező;keresési feltétel)
```

Minden függvény esetében tehát az történik, hogy az első paraméterben megadott táblázaton elvégzi a program a harmadik paraméterben megadott szűrőtartomány alapján a szűrést, de ahelyett, hogy az eredményül kapott sorokat valahová kimásolná, a második paraméterben megadott oszlopra a szűréskor megmaradó sorokra elvégzi a függvény neve által jelzett műveletet.

12.2. Az adatbázisfüggvények felsorolása

A LibreOffice Calc 6.0 verziója a következő adatbázisfüggvényeket ismeri – már elég régóta ezek a függvények érhetőek el:

- DAverage/AB.Darab(): átlagolja a második paraméterben megadott oszlop értékeit.
- DCount/AB.Darab(): a CountIf() függvényt valósítja meg jóval bonyolultabb feltételek megadásának lehetőségével, de csak számként értelmezhető értékeket vesz figyelembe.
- DCountA/AB.Darab2(): bármely nem üres cellát figyelembe véve számolja meg az értékeket.
- DGet/AB.Mező(): az adattartomány egy, a keresési feltételeknek megfelelő cellájának tartalmát határozza meg.
- DMax/AB.Max(): a legnagyobb értéket adja vissza.
- DMin/AB.Min(): a legkisebb értéket adja vissza.
- DProduct/AB.Szozat(): az értékeket összeszorozza (mint a Product/Szorzat() függvény).
- DStdev/AB.Szórás(), DStdevP/AB.Szórás2(): értékek szórását határozza meg.
- DSum/AB.Szum(): Az értékek összegét adja.
- DVar/AB.Var(), DVarP/AB.Var2(): az értékek szórásnégyzetét számolja ki.

12.3. Példa a függvények használatára

A fejezet elején említett példát befejezve, ha a szűrés tényleges végrehajtása nélkül akarjuk megkapni azon újságok átlagos olvasószámát, amelyeknek a sport- vagy pénzügyi rovatát 50 ezernél többen olvasták, akkor az 65. ábrán látható szűrőfeltétel elkészítése után az alábbi képletet kell a kívánt cellába beleírni:

```
=DAverage(A1:G9;B1;J1:K3)
```

A fenti képletben a paraméterek a következőt jelentik:

- Az eredeti táblázat az **A1:G9** tartományban található.
- A **B1** („*Országos*”) cella megadja az átlagolandó oszlopot.
- A szűrőfeltétel a **J1:K3** tartományban található.

13. A szerepelt függvények

Az alábbiakban minden olyan függvény részletes bemutatása megtalálható, amelyek az előző fejezetek valamelyikében szerepeltek.

13.1. Statisztikai függvények

13.1.1. Összegzés

Sum() vagyis Szum()

A **Sum**() vagy magyar változatban **Szum**() függvény a paramétereiben szereplő nem üres, számadatot tartalmazó cellák összegét adja eredményül.

Paraméterül legfeljebb 30 paramétert lehet megadni. Ezek a paraméterek lehetnek számértékek, számértéket adó cellák vagy tartományok. Utóbbi esetben a cella minden számként értelmezhető celláját figyelembe veszi a függvény.

A függvény az üres, illetve a szövegértéket tartalmazó cellákat figyelmen kívül hagyja.

SumIf() vagyis SzumHa()

A **SumIf**() vagy magyar változatban a **Szumha**() függvény adott feltételeknek megfelelő cellákban levő számértékeket tud összeadni. A függvénynek három paramétere van:

```
Sumif(tartomány; feltételek; összegtartomány)
```

Amennyiben a harmadik paraméter hiányzik, akkor az első paramétert tekinti a függvény összegtartománynak is. A függvény az első tartomány celláira megvizsgálja a feltételt. Ha a vizsgált cellára teljesül a feltétel, akkor az összegtartomány azonos sorszámú celláját figyelembe veszi az összeg meghatározásánál.

A feltétel lehet egy konkrét érték, amely esetben ha az első paraméterben ez az érték szerepel az adott cellában, akkor teljesül a feltétel. Másik lehetőség a feltétel megadására egy szöveg formájában van, amely esetben a feltétel elején egy relációsjelnek, majd annak az értéknek kell szerepelnie, amellyel a vizsgált cella értékét össze kell hasonlítani. Például a `"<10"` feltétel azokra a cellákra teljesül, amelyben tíznél kisebb érték szerepel.

