

Óbudai Egyetem

Neumann János Informatikai Kar

Alkalmazott Informatikai Intézet



TUDOMÁNYOS DIÁKKÖRI DOLGOZAT

LUCA - DOKUMENTUM KONVERTER LÁTÁSSÉRÜLTEKNEK

Szerző:

Bäck Konstantin

mérnök informatikus BSc. szak, IV. évf.

Konzulens:

Dr. Vámosy Zoltán

egyetemi docens

Budapest, 2014.

TARTALOMJEGYZÉK

1	Absztrakt	2
2	Bevezetés.....	3
2.1	Látássérültek integrált képzése a közoktatási rendszerben	3
2.2	A látássérültek oktatásában alkalmazott informatikai eszközök	5
2.2.1	OCR programok	5
2.2.2	Beszédszintetizátorok	7
2.2.3	Szabványok, speciális fájlformátumok.....	9
2.3	A projekt célja.....	10
3	LUCA rendszer	12
3.1	LUCA Converter.....	12
3.1.1	Preprocessor	13
3.1.2	Parser	39
3.2	LUCA Reader	59
3.2.1	Document navigator	59
3.2.2	Reader engine	61
4	Eredmények bemutatása és értékelése	62
4.1	LUCA Converter.....	62
4.1.1	Szegmentálás	62
4.1.2	Karakterek azonosítása	65
4.1.3	Összefüggő szövegelemek azonosítása	67
4.1.4	Matematikai kifejezés struktúraelemzés.....	68
4.1.5	PDF állományok feldolgozása.....	70
4.2	LUCA Reader	71
4.2.1	Navigálás	72
4.2.2	Felolvasás	72
5	A továbbfejlesztés lehetőségei	73
5.1	Implementálás	73
5.2	Továbbfejlesztés	73
6	Értékelés.....	75
7	Irodalomjegyzék.....	76
8	Mellékletek.....	78

8.1	Neurális hálózatok alkalmazásával kapcsolatos kísérletek.....	78
8.2	A LUCA Converter szegmentáló eljárásának teszteredményei.....	82
8.3	A LUCA Converter karakter azonosító eljárásának teszteredményei	85
8.4	A LUCA Converter összefüggő szövegelem azonosításának teszteredményei....	87

1 ABSZTRAKT

Egy modern tankönyvben számtalan olyan szerkesztési megoldással találkozhatunk, amely a feldolgozandó anyagban való egyszerű eligazodást, és a megértést segíti elő. A szöveg tagolása, a különböző szövegrészekben alkalmazott karakterek stílusa, színe vagy mérete mind a szöveges információ befogadásának elősegítését szolgálja. Ma Magyarországon az integrált közoktatási rendszer keretein belül tanuló látássérült diákok kevés kivételtől eltekintve a látó társaik számára készített tankönyvek speciális átiratait használják. Ezek az átiratok jellemzően vagy a papír alapú könyvek digitalizálásával létrejött képfájlok, vagy elektronikus dokumentumok (DOCX, PDF) feldolgozásával jönnek létre. Az átiratok létrehozásának célja a szöveges információ kinyerése a forrás fájlokból annak érdekében, hogy a későbbiekben az egy TextToSpeech alkalmazással felolvastatható legyen. Bár az így előállított átiratok hozzáférhetővé teszik a tankönyvek oldalain meglévő szöveges információ nagy részét a látássérültek számára, a szerkesztési megoldásokban kódolt másodlagos információ, és a speciális szövegelemek (pl.: matematikai kifejezések) teljes egészében elvesznek a feldolgozás során.

A dolgozatban ismertetett LUCA rendszer célja a látók számára készített tankönyvek átkonvertálása olyan strukturált szöveges állománnyá, mely mind tartalmában, mind felépítésében megegyezik a forrás dokumentummal. A konvertálás során kiemelt szerepet kap a különböző matematikai kifejezések, és függvények azonosítása és struktúraelemzése.

A dolgozatban bemutatásra kerülnek a látássérült diákok képzéséhez elengedhetetlen tankönyvek beszerzése körül hazánkban jelenleg is tapasztalható anomáliák, valamint azok az eszközök és speciális fájlformátumok, amelyeket ma a leggyakrabban használnak a látók számára készített könyvek konvertálására. Ezen felül a dolgozat részletes ismertetőt nyújt a LUCA rendszer egyes moduljainak működéséről, technikai megvalósításáról, az elért eredményekről, valamint a továbbfejlesztés lehetséges irányairól.

2 BEVEZETÉS

2.1 Látássérültek integrált képzése a közoktatási rendszerben

„Becslések szerint az európai népességnek mintegy 10–15%-a valamilyen fogyatékossgal él. Az Unióban erős társadalmi elkötelezettség bontakozott ki társadalmi integrációjuk mellett. A fogyatékosok természetes együttélése a nem fogyatékosokkal még nem valósult meg társadalmi méretekben, az alapvető emberi jogok, az egyenlő esélyek napjainkban is elérhetetlenek sokuk számára. [...] 1996 óta a fogyatékkal élők társadalmi integrációja bővülő, átfogó stratégia alapján fejlődik. A közoktatásról szóló 1993-as LXXIX. többször módosított törvény hazánkban is zöld utat nyitott az integrált nevelésnek. [...] A sajátos nevelési igényű tanulók jelenléte újszerű kihívásokat hoz a többségi iskolákba. A speciális pedagógiai, módszertani, tanulásszervezési kérdéseken kívül a tárgyi és személyi feltételek megteremtése is feladatot jelent a befogadó intézmények számára.” [1]

A látássérült tanulók integrált képzéséhez szükséges személyi feltételeket az érintett pedagógusok továbbképzésével, gyógypedagógus alkalmazásával, a tanulót segítő mentor megbízásával, valamint az integrált oktatásban résztvevő pedagógusok számára ösztönző bérezési rendszer kialakításával lehet biztosítani. A tárgyi feltételek közé a megfelelő taneszközök (tankönyvek, abakusz, mértani testek stb.), speciális segédeszközök (pl.: Braille írógép) és az akadálymentesített környezet tartozik.

Sajnálatos módon a közoktatásban tanuló látássérült diákok tankönyvi ellátottsága hazánkban nem teljes körűen biztosított. Számukra jellemzően három tankönyv formátum érhető el, melyek a Braille, az MP3 és az elektronikus (DOC, TXT, PDF) tankönyvek. Ezeknek a tankönyveknek az ára a hagyományos tankönyvek árának a többszöröse, előállításuk időigényes, ráadásul (a Braille kivételével) nem illeszkednek a vakok speciális igényeihez. Az ilyen típusú tankönyvek előállítását a Magyar Vakok és Gyengénlátók Országos Szövetsége végzi az oktatási intézmények megrendelése alapján.¹

Fontos megjegyezni, hogy számos korszerűbb, kimondottan a vakok igényeihez illeszkedő dokumentum formátum létezik, ám ezek előállítása a látók számára készített tankönyvek alapján a jelenlegi megoldásoknál sokkal időigényesebb. Ez lehet az egyik oka annak, hogy hazánkban ezek az eszközök a közoktatási rendszerben mostanáig nem tudtak elterjedni. A speciális tankönyvek előállítása a korszerűtlen formátumok alkalmazása ellenére sem egyszerű. A gyártás folyamata lassú, az elkészült anyagok alkalmazhatósága kérdéses.

A látássérültek számára készített tankönyvek minőségét a gyártás gondossága, és az alkalmazott formátum fizikai korlátai együttesen határozzák meg.

¹ Forrás: <http://www.mvgyosz.hu/>

A gyártás gondossága az emberi tényezőkön túl a rendelkezésre álló idő és pénz (tehát a szándék) függvénye. A fizikai korlátok azonban az alkalmazott technikától függnének, tehát az erőforrások pusztán növelésével nem lehet rajtuk változtatni. Jó példa erre az MP3 formátum, amiben szövegrész akkor sem lesz kereshető, ha a gyártás optimális körülmények között történik.

A fent leírtak miatt érdemes áttekinteni az egyes formátumok jellemzőit.

- Braille könyv

Ez a felsorolásban szereplő formátumok közül talán a legjobban használható. Alkalmas a legtöbb tankönyvi információ átadására úgy, hogy feldolgozása gyors, a könyvben történő navigálás egyszerű. Hátránya, hogy előállítása költséges és bonyolult, valamint mérete és súlya miatt nehezen hordozható.

- MP3 könyv

MP3 könyv előállításakor a készítő a feldolgozandó könyvet egyszerűen felolvassa és MP3 formátumban rögzíti. Ennek a formátumnak a hátránya, hogy se navigálni se keresni nem lehet benne, valamint az összetett struktúrák (pl.: matematikai kifejezések) elemi vizsgálatát sem teszi lehetővé, ami a megértést nagyban megnehezíti.

- Elektronikus könyv

Amennyiben az elektronikus könyvet a tankönyvkiadó bocsátja a látássérült tanulók rendelkezésére (pl.: PDF formátumban), az tartalmilag várhatóan azonos lesz a nyomdába kerülő könyvvel. Mivel azonban a képernyőolvasó programok csak a karakterek felolvasását végzik el, a szerkesztési megoldásokban kódolt másodlagos információ (pl.: Címsor szintenként eltérő betűtípus, eltérő háttérszín alkalmazása a megjegyzéseknél, bekezdésekre tagolt szöveg, stb.) teljes egészében elvész. Ezen felül komoly problémát jelent a matematikai kifejezések kezelése is, mivel azok megfelelő felolvasásához strukturális elemzést kellene rajtuk elvégezni.

A látássérültek számára készített tankönyvek előállításának nehézségei miatt azok gyártását sok esetben az oktatási intézmények maguk oldják meg a rendelkezésre álló eszközökkel. Ilyen megoldás a papír alapú könyvek oldalankénti digitalizálása, majd az így előálló képfájlok OCR² alkalmazással történő feldolgozása. A feldolgozás eredménye egy olyan szövegfájl lesz, amit egy általános TextToSpeech rendszerrel fel lehet olvasatni. Nyilvánvaló, hogy a „házi” megoldás komoly hátránnyal indul a „gyári” PDF

² Optical Character Recognition – Optikai karakter felismerés

formátumú könyvekhez képest, mivel ennél a módszernél a karakterfelismerés során is fellép információvesztés.

2.2 A látássérültek oktatásában alkalmazott informatikai eszközök

2.2.1 OCR programok

Az OCR programok alkalmasak arra, hogy képfájlokból szöveges információt nyerjenek ki. Ezek az alkalmazások alapvető fontossággal bírnak a látássérültek életében, mivel a papír alapú dokumentumok digitalizált verzióit ezekkel konvertálhatják olyan formátumúra, ami egy felolvasó program inputja lehet.

Belátható, hogy amennyiben a szöveges információ kinyerésének folyamata során adatvesztés következik be, az output állomány későbbi felhasználhatósága sérül.

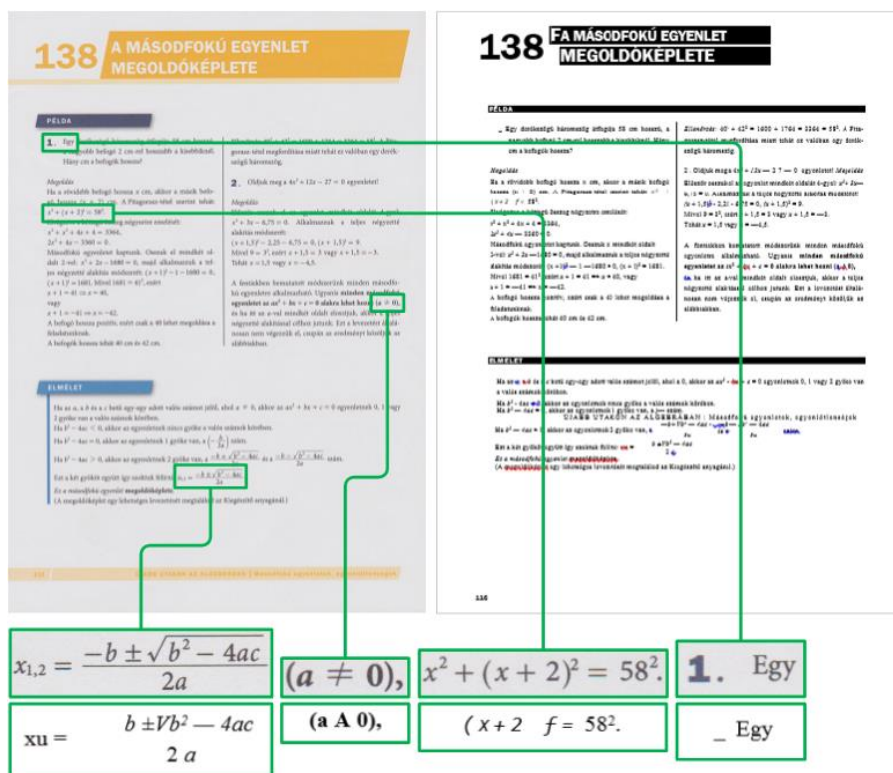
Az alábbiakban két, Magyarországon népszerű, fizetős OCR program működése kerül bemutatásra, melyek a Nuance Omnipage 18 és az ABBYY FineReader 11.

2.2.1.1 Nuance Omnipage 18

Az Omnipage abból a magyar fejlesztésű Recognitából nőtte ki magát, amely évtizedes fejlesztési múlttal büszkélkedhet. A program teljesítményének vizsgálata során egy matematika tankönyv egy oldalának feldolgozását adtuk feladatul. A feldolgozandó oldal kiválasztásakor elsődleges szempont volt, hogy az minél több olyan elemet tartalmazzon, melynek értelmezése problémát jelent a látássérültek számára. Az oldal 300 DPI felbontással került digitalizálásra, a feldolgozás eredménye DOCX formátumban lett elmentve.

A Nuance Omnipage által feldolgozott oldalt az **1. ábra** tartalmazza. Az ábra bal oldalán az eredeti oldal, a jobb oldalon pedig a feldolgozás eredménye látható. Az ábra alsó részén kiemelésre került néhány tipikus feldolgozási hiba, melyből láthatóak az Omnipage gyengeségei.

A feldolgozás eredményével kapcsolatban elmondható, hogy az egyszerű karakterek azonosítása 90% körüli pontossággal történt meg. Hibás azonosítás jellemzően az ékezetes karaktereknél fordult elő. A nagyméretű inhomogén (különböző színű képpontokból álló) karakterek azonosítása egyáltalán nem történt meg, ezek az eredmény fájlban képként jelentek meg. A matematikai kifejezések azonosítását az Omnipage minden olyan esetben elrontotta, ahol egyszerű hatványkitevős szerkezetnél bonyolultabb struktúrát kellett volna azonosítani.



2. ábra - Abby FineReader-rel elvégzett karakterfelismerés matematika tankönyv egy oldalán

A FineReader feldolgozási eredménye számos hasonlóságot mutat az Omnipage-dzsel. Az egyszerű karakterek itt is nagy pontossággal lettek azonosítva, az összetett matematikai kifejezések azonosítása azonban ezúttal is teljesen hibásan történt meg.

Pozitív eredmény az Omnipage-hez képest az inhomogén szegmenszek azonosítása, ami ezúttal többé-kevésbé megtörtént.

2.2.2 Beszédszintetizátorok

A látássérült felhasználók az elektronikus dokumentumokat speciális programokkal olvastatják fel. Ezek az alkalmazások olyan beszédszintetizátorok, melyek az írott szöveget képesek beszéddé alakítani. Számos változatuk létezik mind az ingyenes, mind a fizetős szegmensben. Az alábbiakban egy széles körben elterjedt fizetős alkalmazás a JAWS For Windows kerül terítékre.

2.2.2.1 JAWS For Windows

Az eddig tárgyalt OCR alkalmazásokkal ellentétben a JAWS For Windows-t kimondottan látássérült felhasználók számára fejlesztették. Funkcionalitása alapján képernyőolvasó eszköznek tekinthető, elsődleges feladata az operációs rendszer, és az azon futó alkalmazások használatának elősegítése a képernyőn található szövegek felolvasásával. Nagyon jó minőségű beszédszintetizátorral rendelkezik, amely természetesen alkalmas bármilyen szöveges dokumentum felolvasására.

A program két különböző hangmintát kínál magyar nyelven, a felolvasás tempója mindkét mintánál szabadon beállítható. A szintetizált beszéd minősége kívánnivalót nem hagy maga után, hosszú ideig hallgatva se válik idegesítővé. A JAWS For Windows lehetővé teszi a felolvasó motor vezérlését bármilyen program számára. Ezt a JAWS Script Language segítségével éri el, amellyel olyan scriptfájlok hozhatók létre, amelyek illeszkednek annak a programnak a felületéhez, amelyben a felolvasást el akarjuk végezni.

A JAWS képességeinek vizsgálatához egy olyan dokumentum került létrehozásra, amely tartalmaz különböző szintű címsorokat, táblázatot, felsorolást és matematikai kifejezést (egyenletszerkesztővel szerkesztve). A dokumentum felolvasatása az alábbi tapasztalatokkal zárult:

- Táblázat

Az egyes cellák tartalma megfelelően került felolvasásra. A felolvasás közben a program folyamatosan tájékoztatott az aktuális pozícióról, és a táblázat méretéről.

- Felsorolások

A felsorolás elemeinek felolvasása semmiben nem tér el a normál sorok felolvasásától. Amennyiben a felsorolás sorszámozott, a felsorolás elemei előtt felolvasásra kerül az adott elemet azonosító szám is. Ha a felsorolás elemeit speciális karakter jelöli (pl.: kötőjel) akkor a felolvasáskor nem derül ki, hogy az egyes elemek logikailag összetartoznak.

- Matematikai kifejezések

Az egyszerű matematikai kifejezéseket a program balról jobbra felolvassa. Ilyenkor megfelelően alkalmazza az operátorok neveit, jelzi a nyitó és csukó zárójeleket. Összetett matematikai kifejezés felolvasásakor azt a program képként azonosítja, melynek méretén kívül semmilyen információt nem közöl.

- Fejezet címek felolvasása

A fejezetcímek felolvasása rendben megtörténik, de a felolvasás során nem derül ki, hogy az adott szöveg egy nagyobb egység címe.

Összességében elmondható, hogy a JAWS For Windows mind a felolvasás minőségében, mind a külső eszközök felé kínált interfész rugalmasságában nagyszerűen teljesít, ugyanakkor a speciális szövegelemek felolvasásakor jelentős információvesztés tapasztalható a felolvasandó és a felolvasott szöveg között. Fontos ugyanakkor megjegyezni, hogy a felolvasás során tapasztalt problémák a megfelelő scriptfájlok elkészítésével orvosolhatóak (lásd. 3.2.2).

2.2.3 Szabványok, speciális fájlformátumok

2.2.3.1 DAISY

A DAISY elnevezés a Digital Accessible Information SYstem (Digitálisan Elérhető Információs Rendszer) kifejezés rövidítése. Ez egy olyan hibrid hangoskönyv formátum, amely a dokumentumot írott és felolvasott formában egyaránt tartalmazza úgy, hogy azok szinkronizálva vannak egymással. Az írott szöveg strukturált XHTML alapú, míg a hang MP3 formátumú.

Ez a megoldás a hagyományos MP3 könyvekkel szemben keresetőséget, és mind szekvenciálisan mind hierarchikusan történő bejárhatóságot biztosít.

A DAISY Európa szerte általánosan használt eszköz, irodalmi műveket és folyóiratokat is terjesztenek ebben a formátumban. Számos szoftveres és hardver megoldás létezik DAISY dokumentumok felolvasására és létrehozására, a magyarországi elterjedtsége ugyanakkor elmarad az Európai átlagtól. Ennek számos oka van, melyek közül talán az egyik legfontosabb a kisszámú, magyar nyelven DAISY formátumban elérhető anyag.³

2.2.3.2 MathML

A MathML a Mathematical Markup Language (Matematikai Leíró Nyelv) kifejezés rövidítése. A nyelv kifejlesztésének oka az volt, hogy matematikai kifejezések beágyazását a világhálón fellelhető dokumentumokba szabványos formában lehessen megvalósítani.

A nyelv XML szerkezetű, tehát tökéletesen alkalmas a matematikai kifejezések struktúrájának a leírására. Számos böngésző alapról támogatja a használatát (pl.: Firefox), de pl. az Internet Explorerhez is elérhető MathML plugin.

A MathML tervezésekor a következő célok lettek meghatározva [2]:

- Meg kell valósítani matematikai anyagok kódolását úgy, hogy az alkalmas legyen oktatási és tudományos kommunikáció megvalósítására.
- A kódolásnak tartalmaznia kell az egyes matematikai jelöléseken túl a matematikai tartalmat is.
- Biztosítani kell a konvertálás lehetőségét egyéb matematikai formátumokra mind a formai, mind a tartalmi jellemzők alapján (grafikus kijelzők, beszéd szintetizátorok, TEX, Braille írás, stb.).
- Meg kell valósítani a hosszú matematikai kifejezések megfelelő bejárhatóságát.

³ A csekély számú magyar nyelven elérhető DAISY formátumú kiadványra vonatkozó kijelentés alapja a témakörben elvégzett netes keresés eredményeként kapott elenyésző számú találat.

- Bővíthetőség.
- Sablonok, és más szerkesztési technikák alkalmazása kifejezések szerkesztésekor.
- Egyszerű olvashatóság és könnyű gépi feldolgozás.

2.2.4 Összefoglalás

A látássérültek számára számos olyan eszköz elérhető, melyek segítségével elvben hozzáférhetnek a kifejezetten látók számára készített tartalmakhoz is.

A papír alapú könyveket az OCR eszközökkel felolvashatóvá lehet tenni, majd a karakterfelismerés eredményét képernyőolvasó programokkal fel lehet olvasatni. Léteznek olyan szabványok, amelyek tökéletesen illeszkednek a vakok igényeihez, mivel alkalmasak egy könyv, vagy akár egy a könyvben fellelhető matematikai kifejezés struktúrájának a leírására. Ezen felül elérhetőek olyan felolvasó programok, melyekkel ezek a struktúrák bejárhatóak.

Ennek ellenére kevés az elérhető minőségi konverzió, ami a hazai tankönyvpiacra különösen nagy problémát jelent.

Mivel a karakterfelismerő rendszerek jellemzően a látók igényeinek kielégítésére lettek kifejlesztve, a karakterfelismerés eredménye egy olyan elektronikus dokumentum lesz, amely nem a struktúrájában, hanem a megjelenésében követi a feldolgozott papír alapú könyv szerkezetét. Ebből a dokumentumból pedig csak komoly energia befektetéssel lehet a látássérültek igényeihez illeszkedő anyagot előállítani.

2.3 A projekt célja

A projekt célja egy olyan programcsomag kifejlesztése (LUCA⁴), mely a látássérültek tankönyvi ellátottságával kapcsolatos problémákat megoldja. A programcsomag két elemből áll, melyek közül az egyik (LUCA Converter) végzi a látássérült felhasználók speciális igényeit kielégítő tartalmak előállítását, míg a másik (LUCA Reader) felel ezeknek a tartalmaknak a felolvasásáért.

A látók olvasás közben nem csak az egyes karakterekhez, hanem a karakterek kinézetéhez, és a szöveg struktúrájához is feldolgozást segítő információt rendelnek. Egy szövegrész betűtípusa, betűmérete, stílusa, vagy színe alapján beazonosítható annak dokumentumban betöltött funkciója. Az egyes szövegelemek egymáshoz képest elfoglalt pozíciója alapján pedig azonosíthatóak a logikailag összefüggő tartalmak.

Ahhoz, hogy egy kimondottan a látók számára készített könyvet vakok számára feldolgozhatóvá konvertálhassunk, a karakterekben kódolt elsődleges (tartalmi)

⁴ Szent Lúcia vagy Luca a keresztények által az 5. század óta tisztelt szűz és vértanú. A középkorban közkedvelt szent volt, legendája több művészt megihletett. Lúcia védőszentje a szembetegségben szenvedőknek és vakoknak. (Forrás: Diós István - A szentek élete)

információ azonosítása mellett, el kell végezni a vizuális megjelenésben kódolt másodlagos (formai) információ azonosítását is. A vizuális jellemzők elemzése felfedi a feldolgozandó tankönyv felépítését, ami egy strukturált szöveges állománnyal leírható.

A LUCA Converter feladata tehát az, hogy a szöveges információt kinyerje az inputként átadott dokumentumokból, és az egyes szövegelemek formai jellemzői alapján legenerálja a forrásdokumentum struktúráját. Az azonosított szöveges információk ebbe a struktúrába kerülnek visszaírásra. A LUCA Converter működésével kapcsolatos elvárás a paramétereizhetőség, ami biztosíthatja különböző szerkesztési megoldásokat alkalmazó kiadványok feldolgozását a program módosítása nélkül.

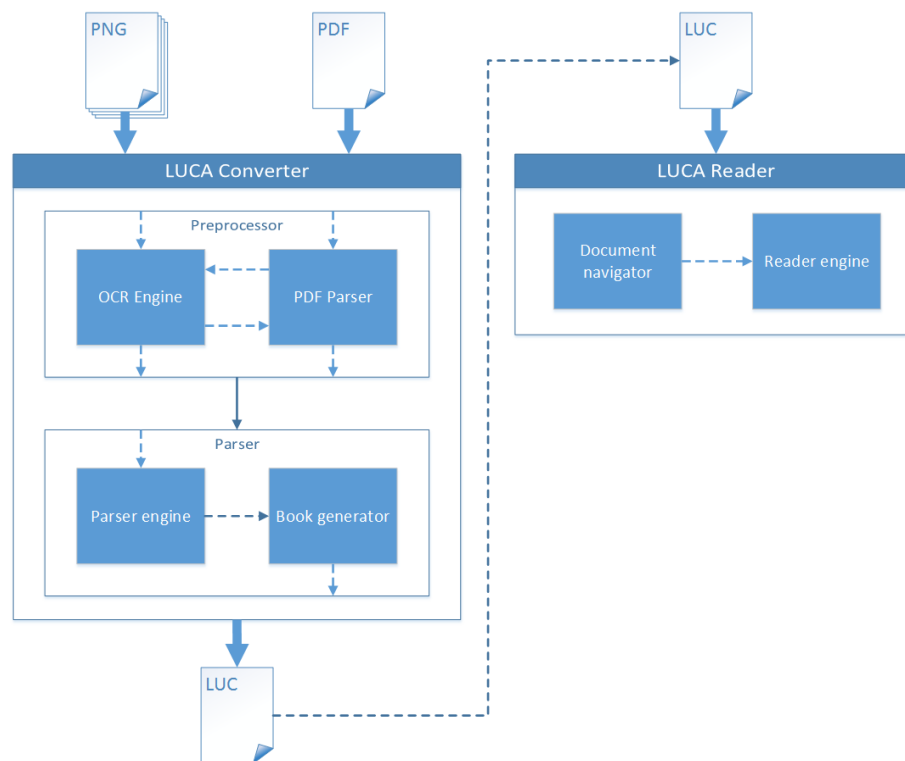
A tankönyvekben megjelenő speciális szövegelemek közül az alábbiak kezelését kell biztosítani:

- Betűtípus, betűméret, szín és háttérszín alapján
 - fő és alfejezetek
 - definíciók
 - megjegyzések
- Az egyes karakterek tartalma és egymáshoz képest elfoglalt pozíciója alapján
 - bekezdések
 - felsorolások
 - számozások
 - matematikai kifejezések
 - függvények

A LUCA Reader alkalmazás feladata a LUCA Converter által előállított szöveges állomány bejárásának és felolvasásának a megvalósítása. A bejáráshoz olyan navigációs eljárást kell kialakítani, ami a strukturált állomány felépítését is figyelembe veszi. Maga a felolvasás a JAWS For Windows alkalmazás beszélő motorjának a felhasználásával valósul meg.

3 LUCA RENDSZER

A LUCA rendszer két, egymástól fizikailag független programból áll, melyek a LUCA Converter és a LUCA Reader. A rendszer felépítését a **3. ábra** tartalmazza.



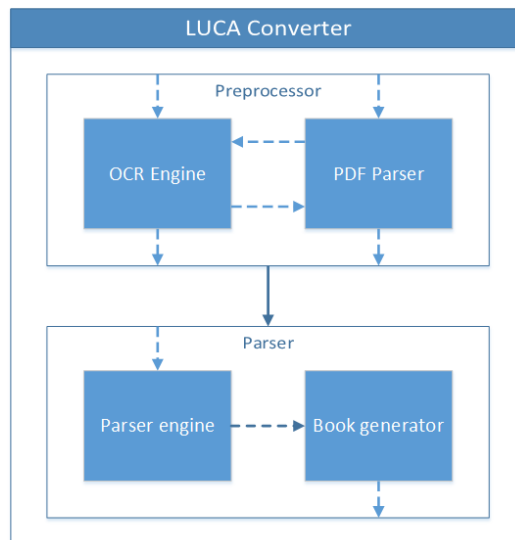
3. ábra - A LUCA rendszer felépítése

A LUCA Converter feladata az inputként megadott dokumentumok strukturált szöveges állománnyá való átalakítása úgy, hogy az output mind felépítésében, mind tartalmában megegyezzen az eredeti dokumentummal. Az input lehet a szkennelt oldalakat tartalmazó képfájlok gyűjteménye (PNG), vagy elektronikus dokumentum (PDF). A program outputja egy XML alapú LUC kiterjesztésű állomány, ami minden olyan információt tartalmaz a forrás dokumentumból, mely a látássérültek számára megkönnyítheti a feldolgozást.

A LUCA Reader feladata a LUCA Converter által előállított LUC kiterjesztésű állományok bejárása és felolvasása.

3.1 LUCA Converter

A LUCA Converter alkalmazás az előfeldolgozó (Preprocessor) és az értelmező (Parser) alrendszerekből áll. A LUCA Converter alkalmazás felépítését a **4. ábra** mutatja.

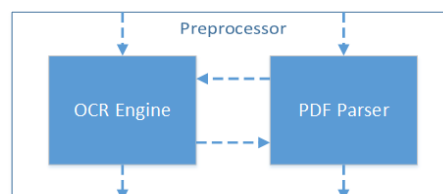


4. ábra - A LUCA Converter felépítése

A Preprocessor alrendszer feladata a szövegben található karakterek azonosítása, és azok formai jellemzőinek meghatározása, míg a Parser alrendszer az előfeldolgozó által meghatározott jellemzőket elemezve felépíti a feldolgozandó dokumentum struktúráját, és előállítja a könyvet leíró állományt.