13.1.2. Átlagolás

Több átlagolás elvégzésére alkalmas függvény is létezik:

Average() vagyis Átlag()

A függvény 1–30 paramétert vár, amelyekben számként értelmezhető – tehát nem üres – cellákat, ilyen cellákat (is) tartalmazó tartományokat vagy konkrét értékeket lehet megadni. Amennyiben a paraméterek között van legalább egy számként értelmezhető érték, akkor a függvény ezen értékek átlagát fogja visszaadni. Ha nincs számként értelmezhető érték a paraméterei között, akkor a nullával osztás miatt hibaüzenetet ad a függvény.

Az üres cellákat és a számként nem értelmezhető értékeket tartalmazó cellákat a függvény az átlag meghatározása során figyelmen kívül hagyja.

AverageA() vagyis ÁtlagA()

Az **Average()** függvénytől eltérően a szövegadatokat nulla értékkel veszi figyelembe.

AverageIf() vagyis ÁtlagHa()

A **SumIf()** függvény testvéreként, az AverageIf() függvény a feltételnek megfelelő cellák átlagát adja eredményül.

`AverageIf(tartomány; feltétel; átlagtartomány)`

Amennyiben csak két paraméter szerepel, akkor az első paraméter lesz egyben az átlagtartomány is. A három paraméter jelentése teljesen megegyezik a **SumIf()** függvény paramétereivel.

13.1.3. Legkisebb és legnagyobb értékek

Min()

A függvény a (legfeljebb 30) paraméterében megadott, számként értelmezhető értékek közül a legkisebbet adja eredményül. A paraméter lehet érték, cella vagy tartomány.

MinA()

A **Min()** függvénynek olyan változata, amely a szövegértékeket nulla értékkel veszi figyelembe.

Max(), MaxA()

A **Max()** függvény a (legfeljebb 30) paraméterében megadott, számként értelmezhető értékek közül a legnagyobb értéket adja eredményül. A paraméter lehet érték, cella vagy tartomány.

A **MaxA()** függvény a szövegértékeket is figyelembe veszi nulla értéknek tekintve.

Small() vagyis Kicsi()

A `Small(tartomány; sorszám)` függvény a `tartomány` számként értelmezhető értékű cellái közül a `sorszám`-adik legkisebb értéket adja eredményül.

Large() vagyis Nagy()

A `Large(tartomány; sorszám)` függvény a `tartomány` számként értelmezhető értékű cellái közül a `sorszám`-adik legnagyobb értéket adja eredményül.

13.1.4. Megszámlálások

Count() vagyis Darab()

A paramétereiben szereplő cellák közül megadja, mennyiben szerepel számként értelmezhető érték. A paraméterek maximális száma 30, egy paraméter lehet számérték, cella vagy tartomány.

CountA() vagyis Darab2()

Nem csak a számértékeket, hanem bármilyen értéket tartalmazó cellát belevesz az összeszámolásba, azaz a paraméterlista által megadott cellák közül megmondja, mennyi cella nem üres. Legfeljebb 30 paramétert lehet megadni.

CountBlank() vagyis DarabÜres()

Megadja a paraméterlistával megadottak közül az üres cellák számát. Ez a függvény is legfeljebb 30 paramétert fogad ed.

CountIf() vagyis DarabTeli()

`CountIf(tartomány; feltétel)` alakban kell megadni, ahol a tartomány a megszámolandó cellák összefüggő tartománya, a feltétel pedig az az érték vagy idézőjeles szöveg, amelynek megfelelő cellák számát adja meg a függvény. A függvénnyel szöveg előfordulási számát is meg lehet határozni, ha a feltételként a kérdéses szöveg szerepel.

13.1.5. Szorzás

Product() vagyis Szorzás()

A legfeljebb 30 számértéket, cellát vagy tartományt tartalmazó paraméterében szereplő valamennyi számérték szorzatát adja eredményül. Nyilvánvalóan nincs értelme a szöveges adatokat nulla értékkel figyelembe venni, hiszen akkor biztosan nulla lenne az eredmény.

SumProduct() vagyis Szorzatösszeg()

Ez a függvény a paraméterében megadott vektorok skaláris szorzatát adja eredményül.

Magyarul: minden paramétere azonos elemszámú egydimenziós tartomány (sor vagy oszlop), amelykből az azonos sorszámúakat összeszorozza, majd veszi az így kapott szorzatok összegét.