3.1.1 Preprocessor

A Preprocessor alrendszer az OCR Engine és a PDF Parser modulokból áll. Az alrendszer felépítését az 5. ábra mutatja.



5. ábra - A LUCA Converter előfeldolgozójának a felépítése

A Preprocessor az inputból kinyeri a szöveges információt, és meghatározza az egyes szövegelemek formai jellemzőit. Ez a feladat teljesen más jellegű megoldást igényel különböző típusú input állományok esetén. Míg képfájlok feldolgozása során egy OCR motort kell megvalósítani a feldolgozáshoz, elektronikus dokumentumoknál (PDF, DOC, RTF) a feldolgozandó fájlból ez az információ közvetlenül kinyerhető. A Preprocessor tehát az input típusától függően fogja egyik, vagy másik modulját felhasználni a munkavégzéshez. Ez a felépítés lehetővé teszi azt, hogy az újabb fájlformátumok kezelését az alrendszer moduláris bővítésével lehessen megvalósítani.

3.1.1.1 OCR Engine

Az OCR Engine az előfeldolgozó alrendszer képfeldolgozásért felelős modulja. Működése során elvégzi a dokumentum szkennelt oldalainak szegmentálását, valamint az egyes szegmensekben kódolt karakterek azonosítását, és vizuális jellemzőinek (betűtípus, méret, szín, stb.) meghatározását. A megfelelő működés érdekében az input állományokat lapszkennelvel kell létrehozni 300 dpi felbontásban, színes beolvasással. Az állományok kiterjesztése kötelezően PNG, ami veszteségmentes tömörítést tesz lehetővé.

Szegmentálás

A szegmentálás során kerülnek azonosításra azok a pixelcsoportok, melyek egy-egy karaktert jelölnek. Ehhez első lépésben elkülönítésre kerülnek a szöveget, és a háttéralkotó képpontok. Ezt követően a szöveget alkotó pixelek szomszédsági vizsgálatával csoportba foglalhatóak azok a képpontok, melyek egy karaktert alkotnak. Ezek a csoportok lesznek a dokumentum feldolgozásának alapját képező szegmensek. Az egyes karakterek azonosítása, és a formai jellemzők meghatározása ezek alapján a szegmensek alapján történik.

Egy hagyományos könyv jellemzően fehér papírra nyomtatott fekete betűkből áll. A korszerű tankönyvek ezzel szemben számtalan speciális szerkesztési megoldást alkalmaznak, melyek segíthetik a tananyag feldolgozását. Ezek a megoldások bár szebbé, és átláthatóbbá teszik a szöveget, a gépi feldolgozást megnehezítik. Annak érdekében, hogy az OCR Engine a tankönyvek feldolgozásakor is megfelelő hatásfokkal működjön, fel kell készíteni az azokban alkalmazott szerkesztési megoldások kezelésére. Ennek megfelelően az alábbi funkcionalitást kell megvalósítania:

- Normál homogén szövegrész azonosítása

Normál homogén szövegrészben a szöveg világos háttér előtt sötét betűszínnel van megjelenítve úgy, hogy az egyes karaktereket, és a karakterek közvetlen környezetét is azonos színű képpontok alkotják.

- Invertált homogén szövegrész azonosítása

Az invertált homogén szövegrész a kiemelés gyakran alkalmazott megoldása. Ebben az esetben a szöveg színe halványabb, mint a környezetéé, és mind a szöveget, mind a közvetlen környezetét azonos színű képpontok alkotják.

- Normál inhomogén szövegrész azonosítása

Színes szövegrészben az egyes karaktereket gyakran különböző színű képpontok alkotják. Mivel az egyes pixelek intenzitás értékei között jelentős eltérés lehet, könnyen a háttér részeként kerülhet azonosításra olyan képpont, amely valójában egy karakter valamely belső pontja.

Normál inhomogén szövegrésznél tehát a szöveg különböző színű, a háttérnél jellemzően sötétebb képpontokból áll, míg a háttér világosabb, azonos színű pixeleket tartalmaz.

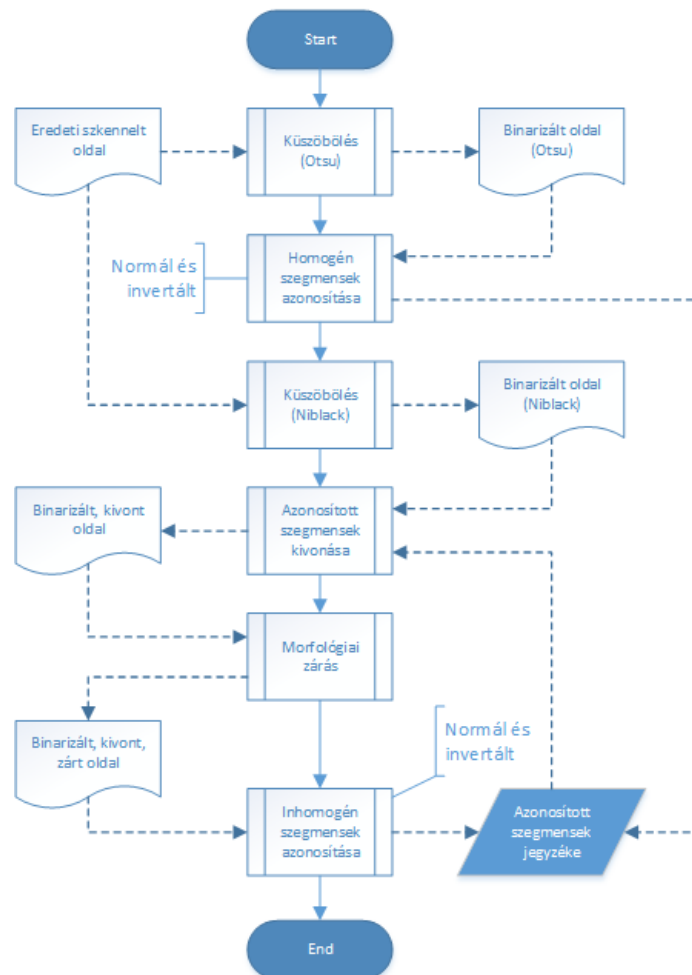
- Invertált inhomogén szövegrész azonosítása

Invertált inhomogén szövegrészben a szöveget alkotó pixelek a környezetnél halványabb színűek és azonos intenzitásúak, míg a környezet sötétebb, és eltérő színű képpontokból áll.

- Színes szövegdozok azonosítása

A dokumentumokban lévő szöveg gyakran színes háttér előtt jelenik meg, ami a szöveghez kiegészítő jelentést társíthat. Ezért fontos a szövegek „alatt” megtalálható szövegdozok azonosítása.

Ahhoz, hogy ezeket a feladatokat az OCR Engine elvégezhesse, egy többszintű szegmentáló eljárás került benne kialakításra, melynek működését a **6. ábra** mutatja.



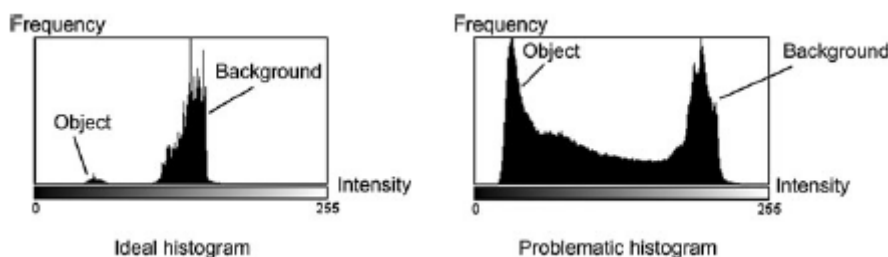
6. ábra - Szegmentálás folyamata

Az eljárás az alábbi lépésekből áll:

Küszöbölés (Otsu)

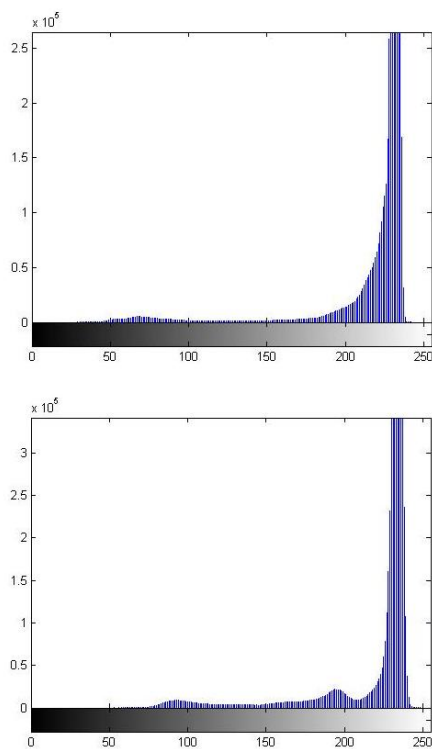
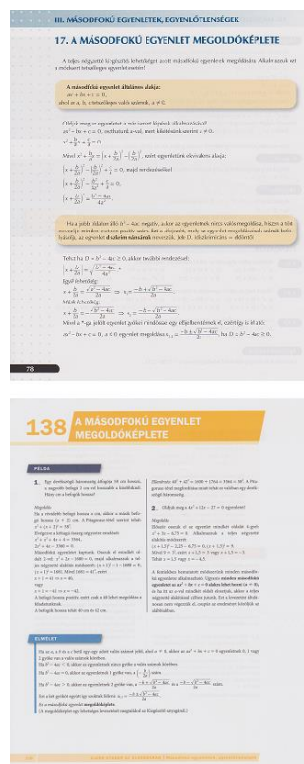
A küszöbölés során célunk a vizsgált objektumok, és a háttér egymástól való egyértelmű megkülönböztetése úgy, hogy az objektumokat alkotó pixelek intenzitását minimálisra, míg a háttér pixeleinek az intenzitását maximálisra állítjuk. [3] Egy OCR alkalmazás esetén az objektumokat az egyes karakterek, az ábrák, és a képek alkotják, míg a háttér maga a papír lesz. A küszöbölés eredménye tehát egy olyan kép, amelyen minden karakter, ábra és kép fekete képpontból áll, míg a papír színe fehér lesz.

Annak eldöntéséhez, hogy mely pixelek tartoznak az egyes objektumokhoz, és melyek a háttérhez, a feldolgozandó képből álló hisztogram kerül felhasználásra. A **7. ábra** két különböző kép hisztogramját tartalmazza. Ideális esetben (Ideal histogram) az objektum, és a háttér egyértelműen elkülöníthető egymástól az egyes intenzitás értékek előfordulási gyakoriságának a vizsgálatával. Problémás esetben (Problematic histogram) azonban nincs egyértelmű határ az objektum és a háttér között, ami az optimális küszöbponthoz megtalálását megnehezíti.



7. ábra - Küszöbölés szempontjából ideális és problémás esetek hisztogramjai [3]

A megfelelő küszöbölési eljárás kiválasztásához tehát meg kell vizsgálnunk, hogy a feldolgozandó tankönyvek szkennelt oldalaiából képzett hisztogramok alapján egyértelműen meghatározható-e az objektumok, és a háttér közötti határ. A **8. ábra** két 10. osztályos matematika tankönyvből szkennelt oldalt, és az azokból generált hisztogramot tartalmaz. A hisztogramokról leolvasható, hogy nincs olyan triviális küszöbérték, amellyel a binarizálás elvégezhető lenne.



8. ábra - Feldolgozandó oldalak, és hisztogramjaik

A megoldást egy olyan algoritmus jelentheti, amely képes automatikusan meghatározni egy, az optimálishoz közeli küszöbértéket.

Az Otsu algoritmus egyike a legjobb küszöbölési eljárásoknak. Az eljárás főleg jó minőségű képek feldolgozása esetén produkál meggyőző eredményt. A rossz minőségű, alacsony kontrasztú, vagy egyenetlen megvilágítású képek esetén az algoritmussal meghatározott küszöbértékkel végzett binarizálás eredménye nem lesz kielégítő. [4] Mivel az OCR modul inputjainak az előállítási módja előre definiált, feltételezhetjük a jó minőségű képek meglétét, tehát az Otsu ebben az esetben alkalmazható.

Maga az Otsu algoritmus egy olyan eljárás, mely meghatározza azt a T küszöbértéket, amivel a binarizálást elvégezve maximális lesz az objektumok és a háttér közötti variancia. Az algoritmus működése során a következő lépéseket az összes lehetséges T küszöbértékre vonatkozóan elvégzi:

1. A háttér és az objektumok pixeleinek gyakoriságának meghatározása, ahol $q_B(T)$ a háttér pixelek gyakorisága, $q_O(T)$ az előtér pixelek gyakorisága, p_i pedig az i -edik szürkeérték gyakorisága a képen (normalizált hisztogram i -edik értéke).

$$q_B(T) = \sum_{i=1}^T p_i \quad (1.1)$$

$$q_O(T) = 1 - q_B(T) \quad (1.2)$$

2. A háttér, az objektumokat és a teljes képet alkotó pixelek középértékének a meghatározása, ahol L a bemeneti kép szürkeárnyalatainak a száma, $\mu_B(T)$ a háttér, $\mu_O(T)$ az objektumok, és μ a teljes kép pixeleinek a középértéke.

$$\mu_B(T) = \frac{\sum_{i=1}^T ip_i}{q_B(T)} \quad (2.1)$$

$$\mu_O(T) = \frac{\sum_{i=T+1}^L ip_i}{q_O(T)} \quad (2.2)$$

$$\mu = \sum_{i=1}^L ip_i \quad (2.3)$$

3. Varianciák számítása, ahol $\sigma_B^2(T)$ a háttér varianciája, $\sigma_O^2(T)$ az objektumok varianciája, $\sigma^2(T)$ a teljes kép varianciája, $\sigma_w^2(T)$ a súlyozott, osztályon belüli variancia, $\sigma_b^2(T)$ pedig az osztályok közötti variancia.

$$\sigma_B^2(T) = \frac{\sum_{i=1}^T (i - \mu_B(T))^2 p_i}{q_B(T)} \quad (3.1)$$

$$\sigma_O^2(T) = \frac{\sum_{i=T+1}^L (i - \mu_O(T))^2 p_i}{q_O(T)} \quad (3.2)$$

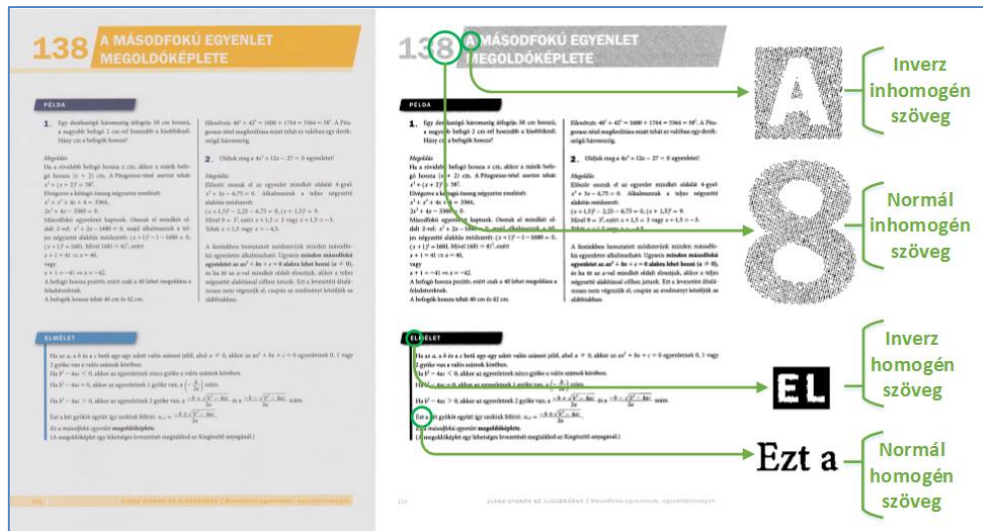
$$\sigma^2(T) = \sum_{i=1}^L (i - \mu)^2 p_i \quad (3.3)$$

$$\sigma_w^2(T) = q_B(T)\sigma_B^2(T) + q_O(T)\sigma_O^2(T) \quad (3.4)$$

$$\sigma_b^2(T) = q_B(T)q_O(T)[\mu_B(T) - \mu_O(T)]^2 \quad (3.5)$$

Az optimális küszöbszám az a T érték lesz, amelynél a $\sigma_w^2(T)$ a minimális, vagy a $\sigma_b^2(T)$ a maximális értéket veszi fel. [5]

A küszöbölés eredményét, és a felhasznált eredeti oldalt a **9. ábra** mutatja. Az ábrán kiemelésre kerültek normál homogén, inverz homogén, normál inhomogén és inverz inhomogén szövegrészek. Látható, hogy az Otsu eljárással elvégzett binarizálás eredményén csak a normál homogén szövegrészek azonosítását lehet azonnal elvégezni. Az inverz homogén, a normál inhomogén és az inverz inhomogén szegmenszek azonosításához további műveletek elvégzésére van szükség.

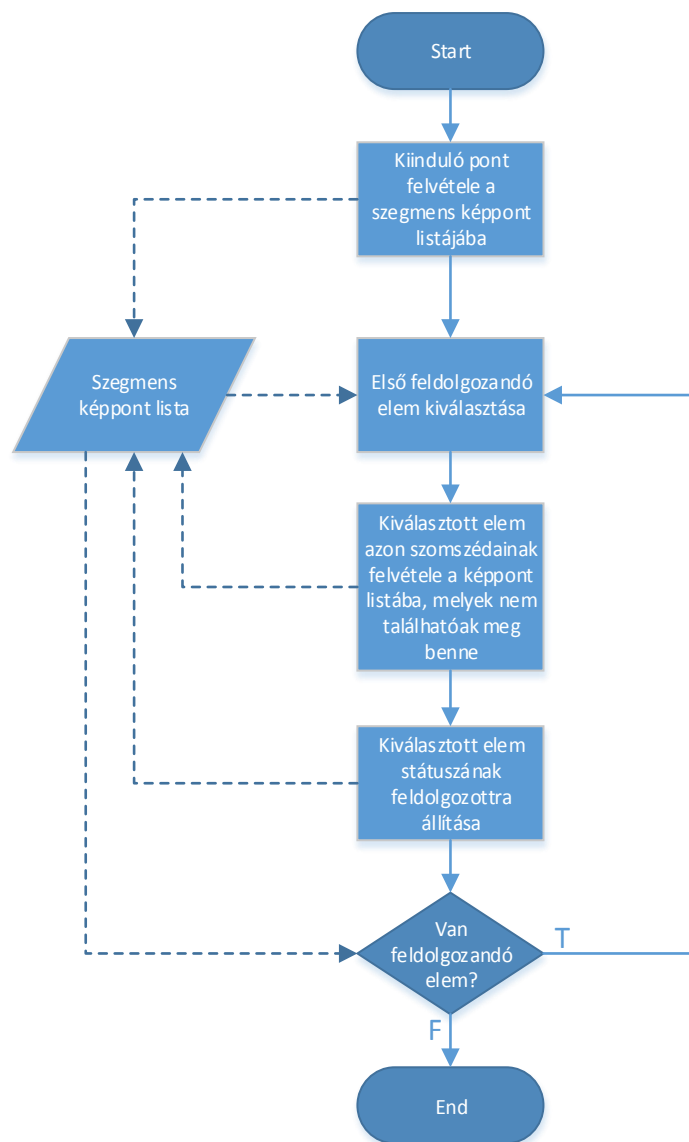


9. ábra - Az Otsu algoritmus felhasználásával elvégzett binarizálás eredménye (Forrás: LUCA Converter)

Normál homogén szegmenszek azonosítása

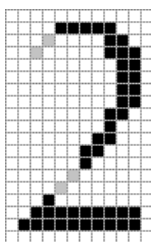
Az Otsu binarizálás eredményeként létrejött kép fájl csak fekete és fehér képpontokat tartalmaz, ahol a fekete képpontok az egyes karakterekhez, a fehérek pedig a háttérhez tartoznak. A szegmenszek azonosítása során csoportosításra kerülnek azok a pixelek, melyek ugyanahhoz a karakterhez tartoznak. Egy csoportba azok a képpontok kerülnek, melyek egy kiválasztott fekete képpontból elérhetőek csak fekete pixelek érintésével.

Az egyes szegmenszeket alkotó pixelek azonosítására a régió alapú szegmentáló eljárások közül a régió növelő (Region Growing) került megvalósításra. Az eljárás működése során képpontokat, vagy alrégiókat csoportosít előre meghatározott kritériumok alapján. Az eljárás működéséhez olyan alappontokra van szükség, amelyekből a régió növelés elvégezhető. Ezek az alappontok az egyes szegmenszek tetszőleges pontjai lehetnek. A régió növelés során ezeknek a képpontoknak a szomszédjait vizsgáljuk meg, majd a szomszédok szomszédjait, és így tovább, amíg el nem fogynak a régióba bevonható szomszédos képpontok. [6] Az eljárás folyamatát a 10. ábra mutatja.



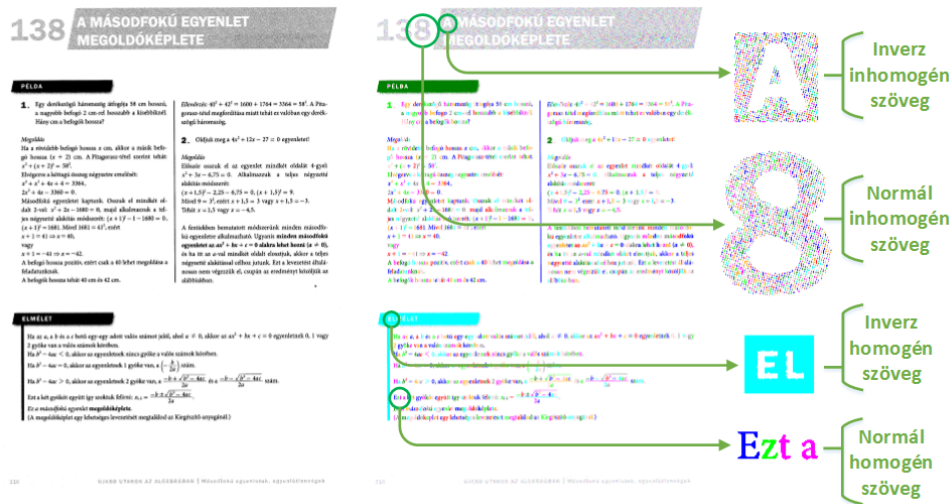
10. ábra - Régió növelő eljárás folyamata

A feldolgozás alatt álló képpont szomszédjainak meghatározásában az adott pixelt körülvevő mind a nyolc képpont részt vesz. Ennek oka az, hogy a kisméretű karaktereket alkotó pixelek gyakran csak a sarkokon érnek össze (11. ábra).



11. ábra - A 8 szomszéd vizsgálatának okai a régiónövelő eljárásban

A régiónövelő eljárás eredményét a **12. ábra** tartalmazza. Az eredményül kapott állományban az egyes szegmensek különböző színekkel lettek megjelölve.



12. ábra - A régiónövelő eljárás alkalmazásával elvégzett szegmentálás eredménye (Forrás: LUCA Converter)

Látható, hogy míg a normál homogén szövegrészek szegmentálása az előzetes elvárásoknak megfelelően került végrehajtásra, addig az inhomogén, és az inverz homogén szövegrészek szegmentálása egyáltalán nem sikerült.

Az inverz homogén szövegrészben a szöveg háttér önálló szegmens lett, míg maga a szöveg háttérként lett azonosítva. Az inhomogén szövegrészeknél az Otsu binarizálás eredményeképpen objektumként azonosított képpontok fizikailag nem érnek össze, ezért a szegmentálásnál minden apróbb képpont csoport önálló szegmensként lett azonosítva. A szegmentálási folyamat további lépései során ezeket a problémákat kell kiküszöbölni.

Inverz homogén szegmensek azonosítása

A rendszer funkcionalitásának megtervezését megelőzően megvizsgált tankönyvekben, a fontosabb szövegrészek kiemelésére gyakran alkalmazott megoldás a sötét színű szövegdobozba világos betűszínnel elhelyezett szöveg volt. Az ily módon kiemelt szövegrészek a binarizált képen fehér színnel fognak megjelenni fekete háttér előtt (9. ábra), ami azt eredményezni, hogy a fekete pixelekből álló szegmensek azonosítására írt eljárás nem fogja azokat megtalálni.

Megoldást jelenthetne a szegmens azonosító eljárás módosítása úgy, hogy az összes fekete színű szegmens azonosítása után végezze el az összes fehér színű szegmens azonosítását is, ez azonban a futási időt jelentősen megnövelné, és nagyszámú hibás szegmensazonosítást eredményezne. Ennek elkerülése érdekében, csak azoknak a sötét szövegdobozoknak a területén kell elvégezni a fehér szegmensek azonosítását, melyeket az első szegmensazonosítási körben sikeresen megtaláltunk. A megvalósítás során azokat

a szegmenseket azonosítjuk szövegdobozként, melyeknek vagy a szélessége, vagy a magassága meghalad egy paraméterként megadható küszöbértéket. A rendszer tesztelése során ez a küszöbérték 50 képpont volt, azonban ez a feldolgozandó könyv függvényében változhat.

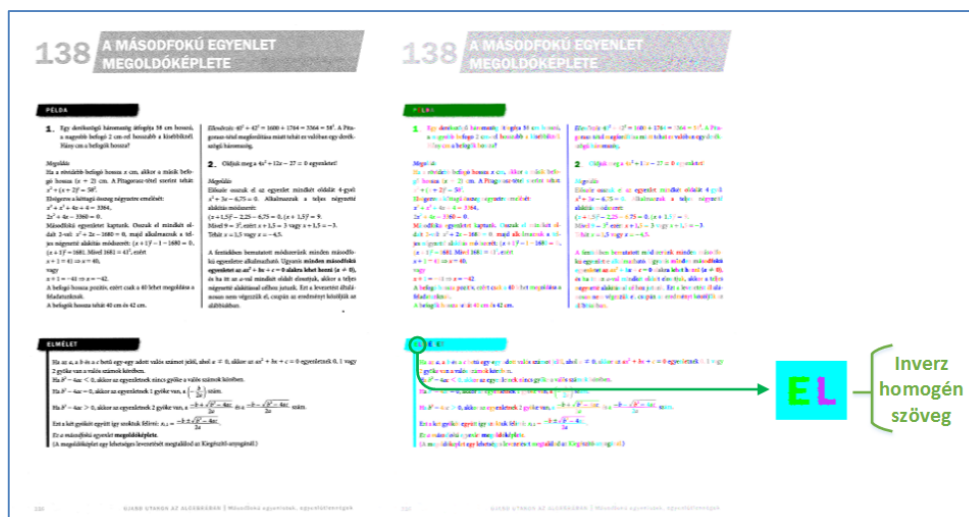
Mivel a szövegdobozok gyakran szabálytalan alakúak, a befoglaló téglalapjukat nem lehet felhasználni a szövegdoboz területén lévő szegmensek meghatározásához. Ennek a problémának a megoldásához a szövegdoboz befoglaló téglalapjára illesztett koordináta-rendszerben kell ábrázolni a szövegdobozt alkotó legalsó és legfelső képpontokat a **13. ábra** jobb oldalán látható módon. Az így létrejött két függvény segítségével eldönthető, hogy az azonosított szegmensek közül melyek vannak teljes terjedelmükkel a szövegdoboz területén. A szövegdobozon kívül eső szegmensek a feldolgozásnak ebben a szakaszában nem vesznek részt.



13. ábra - Nem téglalap alapú szövegdoboz, ábrázolása inverz szegmensek azonosításához

Természetesen ez az eljárás sem garantál tökéletes megoldást a problémára, hiszen így a nagyméretű karaktereket, vagy a sötét tónusú ábrákat is szövegdobozként kezeljük, de mind a futási időnek, mind a tévesen azonosított szegmensek számának a növekedése elhanyagolható mértékű lesz.

Az inverz homogén szövegrész szegmentálásának az eredményét a **14. ábra** tartalmazza.



14. ábra - Az inverz homogén szövegrészek szegmentálásának eredménye (Forrás: LUCA Converter)

Küszöbölés (Niblack)

A szegmentálás eddigi lépései során azonosításra kerültek a normál, és az inverz homogén szegmensek, azonban továbbra sem megoldott az inhomogén szövegrészek kezelése.

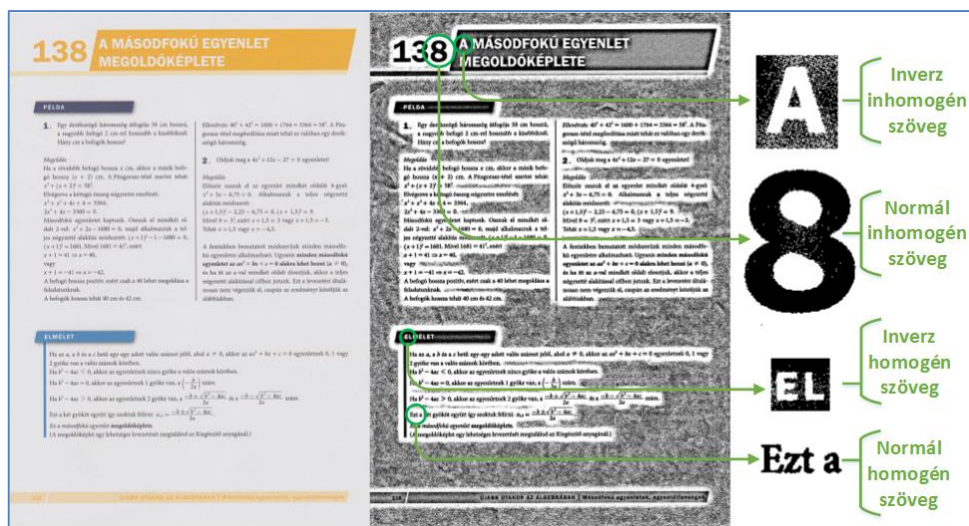
Ahhoz, hogy az inhomogén szegmenseket azonosíthassuk, olyan lokális binarizáló eljárás alkalmazására van szükség, amely az adott képponthoz tartozó küszöbértéket a képpont valamekkora környezetét vizsgálva határozza meg. Az ilyen típusú eljárások közül a legjobb eredményt a Yanowitz Bruckstein és a Niblack metódusok eredményezik. Mivel a Yanowitz Bruckstein mind bonyolultságában, mind a szükséges számítási kapacitásban messze meghaladja a Niblack-et, az OCR Engine-ben ez utóbbi került megvalósításra. [4]

A Niblack eljárás az adott koordináta-hoz tartozó küszöbértéket az alábbi képlettel határozza meg:

$$T(x, y) = m(x, y) + k * s(x, y) \quad (4.1)$$

Ahol $m(x, y)$ a lokális környezet értékeinek az átlaga, az $s(x, y)$ ugyanennek a környezetnek a szórása, a k pedig a szórás figyelembe vételének a mértéke. A környezet méretét úgy kell meghatározni, hogy elég nagy legyen a zajok elnyomásához, ugyanakkor még megőrizze a helyi részleteket. [7]

A Niblack eljárás alkalmazásának eredményét a **15. ábra** tartalmazza. Annak érdekében, hogy az eljárás minden esetben megfelelő hatékonysággal működjön, a paraméterei tetszőlegesen változtathatóak. A mintaoldal feldolgozásakor a környezet mérete 80*80 képpont, a k értéke pedig 0.01 volt.



15. ábra - A Niblack algoritmus felhasználásával elvégzett binarizálás eredménye (Forrás: LUCA Converter)

Az eredményül kapott binarizált képen látható, hogy a normál inhomogén szövegrész teljesen elkülönül a háttértől, így az azonosítása elvégezhető. Az inverz inhomogén szövegrész környezete azonban számos háttérszínű képpontot tartalmaz, így az azonosítása előtt további műveletek elvégzése szükséges.

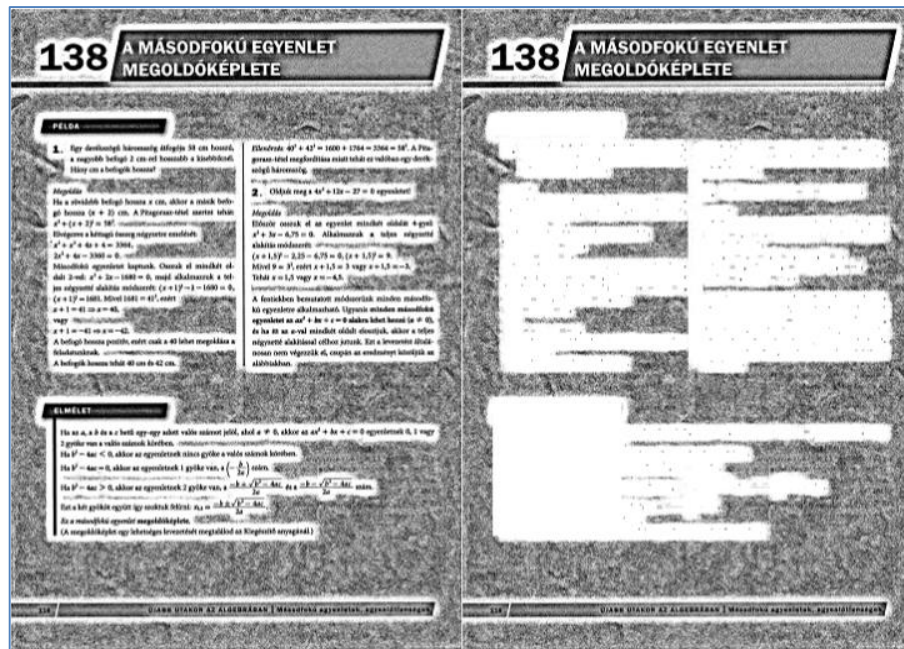
Azonosított szegmensek kivonása

A szegmentálási folyamat során különböző binarizáló eljárásokat alkalmazunk a homogén és az inhomogén szegmensek azonosítására. Ennek oka az, hogy míg az Otsu nagy pontossággal azonosítja a homogén elemeket, a Niblack alkalmasabb az inhomogén szövegrészek azonosítására.

Annak érdekében, hogy a későbbi feldolgozás során az irreleváns adatmennyiséget csökkentsük, a Niblack eljárás alkalmazásának eredményeként kapott küszöbölt képből ki kell vonni azokat a szegmenseket, melyek az Otsu eljárás alkalmazásakor azonosításra kerültek. A kivonás csak abban az esetben végezhető el, ha az adott szegmens a mérete alapján nem az inhomogén szegmenseket alkotó pontthalmaz része.

Ennek megfelelően a kivonás során azoknak a szegmenseknek az alkotó pontjait, melyeknek vagy a magassága, vagy a szélessége meghaladja a 15 képpontot, a Niblack eredményképén maximális intenzitásra állítjuk. A 15 képpontos méret a feldolgozás paramétere, mely tetszőlegesen változtatható.

A kivonás eredményét a 16. ábra mutatja. Az ábra bal oldalán az eredeti Niblack binarizálás, jobb oldalon pedig a kivonás eredménye látható.



16. ábra - A szegmens kivonás eredménye (Forrás: LUCA Converter)

Morfológiai zárás operátor alkalmazása

A Niblack eljárással elvégzett binarizálás az inhomogén szegmenseket nagyjából összefüggővé tette, azonban az inverz inhomogén szegmensek környezetében számos olyan háttérszínű képpont található (15. ábra), amelyek a szegmensek azonosításakor az eredeti körvonalakat torzíthatják. Az inhomogén szegmensek azonosítása előtt tehát ezeket a háttérszínű képpontokat elemi morfológiai műveletek alkalmazásával meg kell szüntetni.

Az elemi morfológiai operátorok közé az erózió, a dilatació, valamint ezek különböző kombinációi tartoznak. [8]

Az X ponthalmaznak Y -al való dilatóltját a következő formula írja le:

$$X \oplus Y = \{a \mid (\hat{Y})_a \cap X \neq \emptyset\} \quad (5.1)$$

Ahol X a feldolgozandó binarizált kép, Y a szerkesztőelem és $\hat{Y}$ a szerkesztőelem saját koordináta rendszerének origójára tükrözve.

A dilatació végrehajtása során a szerkesztőelem koordinátarendszerének origóját eltoljuk a feldolgozandó kép minden elemére és az adott elemet felvesszük a dilatólt halmazba, ha a szerkesztőelemnek legalább egy pontja X -beli ponttal fedésbe kerül. [9]

A dilatáció duálisát, az eróziót a következő formula írja le:

$$X \ominus Y = \{a | (Y)_a \subseteq X\} \quad (5.2)$$

Ahol X a feldolgozandó binarizált kép és Y a szerkesztőelem.

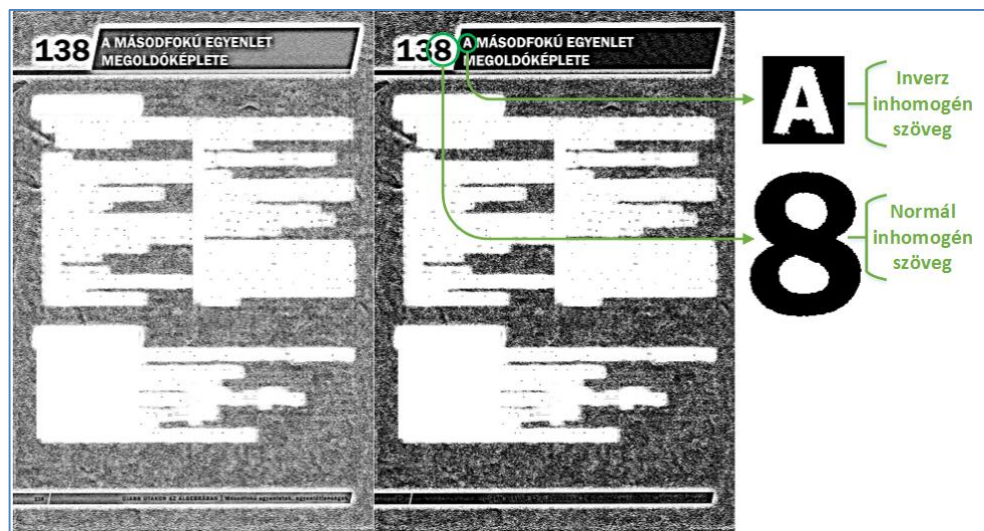
A V halmaz egy pontja akkor kerül be az erodált halmazba, ha a szerkesztőelem koordinátarendszerének origóját az adott pontba eltolva minden egyes Y -beli pont X -belivel kerül fedésbe. [9]

A morfológiai zárás a dilatáció és az erózió műveletének egymás után történő elvégzése:

$$X \cdot Y = (X \oplus Y) \ominus Y \quad (5.3)$$

A morfológiai zárás alkalmas arra, hogy összekösse az egymáshoz közel lévő objektumokat, és kitöltse a felületükön lévő az apróbb lyukakat. Ezen felül a zárás kisimítja az objektumok körvonalait az objektum körvonalán meglévő „keskeny öblök” feltöltésével. [8]

A Niblack binarizálás, és a kivonás eredményeképpen létrejött képen elvégzett morfológiai zárás eredményét a **17. ábra** mutatja. Az ábra bal oldalán látható a szegmenskivonás, jobb oldalon pedig a kivont képen alkalmazott morfológiai zárás eredménye.



17. ábra – A morfológiai zárás eredménye (Forrás: LUCA Converter)

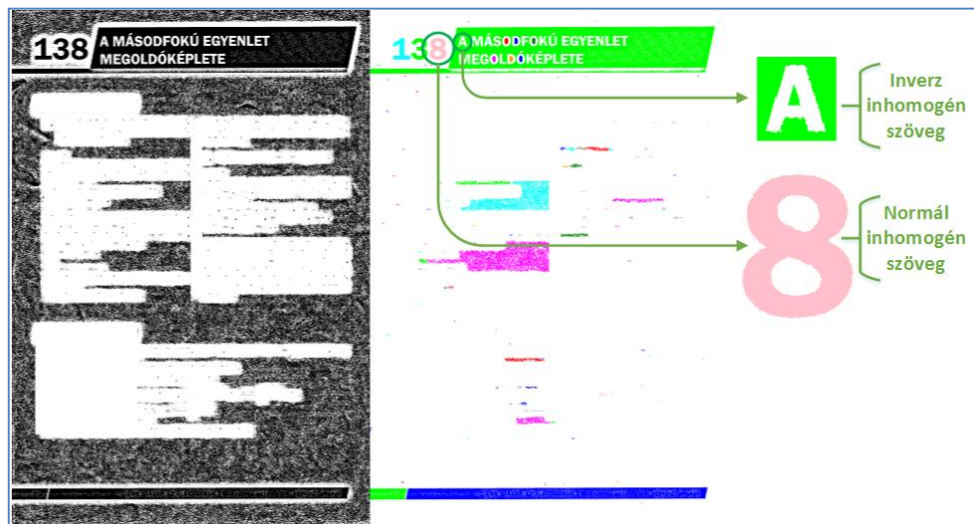
Látható, hogy mind a homogén szegmensek belsejéből, mind az inhomogén szegmensek környezetéből eltűntek a háttérszínű képpontok, így az eredményül kapott kép már alkalmas az inhomogén szegmensek azonosítására.

Normál inhomogén szegmensek azonosítása

A normál inhomogén szegmensek azonosítását a morfológiai zárás eredményeként létrejött binarizált képen lehet elvégezni. Az azonosítás eljárása megegyezik a normál homogén szegmensek azonosításánál leírtakkal. Az eredményül kapott szegmensek közül csak a tartalmilag relevánsak megtartandók, ezért elvetésre kerülnek azok, melyeknek mind a szélessége, mind a magassága egy adott küszöbértéknél kisebb. Elvetésre kerülnek továbbá azok a szegmensek is, melyeknél a befoglaló téglalap területe a feldolgozás alatt álló oldal területének adott százalékát meghaladja. Mind a minimális méret, mind a maximális területarány a feldolgozás paramétere, így értékük szabadon megválasztható.

A mintaoldal feldolgozása során a minimális méret 10 pixel, a maximális területarány pedig 50% volt.

A normál inhomogén szegmensek azonosításának eredményét a **18. ábra** tartalmazza. Az ábra bal oldalán a morfológiai zárás, jobb oldalán pedig a szegmens azonosítás eredménye látható.

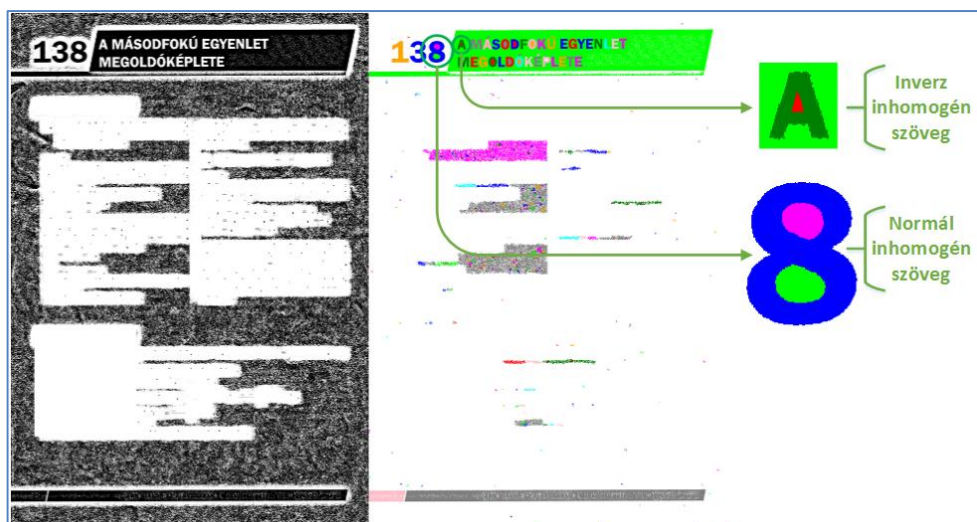


18. ábra - Normál inhomogén szegmensek azonosítása (Forrás: LUCA Converter)

Inverz inhomogén szegmensek azonosítása

Az inverz inhomogén szegmensek azonosítását a morfológiai zárás eredményeként létrejött binarizált képen végezzük el. Az azonosítás eljárása megegyezik az inverz homogén szegmensek azonosításánál leírtakkal.

Az inverz inhomogén szegmensek azonosításának eredményét a **19. ábra** tartalmazza.



19. ábra - Inverz inhomogén szegmensek azonosítása (Forrás: LUCA Converter)

Szimbólumok azonosítása

A szimbólumok azonosítása során az egyes szegmenseket alkotó pixelhalmazok vizsgálatával kerül meghatározásra az azokban kódolt karakter. A szimbólum kódján felül ebben a lépésben a betűtípusának, és pontosan kifejezett méretének a meghatározása is megtörténik.

Kötött karakterek azonosítása

Azokat a karaktereket nevezzük kötöttnek melyek alakja egy adott betűtípuson és méreten belül állandó. Jellemzően ezek a karakterek a betűket és számokat jelölik.

A kötött karakterek azonosítására a rendszer fejlesztése során három különböző megoldás került implementálásra, melyek közül a sablonillesztéses eljárás maradt benn a rendszerben. A másik két megoldás neurális hálózat alkalmazásával végezte el a karakterek azonosítását, melyek rövid ismertetését a melléklet (8.1) tartalmazza.

Sablonillesztéses eljárás alkalmazása

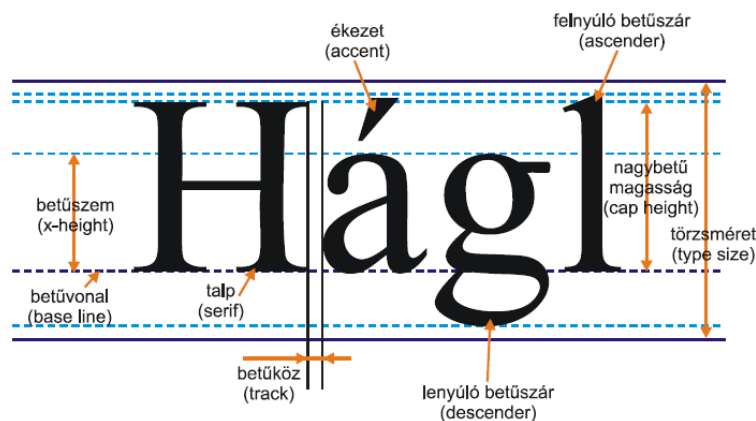
A sablonillesztéses eljárás alkalmazásának lényege, hogy az azonosítandó karakterek képét tartalmazó sablonok kerülnek létrehozásra, amelyekkel a vizsgálandó szegmensset összehasonlítva meghatározható a szegmensben kódolt karakter azonosítóján felül a betűtípusa, és a betűmérete is.

Az egyszerűbb alkalmazhatóság érdekében betűtípusonként és betűméretenként lettek létrehozva olyan sabloncsoportok, melyekben az egyes sablonok tartalmazzák az azonosítható karakterek jellemzőit. Maguk a sablonok olyan mátrixok, melyek a -10 és +10 intervallumban tartalmazhatnak egész számokat.

Sablonméret meghatározása

A sabloncsoportok létrehozásakor meg kell adni, hogy az adott csoport melyik betűtípus, mekkora méretű karaktereit tartalmazza. Sabloncsoporton belül az összes sablon azonos méretű. A sablonméret a sabloncsoporthoz megadott betűméret alapján kerül kiszámításra.

Egy adott betűtípus mérete a törzsméretével adható meg. Se a legmagasabbra felnyúló, se a legmélyebbre lelógó szár nem lóghat ki a törzsméret által meghatározott tartományból (20. ábra) [10].



20. ábra - Nyomdabetűkhöz tartozó fogalmak [10]

A pontokban megadott törzsméretből az egyes sablonok mérete az alábbiak szerint számolható:

- 1 inch = 25,4 mm
- 300 DPI (dots per inch) felbontás mellett tehát 1 pont = 0.085 mm
- Egy szabványos DTP-pont (desktop publishing point) mérete 0,353 mm ami a fentiek alapján 4,167 pontot jelent.
- Ez azt jelenti, hogy pl. 12-es betűméretnél a sablonmátrix magassága és szélessége egyaránt $12 \cdot 4,167 = 50$ lesz.

Sablonok betanítása

Ahhoz, hogy a létrehozott sabloncsoportok karakterazonosításra felhasználhatóak legyenek, el kell végezni a betanításukat. A tanítási folyamat során beállításra kerülnek az egyes sablonok értékei a tanításhoz kiválasztott mintaszegmensek adataival.

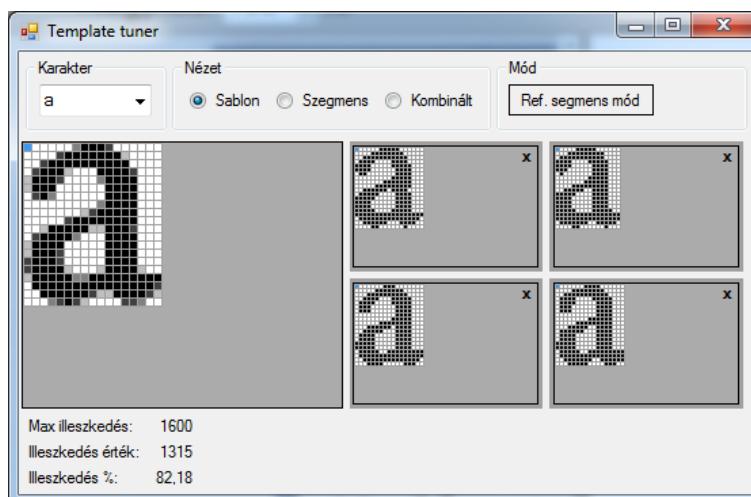
Ha egy szkennelt dokumentum azonos típusú karaktereit közelről megvizsgáljuk, azt fogjuk tapasztalni, hogy a karakterek képe nem teljesen azonos. Az eltérést okozhatják

nyomdai hibák, szkennelési hibák, esetleg a képet terhelő zajok. Ezek az eltérések jellemzően a karakterek határain jelennek meg. Ahhoz, hogy a karakterfelismerés megfelelő hatékonysággal működjön, ezeknek a hibáknak a hatásait figyelembe kell venni.

A tanításhoz három-négy olyan szegmens kerül kiválasztásra a feldolgozandó dokumentumból, amely a tanítandó sablon karakterét tartalmazza. A szegmenseket egyenként kell hozzáadni a sablonhoz. Az első szegmens felvételekor, a szegmens súlypontja kerül a sablon középpontjára. A további szegmensek felvételekor a súlypont úgy lesz elhelyezve a sablonon, hogy az a lehető legnagyobb fedést biztosítsa a korábban a sablonhoz adott szegmensekkel. Ennek érdekében a fedés mértéke kiszámításra kerül úgy, hogy a szegmens súlypontja a sablon középpontjában van, és úgy is, hogy azt a sablon középpontjának közvetlen környezetébe helyezzük. Végül a legnagyobb fedést biztosító pozícióban kerül a szegmens a sablonra.

A tanítás során a sablon azon elemeit, melyek a szegmensben fehérek voltak egyel csökkenteni, a többit pedig egyel növelni kell. A tanítási folyamat utolsó lépéseként a sablon értékei normalizálásra kerülnek a -10, +10 intervallumban.

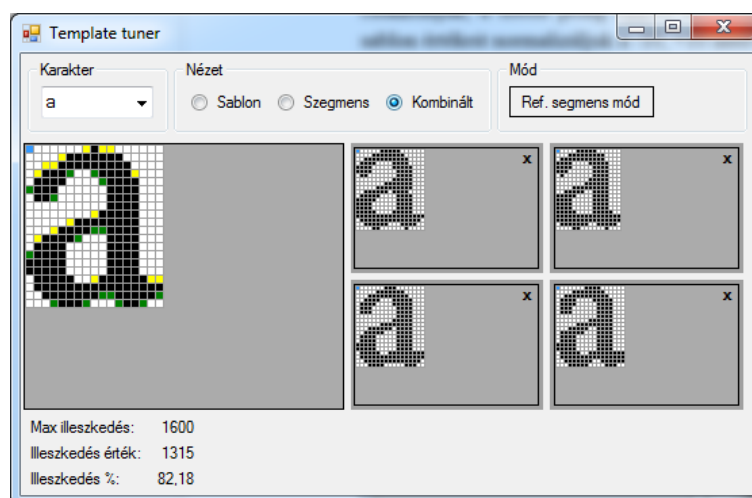
A végeredmény egy olyan mátrix lesz, melynek a -10-et tartalmazó elemei az összes tanítómintán háttérpixel, a +10-es értékek az összes tanítómintán karakter pixel, voltak. A két szélsőérték közé eső számok azt jelzik, hogy a tanítómintákon nem azonos típusú pixel szerepel az adott pozíción. A **21. ábra** az OCR Engine szegmenstanító képernyőjét mutatja. Az ábra jobb oldalán a tanításhoz kiválasztott szegmensek, a balon pedig a tanítás eredménye látható. A bal oldali sablonban fekete színnel vannak jelölve a +10-es szegmensértékek, és fehérrel a -10-esek. A -10 és +10 közötti értékek a szürke különböző árnyalataiban jelennek meg.



21. ábra - Az OCR Engine sablon tanító felülete (Forrás: LUCA Converter)

A tanítás során lehetőséget kell biztosítani a tanítás eredményességének ellenőrzésére. A **22. ábra** mutatja azt a képernyőt, amelyen egy szegmens ráilleszthető a betanított sablonra. A képernyő bal oldalán egymás fölé helyezve látható a vizsgált sablon, és az ellenőrzéshez kiválasztott szegmens. A sárgával jelölt területek részét képezik a sablonnak, de nem tartoznak a vizsgált szegmenshez. A zöld színű területek részét képezik a vizsgált szegmensnek, de nem tartoznak a vizsgált sablonhoz. A fekete képpontok mind a sablonban, mind a szegmensben a karakterhez, a fehérek pedig a háttérhez tartoznak.

A képernyő alsó részén látható az adott sablonnal elérhető maximális, és a vizsgált szegmessel ténylegesen elért illeszkedés mértéke.



22. ábra - Az OCR Engine sablon ellenőrző felülete (Forrás: LUCA Converter)

Karakterek azonosítása

A karakterek azonosítása során a vizsgált szegmens az összes betanított sablonnal összehasonlításra kerül. A vizsgálat végén a szegmensre legjobban hasonlító sablon karakterét kell a szegmenshez rendelni. Azt, hogy az adott karakter melyik betűtípushoz tartozik, mekkora méretű (pontban), és milyen egyéb jellemzőkkel bír, a legjobban illeszkedő sablont tartalmazó sabloncsomag határozza meg.

A sablon és a szegmens összehasonlításakor a szegmens illesztésre kerül a vizsgált sablonra úgy, hogy a súlypontja a sablon közepére essen. Ezt követően meghatározásra kerül az a legkisebb téglalap, melyben az egymás fölött lévő szegmens és sablon összes olyan pontja benne van, amely nem a háttér része. A vizsgálat során csak az ebbe a téglalapba beleeső elemekre kell elvégezni az összehasonlítást.

Az összehasonlítás során elemről elemre vizsgáljuk a megfelelő sablon és szegmens értékeket.

Amennyiben az egymás fölötti szegmens és sablon elemekre igaz az, hogy

- a sablon adott értéke > 0 és a szegmens adott eleme a karakter része vagy
- a sablon adott értéke < 0 és a szegmens adott eleme a háttér része,

akkor a megfelelőségi mutatót növeljük a sablon adott pozíción lévő értékével. Ellenkező esetben ezt a megfelelőségi mutatót ugyanezzel az értékkel csökkentjük.

Látható, hogy a fentiek alapján kiszámított megfelelőségi mutató nem alkalmas arra, hogy különböző méretű karakterek találati pontosságát ezzel összehasonlíthassuk, ezért utolsó lépésként ki kell számolni, hogy a mutató hány százalékát teszi ki a tökéletes találat esetén kapott értéknek.

Egy sablon és egy szegmens összehasonlításának eredménye tehát azt fogja megadni, hogy azok hány százalékban illeszkednek egymásra.

Az eljárás tesztelése során számos esetben a szemmel láthatóan nagyon hasonló szegmensek és sablonok összehasonlításának eredménye elmaradt a várt értéktől. Az egyes esetek vizsgálata során megállapítást nyert, hogy a vizsgált szegmens súlypontját a sablon közepére helyezve nem volt maximális a szegmens és a sablon fedése. A jelenséget az okozza, hogy a szegmens középpontjától távolabb elhelyezkedő pixelhiba a súlypontot az optimális helyzetből kimozdíthatja, ami rontja az összehasonlítás eredményét. Mivel ez az elmozdulás a vizsgált esetekben kivétel nélkül egy pixel volt, az összehasonlító eljárás módosítva lett úgy, hogy a szegmens súlypontját nem csak a sablon közepére, hanem annak egy pixeles környezetében mindenhova elhelyezi, és így is végrehajtja az összehasonlítást. Az így elvégzett kilenc vizsgálat eredményéből a legnagyobb érték tekinthető végeredménynek.

Mivel ez a megoldás a vizsgálat idejét gyakorlatilag kilencszerezésre növeli, csökkenteni kell azoknak az összehasonlításoknak a számát, melyek biztosan nem hoznak pozitív eredményt. Ennek megfelelően egy szegmens egy sablonnal csak akkor kerül összehasonlításra, ha a szegmens pixelszáma, és a sablon pozitív értékeinek a száma nem tér el 15%-nál nagyobb mértékben.

Ligatúrák azonosítása

A ligatúra két vagy több betű egybeolvasztásával keletkezik. Lényege, hogy bizonyos betűpárokat nem két különálló karakterrel kell szedni, hanem egygyel. A magyar nyelvben ligatúrát az fi, ff, fj, fl, ffi ffl betűkapcsolatoknál szoktak alkalmazni. [10]

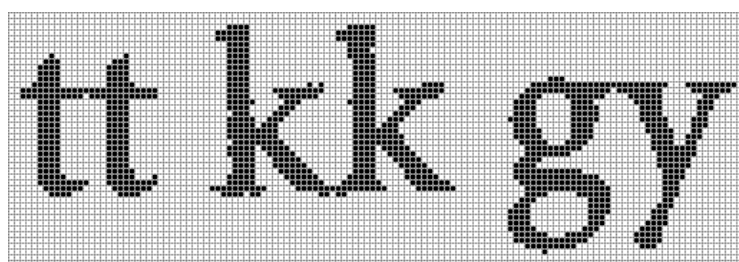
A ligatúrák megjelenési formája betűtípusonként eltérő lehet. A **23. ábra** jellemző fi és fl ligatúrákat tartalmaz. Az ábra felső sora a betűkapcsolatot különálló betűvel, az alsó pedig ligatúrával mutatja.



23. ábra - Ligatúrák formavilága különböző betűtípusoknál. [11]

Látható, hogy a ligatúrák ugyanolyan kötött karakterek, mint az egyedül álló betűk. Ahhoz, hogy ezek azonosíthatóak legyenek, az egyes betűtípusok sablonkészletét kell kiegészíteni a ligatúra karakterekkel. A ligatúrák képéhez tehát nem egy, hanem több karakter van hozzárendelve.

Ez a gyakorlat nem csak a ligatúrák, hanem az egyéb karakter összemosódások kezelést is megoldja. Számos betűtípusnál tapasztalható, hogy bizonyos karakter párok azonos szegmensbe kerülnek, aminek oka az egymáshoz közel lévő karakterek szárainak összeérése. Talpas betűtípusok esetén ez különösen gyakran előfordul a tt, kk és gy betűkombinációknál (24. ábra).



24. ábra - Összefolyó karakterpárok Times New Roman betűtípusnál (Forrás: LUCA Converter)

A ligatúrákhoz hasonlóan ezeket a karakterpárokat is fel lehet venni az azonosító sablonok közé, amivel elkerülhető a téves azonosításuk.

Egyszerű szimbólumok azonosítása

Egyszerűek azok a szimbólumok, melyek azonosíthatóak alapvető jellemzőik alapján. Ilyenek a vízszintes és függőleges vonalak, és a kisméretű karakterekhez tartozó pontok.

Függőleges vonal

Függőleges vonalat egy tankönyvben vagy a különböző jellegű szövegrészek elválasztásához, vagy matematika könyv esetén pl. az abszolút érték jelölésére használnak.

Az **1. táblázat** tartalmazza néhány olyan szimbólum oldalarányát, melyet vagy függőleges vonallal jelölünk, vagy a befoglaló téglalapjának a magassága jelentősen nagyobb a szélességénél.