A `Sumproduct(A2:A10; C2:C10; E3:E11)` függvény valójában a következő képlet végrehajtását jelenti:

`=A2*C2*E3+A3*C3*E4 + … + A10*C10*E11`

Legfeljebb 30 paramétere lehet a függvénynek, amelyek mindegyike azonos számú cellát megadó tartomány kell legyen. Hogy az egyes tartományok sorokat vagy oszlopokat adnak meg, az a függvényen belül keverhető, de mindegyik tartománynak vagy sort, vagy oszlopot kell megadnia.

13.1.6. Kerekítések

A három kerekítő függvény azonosan paraméterezendő. Első paraméter a kerekítendő számérték (vagy számot tartalmazó cella), a második paraméter pedig a helyiértékek száma, amilyen pontossággal a kerekítést végezni kell. Ez lehet negatív szám is, amely esetben egésznél nagyobb helyiértékre kell kerekíteni.

Round(szám; pontosság)

Magyarul: `Kerekítés()`. A `szám`ot matematikai kerekítést használva kerekíti `pontosság` helyiérték pontosan.

RoundDown()

Magyarul: `Kerek.Le()`. Mindig lefelé kerekít.

RoundUp()

Magyarul: `Kerek.Fel()`. Mindig felfelé kerekít.

13.2. Dátum- és időkezelés

`Today()` magyarul `Ma()` függvény megadja az aktuális dátumot. Mivel mindig a képlet értékének kiszámolásakor érvényes dátumot adja, így a dokumentum megnyitásakor is mindig az aktuális dátumot fogja adni.

`Now()` magyarul `Most()` függvény ugyanilyen módon az aktuális időpontot adja a függvényt tartalmazó képlet minden kiszámításakor.

13.2.1. A dátum- és időkomponensek meghatározása

A dátum év komponensét a `Year()` magyarul `Év()`, a hónap komponensét a `Month()` magyarul `Hónap()`, a nap komponensét pedig a `Day()` magyarul `Nap()` függvénnyel lehet meghatározni. Egy időadat esetén az órát az `Hour()` magyarul `Óra()`, a perceket a `Minute()` magyarul `Perc()` és a másodperceket a `Second()` magyarul `Másodperc()` függvényel lehet megkapni.

Mind a hat függvény esetében egyetlen paramétert kell megadni. Ez lehet egy számérték vagy egy számértéket tartalmazó cella neve. Ezt a számértéket fogja a függvény dátumként vagy időként kezelve visszaadni a megfelelő komponenst.

13.2.2. Dátum érték előállítása

A `Date(év; hónap; nap)` magyarul `Dátum(év; hónap; nap)` függvény segítségével a dátum három komponenséből lehet az adott dátumot reprezentáló egész számot előállítani.

A LibreOffice 6.0 Calc 1583 és 9956 közötti évszámokat tartalmazó dátumokat tud kezelni, így az első paraméterbe írt számnak ebben az intervallumban kell lennie. Ha kétjegyű számot adunk meg, akkor az 19xx vagy 20xx évszámokat fog jelenteni.

13.2.3. Idő érték előállítása

A `Time(óra; perc; másodperc)` magyarul `Idő(óra; perc; másodperc)` függvény segítségével az idő három komponenséből lehet egy adott időadatot reprezentáló valós számot előállítani.

13.2.4. Dátumok közti különbség számítása

A `DateDif(kezdő dátum; befejező dátum; intervallum)` magyarul `Dátumtólig()` függvény két dátum közt eltelt teljes napok, teljes hónapok, teljes évek számának meghatározására szolgál. A harmadik paramétere a következő szövegek valamelyike lehet:

- `"d"`: napokban adja meg a különbséget;
- `"m"`: a két dátum közti teljes hónapok számát adja;
- `"y"`: a két dátum közti teljes évek számát adja;
- `"ym"`: teljes hónapok száma a kezdő dátum és avégdátum különbségéből az éveket kivonva (azaz figyelmen kívül hagyva);
- `"md"`: teljes napok száma a két dátum különbségéből az éveket és hónapokat kivonva (azaz figyelmen kívül hagyva);
- `"yd"`: teljes napok száma a két dátum különbségéből az éveket figyelmen kívül hagyva.

13.2.5. DaysInMonth(dátum)

Magyar neve nincs a függvénynek, tehát magyarul is ugyanezt kell írni.

A megadott dátum hónapjában levő napok számát adja eredményül.