Szimbólum	Magasság (pixel)	Szélesség (pixel)	Oldal arány
Abszolút érték	34	3	11,33
	38	3	12,66
	40	3	13,33
Tartalom elválasztó	2418	6	403
Zárójel	117	18	6,5
Kapcsos zárójel	116	17	6,82
Talpas (Times New Roman) 'l' betű	30	10	3
Talp nélküli (Arial Narrow) 'l' betű	33	5	6,6
	32	4	8

1. táblázat - Nagy oldalarányú (magas) szimbólumok (forrás: LUCA Converter)

Míg a vizsgálathoz felhasznált abszolút érték jel kiválasztásakor az volt a szempont, hogy az minél kisebb legyen, a zárójelnél és a kapcsos zárójelnél olyat választottam, amelynek a magassága sokkal nagyobb, mint az összehasonlításhoz kiválasztott abszolút érték jel magassága.

Ennek ellenére látható, hogy azoknál a szegmenseknél, melyeket függőleges vonalként kell felismerni, az oldalarány tíz fölötti, míg a többi esetben ez az arány tíz alatt marad, ezzel a küszöbvel tehát megkülönböztethetők a függőleges vonalak a hasonló szimbólumoktól. Problémát egyedül a talp nélküli betűtípusok kis 'l' karaktere jelenthet, mivel itt a mért nyolcas arány közel van a küszöbértékhez.

Annak érdekében, hogy az egyszerű szimbólumok azonosítása különböző kiadványok feldolgozásakor egyaránt jól működjön, a függőleges vonalat az egyéb karakterektől elválasztó küszöbérték külső paraméterként beállítható.

Vízszintes vonal

A vízszintes vonalak azonosítása elvileg sokkal könnyebb a függőlegesek azonosításánál, mivel nincs olyan karakter, amellyel a vízszintes vonal összetéveszthető lenne.

A **2. táblázat** tartalmazza néhány olyan szimbólum oldalarányát, melyet vízszintes vonallal jelölünk.

Szimbólum	Magasság (pixel)	Szélesség (pixel)	Oldal arány
Törtvonal	4	161	40,25
Kivonás	2	22	11
	3	21	7
Egyenlőségjel egyik vonala	4	22	5,5
	3	22	7,33
Kötőjel	3	11	3,66
	3	11	3,66

2. táblázat - Nagy oldalarányú (széles) szimbólumok (forrás: LUCA Converter)

A vízszintes vonallal jelölt szimbólumok közül a legrövidebb a kötőjel volt, melynek az oldalaránya nem éri el a 4-et. Ahhoz, hogy a kötőjel biztosan vízszintes vonalként legyen azonosítva, maximum 3-as küszöbértéket lehet alkalmazni az oldalarány vizsgálatnál. Ez az érték viszont már annyira alacsony, hogy alkalmazásával könnyen olyan szegmensek is vízszintes vonalként lennének azonosítva, amelyek valójában nem azok (pl.: Széles keretet egy szövegrész körül).

Ennek a hibának a kiküszöböléséhez vizsgálni kell azt is, hogy az azonosítandó szegmens milyen arányban tartalmaz előtér és háttérpixeleket, valamint hogy mekkora a tényleges mérete. Amennyiben egy szegmens oldalaránya nagyobb, mint 3, a magassága nem éri el a 10 pixelt, valamint az előtér pixelek aránya a szegmensben meghaladja a 70%-ot, az vízszintes vonalként azonosítható.

A függőleges vonalak azonosításánál leírtakhoz hasonlóan a vizsgálat küszöbszámai ebben az esetben is szabadon megválasztható külső paraméterek.

Pont

A dokumentumokban lévő pontok nagysága a betűmérettel növekszik. Automatikusan pontnak veszünk minden olyan szegmenset, amelynek a mérete, és az előtérpixel-háttérpixel aránya a **3. táblázatban** leírtaknak megfelel.

Magasság (pixel)	Szélesség (pixel)	Előtér-háttér arány (legalább)
2	2	75
2	3	75
3	2	75
3	3	75

3. táblázat - Pontként azonosítható szegmensek jellemzői

Ékezetes karakterek azonosítása

A kötött karakterek és az egyszerű szimbólumok azonosítása után minden információ adott az ékezetes karakterek azonosításához, melynek során cél a magyar nyelv ékezetes betűinek, és más nyelvek jelkészletéhez tartozó, de magyar nyelvű szövegben is gyakran előforduló ékezetes kis és nagybetűk felismerése (á é í ó ö ú ü ù ä Á É Í Ó Ő Ő Ú Ü Ű Ä).

Az ékezetes karakterek azonosítása az alap karakter fölötti térközbe eső, ékezetként vagy pontként azonosított szegmensek száma alapján történik. Az ékezeteknek vagy pontoknak a vizsgált térközben elfoglalt pozíciója a vizsgálat során nem játszik szerepet. Csak azoknak a karaktereknek az azonosítása egészül ki az ékezetek vizsgálatával, melyek jelentését az ékezetek, vagy pontok megváltoztathatják (a e i o u A E I O U).

A vizsgált térköz szélességét az alap karakter befoglaló téglalapjának szélessége, magasságát pedig az alap karakter törzsméretének (lásd: **20. ábra**) 33%-a adja. A befoglaló téglalapot, és az ékezetek szempontjából releváns térrészt ékezetes karaktereknél a **25. ábra** mutatja.



25. ábra - Néhány ékezetes karakter a befoglaló téglalap és az ékezet vizsgálatban szereplő térköz megjelölésével

3.1.1.2 PDF Parser

A PDF Parser modul létrehozását az elektronikus tananyagok elterjedése indokolta. Számos esetben a papír alapú tankönyvek mellett (helyett) hozzáférhetőek elektronikus anyagok is. Ezek az elektronikus anyagok az esetek nagy részében PDF formátumúak, melyek a tananyagot szöveges információ és képek kombinációjaként tartalmazzák. A PDF nyílt szabványú fájlformátum, amely alkalmas eszköz és felbontás független módon szöveges és képi információk leírására. Maga a PDF állomány négy különböző típusú

részből áll, melyek a fejléc (file header), a törzs (file body), a kereszthivatkozási táblázat (cross-reference table), és a lezáró rész (trailer).

- Fejléc (header)

A PDF fájl fejléce a PDF verziószámát tartalmazza.

- Törzs (body)

A törzs a PDF objektumok felsorolását tartalmazza. Minden egyes objektumhoz tartozik egy egyedi azonosító, és egy generációs szám. Ez a két szám egyértelműen azonosítja az objektumokat a fájlban belül, így azok a fájl bármely pontjáról hivatkozhatók.

- Kereszthivatkozási táblázat (cross-reference table)

A kereszthivatkozási táblázat tartalmazza a törzsben leírt objektumok első bájtyjának indexét (offset cím).

- Lezáró rész (trailer)

A lezáró rész speciális név-érték listát tartalmaz, ahol sok egyéb információ mellett megadásra kerül pl. a PDF fájl fájlazonosítója, és a PDF logikai felépítését leíró objektumok gyökérelemének offset címe.

A PDF Parser modulba a PDF fájlok tartalmi és formai jellemzőinek a kinyeréséhez egy nyílt forráskódú, szabadon felhasználható .NET library, az iTextSharp került beépítésre. Ez az eszköz lehetővé teszi a PDF fájlok feldolgozását a formátum felépítésének ismerete nélkül.

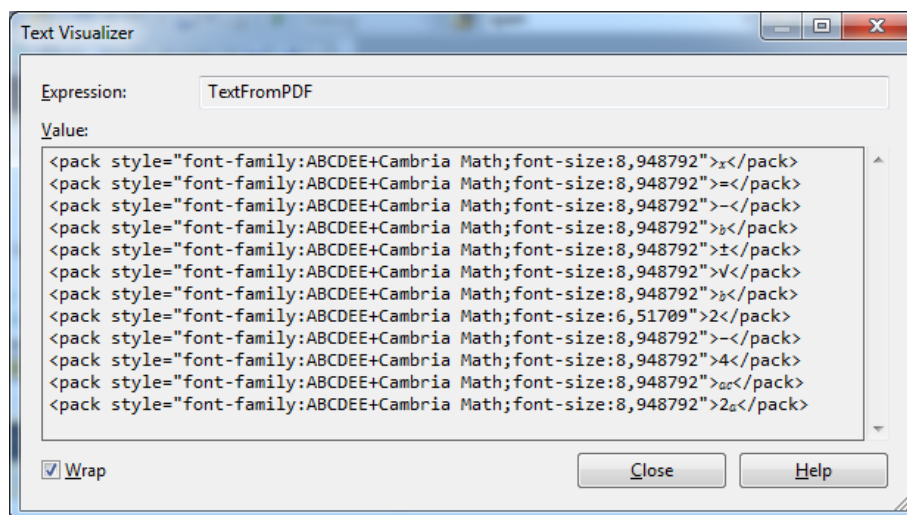
A feldolgozás célja az, hogy a PDF állományból az OCR Engine outputjával megegyező struktúra jöjjön létre annak érdekében, hogy a Parser alrendszer az input állomány formátumától függetlenül dolgozhasson. Ennek a megoldásnak a nyilvánvaló hátránya az, hogy a PDF fájl strukturális felépítését figyelmen kívül hagyja, így a forrásfájlban már eleve csoportosított szöveget is önálló karakterekként kezeli. A továbbiakban a Parser alrendszer feladata lesz ezeknek az önálló karaktereknek a szavakba, sorokba és bekezdésekbe szervezése.

A PDF állományok feldolgozása két lépésben kerül végrehajtásra. Első körben a karakterek kerülnek kinyerésre az input állományból, második körben pedig az állományba beágyazott képek feldolgozása hajtódik végre.

Karakterek azonosítása

A PDF állományban található karakterek azonosítása az iTextSharp library PDFReader osztályával történik. A PDFReader osztály feladata az egyes oldalakon található szövegrészek meghatározása a szövegrész pozíciójával, a betű típusával és méretével együtt.

A PDFReader az állomány feldolgozásakor az összefüggő karaktereket csomagokban adja vissza. Az egyes csomagok tartalmazzák a karaktereket, és a karakterek jellemzőit is. [12] A **26. ábra** a másodfokú egyenlet megoldó képletét tartalmazó PDF állományból kivont karaktereket tartalmazza. Jól látható, hogy az azonos betűtípusú, összefüggő karakterek azonos csomagba kerültek (pl.: 2a, ac).



26. ábra - Másodfokú egyenlet megoldó képletét tartalmazó PDF karakter csomagjai

Ahhoz, hogy a Parser Engine számára megfelelő input állhasson elő a PDF feldolgozás során, a karakterek betűtípusán és méretén kívül a pozíciójukat is meg kell határozni. Bár a karakter csoport nem tartalmazza az egyes karakterek pozícióit, a különböző karakterek szélessége a betűtípus és a méret alapján lekérdezhető. Ennek a szélességnek, és a karakter csoport befoglaló téglalap bal felső pontjának ismeretében az egyes karakterek pozíciója már számolható.

A PDF Parser tervezésekor megvizsgált elektronikus tankönyv kivonatban (Juhász István, Orosz Gyula, Paróczay József, Szászné Dr. Simon Judit MATEMATIKA 10. - Az érthető matematika⁵) alkalmazott megoldás a törtvonal létrehozására, a számlálóban található karakterek aláhúzása. Ennek a megoldásnak a kezeléséhez a PDF Parser outputjába az egyes karakterek szegmensein felül egy vízszintes vonalként azonosított szegmenset is létre kell hozni, melynek szélessége megegyezik a szöveg befoglaló téglalapjának szélességével, pozíciója pedig a befoglaló téglalap bal alsó pontjára esik.

Beágyazott képben lévő szövegrész azonosítása

Abban az esetben, ha egy PDF állomány valamilyen más formátumú (pl.: DOCX) dokumentum átkonvertálásával jön létre, opcionálisan választható lehetőség a PDF-be

⁵ Forrás: <http://www.ntk.hu/kozepiskola/tananyagok/kiadvanyok/matematika>

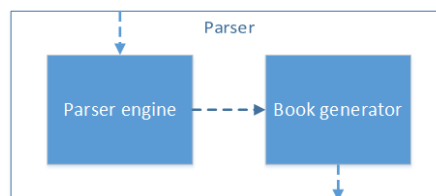
nem beágyazható karakterek bitképként való megjelenítése. Ilyenkor a létrehozott PDF állományban az adott szövegrész beágyazott képként fog megjelenni, aminek feldolgozását a karakterek azonosításánál leírtak szerint nem lehet elvégezni.

A beágyazott képen lévő szövegrész azonosításához tehát először meg kell találni a PDF állomány beágyazott képeit. A képek megkeresése a karakterek azonosításánál leírtakhoz hasonlóan az iTextSharp library PDFReader osztályával történik azzal a különbséggel, hogy most az IMAGE típusú STREAM összetevőket keressük a PDF struktúrában. [12]

Az egyes képek feldolgozásához a Preprocessor alrendszer OCR Engine modulja kerül felhasználásra, melynek inputja lehet egy vagy több képfájl, outputja pedig az azonosított szegmensek halmaza. A használat során figyelembe kell venni, hogy az egyes szegmensek pozíciója a képfájlon belüli pozíció lesz, tehát a PDF Parser output előállításánál ezeket az értékeket korrigálni kell a kép dokumentum oldalon belül elfoglalt pozíciójával.

3.1.2 Parser

A Parser alrendszer a Parser engine és a Book generator modulokból áll. A Parser engine feladata az azonosított szegmensállomány adatai alapján a dokumentum struktúrájának azonosítása. A Book generator a Parser engine által szolgáltatott adatok alapján létrehozza a LUC kiterjesztésű könyv fájlt. A Parser alrendszer felépítését a **27. ábra** tartalmazza.



27. ábra - A LUCA Converter értelmezőjének a felépítése

3.1.2.1 Parser engine

A Parser engine több különböző szempontot figyelembe véve csoportokba szervezi az egyes szegmenseket. A csoportok kialakítása során mind a formai jellemzők, mind az elhelyezkedés szerepet játszik. A csoportosításnak négy szintje került kialakításra.

- Az első szint a bekezdéseké. Azok a szegmensek kerülnek egy bekezdés csoportba, melyek az oldalon elfoglalt pozíciójuk alapján összefüggő szövegrészt alkotnak. A bekezdés csoport elemeit alkotó szegmensek sorokba szervezve vannak a bekezdésen belül nyilván tartva.

- A bekezdések sorokra vannak bontva. Azok a szegmensek kerülnek egy sor csoportba, melyek az oldalon elfoglalt pozíciójuk alapján a bekezdésen belül különálló sort alkotnak.
- A sor csoportok elemeit szó csoportok alkotják, melyekbe értelem szerűen az azonos szavakat alkotó szegmensek kerülnek.
- A negyedik szint a hasábok, vagy bekezdéscsoportok szintje. Ennek a szintnek a feladata a hasábokra tördelt dokumentumok megfelelő sorrendben történő feldolgozásának a biztosítása. A hasábokat alkotó szegmensek bekezdésekbe szervezve vannak nyilvántartva.

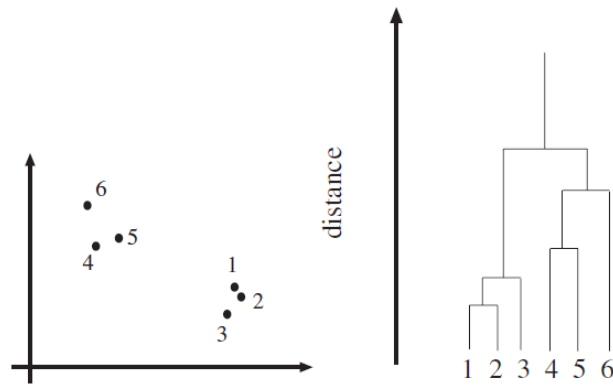
A három csoportosítási szint azonosításának alapját a térközök hierarchiája biztosítja, mely a tipográfia egyik alapvetése. Eszerint a tipográfiában a térközök ugyanolyan fontosak, mint a betűk, és létezik egy alapvető hierarchiájuk. A legkisebb a betűköz, nagyobb a szóköz, még nagyobb a sorköz, annál is nagyobb a hasábköz, és a legnagyobb a margó. [11] Ez a struktúra amellet, hogy biztosítja a dokumentum kényelmes olvashatóságát elősegíti a bekezdések, sorok, szavak és hasábok azonosítását is.

Bekezdések azonosítása

A bekezdések azonosításának folyamata során, az oldalon klaszterek kerülnek kialakításra. A klaszterek száma megegyezik a feldolgozás alatt lévő oldalon található bekezdések számával. Az egyes klaszterekbe az adott bekezdést alkotó szegmensek kerülnek.

Az egyszerű klaszterező eljárásnak két változata létezik, az egyik a megosztó, a másik az összevonó algoritmus. Az összevonó algoritmus kiinduló állapotában az összes szegmens önálló klasztert képez, amelyek az eljárás során rekurzívan összevonásra kerülnek. Az összevonás addig folytatódik, míg létre nem jön az összes szegmenseket tartalmazó, a továbbiakban már nem összevonható klaszter. A megosztó algoritmus ezzel szemben egy, az összes szegmenseket tartalmazó klasztert bont elemeire addig, míg a szétbontás tovább már nem folytatható. [13]

A Parser engine-ben az összevonó klaszterező algoritmus került megvalósításra, melynek működését a **28. ábra** szemlélteti. A kép bal oldalán a feldolgozandó pontthalmaz, a jobb oldalán pedig a klaszterezés egyes szintjeit tartalmazó dendrogram látható.



28. ábra - Összevonó klaszterező eljárás működése [13]

A dendrogram legalsó szintjén még minden egyes pont külön klasztert alkot. A második szinten az egymáshoz legközelebb álló klaszterek kerültek összevonásra (1 és 2). A harmadik szinten az 1 és 2 pontokat tartalmazó klaszter került a 3-al összevonásra. A feldolgozás hatodik szintjén már csak egy klaszter létezik, amely mind a hat pontot tartalmazza.

Ahhoz, hogy a fent ismertetett klaszterező eljárás átültethető legyen a Parser engine-be, tisztázni kell az alábbi kérdéseket:

- Hogyan értelmezendő a távolság fogalma az egyes karaktereket tartalmazó szegmensek esetében?

A távolság fogalmának értelmezésére rögtön három megoldás is kínálkozik:

- A szegmenseket pontszerű objektumoknak vesszük. Ebben az esetben a szegmensek közötti távolságot a szegmenseket reprezentáló pontok távolsága adja. Ennél a megoldásnál a szegmenseket befoglaló téglalapjaik mértani középpontjával, vagy a súlypontjukkal írhatjuk le.

A tervezés során ez az eljárás elvetésre került, mivel a különböző szegmensméretek sok esetben hibásan elvégzett klaszterezést eredményeztek.

- A szegmenseket képpontok halmazának vesszük. Ebben az esetben a szegmensek távolságát az egymáshoz legközelebb álló képpontjaik távolsága adja.

Bár ez a megoldás biztosíthatja a legnagyobb pontosságot két szegmens távolságának meghatározásakor, a számításigénye miatt egy közel ennyire pontos, ám nagyságrendekkel gyorsabb eljárást kell alkalmazni.

- A szegmenseket egyszerű síkidomként (téglalap) írjuk le. Minden azonosított szegmenshez hozzárendelhető az a legkisebb téglalap, amelynek területe magába foglalja az adott szegmens összes képpontját.

Ennél a megoldásnál két szegmens távolságát a befoglaló téglalapjaik egymáshoz legközelebb lévő oldalai között mért távolság adja.

A Parser engine-ben ez utóbbi megoldás került megvalósításra, mivel amellet, hogy egyszerűen megvalósítható, nagy pontosságot is biztosít.

- Honnan tudható, hogy a dendrogram mely szintjén kerülnek az azonosítandó bekezdések különálló klaszterekbe?

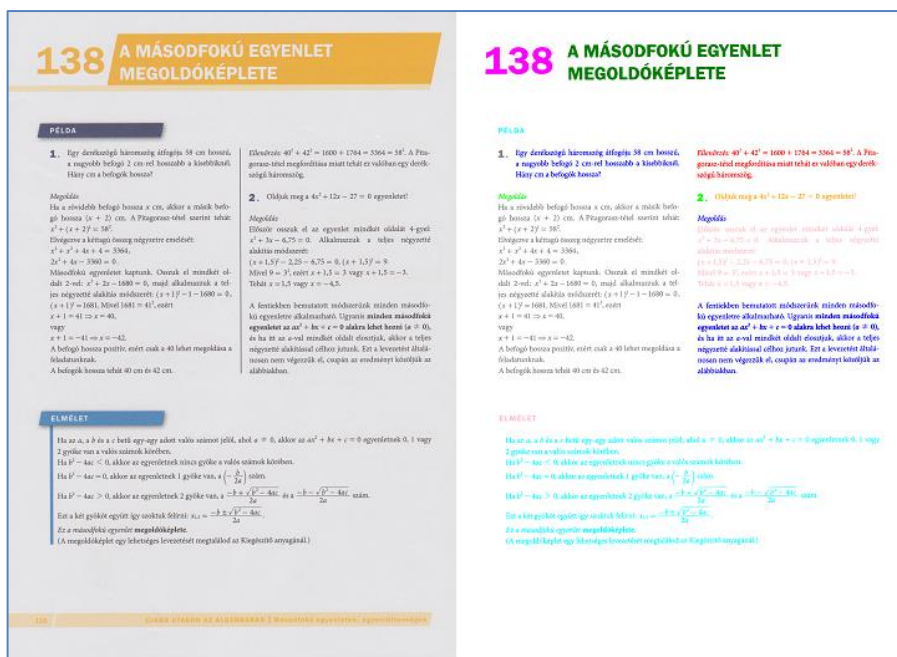
Ahogy az a klaszterező eljárást bemutató ábrán (**28. ábra**) is látható, a klaszterezés eredménye egy olyan hierarchikus struktúra lesz, mely a klaszterezés több szintjét tartalmazza. A legalsó szinten minden szegmens különálló klasztert képez, a legfelsőben pedig egy klaszter tartalmazza az összes szegmenset. Az egyes bekezdéseket reprezentáló klaszterek, valahol ezek között a szintek között találhatók.

Minden egyes szinthez hozzárendelhető az a távolságérték, amelyen az adott szint klaszterei összevonásra kerültek. A megfelelő szint meghatározásakor ez a távolságérték kerül felhasználásra. Általánosan elmondható, hogy a bekezdések azon a klaszter szinten találhatók meg, melynél szignifikáns távolságérték növekedés tapasztalható az alsóbb szintek távolságértékeiéhez képest. Bár léteznek automatikus megoldások ennek a szignifikáns értéknövekedésnek az meghatározására, egyszerűbbnek és gyorsabbnak mutatkozott az a megoldás, hogy külső paraméterként kerüljön megadásra a különálló bekezdések hozzávetőleges távolsága.

A bekezdések azonosítása tehát az egyes szegmensek távolsága alapján történik meg, azonban ez a távolság betűtípusonként és betűméretenként eltérő lehet. A nagyobb betűmérettel rendszerint nagyobb bekezdés távolság is jár. Ha ezt a tényezőt figyelmen kívül hagyjuk, akkor pl. a.12-es betűméretre optimalizált bekezdés azonosítás a 36-os betűméret egyes szavait azonosítaná önálló bekezdésként.

Ennek a problémának a kezeléséhez betűtípusonként nyilván kell tartani a bekezdéstávolságot, és a bekezdések azonosítását betűtípusonként külön-külön kell elvégezni. Mivel a betűtípusok bekezdésen belül is változhatnak, az azonosítás utolsó lépéseként összevonásra kerülnek azok a bekezdések, melyek befoglaló téglalapjai összeérnek.

A bekezdések azonosításának eredményét a **29. ábra** tartalmazza. Az ábra bal oldalán az eredeti oldal, jobb oldalán pedig az azonosított bekezdések láthatóak. Az egyes bekezdések eltérő színekkel vannak jelölve.



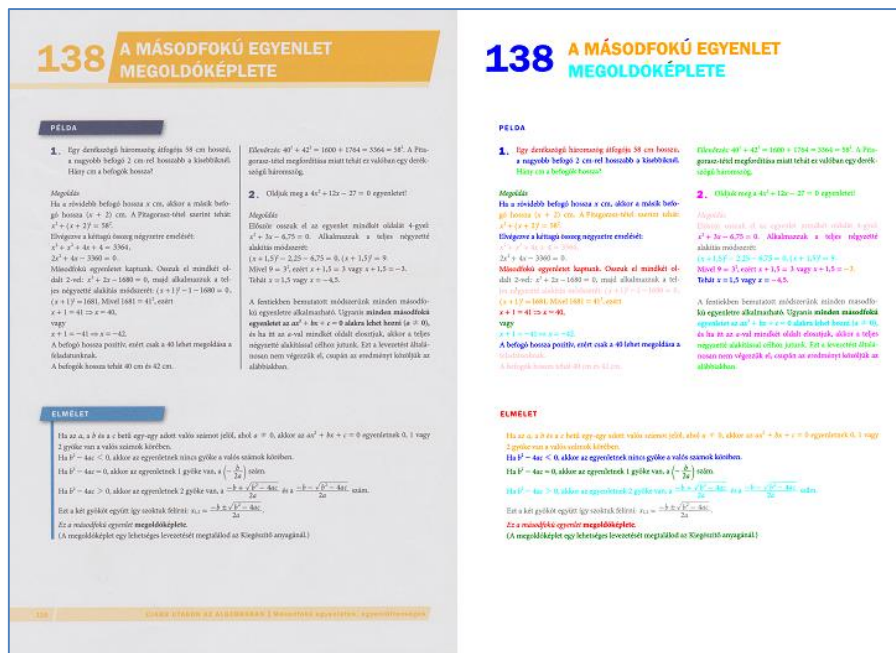
29. ábra - Bekezdések azonosításának eredménye (Forrás: LUCA Converter)

Sorok azonosítása

A térközök hierarchikus felépítése alapján arra lehet számítani, hogy a feldolgozandó dokumentumokban a sortávolság nagyobb lesz a szótávolságnál. Ez az esetek nagy részében megfelel a valóságnak, azonban ha két egymást követő sorban egymás felett helyezkednek el lelógó (pl.: p, j, q) és felnyúló (pl.: f, t, k) karakterek, akkor a két sor között könnyen az átlagos szótávolságnál kisebb sortávolság keletkezik. Ez a jelenség megakadályozza azt, hogy a bekezdések azonosítása során létrejött dendogramban keressük az egyes sorokat tartalmazó klasztereket.

A sorok azonosításához egy újabb dendogram kerül felépítésre, melynek előállítása során csak két irányban (jobb és bal) történik meg a környezetben található szegmensek vizsgálata. Kivételt képeznek a törtvonalként azonosított szegmensek, melyeknél továbbra is mind a négy irányban kell vizsgálni a környezetet. Ennek oka az, hogy így a soron belül található tört kifejezések számlálója és nevezője is az adott soron belül kerül azonosításra. A sorok azonosításakor tehát az átlagos szótávolságot meghaladó távolság értéknél kell a dendogram elemeit azonosítani. Mivel ez az érték is betűtípus és betűméret függő, a bekezdés távolsághoz hasonlóan ez is az adott karakter típusának függvénye.

A sorok azonosításának eredményét a **30. ábra** tartalmazza. Az ábra bal oldalán az eredeti oldal, jobb oldalán pedig az azonosított sorok láthatóak. Az egyes sorok eltérő színekkel vannak jelölve.

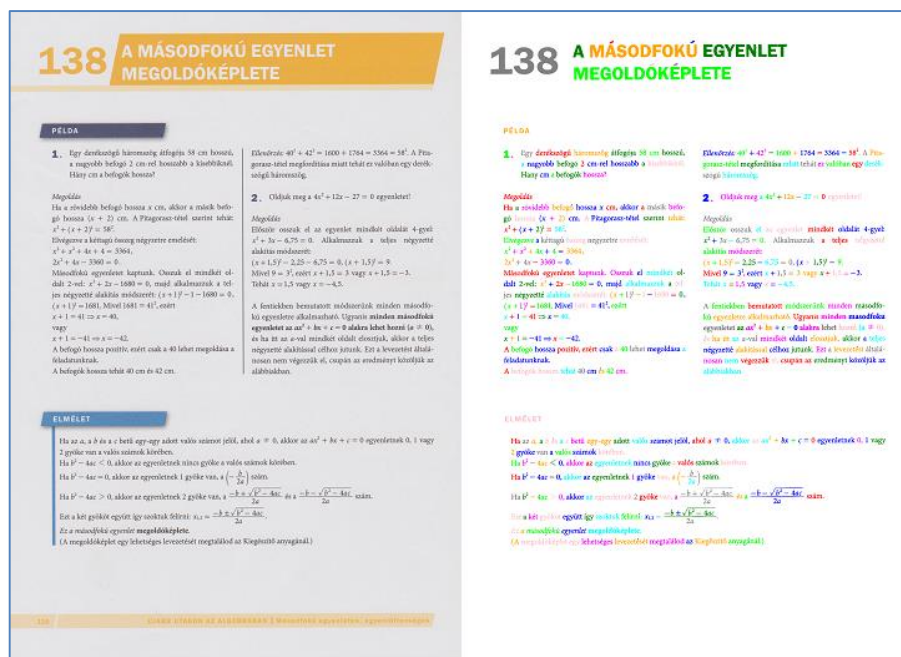


30. ábra - Sorok azonosításának eredménye (Forrás: LUCA Converter)

Szavak azonosítása

A sorok azonosításakor létrejött dendrogram megfelelő szintje tartalmazza a szavak klaszterit is, így újabb dendrogram felépítésére már nincs szükség. A szavak klaszterei azon a dendrogram szinten találhatóak, melyen a távolság érték meghaladja az átlagos betűtávolságot. Természetesen a betűtávolság is betűtípus és méret függő, ezért ez is paraméterként jelenik meg a feldolgozás során.

A szavak azonosításának eredményét a **31. ábra** tartalmazza. Az ábra bal oldalán az eredeti oldal, jobb oldalán pedig az azonosított szavak láthatóak. Az egyes szavak eltérő színekkel vannak jelölve.

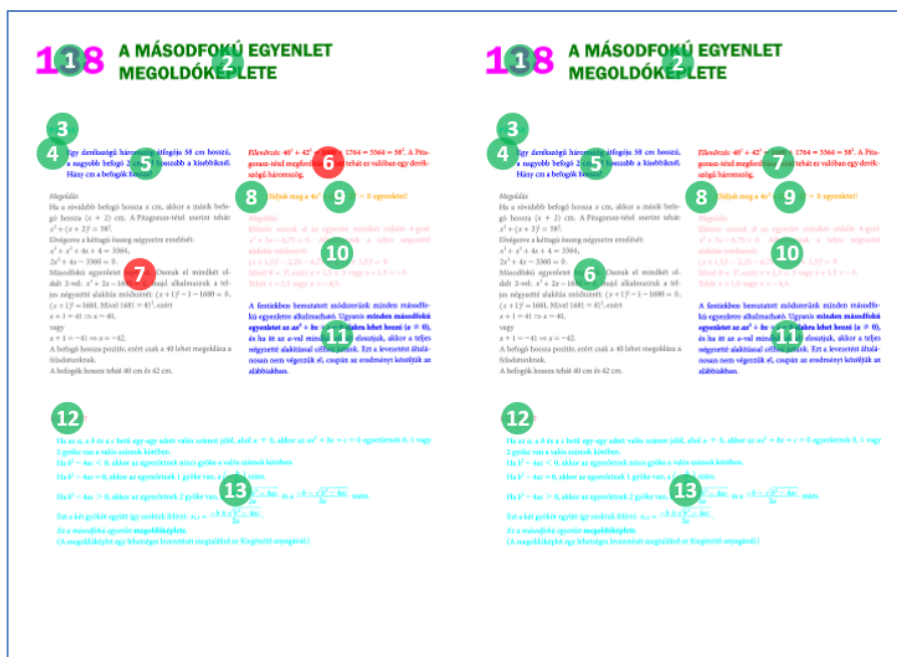


31. ábra - Szavak azonosításának eredménye (Forrás: LUCA Converter)

Hasábok azonosítása

Amennyiben az alkalmazott betűméret aránya a lapmérethez képest kicsi, a tördeléskor túl hosszú sorok keletkeznek, melyek az olvashatóságot rontják. Erre a problémára jelent megoldást a hasábolás. Ilyenkor az oldalt függőlegesen oszlopokra bontják, és ezekbe az oszlopokba kerül betördelésre a szöveg. [10]

Amikor a Book generator modul létrehozza a könyv struktúrájával megegyező felépítésű output fájlt (lásd. 3.1.2.2 Book generator) a Parser engine által azonosított bekezdéseket az oldalon való elhelyezkedésük szerint fentről lefele, és balról jobbra haladva fogja feldolgozni, ami hibát eredményezhet. A 32. ábra egy hasábolat oldal bekezdéseinek a feldolgozási sorrendjét mutatja a hasábok kezelésével (jobb oldali ábra) és a hasábok kezelése nélkül (bal oldali ábra).



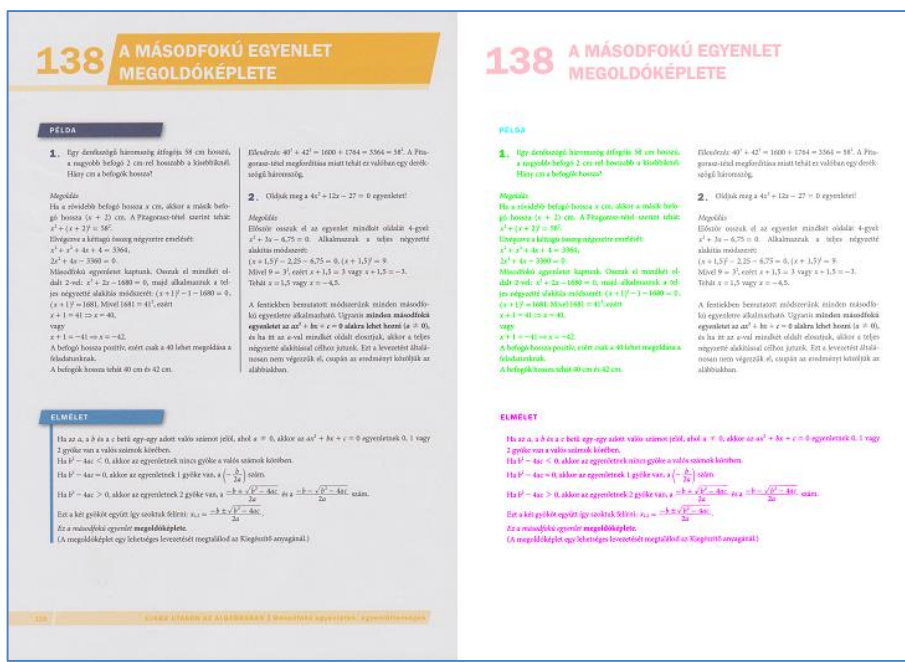
32. ábra - Hasábolt oldal bekezdéseinek hibás és helyes feldolgozási sorrendje

Ahhoz tehát, hogy a Book generator az output létrehozásakor megfelelő sorrendben dolgozza fel az egyes bekezdéseket, információval kell rendelkeznie a hasábok elhelyezkedéséről.

Mivel a hasábok méretének elég nagyok kell lennie ahhoz, hogy a hasábok jól elkülönüljenek egymástól [10] a bekezdések azonosításánál tárgyalt hierarchikus klaszterezés alkalmas eszköz a hasábok azonosítására.

A 33. ábra a hasábok azonosításának eredményét mutatja.

Egy oldal feldolgozását a helyes sorrend biztosításához hasábonként kell elvégezni. A hasábok fentről lefelé és balról jobbra haladva kerülnek feldolgozásra. Az egyes hasábokon belül az azonosított bekezdések feldolgozása is a fentről lefelé és balról jobbra haladva történik.



33. ábra - Hasáb azonosítás eredménye (Forrás: LUCA Converter)

Matematikai kifejezések azonosítása

A matematikai kifejezések azonosítása során egyrészt meg kell találni azokat az összefüggő szövegrészeket, amelyek ilyen kifejezést rejtjenek, másrészt el kell végezni a megtalált kifejezések struktúraelemzését.

A matematikai kifejezéseket a Parser engine által azonosított szavak tartalmi vizsgálatával lehet meghatározni. Optimális esetben a szó, mint legkisebb csoport foglalja magában az egyes matematikai kifejezéseket, azonban a gyakorlat azt mutatja, hogy egy kifejezés rendszerint több különálló szóra esik szét. Ahhoz tehát, hogy egy összefüggő matematikai kifejezés azonosítható legyen, meg kell keresni azokat a szavakat, melyek potenciálisan egy matematikai kifejezés részei, majd ezek közül az egymással közvetlen szomszédságban lévőket össze kell vonni. Mivel a sorok azonosításának eljárása úgy lett kialakítva, hogy a törtes kifejezéseknél mind a törtvonal, mind a számláló és a nevező egy sorba kerül, a kifejezés elemeket tartalmazó szavak összevonásakor elég a soron belüli bal és jobb szomszédokat vizsgálni.

Annak megállapítása, hogy melyik szó lehet egy matematikai kifejezés eleme, a szavakat alkotó szegmensek vizsgálatával történik. Azokat a szavakat kell matematikai kifejezés elemeként kezelni, melyek 100%-ban az alábbi karakterek, vagy karaktersorozatok valamelyikét tartalmazzák:

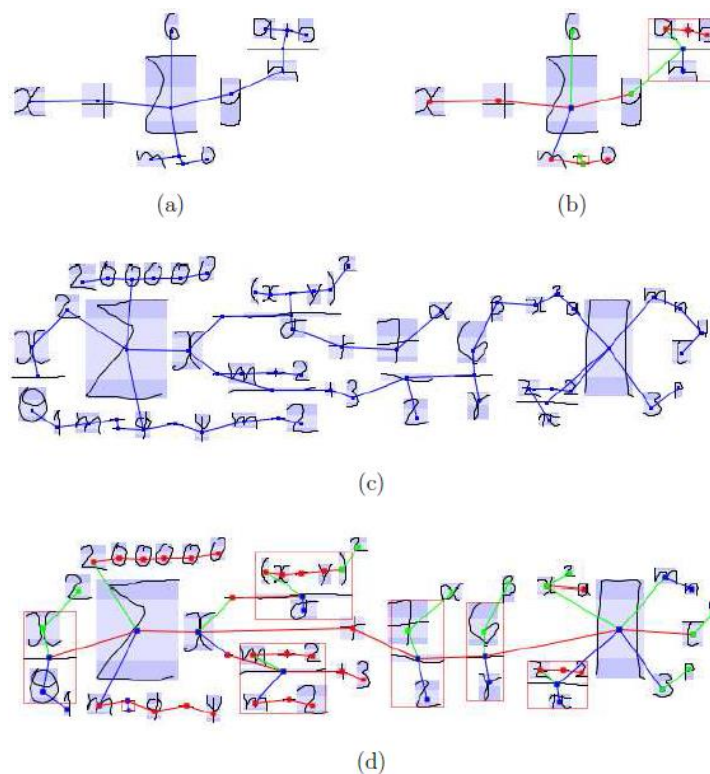
- Számjegyek: 0..9
- Kisbetűs karakterek: a, b, c, x, y, z, m, n
- Operátorok: +, -, =, <, >, *, ;

- Speciális karakter sorozat: sin, cos, tan, tg, ctg, lg, log
- Egyéb: törtvonal, (), [], $\sqrt{\quad}$, Σ
- Paraméterként megadható betűtípussal szedett szövegrész

A rendszer rugalmas felhasználhatósága érdekében a fenti lista dinamikusan bővíthető, módosítható.

A matematikai kifejezések szerkezeti azonosításának és leírásának a kulcsa Matsakis [14] szerint egy minimális feszítőfa (MST) generálása. Maga a matematikai kifejezés leírható összefüggő, irányítatlan, súlyozott gráffal, melyben az egyes csúcspontok a kifejezés elemei (operátorok, operandusok), melyeket a gráf élei kötnek össze. Két csomópont közötti él súlyértéke megegyezik a csúcspontok távolságával. Ebben a gráfban kell egy olyan feszítőfát létrehozni, melynek csúcsai megegyeznek az összefüggő gráf csúcsaival, élei pedig a gráf éleinek részhalmazát alkotják úgy, hogy ciklust nem tartalmaz. A minimális feszítőfa éleinek súlyértéke az összes kialakítható feszítőfa közül a legalacsonyabb.

A minimális feszítőfa alkalmazásának problémáit a **34. ábra** szemlélteti.



34. ábra - (a) Minimális feszítőfa egyszerű kifejezésre; (b) Kívánt kapcsolati fa egyszerű kifejezésre; (c) Minimális feszítőfa összetett kifejezésre; (d) Kívánt kapcsolati fa összetett kifejezésre [15]

Látható, hogy még az egyszerű matematikai kifejezés esetében is eltér a minimális feszítőfa a kifejezés struktúráját leíró kívánt feszítőfától.

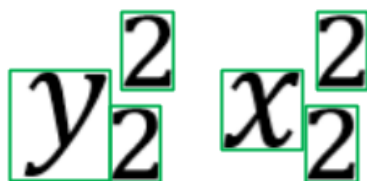
Jelenleg a Preprocessor modul által azonosított szegmensek az alábbi jellemzőkkel rendelkeznek:

- Pozíció (befoglaló téglalap bal felső csúcsának koordinátái)
- Méret (befoglaló téglalap szélessége és magassága)
- Súlypont (a szegmens pixelei alapján számítva)
- Közeppont (a befoglaló téglalap paraméterei alapján számolva)
- Azonosított karakter (kötött karakter, vagy szimbólum)
- Betűtípus (pl.: Arial, Times New Roman, stb.)
- Betűméret (pontokban kifejezve)
- Betűjellemzők (kurzív, félkövér, kurzív-félkövér, stb)

A matematikai kifejezések azonosításához ezeknek a már ismert jellemzőknek a körét újabbak bevezetésével kell bővíteni.

Tényleges középpont helyzete

A matematikai kifejezések struktúraelemzésének alapja a kifejezést alkotó szegmensek egymáshoz képest elfoglalt pozíciója. A pozíció meghatározásakor a szegmenseket befoglaló téglalapot körülvevő teret régiókra kell bontani. Két szegmens viszonyának meghatározásakor többek között azt kell vizsgálni, hogy egymáshoz képest melyik régióban találhatóak. Ezek a térrészek különböző típusú karaktereknél máshol helyezkednek el. Míg az y karakter alsó indexének bal felső pontja az y középpontjánál magasabban helyezkedik el, az x karakternél az alsó index bal felső pontja nagyjából x középpontjában található (35. ábra).



35. ábra - Környező térrész helyzetének változása különböző típusú karakterek esetében

Ahhoz tehát, hogy a szegmens körüli térrész felosztható legyen, a szegmens jellemzői közé fel kell venni egy olyan tulajdonságot, amely megadja a szegmens tényleges

középpontjának helyzetét. A szegmens tényleges középpontjának helyzete egy háromállású jellemzővel leírható:

- Normál (central)

Ebben az esetben a tényleges középpont a szegmens befoglaló téglalapjának középpontjával egybe esik.

- Lelógó (descendent)

Ezeknek a karaktereknek a tényleges középpontja a szegmens befoglaló téglalapjának középpontjánál magasabban helyezkedik el. Ezek tipikusan a lelógó szárú karakterek.

- Felnyúló (ascendent)

Ebben az esetben a karakter tényleges középpontja a befoglaló téglalap középpontjánál alacsonyabban helyezkedik el. Ezek jellemzően a számjegyek, és a felnyúló szárú karakterek.

A matematikai kifejezésekben előforduló karakterek besorolását a tényleges középpont helyzete alapján a **4. táblázat** tartalmazza. Az, hogy egy karakter melyik kategóriába tartozik, betűtípusonként változhat, így ez az információ az adott betűtípust azonosító sabloncsoport tulajdonságaként nyilvántartandó.

	central	ascendent	descendent
non-scripted	$+ - * / (\infty \rightarrow$		
superscripted	$e \pi \sqrt{} \log \sin \cos \tan$	0 1 2 3 4 5 6 7 8 9	
scripted	$a c x z)$	$b d \partial \Delta \nabla$	$g y$
sum-like	$\Sigma \int \Pi$		
lim-like	lim max min		

4. táblázat - Matematikai kifejezésben szereplő gyakoribb elemek csoportosítása a tényleges szimbólumközéppont helyzete alapján [15]

Szegmens körüli térrész felosztása

Az, hogy egy szegmens körül mely térrészeket kell a kifejezés struktúraelemzése során vizsgálni, az függ az azonosított karaktertől, a tényleges középpont helyzetétől, valamint a szegmens méretétől.

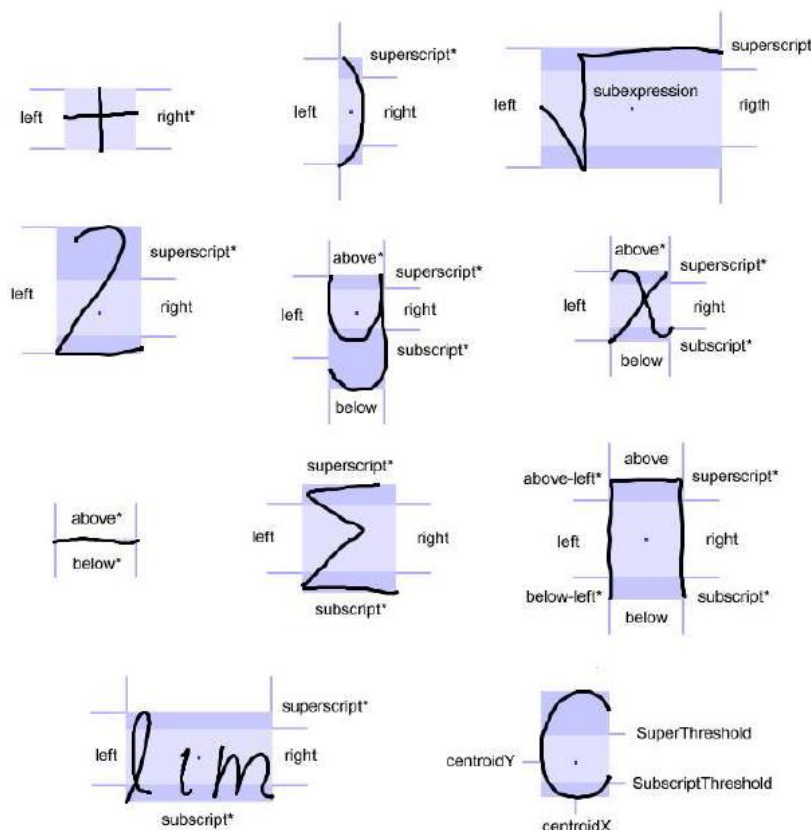
A térrészek kialakításához a szegmens befoglaló téglalapját három tartományra (alsó, középső és felső) kell osztani. A középső tartomány középpontja a szegmens tényleges

középpontja lesz, a tartomány magassága pedig a befoglaló téglalap magasságából számolható az **5. táblázatban** leírt módon. A középső tartomány ismeretében az alsó és felső tartomány már adott.

	super threshold	sub threshold	centroid
ascendent	$y_s + 0.8H_s$	$y_s + 0.2H_s$	$(x + 0.5W_s, y + 0.33H_s)$
descendent	$y_s + 0.9H_s$	$y_s + 0.6H_s$	$(x + 0.5W_s, y + 0.66H_s)$
central	$y_s + 0.8H_s$	$y_s + 0.2H_s$	$(x + 0.5W_s, y_s + 0.5H_s)$

5. táblázat - Szegmens középső tartomány kiterjedésének meghatározása [15]

Az egyes szegmensek befoglaló téglalapjainak a felosztását, és a szegmens körüli térrészek kialakítását a **36. ábra** mutatja. Ahogy az látható, a karakter típusától függően csak azok a térrészek kerülnek kialakításra, melyekben kapcsolódó szegmensek jelenhetnek meg. Értelemszerűen pl. egy + operátornak se felső, se alsó indexben nem lehet kapcsolódó eleme.



36. ábra - Szegmens körüli térrész felosztása kézzel írott karakterek esetén [15]

A szegmensek körül az alábbi térrészek alakíthatóak ki:

- Felső (above)
- Alsó (below)
- Bal (left)
- Jobb (right)
- Felső index (superscript)
- Alsó index (subscript)
- Bal-felső (above-left)
- Bal-alsó (below-left)
- Belső (inside)

Az, hogy egyes karakterekhez milyen vizsgálendő térrészeket kell kialakítani csak az azonosított karaktertől és nem annak betűtípustól függ. Ennek megfelelően ez az információ a Parser engine működéséhez szükséges adatok között kerül eltárolásra.

Szimbólum dominancia

Matematikai kifejezés struktúrájának a felépítéséhez meg kell állapítanunk a szimbólumok között egyfajta dominanciát (alá-fölrendeltségi viszonyt). Egy egyszerű hatványkitevős kifejezés esetén az alap dominálja a kitevőt, a feldolgozás során az alap lesz a kiindulási pont, innen kezdődik a struktúra felfejtése.

Ernesto [15] szerint két elem közötti dominancia viszonyt három tényező vizsgálatával lehet eldönteni:

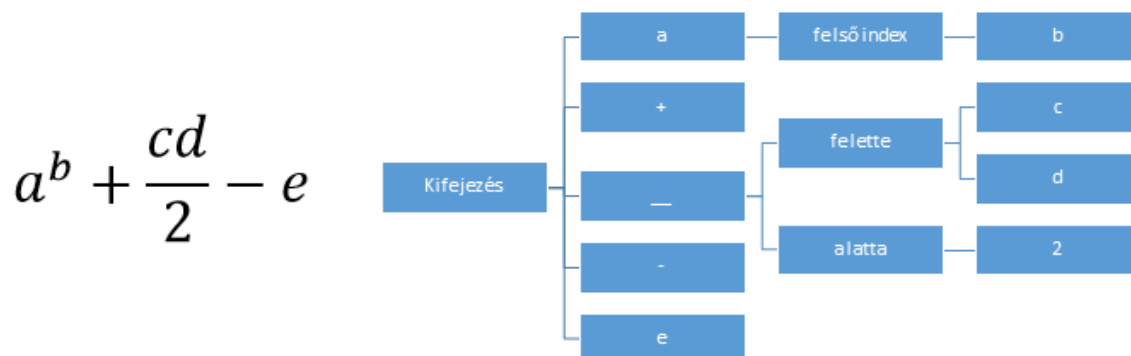
- Az a szegmens dominálja b szegmenset, ha b megtalálható a valamely dedikált térrészában úgy, hogy a nem eleme b valamely dedikált térrészában.
- Az a szegmens dominálja b szegmenset, ha b megtalálható a valamely dedikált térrészában úgy, hogy a eleme b valamely dedikált térrészában és a szegmens mérete nagyobb, mint b szegmens mérete.
- Az a szegmens dominálja b szegmenset, ha b megtalálható a valamely dedikált térrészában úgy, hogy a eleme b valamely dedikált térrészában és a szegmens mérete megegyezik, b szegmens méretével, úgy, hogy a szegmens operátor, b szegmens pedig operandus.

Matematikai kifejezés leírása

A matematikai kifejezések leírására Zanibbi [16] által alkalmazott struktúra az alapvonal fa (Baseline Structure Tree - BST). Ennek a leírásnak a lényege, hogy a matematikai kifejezésben található szimbólumok klaszterezésre kerülnek úgy, hogy az egy sorban lévő

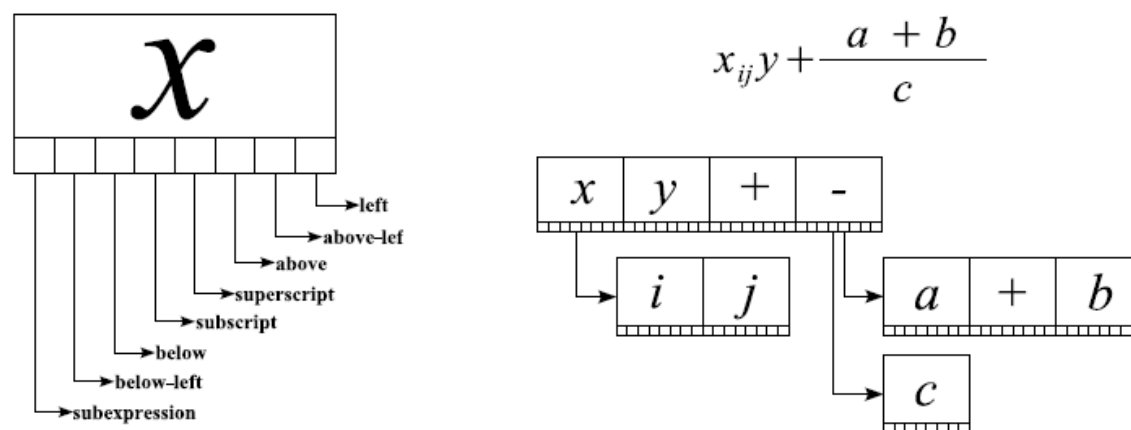
elemek kerülnek egy csoportba. Ezt követően a szimbólum dominancia szabályainak alkalmazásával meghatározható az egyes csoportok közötti alá-fölérendeltségi viszony.

A kifejezésfa felépítésének első szintjét annak a klaszternek az elemei alkotják, melyek nincsenek alárendeltségi viszonyban más alapvonalbeli elemmel. A második szint az első szint elemeivel közvetlen alárendeltségi viszonyban lévő klasztereket tartalmazza, és így tovább (37. ábra).



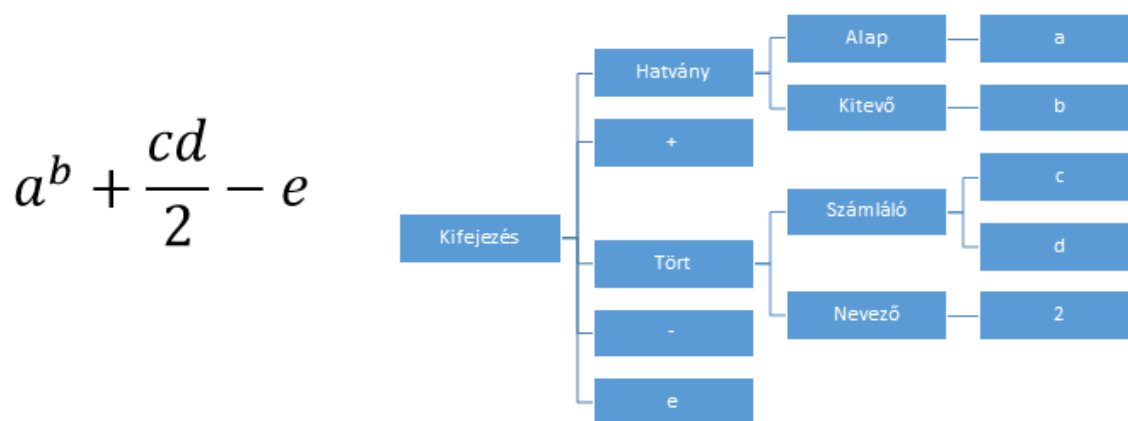
37. ábra –Alapvonal fa létrehozása matematikai kifejezésből

Az alapvonal fa leírásához alkalmazandó adatszerkezetet Tapia [15] alapján a 38. ábra szemlélteti. Ez alapján a matematikai kifejezések esetében az egyes szegmensek kiegészülnek nyolc darab mutatóval, melyek a szegmens környezetének egyes térrészeiben található szegmens klaszterekre mutatnak.



38. ábra - Alapvonal fa felépítéséhez használandó adatszerkezet, és alkalmazása [15]

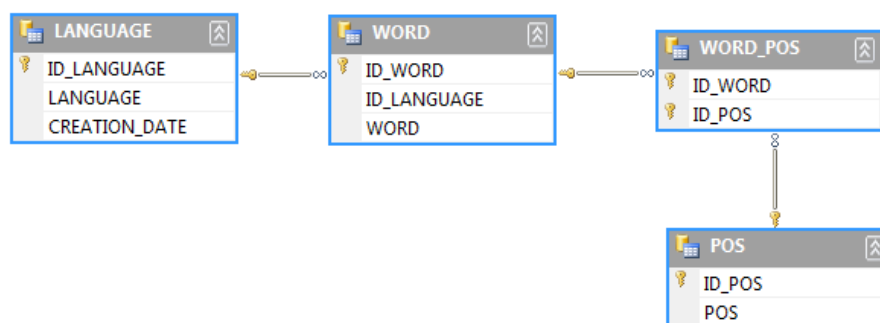
Az alapvonal fa átalakításával már egyszerűen létrehozható egy olyan kifejezésfa, amely alkalmas a matematikai kifejezés bejárására (39. ábra).



39. ábra – Kifejezés fa létrehozása matematikai kifejezésből

Karakterazonosítási hiba kezelése normál bekezdés egy szavában

Az OCR Engine az egyes szegmens azonosítását sablonillesztéssel végzi. Az eljárás során minden egyes szegmens adatai között a rá legjobban hasonlító három sablon karaktere, és az illeszkedés mértéke kerül rögzítésre. A hiba azonosítása és javítása ezeknek a rögzített adatoknak, valamint a rendszer szótár adatbázisának adatai alapján kerül végrehajtásra. A szótár adatbázis felépítését a **40. ábra** mutatja.



40. ábra - Szótár adatbázis felépítése

A szótár adatbázis az alábbi táblákat tartalmazza:

- WORD

Szótár tábla, mely ellenőrizhetővé teszi a karakter azonosítások eredményét.

- LANGUAGE

A különböző nyelvi változatok törzstáblája. Biztosítja a LUCA Converter többnyelvűsíthetőségének a lehetőségét.

- POS és WORD_POS

A szófajok törzstáblája, valamint a szavak és szófajok kapcsolótáblája. Segítségükkel a későbbiekben mondat szerkezet elemzés végezhető az azonosított szövegben.

A Parser Engine kétféle karakterazonosítási hiba kezelésére lett felkészítve.

- Gyenge azonosítás

A PDF Parser által szóként azonosított szegmenscsoport valamely tagjánál a legjobban illeszkedő sablon hasonlósága nem éri el a 70%-ot. Ebben az esetben az azonosítás eredménye nem elég erős ahhoz, hogy további vizsgálat nélkül az elfogadható legyen.

- Többszörös azonosítás

A PDF Parser által szóként azonosított szegmenscsoport valamely tagjánál a legjobban illeszkedő két sablon hasonlósága közötti különbség nem éri el az 5%-ot. Ebben az esetben két potenciális eredmény is van, amelyekről el kell dönteni azt, hogy melyik a valós.

A hiba javítását mindkét esetben ugyanazzal a módszerrel kell elvégezni. A nem egyértelműen azonosított karaktereket tartalmazó szó összes olyan változatát keresni kell a szótárban, mely az azonosítás eredményeképpen szóba jöhet. Amennyiben a vizsgált változatokra pontosan egy találat van a szótárban, az automatikusan elfogadásra kerül. Több találat esetén a felhasználó feladata eldönteni, hogy melyik változatot fogadja el. Amennyiben nincs találat, a felhasználó a felületen kézzel végezheti el az azonosítás javítását.

3.1.2.2 Book generator

A Book generator modul feladata a LUCA Converter rendszer outputjának az előállítás. Az outputtal szemben támasztott követelmény mindenekelőtt az, hogy mind tartalmában, mind szerkezetében egyezzen meg a feldolgozott tankönyvvel. Ezen felül fontos szempont, hogy a szerkezeti jellegzetességek, és a tartalmi elemek a LUCA Reader-rel történő felolvasás során egyszerűen elérhetőek legyenek.