13.2.6. DaysInYear(dátum)

Magyarul is ugyanezt a nevet kell írni.

A megadott dátum évében levő napok számát adja eredményül.

13.2.7. EasterSunday(év)

Megadja, hogy az adott évben milyen dátumra esik Húsvét vasárnapja. A függvénynek nincs magyar neve.

13.3. Logikai függvények

A logikai függvények közül a legfontosabb az **If()**, magyarul **Ha()** függvény:

`If(feltétel; igaz-érték; hamis-érték)`

Az első paraméter logikai értékétől függ, hogy a második vagy a harmadik paraméter értéke lesz-e a függvény értéke. Ha a feltétel paraméter logikai értéke igaz, akkor a második paraméter, különben a harmadik lesz a függvény értéke. A második és a harmadik paraméter értéke nem feltétlenül azonos típusú, de ha eltérőek – például egyik szám, a másik szöveg –, akkor erre a függvény értékének felhasználásánál figyelni kell!

HAMIS konstanst adó értéke van a **False()** magyarul **Hamis()** függvénynek.

IGAT konstanst adó értéke van a **True()** magyarul **Igaz()** függvénynek.

Egy logikai érték tagadása a **Not()** magyarul **Nem()** függvénnyel történik, amely a paraméterében szereplő logikai értékkel ellentétes logikai értéket ad eredményül.

Az **And()** magyarul **És()** függvény akkor ad igaz értéket, ha minden paraméterében szereplő kifejezés logikai értéke igaz.

Az **Or()** magyarul **Vagy()** függvény akkor ad igaz értéket, ha legalább az egyik paraméterében szereplő kifejezés logikai értéke igaz.

Kizáró vagy művelet megadására alkalmas a magyar névvel nem rendelkező **Xor()** függvény. Ez eredeti definíciója szerint akkor ad igaz értéket, ha a két paraméterének logikai értéke ellentétes – azaz egyik igaz, másik hamis –, ám a kettőnél több paraméteres változatra kibővített definíciója szerint akkor ad igaz értéket, ha a páratlan számú paramétere igaz értékű.

13.4. Keresőfüggvények

A leggyakrabban használt keresőfüggvény szerepelt a fentiekben, de még van több is. Ezek a *Függvénytündér* „*Munkafüzet*” kategóriájában találhatóak meg. Ímer a legfontosabbak:

- `VLookup`(*keresési érték*; *tábla*; *oszlopsorszám*; *tartományban keresés*): Magyarul: `Fkeres()` A függvény megkeresi az első paraméterében szereplő értéket a második paraméterben megadott *tábla* első oszlopában. A keresés eredménye kiválasztja valamelyik sort, majd ennek a sornak a harmadik paraméterben megadott sorszámú oszlopában levő cella értékét adja vissza a függvény. Amennyiben a negyedik paraméter nem szerepel – vagy szerepel, de *IGAZ* értékkel –, akkor a *tábla* első oszlopát mint növekvő intervallumok alsó határait kezeli a függvény. Amennyiben a negyedik paraméter *HAMIS* értékkel van megadva, akkor viszont a függvény csak a *tábla* első oszlopában ténylegesen szereplő értékeket találja meg, amelyek ebben az esetben bármilyen sorrendben jelen lehetnek. Ha az első paraméterben olyan érték szerepel, amely nem található meg a keresési tartomány (tábla paraméter) első oszlopában, akkor a függvény a „`#HIÁNYZIK`” hibajelzést adja.
- `HLookup`(*keresési érték*; *tábla*; *sorszám*; *tartományban keresés*): Magyarul: `Vkeres()` A VLookup() függvény megfelelője sorokban történő keresésre: egyszerűen meg kell cserélni a sorok és az oszlopok szerepét a fenti definícióban. Azaz a függvény első paraméterében megadott értéket a *tábla* első sorában keresi balról jobbra haladva. Amelyik oszlopban megtalálja, annak az oszlopnak a harmadik paraméterben megadott sorszámú sorában levő értéket adja eredményül. A negyedik paraméter jelentése azonos a VLookup() függvény negyedik paraméterével.
- `Lookup`(*keresési érték*; *keresési vektor*; *eredményvektor*): Az első paramétert megkeresi a második paraméterben megadott egy soros vagy egy oszlopos tartományban. A függvény értéke az így kapott sorszámú elem lesz a harmadik paraméterben szereplő, szintén egy soros vagy egy oszlopos tartományon belül.