A Book generátor által létrehozott LUC kiterjesztésű állomány XML dokumentumként tárolja a könyvek adatait. „Az XML dokumentumok úgynevezett önleíró vagy öndefiniáló szöveges információk. Az önleíró vagy önmeghatározó tulajdonságon azt értjük, hogy egy XML dokumentum speciálisan meghatározott formában saját maga leírja az információ szerkezeti felépítését. Ebben valójában semmilyen új gondolat nincs az írott ismereti anyag ilyen tulajdonságaihoz képest (az adathordozó itt és most a papír), hiszen minden hírlapról, könyvről, levélről ugyanez elmondható. A különbség annyi,

hogy míg a hagyományos papír adathordozó esetén az információ tagolása, szerkezeti képe fejezetekre, paragrafusokra, bekezdésekre tagolással (fejezetcímek, alcímek, üres sorok, bekezdések, egyéb tabulációk) történik. Digitális formában ezek a szerkezeti formát biztosító elemek nem célra vezetők, mert az ilyen tipográfiai megoldások az emberi szem számára jól érzékelhetők, számítógépes érzékelésük azonban nem minden esetben (esetenként egyáltalán nem) lehetséges. [...] Az XML nyelvi szerkezeti biztosítják a szemmel jól érzékelhető, de számítógépes programmal már kevésbé felismerhető és azonosítható szerkezeti formázást biztosító elemek pontos meghatározásának a lehetőségét. Ennek eredményeképpen olyan dokumentumhoz jutunk, amelynek szerkezeti felépítése, tagoltsága mind az emberi szem, mind a számítógépes programok számára jól felismerhető és azonosítható. Nem mellékes körülmény az sem, hogy teszi mindezt úgy, hogy eközben csak ábrázolható (az emberi kézírásban is alkalmazható és alkalmazott) karaktereket, szimbólumokat használ. Ez egyben azt is jelenti, hogy az XML dokumentumok készítése nem igényel semmiféle speciális dokumentáló rendszert, speciális szövegszerkesztő programot, jóllehet ma már ilyenek szinte minden software környezetben elérhetők. Egy XML dokumentumban leírt ismeretanyag minden nehézség nélkül érthetővé válik egy XML-t nem ismerő, de az ismeretanyag témakörében járatos, vagy legalábbis érintőlegesen tájékozott ember számára is, legfeljebb nem érti, hogy bizonyos, az „érdemi” szövegtől jól elkülönülő furcsa szimbólumok (szerkezeti formázás – markup) miért törlik meg helyenként az érdemi szöveget.” [17]

Az XML-t a legtöbb szövegszerkesztő alkalmazás kezeli, szabványos és hordozható. Ezeknek a tulajdonságoknak köszönhetően bárki könnyen módosíthatja, vagy kiegészítheti a korábban feldolgozott tankönyvek állományait, vagy akár egy egyszerű notepad-al létrehozhat új könyveket, amely a LUCA Reader segítségével felolvastatható.

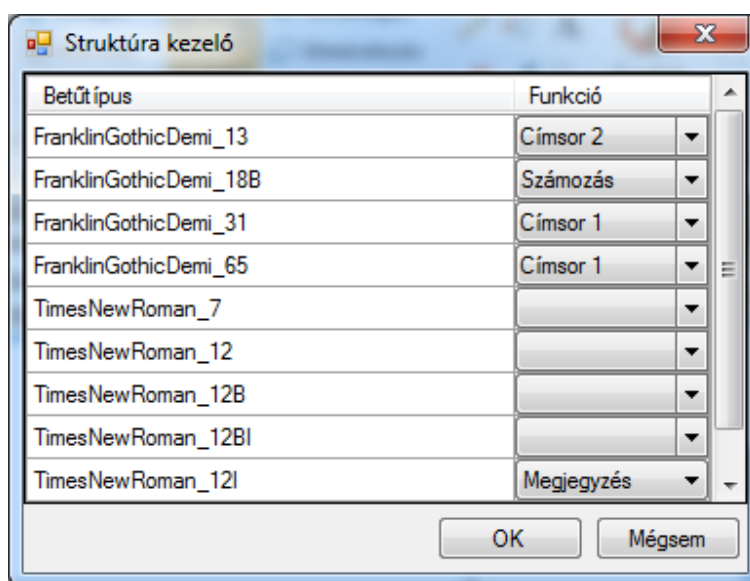
A LUC állományok előállítása a Preprocessor modul által azonosított szegmensek, és a Parser engine modul által azonosított szegmens klaszterek (hasábok, bekezdések, stb.) alapján történik.

A Parser engine az összefüggő szövegelemeket oldalanként azonosítja, tehát az outputja oldalakra tagolva tartalmazza azokat. Ennek megfelelően a Book generator is oldalanként állítja elő a LUC állományt, amiben azonban már nem releváns információ az oldalszám. Ahhoz, hogy a feldolgozott könyv struktúrájával megegyező felépítésű állomány jöhessen létre, alapvető fontosságú az egyes oldalakon található szövegelemek feldolgozásának megfelelő sorrendje. A feldolgozás alapegysége a hasáb. Egy oldalon belül a hasábok fentről-lefelé és balról jobbra kerülnek feldolgozásra. A hasábokon belül a bekezdések sorrendjének a meghatározásakor ugyanez a módszer kerül alkalmazásra. A bekezdések sorai értelem szerűen fentről lefele, a szavak a sorokon belül balról jobbra lesznek feldolgozva.

Címsorok kezelése

Egy tetszőleges könyv felépítésének gerincét a különböző szintű címsorok adják. Mivel az egyes címsor szintek címeit rendszerint eltérő betűtípus és (vagy) betűméret jelzi, ennek az információnak a segítségével elvégezhető a könyv alapstruktúrájának a meghatározása.

A Book generator modul struktúra kezelő ablakában van arra lehetőség, hogy a Preprocessor modul működése során azonosított betűtípusokhoz címsor szintet rendeljünk (**41. ábra**).



41. ábra - Book generator struktúra kezelő felület

Ugyanaz a címsor szint több különböző betűtípushoz is tartozhat. A LUC állomány előállításakor az adott betűtípussal írt bekezdés szövege lesz a megfelelő szinthez tartozó fejezet, vagy alfejezet címe.

Felsorolások, számozások kezelése

A felsorolások, és számozások változatos formában jelenhetnek meg a dokumentumokban. A LUCA Converter funkcionalitásának kialakításakor mintául használt tankönyvben (I. Barcza, I. Basa és M. K. Tamásné, Matematika 10 Kísérleti Tankönyv - Első Kötet, Budapest: Oktatási és Fejlesztő Intézet, 2014) a sorszámok a szöveg többi részétől eltérő betűtípussal kerülnek megjelenítésre. Ennek következtében az egyes sorszámok a bekezdések azonosításakor önálló bekezdésként fognak megjelenni (**29. ábra**), így amennyiben a számozás betűtípusához a struktúra kezelő felületen (**41. ábra**) hozzárendelésre kerül a „Számozás” jelölő, az output fájlban létrejön a megfelelő struktúra.

A LUCA Converter korábbi verzióinak fejlesztésekor használt tankönyvben (J. Kosztolányi, I. Kovács, K. Pintér, J. Urbán és I. Vincze, Sokszínű Matematika 12, Szeged: Mozaik Kiadó, 2012) a sorszámozás a bekezdések azonosítást követően nem különül el a sorszámot követő szövegtől. A bekezdések és a sorok azonosításának eredményét a **42. ábra** mutatja. Ebben az esetben a számozás azonosításához a bekezdést alkotó sorok első szavait kell vizsgálni.

5. Egyenletek, egyenlőtlenségek	5. Egyenletek, egyenlőtlenségek
Elsőfokú egyenletekre, egyenlőtlenségekre vezető feladatok	Elsőfokú egyenletekre, egyenlőtlenségekre vezető feladatok
1. A szövegben logikai összefüggéseket keresünk, és megválasztjuk az ismeretlent. (Az ismeretlent leggyakrabban a kérdés alapján célszerű megválasztani.) 2. Felírjuk az egyenletet. 3. Megoldjuk. 4. Ellenőrizzük a megoldást a szöveg alapján. 5. Megfogalmazzuk a választ.	1. A szövegben logikai összefüggéseket keresünk, és megválasztjuk az ismeretlent. (Az ismeretlent leggyakrabban a kérdés alapján célszerű megválasztani.) 2. Felírjuk az egyenletet. 3. Megoldjuk. 4. Ellenőrizzük a megoldást a szöveg alapján. 5. Megfogalmazzuk a választ.

42. ábra - Bekezdések és sorok azonosításának eredménye

Amennyiben az egyes sorok első szavai növekvő számsort alkotnak, vagy felsorolás jelölő karakterből állnak, az adott bekezdés sorai számozásként, vagy felsorolásként kerülnek be az output állományba.

Kiegészítő jelentés kezelése

Egy hagyományos tankönyv esetében gyakori megoldás, hogy bizonyos típusú tartalmakat ugyanazzal a betűtípussal jelölnek (pl.: megjegyzés, definíció, érdekesség, fontos tudnivaló). Mivel a felolvasás során ezek az információk is segíthetik a szövegben való eligazodást, fontos, hogy bekerüljenek a rendszer output állományába. Ehhez arra van szükség, hogy a struktúra kezelő felületen (**41. ábra**) az adott betűtípushoz hozzá legyen rendelve a megfelelő kiegészítő jelentés.

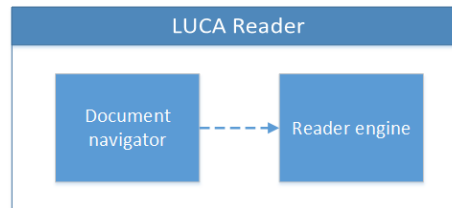
A kiegészítő jelentés az output állományban az adott betűtípusból álló bekezdés attribútumai között fog megjelenni.

Matematikai kifejezések kezelése

A matematikai kifejezések önálló struktúrát alkotnak a könyv struktúráján belül. Az XML állományba a matematikai kifejezésből létrehozott kifejezésfa kerül leírásra (lásd: Matematikai kifejezés leírása).

3.2 LUCA Reader

A LUCA Reader alrendszer feladata a LUCA Converter által előállított strukturált szöveges állomány bejárása és felolvasása, felépítését a **43. ábra** szemlélteti.



43. ábra - A LUCA Reader felépítése

A LUCA Reader a Document navigator és a Reader engine modulokból áll. A Document navigator végzi a strukturált szöveges állomány bejárását, a Reader engine feladata pedig az aktuális szövegrész felolvasása.

3.2.1 Document navigator

A Document navigator modul feladata a LUCA Converter alrendszer által előállított LUC kiterjesztésű állomány struktúrájának bejárása. A modul az XML alapú állományok tartalmát egy fa struktúrát megjelenítő objektumban helyezi el. Működése során a Document navigator ennek a fának a bejárását teszi lehetővé úgy, hogy a feldolgozás mélysége szabadon megválasztható. Ez a megoldás lehetővé teszi a felhasználó számára azt, hogy könnyen a dokumentum kívánt részéhez navigáljon, ahonnan elkezdheti a dokumentum felolvasását. Automatikus felolvasás módban az éppen aktuális pozíciótól kezdődően történik meg a dokumentum teljes felolvasása, mely során a könyv struktúrájának minden szintje feldolgozásra kerül.

A **44. ábra** egy, a LUCA Converter által előállított LUC állomány kivonatát tartalmazza. Látható, hogy az állományban leírt könyv egyetlen fejezete további alfejezetekre, és bekezdésekre tagolódik, és tartalmaz két matematikai kifejezés struktúráját is.

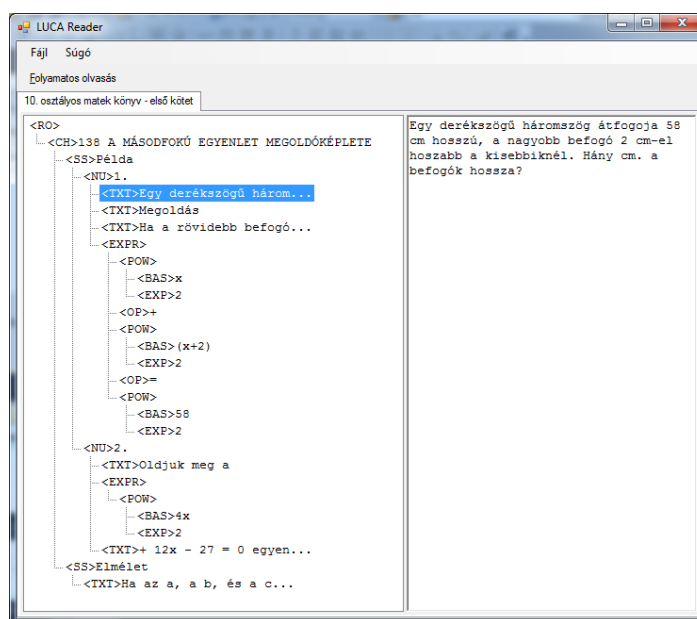
```

1 <book title = "10. osztályos matek könyv - első kötet" publisher = "Oktatókutatató és fejlesztő intézet"
  yearofpublication = "2014">
2   <author>Barcza István</author>
3   <author>Basa István</author>
4   <author>Tamásné Kollár Magdolna</author>
5   <collector title = "138 A MÁSODFOKÚ EGYENLET MEGOLDÓKÉPLETE" type = "chapter">
6     <collector title = "Példa" type = "subsection">
7       <collector title = "1." type = "numbering">
8         <text>Egy derékszögű háromszög átfogója 58 cm hosszú, a nagyobb befogó 2 cm-el hosszabb a kisebbiknél.
9         <text>Hány cm. a befogók hossza?</text>
10        <text>Megoldás</text>
11        <text>Ha a rövidebb befogó hossza x cm, akkor a másik befogó hossza (x+2) cm. A Pitagorasz tétel
12        szerint tehát: </text>
13        <expr>
14          <pow>
15            <base>x</base><exp>2</exp>
16          </pow>
17          <op>+</op>
18          <pow>
19            <base>(x+2)</base><exp>2</exp>
20          </pow>
21          <op>=</op>
22          <pow>
23            <base>58</base><exp>2</exp>
24          </pow>
25        </expr>
26      </collector>
27      <collector title = "2." type = "numbering">
28        <text>Oldjuk meg a </text><expr><pow><base>4x</base><exp>2</exp></pow></expr> <text>+ 12x - 27 = 0
29        egyenletet!</text>
30      </collector>
31    </collector>
32    <collector title = "Elmélet" type = "subsection">
33      <text>Ha az a, a b, és a c betű egy-egy valós számot jelöl...</text>
34    </collector>
35  </book>

```

44. ábra - Matematika tankönyvet leíró LUC állomány kivonata

A 45. ábra, a LUCA Reader dokumentum navigációs felületét mutatja. A képernyő bal oldalán a felolvasandó dokumentum (44. ábra) fa struktúrába történő leképezése látható, míg a jobb oldalon jelenik meg az aktuálisan felolvasandó szöveg.



45. ábra - A LUCA Reader alrendszer dokumentum navigációs felülete

3.2.2 Reader engine

A Reader engine az adott dokumentumrész felolvasását a JAWS For Windows alkalmazás beszéd szintetizátorának a felhasználásával végzi, amelyhez rugalmasan alkalmazható interfészt biztosít a gyártó.

A JAWS For Windows lehetővé teszi a felolvasó motor vezérlését bármilyen program számára. Ezt a JAWS Script Language segítségével éri el, amellyel olyan scriptfájlok hozhatók létre, amelyek illeszkednek annak a programnak a felületéhez, amelyben a felolvasást el akarjuk végezni.

„Egy scriptfájl egy, vagy több scriptet tartalmazhat. A JAWS script language két alapvető típusát különbözteti meg a scripteknek. Ezek az „Default” és az „Application” típusok.

A JAWS Script Source (JSS) fájlok tartalmazzák a script-ek és függvények forrását. A Default.jss olyan alapvető funkciókat tartalmaz, amelyek lehetővé teszik a képernyőolvasást bármilyen Windows-alapú program számára. A „Default” script fájl a JAWS minden indulásakor betöltődik, és folyamatosan a háttérben van. Az „Application” script fájlok erre a Default.jss-re töltődnek rá akkor, amikor a hozzájuk tartozó alkalmazás elindul. Az „Application” script fájl neve megegyezik az alkalmazás nevével (pl.: Calc.jss és Calc.exe). Ebben a konkrét esetben a Calc.exe indításakor töltődik be a Calc.jss, és ekkor válnak aktívvá a benne megírt scriptek. A program bezárásakor, vagy fókuszváltáskor a Calc.jss kikerül a memóriából, és a Default.jss vezérli tovább a felolvasásokat.

A JAWS állandóan monitorozza a billentyűzetet, és figyeli a leütéseket. Billentyű leütésekor először az „Application” script fájlban keres az eseményhez tartozó műveletet. Ha itt nincs találat, akkor a Default.jss-ben keres tovább. Ha a Default.jss sem ad találatot, akkor az eseményt a JAWS nem kezeli.” [18]

Ahogy a felhasználó a dokumentum navigációs ablak (**45. ábra**) bal oldalán található fa struktúrában lépked az egyes szövegelemek között, úgy változik az éppen felolvasandó szöveg a képernyő jobb oldalán található szövegdobozban. Ez a szöveg kerül tájékoztató információkkal kiegészítve a JAWS For Windows beszéd szintetizátorának elküldésre.

Maga a tájékoztató információ utal a felolvasott szöveg minőségére. Pl.: fejezetcím felolvasása előtt a „Fej” szócska jelöli azt, hogy most egy új fejezet kezdődik. A szöveges tájékoztató információ adott esetben kiváltható akár különböző hangjelzésekkel is.

4 EREDMÉNYEK BEMUTATÁSA ÉS ÉRTÉKELÉSE

A projekt megvalósítása során elért eredmények értékeléséhez az elkészült főbb rendszerelemek tesztelésre kerültek. A tesztelés eredményei az alábbiakban alrendszerre és modulra bontva kerülnek bemutatásra.

4.1 LUCA Converter

A LUCA Converter funkcionalitásának teszteléséhez két matematika tankönyv került kiválasztásra. Az egyik könyv az Oktatási és Fejlesztő Intézet által 2014-ben kiadott Matematika 10. című kísérleti tankönyv első kötete (I. Barcza, I. Basa és M. K. Tamásné), a másik a Mozaik kiadó által 2012-ben kiadott Sokszínű Matematikai 12. című könyv (J. Kosztolányi, I. Kovács, K. Pintér, J. Urbán, I. Vince). A könyvek kiválasztásakor elsődleges szempont volt, hogy a korszerű tankönyvekben jellemzően alkalmazott szerkesztési megoldások minél szélesebb spektrumát lefedjék.

A tesztelést szegmentálás, karakterfelismerés, összefüggő szövegelem azonosítás, és matematikai kifejezés struktúraelemzés témakörökben végeztem el.

A szegmentálás, és az összefüggő szövegelem azonosítás vizsgálatához mind a két könyvből három-három oldal került feldolgozásra, majd a feldolgozás eredménye lett összehasonlítva a forrásdokumentummal. A karakter azonosítás tesztelése azon a könyvön lett végrehajtva, amelyen a LUCA Converter a szegmentálás során gyengébb teljesítményt produkált.

A matematikai kifejezések struktúraelemzésének teszteléséhez három összetett kifejezés került feldolgozásra. A feldolgozás eredménye olyan fa struktúrájú leírás, amely a feldolgozandó kifejezés képével összehasonlítható.

4.1.1 Szegmentálás

A tesztelés célja annak megvizsgálása, hogy a LUCA Converter szegmentáló eljárása milyen hatékonysággal végzi el az összefüggő képpont csoportok azonosítását. A szegmensazonosítás hibásnak tekintendő a következő esetekben:

- Összeérő szegmensek egy elemként kerültek azonosításra
- Egy szegmens szakadás miatt több elemként került azonosításra
- A karakter azonosítását megnehezítő, bizonytalan kontúrral került a szegmens azonosításra

A tesztelés során a Matematika 10 tankönyv 32-34 oldalai, és a Sokszínű Matematika 12 tankönyv 178-180 oldalai kerültek feldolgozásra. A tesztelés során létrejött ellenőrző képeket a **8.2 melléklet** tartalmazza.

A Matematika 10 tankönyv 32. oldalán 2457 db szegmens került azonosításra. A szegmensazonosítás statisztikáját a **6. táblázat** tartalmazza.

Eredmény	Darabszám	Arány (%)
Pontos azonosítás	2331	94,87
Szegmens összeérésből adódó hibás azonosítás	82	3,34
Szegmens szakadásból adódó hibás azonosítás	0	0,00
Pontatlan kontúrral rendelkező szegmensek	44	1,79

6. táblázat - Matematika 10 tankönyv 32. oldal szegmentálás eredmény statisztika

A Matematika 10 tankönyv 33. oldalán összesen 2358 db szegmens került azonosításra. A hibás szegmensazonosítások statisztikáját a **7. táblázat** tartalmazza.

Eredmény	Darabszám	Arány (%)
Pontos azonosítás	2230	94,57
Szegmens összeérésből adódó hibás azonosítás	71	3,01
Szegmens szakadásból adódó hibás azonosítás	12	0,51
Pontatlan kontúrral rendelkező szegmensek	45	1,91

7. táblázat - Matematika 10 tankönyv 33. oldal szegmentálás eredmény statisztika

A Matematika 10 tankönyv 34. oldalán összesen 1898 db szegmens került azonosításra. A hibás szegmensazonosítások statisztikáját a **8. táblázat** tartalmazza.

Eredmény	Darabszám	Arány (%)
Pontos azonosítás	1758	92,62
Szegmens összeérésből adódó hibás azonosítás	97	5,11
Szegmens szakadásból adódó hibás azonosítás	0	0,00
Pontatlan kontúrral rendelkező szegmensek	43	2,27

8. táblázat - Matematika 10 tankönyv 34. oldal szegmentálás eredmény statisztika

A Sokszínű matematika 12 tankönyv 178. oldalán összesen 1327 db szegmens került azonosításra. A hibás szegmensazonosítások statisztikáját a **9. táblázat** tartalmazza.

Eredmény	Darabszám	Arány (%)
Pontos azonosítás	1324	99,77
Szegmens összeérésből adódó hibás azonosítás	3	0,23
Szegmens szakadásból adódó hibás azonosítás	0	0,00
Pontatlan kontúrral rendelkező szegmensek	0	0,00

9. táblázat - Sokszínű matematika 12 tankönyv 178. oldal szegmentálás eredmény statisztika

A Sokszínű matematika 12 tankönyv 179. oldalán összesen 1478 db szegmens került azonosításra. A hibás szegmensazonosítások statisztikáját a **10. táblázat** tartalmazza.

Eredmény	Darabszám	Arány (%)
Pontos azonosítás	1477	99,93
Szegmens összeérésből adódó hibás azonosítás	1	0,07
Szegmens szakadásból adódó hibás azonosítás	0	0,00
Pontatlan kontúrral rendelkező szegmensek	0	0,00

10. táblázat - Sokszínű matematika 12 tankönyv 179. oldal szegmentálás eredmény statisztika

A Sokszínű matematika 12 tankönyv 180. oldalán összesen 1361 db szegmens került azonosításra. A hibás szegmensazonosítások statisztikáját a **11. táblázat** tartalmazza.

Eredmény	Darabszám	Arány (%)
Pontos azonosítás	1354	99,49
Szegmens összeérésből adódó hibás azonosítás	6	0,44
Szegmens szakadásból adódó hibás azonosítás	1	0,07
Pontatlan kontúrral rendelkező szegmensek	0	0,00

11. táblázat - Sokszínű matematika 12 tankönyv 180. oldal szegmentálás eredmény statisztika

A szegmensek azonosításának eljárásával kapcsolatban kétféle probléma tapasztalható.

- Különálló karakterek egy szegmensként történő azonosítása

Az input fájlok esetében 300 DPI az előírt szkennelési felbontás, ami az egymáshoz közel található szegmenseknél könnyen azok összeérését eredményezi. Ilyenkor az egyébként különálló karakterek egy szegmensként kerülnek azonosításra. Ez a hibatípus az összes szegmensazonosítás kevesebb, mint 4%-ában fordul elő, betűtípustól és mérettől függően.

Ez a probléma jelenleg kezelhető az összeérő szegmenscsoportok tanító sablonok közé való felvételével, ám ez a megoldás jelentős többlet beavatkozást igényel a felhasználó részéről. Az összeérő szegmensek problematikájának kezeléséhez meg kell valósítani azok automatikus szétbontását.

- Kisméretű inhomogén szegmensek azonosítási hibája

Az inhomogén szegmensek azonosítását végző eljárás a nagyméretű szegmensek azonosítását csaknem 100%-os pontossággal végzi el, kis szegmensek esetén ez az arány azonban 30%-ra csökken.

A problémát az okozza, hogy a binarizálást végző Niblack eljárás paraméterei a nagyméretű inhomogén szegmensek azonosítására vannak optimalizálva. A megoldáshoz meg kell valósítani a lapon belül eltérő paraméterekkel történő binarizálást.

4.1.2 *Karakterek azonosítása*

A tesztelés célja annak megvizsgálása, hogy a LUCA Converter karakter azonosító eljárása milyen hatékonysággal végzi el az egyes szegmensekben kódolt karakterek azonosítását. A karakter azonosítás hibásnak tekintendő a következő esetekben:

- Az azonosítás során a karakter, vagy a karakter típus hibásan, vagy 50%-ot el nem érő találati aránnyal került azonosításra
- Mind a karakter, mind a betűtípus megfelelően lett azonosítva, de az azonosítás erőssége csak 50 és 70% közötti.

A karakterek azonosításának vizsgálatához a szegmentálás tesztelésekor vizsgált oldalakon azonosított szegmensek kerültek feldolgozásra. A tesztelés során létrejött ellenőrző képeket a **8.3 melléklet** tartalmazza. Az ellenőrző képeken zöld színnel kerültek feltüntetésre a pontosan azonosított karakterek, pirossal a hibás, kékkel a gyenge azonosítások.

A Matematika 10 tankönyv 32. oldalán összesen 2457 db szegmens került azonosításra. A hibás karakterazonosítások statisztikáját a **12. táblázat** tartalmazza.

Eredmény	Darabszám	Arány (%)
Pontosan azonosított karakterek	2289	93,16
Gyengén azonosított karakterek	52	2,12
Hibásan azonosított karakterek	116	4,72

12. táblázat - Matematika 10 tankönyv 32. oldal karakter azonosítás eredmény statisztika

A Matematika 10 tankönyv 33. oldalán összesen 2358 db szegmens került azonosításra. A hibás karakterazonosítások statisztikáját a **13. táblázat** tartalmazza.

Eredmény	Darabszám	Arány (%)
Pontosan azonosított karakterek	2169	91,98
Gyengén azonosított karakterek	66	2,80
Hibásan azonosított karakterek	123	5,22

13. táblázat - Matematika 10 tankönyv 33. oldal karakter azonosítás eredmény statisztika

A Matematika 10 tankönyv 34. oldalán összesen 1898 db szegmens került azonosításra. A hibás karakterazonosítások statisztikáját a **14. táblázat** tartalmazza.

Eredmény	Darabszám	Arány (%)
Pontosan azonosított karakterek	1703	89,73
Gyengén azonosított karakterek	37	1,95
Hibásan azonosított karakterek	158	8,32

14. táblázat - Matematika 10 tankönyv 34. oldal karakter azonosítás eredmény statisztika

A karakterek azonosításának eljárásával kapcsolatban kétféle probléma tapasztalható.

- Hibásan azonosított szegmenseknél bekövetkező hibás karakterazonosítás
A szegmensek azonosításával kapcsolatos legjellemzőbb hiba az összeérő szegmensek egy elemként történő azonosítása volt. Könnyen belátható, hogy a hibásan azonosított szegmensek esetében a karakterazonosítás is hibás lesz.
A szegmentáló eljárás további fejlesztésével tehát a karakterazonosítás határfoka is jelentősen növelhető.

- Pontatlan sablon tanítás

A karakterazonosítás másik jellemző problémája a pontatlanul elvégzett sablontanításból fakad. Ilyenkor a hibás sablon nem képes az elvárt pontossággal eredményt szolgáltatni a felhasznált tanítóminták kis száma, vagy a rosszul megválasztott tanítóminták miatt.

A feldolgozás hatékonyságának javításához a többihez képest szignifikánsan több hibát produkáló sablonok újratanítását kell elvégezni.

4.1.3 *Összefüggő szövegelemek azonosítása*

A tesztelés célja annak megállapítása, hogy a LUCA Converter összefüggő szövegelem azonosító eljárása milyen hatékonysággal végzi el a logikailag összetartozó szegmensek csoportba foglalását.

Az összefüggő szövegelem azonosítása hibásnak tekintendő a következő esetekben:

- Az adott csoportból olyan szegmens marad ki, amely logikailag hozzá tartozik.
- Adott csoportba belekerül olyan szegmens, amely logikailag nem tartozik bele.

A tesztelés során a Matematika 10 tankönyv 32-34 oldalai, és a Sokszínű Matematika 12 tankönyv 178-180 oldalai kerültek feldolgozásra. A tesztelés során létrejött ellenőrző képeket a **8.4 melléklet** tartalmazza. Az összes ábránál azonos színnel vannak jelölve az ugyanabba az egységbe bekerült szegmensek.

A hasábok azonosítása csak a Matematika 10 tankönyv esetén került végrehajtásra, mivel a Sokszínű Matematika 12 nem tartalmaz hasábolt szöveget. A hasábok a teszteléshez felhasznált oldalak esetében hiba nélkül kerültek azonosításra.

A bekezdések azonosításakor öt esetben olyan szövegrészek is kikerültek a bekezdés csoportból, amelyek logikailag bele tartoztak volna. A hibát mind az öt esetben az okozta, hogy a bekezdés elemei közötti elemtávolság meghaladta a bekezdés távolságot. Ez jellemzően matematikai kifejezések környezetében, vagy táblázatok esetében fordul elő. Ahhoz, hogy ez a jelenség elkerülhető legyen, az egymás szomszédságában lévő bekezdések tartalmi elemzésével kell a logikai összetartozás tényét megállapítani.

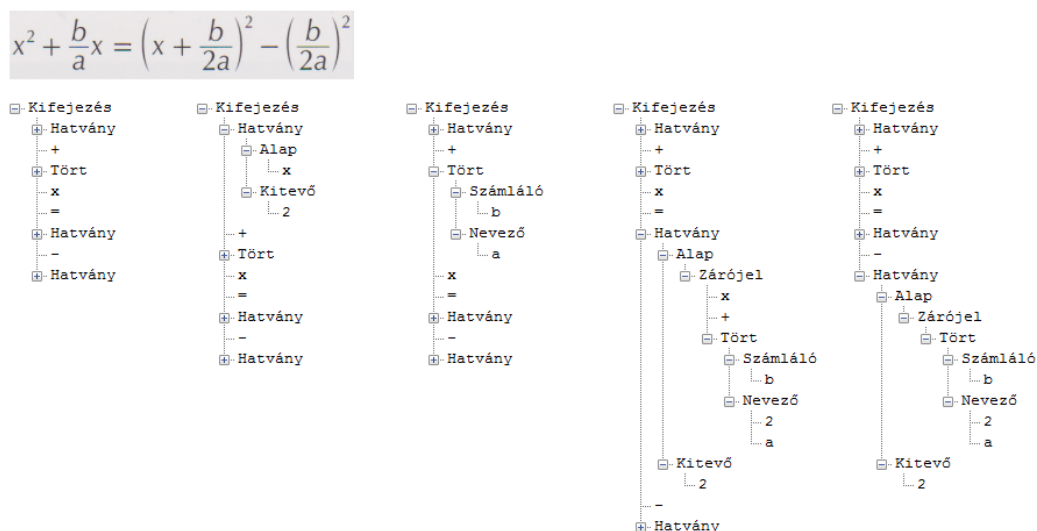
Sorok azonosításakor tizenkét esetben történt hibás azonosítás. A hibát az összes esetben olyan matematikai kifejezések okozták, melyek törtes tagot is tartalmaztak. Ezeknél a tételeknél a számláló, a nevező, vagy mind a két elem a kifejezés többi részétől különböző sorba került. A sorok azonosításakor az egyes szegmensek közötti távolságok meghatározása során az Y tengely irányába tett elmozdulás kisebb súllyal esik latba, mint az X tengely irányú elmozdulás. Ez alól csak a vízszintes vonalak képeznek kivételt, mivel azoknál minden irány egyformán vesz rész a távolság meghatározásában. Ez a módszer

biztosítja azt, hogy egy törtszám számlálója, és nevezője is ugyanabba a sorba kerüljön. A hibát az okozta, hogy a törtvonal, valamint a számláló és a nevező között az átlagos sortávolságnál nagyobb térköz található, ami megfelelő paraméterezéssel javítható.

A szavak azonosítása normál szövegnél gyakorlatilag hiba nélkül megtörténik, ám a matematikai kifejezések az esetek túlnyomó részében több szóra esnek szét. Ennek oka az, hogy a matematikai kifejezésekben nem érvényesülnek azok a szótávolságok, amelyek a normál szövegrészre jellemzőek. A hiba javításához az egymás szomszédságában található szavak tartalmi elemzése alapján kell azok összetartozását vizsgálni.

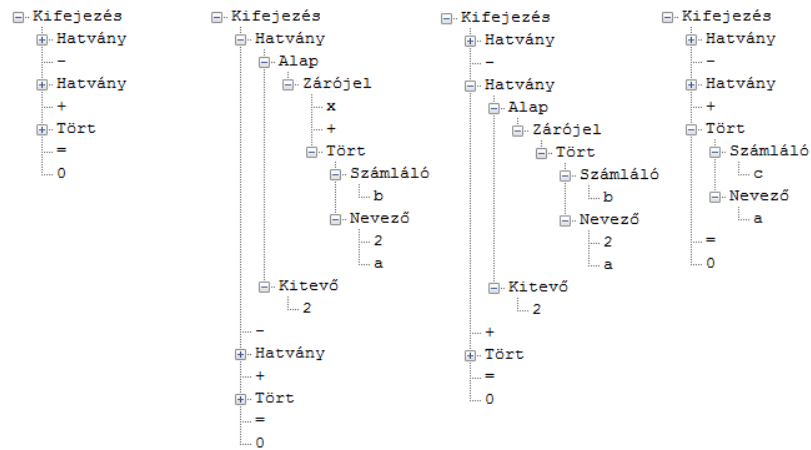
4.1.4 Matematikai kifejezés struktúraelemzés

A tesztelés célja annak vizsgálata, hogy a LUCA Converter milyen hatásokkal végzi el az összetett matematikai kifejezések struktúraelemzését. A struktúraelemzés vizsgálatához a tesztelés korábbi részeiben is felhasznált tankönyvekből került öt kifejezés kiválasztásra, melyekből előállításra kerültek a kifejezéseket leíró fa struktúrák. A kiválasztott kifejezéseket, és az előállított fa struktúrákat a **46. ábra**, a **47. ábra** a **48. ábra**, a **49. ábra** és az **50. ábra** tartalmazza. Az ábrák felső részében minden esetben a feldolgozott kifejezés, alatta pedig a LUCA Parser által létrehozott kifejezésfa látható.



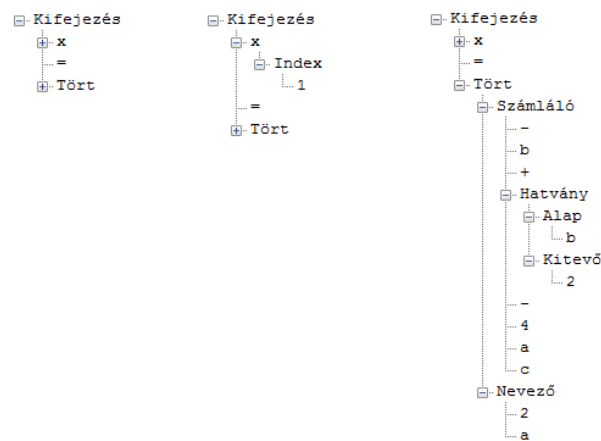
46. ábra - Összetett matematikai kifejezés struktúraelemzése

$$\left(x + \frac{b}{2a}\right)^2 - \left(\frac{b}{2a}\right)^2 + \frac{c}{a} = 0$$



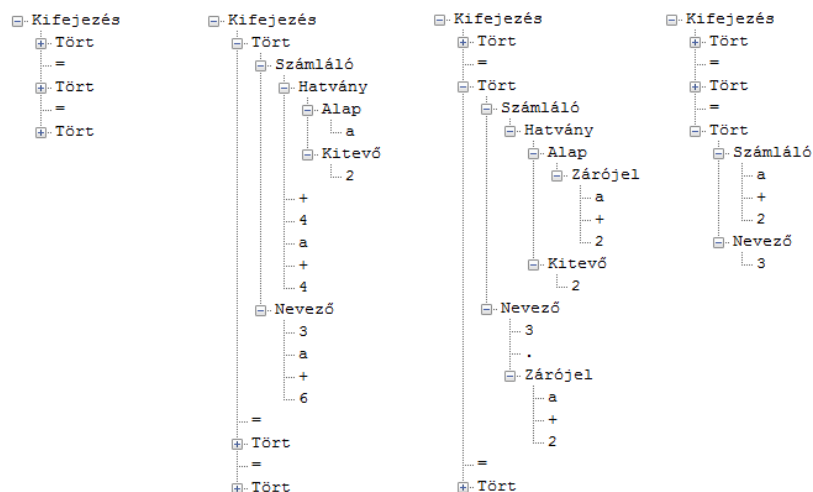
47. ábra - Összetett matematikai kifejezés struktúraelemzése

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$



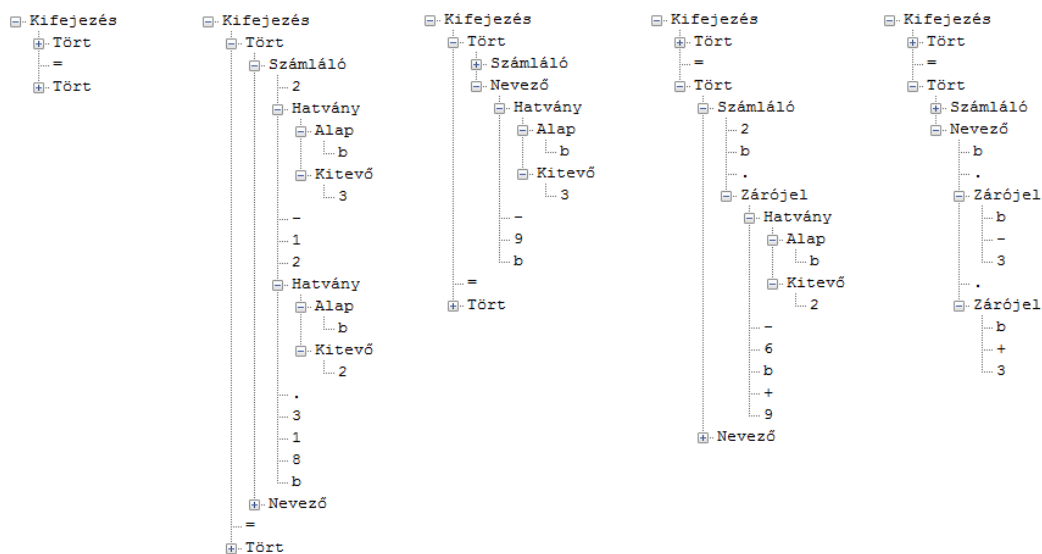
48. ábra - Összetett matematikai kifejezés struktúraelemzése

$$\frac{a^2 + 4a + 4}{3a + 6} = \frac{(a + 2)^2}{3 \cdot (a + 2)} = \frac{a + 2}{3}$$



49. ábra - Összetett matematikai kifejezés struktúraelemzése

$$\frac{2b^3 - 12b^2 \cdot 318b}{b^3 - 9b} = \frac{2b \cdot (b^2 - 6b + 9)}{b \cdot (b - 3) \cdot (b + 3)}$$



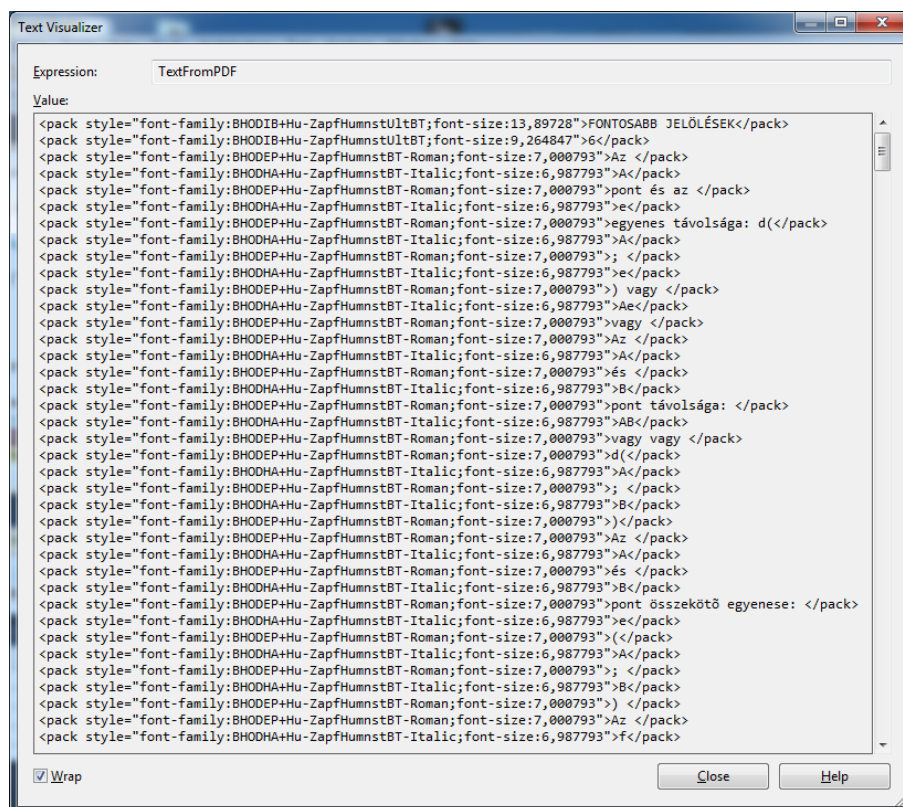
50. ábra - Összetett matematikai kifejezés struktúraelemzése

A vizsgált tesztesetekben a matematikai kifejezések feldolgozása hiba nélkül megtörtént.

4.1.5 PDF állományok feldolgozása

A PDF állományok feldolgozásának teszteléséhez egy 10. osztályos matematika tankönyv (Juhász István, Orosz Gyula, Paróczay József, Szászné Dr. Simon Judit MATEMATIKA 10. - Az érthető matematika) 6. oldala került feldolgozásra.

A feldolgozás eredményének egy részét a **51. ábra** mutatja.



51. ábra - PDF állomány feldolgozásának az eredménye

Az eredmények részletes elemzése alapján elmondható, hogy a normál karakterek azonosítása a PDF állományban 100%-ban megtörtént, azonban a matematikai jelölések, és az egyszerű szimbólumok nem jelentek meg se a karakterek, se a képek feldolgozásának eredményei között. Ennek következtében pl. a törtes kifejezések elemei közül pont a törtvonal marad le, ami a kifejezés elemzését lehetetlenné teszi. Egyelőre nem egyértelmű, hogy a hibát az iTextSharp library hibája, vagy annak a helytelen felhasználása okozza. A javításhoz így vagy az iTextSharp library-t kell másik eszközre cserélni, vagy a felhasználás módját kell megváltoztatni.

4.2 LUCA Reader

A LUCA Reader program alkalmas arra, hogy a LUCA Converter által előállított LUC állomány tartalmát egy fa struktúrában megjelenítse, és lehetővé tegye annak bejárását. A tesztelés a tervezési fázisban létrehozott LUC állomány került a programmal felolvasatásra.

4.2.1 Navigálás

A tesztelés során mind a különböző szintű címsorok, mind a matematikai kifejezések elemei helyesen kerültek megjelenítésre. A navigáció mind a struktúra hosszában, mind a mélységében az elvárásoknak megfelelően megtörténik. A navigáció során a struktúra aktuális eleméhez tartozó szöveg a képernyőn erre a célra létrehozott textbox-ban megjelenik.

4.2.2 Felolvasás

Mivel a LUCA Reader-hez még nem készült el a saját JAWS script fájl, a felolvasás a struktúrát leíró fa, és nem a felolvasandó szöveget tartalmazó textbox adatai alapján történik.

5 A TOVÁBBFEJLESZTÉS LEHETŐSÉGEI

A projekttel kapcsolatos további fejlesztési feladatok az implementálás és továbbfejlesztés kategóriákba kerültek besorolásra.

5.1 Implementálás

Az implementálás feladatai közé azoknak a funkcióknak a leprogramozása került, melyek megjelennek a rendszer leírásában, ugyanakkor még nem kerültek megvalósításra.

- OCR Engine
 - Ékezetes karakterek azonosítása.
- PDF Parser
 - A PDF állományokból kinyert információk átkonvertálása a LUCA Converter által használt adatstruktúrába.
- Book generator
- Reader engine
 - A LUCA Reader működésére optimalizálta JAWS script létrehozása.

5.2 Továbbfejlesztés

Továbbfejlesztésnek azok a nagyobb volumenű feladatok tekinthetők, melyek nem jelennek meg a rendszer leírásában, ugyanakkor nagymértékben javítanak a program felhasználhatóságát.

- OCR Engine
 - Ahogy az az eredmények ismertetésében bemutatásra került, a LUCA Converter OCR Engine modulja 90% feletti pontossággal azonosítja a kötött karaktereket és az egyszerű szimbólumokat. Azoknak a jeleknek az azonosítása, melyek nem valamely betűtípus jelkészletéből kerülnek ki (pl. különféle zárójelek, gyökjelek, stb.) ugyanakkor nem megoldott. Bár ezekre a szimbólumokra is létrehozható tanító sablon (jelenleg ez történik), ez a megoldás hosszútávon nem tartható. A probléma megoldásához olyan karakterazonosítási módszer kialakítására van szükség, mely a szegmensek formai jellemzői alapján sablonok alkalmazása nélkül képes az azonosítás elvégzésére.
- LUCA Reader
 - Jelenleg a LUCA Reader-rel kapcsolatos egyik legnagyobb korlátot a JAWS For Windows beszédszintetizátorának a felhasználása jelenti. Mivel a JAWS fizetős termék, csak olyan munkaállomáson lehet

működtetni a LUCA Reader-t, melynek tulajdonosa megvásárolta azt. Ennek a korlátnak a megszüntetéséhez meg kell vizsgálni nyílt forráskódú (pl.: NVDA) rendszerek felhasználásának, vagy saját fejlesztésű fonéma alapú beszélő motor fejlesztésének a lehetőségét.

- Egy felolvasó alkalmazás felhasználhatóságát nagymértékben befolyásolja a szó szerint értendő hordozhatósága. Csak akkor terjedhet el széles körben a LUCA Converter által nyújtott szolgáltatás, ha elkészül a LUCA Reader mobil platformos átírata.

6 ÉRTÉKELES

A látássérült tanulók számára helyzetüknél fogva sokkal nagyobb terhet jelent tanulmányaik elsajátítása, mint látó társaiknak. Ahhoz, hogy ez a hátrány csökkenthető legyen, alapvető fontosságú olyan taneszközök biztosítása, melyek a látássérültek speciális igényeit kielégítik. Bár számos félmegoldás létezik, összességében kijelenthető, hogy Magyarországon mind az elérhető taneszközök minősége, mind az elosztórendszer hatékonysága számos kívánnivalót hagy maga után.

A LUCA rendszer tervezésekor a cél egy olyan eszköz létrehozása volt, amely valódi megoldást nyújt a látássérültek tankönyvi ellátottságával kapcsolatos problémákra. A funkcionalitásának tervezése a leendő felhasználók (látássérült diákok és tanáraik) igényeinek felmérése alapján történt.

A rendszer fejlesztésének két éve alatt létrehozásra került egy olyan rugalmasan paraméterezhető alkalmazás, amely képes digitalizált kiadványok hatékony feldolgozására. A feldolgozás részfeladatait saját fejlesztésű eljárások hajtják végre, melyek hatékony működést biztosítanak különböző szerkesztési megoldásokat alkalmazó kiadványok feldolgozásakor. A LUCA rendszer jelenlegi készültségi fokában képes papír alapú (szkennelt) kiadványok szövegelemeinek szegmentálására, a szegmensekben kódolt információ kinyerésére, valamint az egyes szegmensek tartalma, formai jellemzői és pozíciója alapján a feldolgozandó dokumentum struktúrájának felépítésére. Ezen felül a rendszer azonosítja, és strukturálisan elemzi a dokumentumban esetlegesen megtalálható matematikai kifejezéseket.

Az elkészült rendszerelemek teszteléséhez az Oktatási és Fejlesztő Intézet által 2014-ben kiadott Matematika 10. című kísérleti tankönyv (I. Barcza, I. Basa és M. K. Tamásné) első kötete, valamint a Mozaik kiadó által 2012-ben kiadott Sokszínű Matematikai 12. (J. Kosztolányi, I. Kovács, K. Pintér, J. Urbán, I. Vince) című könyvek kerültek kiválasztásra. A könyvek kiválasztásakor elsődleges szempont volt, hogy a korszerű tankönyvekben jellemzően alkalmazott szerkesztési megoldások minél szélesebb spektrumát lefedjék.

A tesztelés során a szegmentálás átlagosan 95%-os, a karakterazonosítás 90%-os, míg a dokumentum struktúraelemzése (hasábok, bekezdések, sorok és szavak azonosítása) 92%-os pontossággal került végrehajtásra. Az összetett matematikai kifejezések struktúraelemzése hibátlanul azonosított szegmensek esetén 100%-os volt. A tesztelés adatainak vizsgálata alapján meghatározásra kerültek azok a fejlesztési irányok, melyek a hibaarány további csökkenését eredményezhetik.

Az elkészült rendszerterv, és az implementált rendszerelemek teszteredményei alapján elmondható, hogy a LUCA rövid időn belül alkalmassá tehető arra, hogy a bevezetőben meghatározott funkcióját betöltse, és alapjaiban változtassa meg a látássérültek tankönyvi ellátásának hazai rendszerét.

7 IRODALOMJEGYZÉK

- [1] Á. Szomorjai, Ajánlások vak és aliglátó gyermekek, tanulók kompetencia alapú fejlesztéséhez, Budapest: sul Nova Közoktatás-fejlesztési és Pedagógus-továbbképzési Kht., 2006.
- [2] R. Ausbrooks, S. Buswell, D. Carlisle, G. Chavchanidze, S. Dalmás, S. Devitt és A. Diaz, „Mathematical Markup Language (MathML) Version 3.0 2nd Edition,” 04 2014. [Online]. Available: <http://www.w3.org/TR/MathML3/>.
- [3] T. B. Moeslund, *Introduction to Video and Image Processing*, London: Springer, 2012.
- [4] M. H. Som, M. J. Zain és J. A. Ghazali, „Application of threshold techniques for readability improvement of jawi historical manuscript images,” %1. kötet *Advanced Computing: An International Journal*, 2011.
- [5] N. Otsu, „A Threshold Selection Method from Gray-Level Histograms,” *IEEE Transactions on Systems, Man, and Cybernetics*, 1977.
- [6] R. C. Gonzalez és R. E. Woods, *Digital Image Processing*, New Jersey: Prentice-Hall, Inc., 2002.
- [7] G. Leedham, C. Yan, K. Takru, J. H. N. Tan és L. Mian, „Comparison of Some Thresholding Algorithms for Text/Background Segmentation in Difficult Document Images,” School of Computer Engineering, Nanyang Technological University, 2003.
- [8] M. Sonka, V. Hlavac és R. Boyle, *Image Processing, Analysis, and Machine Vision*, United States of America: Thomson Learning, 2008.
- [9] K. Palágyi, *Képfeldolgozás haladóknak*, Budapest: Typotex Kiadó, 2011.
- [10] I. Mohai, „Tipográfiai alapismeretek,” Székesfehérvár, 2006.
- [11] P. Virágvölgyi, *A tipográfia mestersége számítógéppel*, Budapest: Osiris kiadó, 2004.
- [12] B. Lowagie, *iText in Action*, Stamford: Manning Publications Co., 2011.
- [13] D. A. Forsyth és J. Ponce, *Computer Vision: A Modern Approach*, New Jersey: Pearson Education, 2011.
- [14] N. E. Matsakis, *Recognition of Handwritten Mathematical Expressions*, 1999.

- [15] E. Tapia, *Understanding Mathematics: A System for the Recognition of On-Line Handwritten Mathematical Expressions*, Berlin, 2004.
- [16] R. Zanibbi, D. Blostein és J. Cordy, *Recognizing mathematical expressions using tree transformation*, Rochester: RIT Scholar Works, 2002.
- [17] Á. Nagy, A. Bessenyei, P. Plósz, S. Mikó, A. G. Nagy és Z. Parragi, „XML,” ELTE IK, Programozási Nyelvek és Fordítóprogramok Tanszék, 2010. [Online]. Available: <http://nyelvek.inf.elte.hu/leirasok/XML/index.php>. [Hozzáférés dátuma: 15 10 2014].
- [18] K. A. Gould, „Everything You Always Wanted to Know About Writing JAWS Scripts, But Didn't Know Whom to Ask,” Freedom Scientific Inc., 2000. [Online]. Available: http://www.blind-geek-zone.net/Docs/JAWS_ScriptingManual.txt. [Hozzáférés dátuma: 05 10 2014].
- [19] M. Altrichter, G. Horváth, B. Pataki, G. Strausz, G. Takács és J. Valyon, *Neurális hálózatok*, Budapest: Panem Könyvkiadó Kft., 2006.

8 MELLÉKLETEK

8.1 Neurális hálózatok alkalmazásával kapcsolatos kísérletek

A LUCA Converter OCR Engine moduljának fejlesztésekor a kötött karakterek azonosítására három különböző megoldás került implementálásra, melyek közül végül a sablonillesztéses eljárás maradt a rendszerben. A másik két megoldás neurális hálózat alkalmazásával végezte el az azonosítást, azonban ezek hatásfoka nem bizonyult kielégítőnek.

Több kimenetű neurális hálózat alkalmazása

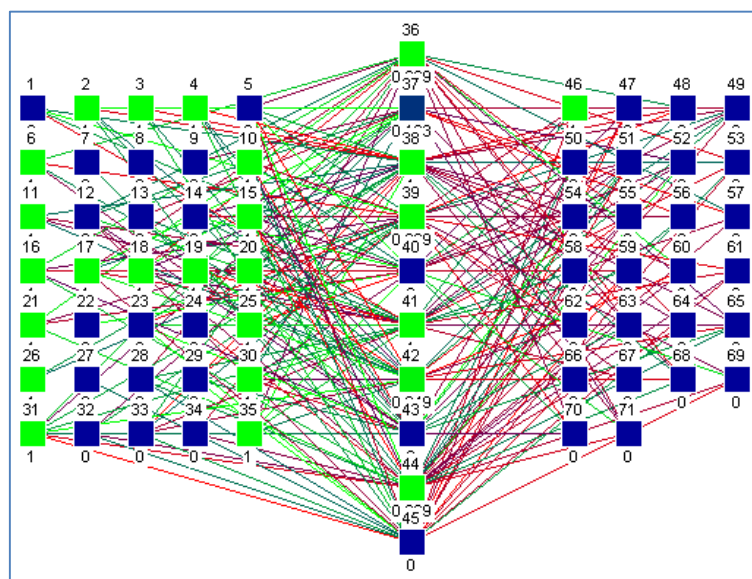
Számos OCR eszköz alkalmaz neurális hálózatot az egyes karakterek azonosítására. Neurális hálózatnak nevezzük azt a hardver vagy szoftver megvalósítású párhuzamos, elosztott működésre képes információ feldolgozó eszközt, amely:

- azonos, vagy hasonló típusú - általában nagyszámú - lokális feldolgozást végző műveleti elem, neuron (processing, element, neuron) többnyire rendezett topológiájú, nagymértékben összekapcsolt rendszeréből áll,
- rendelkezik tanulási algoritmussal (learning algorithm), mely általában minta alapján való tanulást jelent, és amely az információfeldolgozás módját határozza meg,
- rendelkezik a megtanult információ felhasználását lehetővé tevő információ előhívási, vagy röviden előhívási algoritmussal (recall algorithm). [19]

A neurális hálózatok működésénél tipikusan két fázist különböztethetünk meg. Az első fázis, melyet tanulási fázisnak nevezünk, a hálózat kialakítására szolgál, melynek során a hálózatba valamilyen módon beépítjük, eltároljuk a rendelkezésre álló mintákban rejtve meglevő információt. Eredményként egy információ-feldolgozó rendszert kapunk, melynek használatára általában a második fázisban, az előhívási fázisban kerül sor. A két fázis a legtöbb esetben, időben szétválik. [19]

A szövegelemek azonosításához alkalmazott neurális háló feladata az, hogy az egyes szegmenseket alkotó pixelek halmazából meghatározza a szegmensben kódolt karakter azonosítóját.

Az alkalmazott neurális háló három rétegből áll, melyek közül az első a bemeneti réteg, a második a rejtett réteg, míg a harmadik a kimeneti réteg (52. ábra). A bemeneti rétegre helyezük az azonosítandó szegmens pixelmátrixát, és a kimeneti rétegen fogjuk megkapni az azonosított karakter kódját. A köztes réteg végzi el a szegmens azonosítását. Topológiáját tekintve a hálózat egyirányú, és előrecsatolt, tanító eljárása a Back-Propagation (hiba visszaterjesztéses) algoritmus, munkafüggvénye pedig a szigmoid.



52. ábra - Szimbólumazonosító több kimenetű neurális háló topológia (forrás: JavaNNS)

A neurális háló kimeneti rétegének neuronjai a számítás eredményeképpen -0.5 és 0.5 közötti értéket kapnak eredményül. Az a neuron jelzi a bemeneti rétegre helyezett szegmens kódját, amelynek értéke szignifikánsan nagyobb, mint az összes többi neuron értéke. A kimeneti réteg két legmagasabb értékű neuronja közötti eltérés tehát mutatója lehet a neurális háló által szolgáltatott eredmény pontosságának.

A megvalósításhoz az Aforge.Net keretrendszer Neural Networks Computation eszköztár került felhasználásra. A hálózat betanításához a feldolgozandó könyv azonosított szegmensei lettek felhasználva úgy, hogy azok a neurális hálózat bemeneti rétegének méretére lettek kicsinyítve, és a tanítás során minden egyes bemeneti értékhez megadásra került a kimeneti rétegen megjelenítendő elvárt érték.

Ennél a megoldásnál a neurális hálózatnak 26 kisbetűt, 26 nagybetűt és 10 számjegyet kell megkülönböztetnie, tehát a kimeneti réteg 62 neuronból áll. A bemeneti réteg elemszámát a vizsgált szegmens befoglaló téglalapjának a mérete határozza meg, azonban mivel minden egyes szegmens különböző méretű, a neurális hálózat tanításakor (és természetesen a karakterek azonosításkor is) azokat transzformálni kell úgy, hogy a befoglaló téglalapjuk azonos méretű legyen.

A fejlesztés során 15×15 -ös bemeneti réteget lett alkalmazva (225 neuron), mivel a dokumentumokban leggyakrabban előforduló 11 és 12 pontos karaktereket ekkora méretűre úgy lehet transzformálni, hogy az eredmény megőrzi a karakterek formai jellegzetességeit. Ennél kisebb méretnél már komoly gondot jelenthet a karakterek átméretezése során elszenvedett információvesztés.

A karakterek átméretezése az alábbi képlettel végezhető el:

$$g(x, y) = f\left(\text{round}\left(\frac{1}{\lambda_x}x\right), \text{round}\left(\frac{1}{\lambda_y}y\right)\right) \quad (6.1)$$

Ahol g az átskálázott kép, f az eredeti kép, λ_x az eredeti és az átskálázott kép szélességének, míg λ_y a magasságának a hányadosa.

A több kimenetű neurális hálózattal kapcsolatban már a hálózat tanítása során számos probléma fölmerült:

- A neurális hálózat nagy mérete, és a felismerendő karakterek nagy száma miatt a tanítási folyamat hosszadalmas, a tanítás sikeressége esetleges.
- A bemeneti réteg elemszáma korlátos, nagyobb méretű karaktereknél (pl.: címsor) a kicsinyítés olyan mértékű, hogy az elkeni a karakterek formai jellegzetességeit.
Erre a problémára megoldást jelenthet, ha a különböző méretű karakterek azonosítására különböző neurális hálózatokat tanítunk be, de ez jelentős többletmunkával jár.
- A neurális hálózattal az egyes karakterek azonosítása elvégezhető, de a betűtípus, és a betű egyéb formai jellemzői (dőlt, félkövér, stb.) nem beazonosíthatóak.

Mivel már a fejlesztés korai fázisában kiderült, hogy ez a megoldás nem lesz működőképes, más megoldást kellett keresni a karakterek azonosítására.