- `Match`(*keresési érték*; *keresési tartomány*; *rendezettség típusa*): Magyarul: `Hol.Van()` A függvény a VLookup() és a HLookup() függvény általánosítására alkalmas: Visszaadja a sorszámát a keresési tartomány azon cellájának, amely megfelel az első paraméterben szereplő értéknek. A harmadik paraméter meghatározza, hogy hogyan kell a második paraméterben szereplő egy soros vagy egy oszlopos tartomány rendezettségét figyelembe venni:
  - Ha az érték 1, akkor a tartomány növekvő rendezésű intervallumokat ad meg. Ekkor a VLookup() és a HLookup() függvény háromparaméteres változatával azonos módon történik a keresés.
  - Ha az érték 0, akkor a tartomány nincs rendezve, csak pontos egyezés lehetséges. Ekkor a keresés módja olyan, mint a VLookup() és HLookup() esetén a negyedik paraméter *HAMIS* értéke esetén.
  - Ha az érték -1, akkor a tartomány csökkenő rendezésű intervallumokat reprezentál. Ekkor a keresés a növekvő rendezéshez hasonlóan viselkedik, de ellenkező irányba haladva.
- `Index`(*tartomány*; *sorszám*; *oszlopszám*, *tartományszám*): Az első paraméterében megadott tartománynak a második és a harmadik paraméterben megadott koordinátákon levő cellájá-

nak értékét adja eredményül. A negyedik (opcionális) paraméter akkor használható, ha az első paraméterben egy több részből álló tartomány *neve* szerepel – ilyen tartományt csak előre definiált névvel lehet megadni – használandó tartományként. Ebben az esetben ez a negyedik paraméter annak a résztartománynak a sorszámát adja, amelyben a cellát a második és a harmadik paraméter meghatározza. A függvényt a Match() függvénnyel együtt használva lehet olyan érték-keresést végezni, amelyre a VLookup() és a HLookup() függvények a korlátaik miatt nem alkalmasak.

13.5. Adatbázisfüggvények

A következő adatbázisfüggvények használhatóak:

- DAverage/AB.Darab(): átlagolja a második paraméterben megadott oszlop értékeit.
- DCount/AB.Darab(): a CountIf() függvényt valósítja meg jóval bonyolultabb feltételek megadásának lehetőségével, de csak számként értelmezhető értékeket vesz figyelembe.
- DCountA/AB.Darab2(): bármely nem üres cellát figyelembe véve számolja meg az értékeket.
- DGet/AB.Mező(): az adattartomány egy, a keresési feltételeknek megfelelő cellájának tartalmát határozza meg.
- DMax/AB.Max(): a legnagyobb értéket adja vissza.
- DMin/AB.Min(): a legkisebb értéket adja vissza.
- DProduct/AB.Szozat(): az értékeket összeszorozza (mint a Product/Szorzat() függvény).
- DStdev/AB.Szórás(), DStdevP/AB.Szórás2(): értékek szórását határozza meg.
- DSum/AB.Szum(): Az értékek összegét adja.
- DVar/AB.Var(), DVarP/AB.Var2(): az értékek szórásnégyzetét számolja ki.

A függvények paramétere mind azonos:

```
függvény(adatbázis; adatbázismező;keresési feltétel)
```

A paraméterek jelentése:

- **Adatbázis**: az a táblázat, amelyen dolgoznia kell a függvénynek.
- **Adatbázismező**: Kétféleképpen is megadható. Az egyik lehetőség, hogy megadom azt a fejléc-cellát, amelyiknek az oszlopán kell a függvény műveletét elvégezni; ekkor a szöveg lesz a lényeg, ami a paraméterben van, tehát elvileg maga az érték is megadható. A másik lehetőség, hogy megadjuk egész számmal az oszlop sorszámát – ahogyan a VLookup() függvénynél a harmadik paraméterben.
- **Keresési feltétel**: A keresési feltétel az a tartomány, amely a szűrési feltételeket tartalmazza.

A függvények működése egyforma:

1. Szűrést készít a memóriában – tehát a munkafüzetben nem lesz nyoma – az `adatbázis` paraméter által megadott tartományból, keresési szűrőtartománynak a `keresési feltétel` paraméterben megadott tartományt használva.
2. A keletkezett táblázat sorait figyelembe véve elvégzi a függvény nevében szereplő műveletet az `adatbázismező` paraméter által megadott oszlopon.

Felhasznált irodalom

Függvénymutató