Két kimenetű neurális hálózat rendszer alkalmazása

A több kimenetű neurális hálózathoz az egyik legnagyobb problémát az jelentette, hogy az azonosítandó karakterek nagy száma miatt a tanítási folyamat eredménye bizonytalan.

A problémára megoldást jelenthet, ha egy neurális hálózat betanítása és alkalmazása helyett annyi neurális hálózatot kerül betanításra, ahány különböző felismerendő szimbólumunk van.

Ennél a megoldásnál mindegyik hálózathoz hozzárendeljük azt a karaktert, amelynek a felismerésére betanítottuk. A hálózatok topológiája hasonlít a több kimenetű hálózathoz ismertetettel, mindössze annyi a különbség, hogy itt a kimeneten az összes hálózathoz mindössze két neuron található (53. ábra). Amennyiben az input rétegre illesztett bemenet megegyezik a betanított karakterrel, az első kimeneti neuron értéke 0.5-höz, míg a második -0.5-höz közelít. Ellenkező esetben az első neuron értéke közelít -0.5-höz, és a második 0.5-höz.

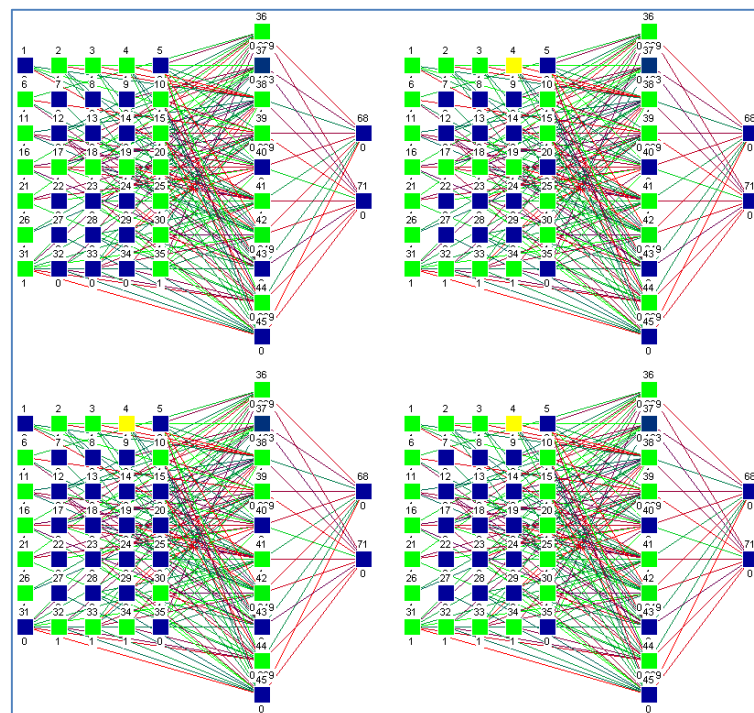
A megvalósítás során ezúttal is az Aforge.Net keretrendszer Neural Networks Computation eszköztárát került alkalmazásra. A hálózat betanításához a feldolgozandó könyv azonosított szegmensei lettek felhasználva a neurális hálózat bemeneti rétegének méretéhez kicsinyítve. A tanítás során minden egyes bemeneti értékhez megadásra kerültek a kimeneti rétegen megjelenítendő elvárt értékek.

Egy szegmens vizsgálatakor azt 15*15-ös méretűre kell transzformálni annak érdekében, hogy ráhelyezhető legyen az összes neurális hálózat bemeneti rétegére. A kiértékelés során azt kell vizsgálni, hogy melyik neurális hálózat kimeneti rétegének két neuronja közelíti leginkább a találat esetén elvárt értékeket.

A vizsgált szegmens által jelölt szimbólum így az a karakter lesz, amelyet a legjobb eredményt produkáló hálózathoz rendeltünk.

Bár a több kimenetű neurális hálózathoz képest számottevő előrelépést jelentett ez a megoldás, továbbra is számos megoldandó probléma maradt:

- Az eljárással elérhető találati arány, kb. 70%-os volt, ami elmaradt az elvárt pontosságtól.
- A különböző méretű karakterek azonosítása érdekében különböző elemszámú bemeneti réteggel rendelkező hálózat rendszert kell betanítani. Bár egy két kimenetű hálózat betanítása sokkal gyorsabb folyamat, mint egy több kimenetű hálózat betanítása, összességében ez a módszer mégis sokkal több időt vesz igénybe a hálózatok nagy száma miatt.
- Az eljárással továbbra se lehet az egyes betűtípusokat meghatározni.



53. ábra - Szimbólumazonosító két kimenetű neurális háló topológia (forrás: JavaNNS)

8.2 A LUCA Converter szegmentáló eljárásának teszteredményei

109 ATLAG, SZÁMTANI KÖZÉP

BEVEZETŐ

A feladatlap ötletét merész és bátorjaként az „atlag”, „átlagosság” szavakból. Hisz Val azt mondja, hogy a másik ember átlagosan havi 145 ezer Ft-ot a nettó jövedelménél (mivel mindenki azonos a gondok, tehát 12-143 = 1740 ezer (1 millió 740 ezer) forint volt az a sorús kapott nettó jövedelménél összege. Azért nemcsak ezt tudtuk meg, hogy valójában mekkora nettó jövedelmek kaptak Vali barátai. Láthat, hogy minden hónapban 145 ezer forintot, de is lehet, hogy 5 hónapig kapott havi 110 ezer forintot, majd 7 hónapig kapott

havi 175 ezer forintot, 50 ez a lehet, hogy 11 hónapig kapott egyelőre általánosan jövedelmén sem volt, de volt egy hónap, amelyben éppen 1 millió 740 ezer forint nettó jövedelmét emelt.

Sokfeltehetően átlagosabban az egyes feladatok netójövedelmére vagy végül a havi átlag 145 ezer Ft legrosszabbul ötletet kaptunk azonos, hogy ha a havi nettó jövedelmek összege, majd az összeg elosztjuk 12-vel, akkor a havi átlag 145 ezer lesz.

PELDA

Aranka juttatás 150 ezer, Berta havi 164 ezer, mások havi 134 ezer forint volt a nettó jövedelménél. Berta minden hónapban ugyanannyi a nettó jövedelménél, az első jövedelménél összegeit megkapjuk. Aranka:

a) Mennyi volt az első jövedelménél Anna jövedelménél?

b) Mekkora volt Berta havi nettó jövedelménél?

Megoldás

a) Anna első jövedelménél összege: $150 + 164 + 134 = 448$ ezer forint volt.

FELELD

1. Anna az első jövedelménél átlagosan havi nettó 150 ezer forint jövedelménél (itt az, ami az havi átlag átlagosan havi 150 ezer forint jövedelménél). Az első jövedelménél havi átlagban 174 ezer forint jövedelménél volt.
2. Mekkora volt Anna első nettó jövedelménél?
3. Csak minden hónapban ugyanannyi jövedelménél. Egy év alatt ugyanannyi havi a nettó jövedelménél, mint Aranka. Mennyi pénzt kapott havi jövedelménél?
4. Duci havi nettó jövedelménél 150 ezer, a 150 ezer és a 174 ezer számtani közepén volt

b) Berta havi nettó jövedelménél $150 + 164 + 134 = 448$ ezer forint volt, ami Anna első jövedelménél átlagos jövedelménél.

Megoldás

a) $150 + 164 + 134$ jövedelménél a 150, a 164 és a 134 átlaga, ma, más szóval számtani közepén (lásd meg. 39. oldal, 39. és 38. lecke).

109 ATLAG, SZÁMTANI KÖZÉP

BEVEZETŐ

A feladatlap ötletét merész és bátorjaként az „atlag”, „átlagosság” szavakból. Hisz Val azt mondja, hogy a másik ember átlagosan havi 145 ezer Ft-ot a nettó jövedelménél (mivel mindenki azonos a gondok, tehát 12-143 = 1740 ezer (1 millió 740 ezer) forint volt az a sorús kapott nettó jövedelménél összege. Azért nemcsak ezt tudtuk meg, hogy valójában mekkora nettó jövedelmek kaptak Vali barátai. Láthat, hogy minden hónapban 145 ezer forintot, de is lehet, hogy 5 hónapig kapott havi 110 ezer forintot, majd 7 hónapig kapott

havi 175 ezer forintot, 50 ez a lehet, hogy 11 hónapig kapott egyelőre általánosan jövedelmén sem volt, de volt egy hónap, amelyben éppen 1 millió 740 ezer forint nettó jövedelmét emelt.

Sokfeltehetően átlagosabban az egyes feladatok netójövedelmére vagy végül a havi átlag 145 ezer Ft legrosszabbul ötletet kaptunk azonos, hogy ha a havi nettó jövedelmek összege, majd az összeg elosztjuk 12-vel, akkor a havi átlag 145 ezer lesz.

PELDA

Aranka juttatás 150 ezer, Berta havi 164 ezer, mások havi 134 ezer forint volt a nettó jövedelménél. Berta minden hónapban ugyanannyi a nettó jövedelménél, az első jövedelménél összegeit megkapjuk. Aranka:

a) Mennyi volt az első jövedelménél Anna jövedelménél?

b) Mekkora volt Berta havi nettó jövedelménél?

Megoldás

a) Anna első jövedelménél összege: $150 + 164 + 134 = 448$ ezer forint volt.

b) Berta havi nettó jövedelménél $150 + 164 + 134 = 448$ ezer forint volt, ami Anna első jövedelménél átlagos jövedelménél.

Megoldás

a) $150 + 164 + 134$ jövedelménél a 150, a 164 és a 134 átlaga, ma, más szóval számtani közepén (lásd meg. 39. oldal, 39. és 38. lecke).

FELELD

1. Anna az első jövedelménél átlagosan havi nettó 150 ezer forint jövedelménél (itt az, ami az havi átlag átlagosan havi 150 ezer forint jövedelménél). Az első jövedelménél havi átlagban 174 ezer forint jövedelménél volt.
2. Mekkora volt Anna első nettó jövedelménél?
3. Csak minden hónapban ugyanannyi jövedelménél. Egy év alatt ugyanannyi havi a nettó jövedelménél, mint Aranka. Mennyi pénzt kapott havi jövedelménél?
4. Duci havi nettó jövedelménél 150 ezer, a 150 ezer és a 174 ezer számtani közepén volt

1. Aranka juttatás 150 ezer, Berta havi 164 ezer, mások havi 134 ezer forint jövedelménél. Berta minden hónapban ugyanannyi jövedelménél, az első jövedelménél összegeit megkapjuk. Aranka:
2. Duci havi nettó jövedelménél 150 ezer, a 150 ezer és a 174 ezer számtani közepén volt

109

MATEMATIKA A MINDENHÓNAPBAN | 9. évfolyam, 2020/2021

109

MATEMATIKA A MINDENHÓNAPBAN | 9. évfolyam, 2020/2021

54. ábra - Matematika 10 tankönyv 32. oldal szegmentálásának eredménye

akkor nem fogadjuk el az élelel csomagolt anyagot. A táblázat tartalmazza a mérési eredményeket.

Mérés ideje (óra:perc)	Összeg
23:26	4
23:09	2
24:21	3
24:50	1

a) Mennyi a víz mennyiség (számszázharmincöttes átlagot képezz!)
b) Előfordult-e az élelel csomagolt anyag?

3. A Kati bátyján kedélyes viza alakult ki egy most bevezetett kerekül „akadémikus” jövedelmének alakulásánál.

Ar. bátyjának azt javasolta, hogy az akadémikus „jövendő” átlag két hónap múlva emelkedjen 20%-kal, majd további két hónap múlva további 10%-os emelkedést tapasztaljon. Az akadémikus bátyjának ezután további öt hónapos elhalasztással, azaz 19%-os emelkedéssel hajlandó volt. Szerinte így a jövedelmének egy hónap után a termék ára. Tegyük fel, hogy a termék ára 1000 Ft.

4. A Kati bátyján kedélyes viza alakult ki egy most bevezetett kerekül „akadémikus” jövedelmének alakulásánál.

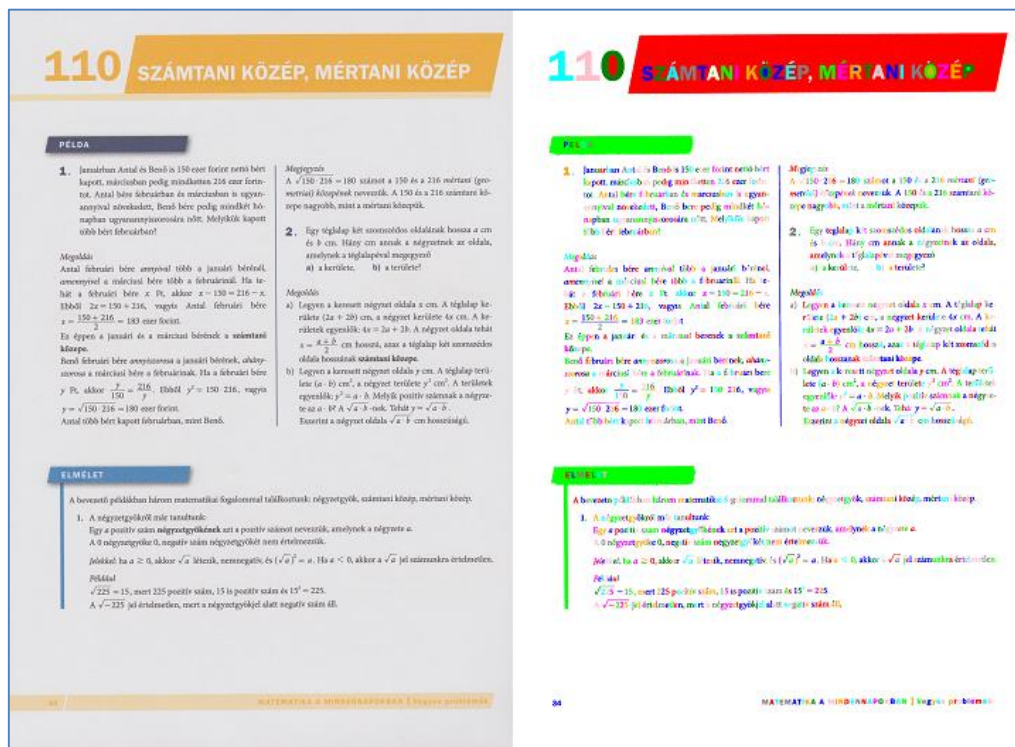
Ar. bátyjának azt javasolta, hogy az akadémikus „jövendő” átlag két hónap múlva emelkedjen 20%-kal, majd további két hónap múlva további 10%-os emelkedést tapasztaljon. Az akadémikus bátyjának ezután további öt hónapos elhalasztással, azaz 19%-os emelkedéssel hajlandó volt. Szerinte így a jövedelmének egy hónap után a termék ára. Tegyük fel, hogy a termék ára 1000 Ft.

HÁZI FELADAT

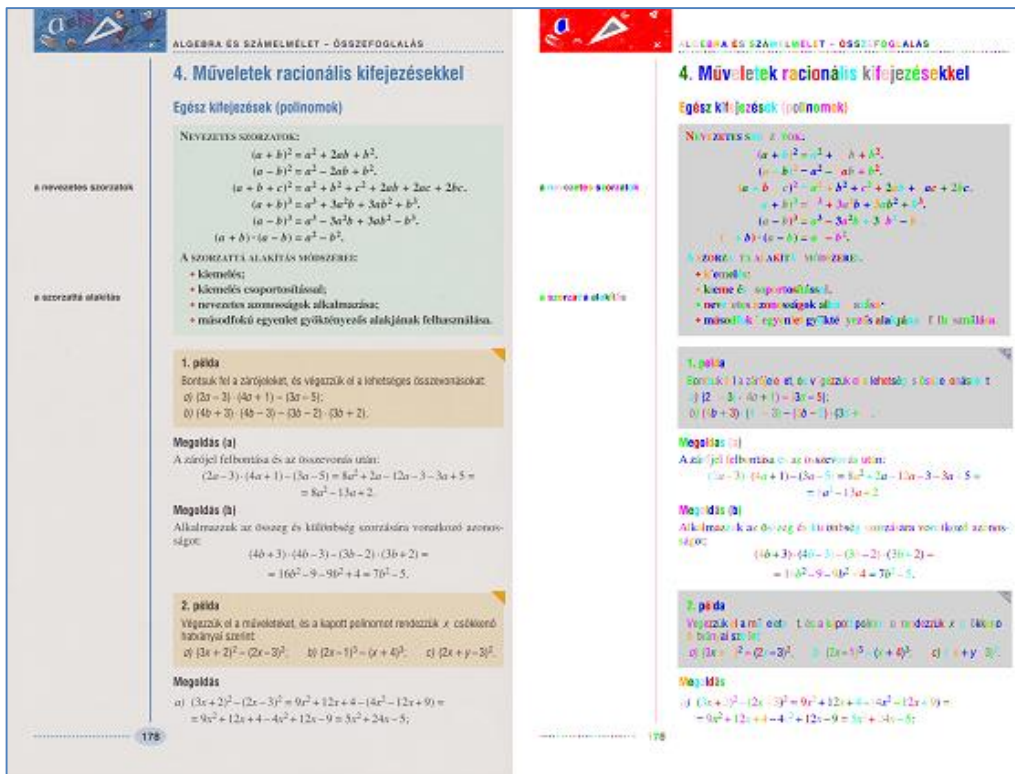
1. A táblázatban az adatszolgáltatás körében a munkaadók adatai. Egy rendszerben való a körbeírás.

Állás	Állás	Állás
1. Állás	2. Állás	3. Állás
4. Állás	5. Állás	6. Állás
7. Állás	8. Állás	9. Állás
10. Állás	11. Állás	12. Állás
13. Állás	14. Állás	15. Állás
16. Állás	17. Állás	18. Állás
19. Állás	20. Állás	21. Állás
22. Állás	23. Állás	24. Állás
25. Állás	26. Állás	27. Állás
28. Állás	29. Állás	30. Állás
31. Állás	32. Állás	33. Állás
34. Állás	35. Állás	36. Állás
37. Állás	38. Állás	39. Állás
40. Állás	41. Állás	42. Állás
43. Állás	44. Állás	45. Állás
46. Állás	47. Állás	48. Állás
49. Állás	50. Állás	51. Állás
52. Állás	53. Állás	54. Állás
55. Állás	56. Állás	57. Állás
58. Állás	59. Állás	60. Állás
61. Állás	62. Állás	63. Állás
64. Állás	65. Állás	66. Állás
67. Állás	68. Állás	69. Állás
70. Állás	71. Állás	72. Állás
73. Állás	74. Állás	75. Állás
76. Állás	77. Állás	78. Állás
79. Állás	80. Állás	81. Állás
82. Állás	83. Állás	84. Állás
85. Állás	86. Állás	87. Állás
88. Állás	89. Állás	90. Állás
91. Állás	92. Állás	93. Állás
94. Állás	95. Állás	96. Állás
97. Állás	98. Állás	99. Állás
100. Állás	101. Állás	102. Állás
103. Állás	104. Állás	105. Állás
106. Állás	107. Állás	108. Állás
109. Állás	110. Állás	111. Állás
112. Állás	113. Állás	114. Állás
115. Állás	116. Állás	117. Állás
118. Állás	119. Állás	120. Állás
121. Állás	122. Állás	123. Állás
124. Állás	125. Állás	126. Állás
127. Állás	128. Állás	129. Állás
130. Állás	131. Állás	132. Állás
133. Állás	134. Állás	135. Állás
136. Állás	137. Állás	138. Állás
139. Állás	140. Állás	141. Állás
142. Állás	143. Állás	144. Állás
145. Állás	146. Állás	147. Állás
148. Állás	149. Állás	150. Állás
151. Állás	152. Állás	153. Állás
154. Állás	155. Állás	156. Állás
157. Állás	158. Állás	159. Állás
160. Állás	161. Állás	162. Állás
163. Állás	164. Állás	165. Állás
166. Állás	167. Állás	168. Állás
169. Állás	170. Állás	171. Állás
172. Állás	173. Állás	174. Állás
175. Állás	176. Állás	177. Állás
178. Állás	179. Állás	180. Állás
181. Állás	182. Állás	183. Állás
184. Állás	185. Állás	186. Állás
187. Állás	188. Állás	189. Állás
190. Állás	191. Állás	192. Állás
193. Állás	194. Állás	195. Állás
196. Állás	197. Állás	198. Állás
199. Állás	200. Állás	201. Állás
202. Állás	203. Állás	204. Állás
205. Állás	206. Állás	207. Állás
208. Állás	209. Állás	210. Állás
211. Állás	212. Állás	213. Állás
214. Állás	215. Állás	216. Állás
217. Állás	218. Állás	219. Állás
220. Állás	221. Állás	

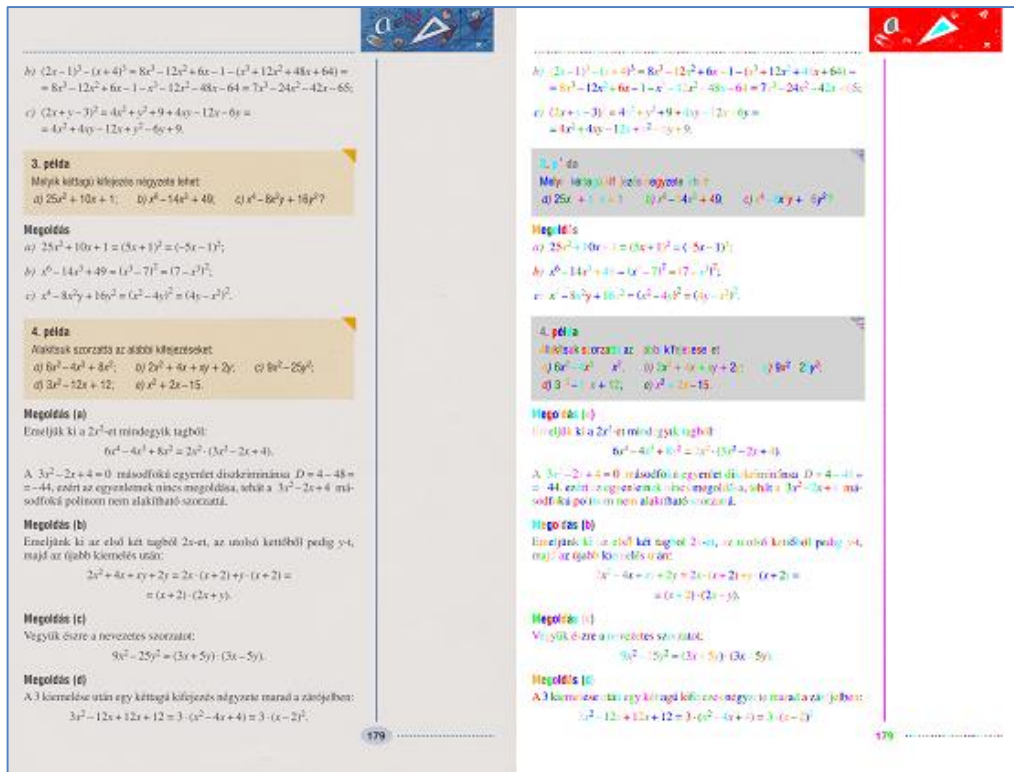
55. ábra - Matematika 10 tankönyv 33. oldal szegmentálásának eredménye



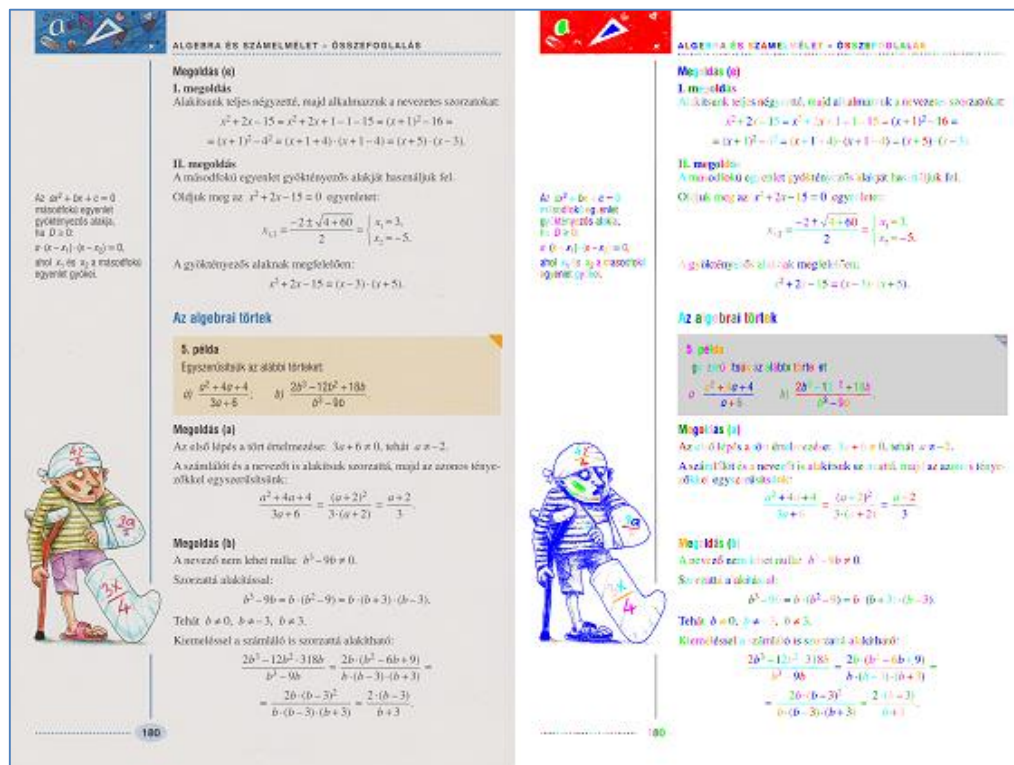
56. ábra - Matematika 10 tankönyv 34. oldal szegmentálásának eredménye



57. ábra - Sokszínű matematika 12 tankönyv 178. oldal szegmentálásának eredménye



58. ábra - Sokszínű matematika 12 tankönyv 179. oldal szegmentálásának eredménye



59. ábra - Sokszínű matematika 12 tankönyv 180. oldal szegmentálásának eredménye

85

110 SZÁMTANI KÖZÉP, MÉRTANI KÖZÉP

PELDÁ

1. Januárban Antal és Benő is 150 ezer forint nettó bért kapott, márciusban pedig mindketten 216 ezer forintot. Antal bére februárban és márciusban is ugyanannyival növekedett, Benő bére pedig mindkét hónapban ugyanannyiszorosára nőtt. Melyikük kapott több bért februárban?

Megoldás

Antal februári bére annyival több a januári béreénél, annennyivel a márciusi bére több a februáriénál. Ha tehát a februári bére x Ft, akkor $x + 150 = 216 - x$. Ebből $2x = 150 + 216$, vagyis Antal februári bére $x = \frac{150 + 216}{2} = 183$ ezer forint.

Ez éppen a januári és a márciusi bérének a számtani közepe.

Benő februári bére annyiszorosára a januári béreétől, ahány-szerosa a márciusi bére a februáriénak. Ha a februári bére y Ft, akkor $\frac{y}{150} = \frac{216}{y}$. Ebből $y^2 = 150 \cdot 216$, vagyis $y = \sqrt{150 \cdot 216} = 180$ ezer forint.

Antal több bért kapott februárban, mint Benő.

Megjegyzés

A $\sqrt{150 \cdot 216} = 180$ számot a 150 és a 216 mértani (geometriai) közepének nevezzük. A 150 és a 216 számtani közepé nagyobb, mint a mértani közepük.

2. Egy téglalap két szomszédos oldalának hossza a cm és b cm. Hány cm annak a négyzetnek az oldala, amelynek a téglalapével megegyező a) a kerülete, b) a területe?

Megoldás

a) Legyen a keresett négyzet oldala x cm. A téglalap kerülete $(2a + 2b)$ cm, a négyzet kerülete $4x$ cm. A kerületek egyenlősége $4x = 2a + 2b$. A négyzet oldala tehát $x = \frac{a + b}{2}$ cm, ami az a és b számtani közepe.

b) Legyen a keresett négyzet oldala y cm. A téglalap területe $(a \cdot b)$ cm², a négyzet területe y^2 cm². A területek egyenlősége $y^2 = a \cdot b$. Melyik pozitív számnak a négyzetese az $a \cdot b$? A $\sqrt{a \cdot b}$ -nak. Tehát $y = \sqrt{a \cdot b}$. Ezért a négyzet oldala $\sqrt{a \cdot b}$ cm hosszúságú.

ELMÉLET

A következő példákban három matematikai fogalommal találkoztunk: négyzetgyök, számtani közép, mértani közép.

1. A négyzetgyökök mire használunk?

Egy a pozitív szám négyzetgyökének azt a pozitív számot nevezzük, amelynek a négyzete a . A 0 négyzetgyöke 0, negatív szám négyzetgyökét nem értelmezzük.

Például ha $a \geq 0$, akkor $\sqrt{a}$ létezik, nemnegatív, és $(\sqrt{a})^2 = a$. Ha $a < 0$, akkor $\sqrt{a}$ jel nemrekeszt értelmezés.

Példák

$\sqrt{225} = 15$, mert 225 pozitív szám, 15 is pozitív szám és $15^2 = 225$.
A $\sqrt{-225}$ jel értelmezés, mert a négyzetgyökök alatt negatív szám áll.

110 SZÁMTANI KÖZÉP, MÉRTANI KÖZÉP

PELDÁ

1. Januárban Antal és Benő is 150 ezer forint nettó bért kapott, márciusban pedig mindketten 216 ezer forintot. Antal bére februárban és márciusban is ugyanannyival növekedett, Benő bére pedig mindkét hónapban ugyanannyiszorosára nőtt. Melyikük kapott több bért februárban?

Megoldás

Antal februári bére annyival több a januári béreénél, annennyivel a márciusi bére több a februáriénál. Ha tehát a februári bére x Ft, akkor $x + 150 = 216 - x$. Ebből $2x = 150 + 216$, vagyis Antal februári bére $x = \frac{150 + 216}{2} = 183$ ezer forint.

Ez éppen a januári és a márciusi bérének a számtani közepe.

Benő februári bére annyiszorosára a januári béreétől, ahány-szerosa a márciusi bére a februáriénak. Ha a februári bére y Ft, akkor $\frac{y}{150} = \frac{216}{y}$. Ebből $y^2 = 150 \cdot 216$, vagyis $y = \sqrt{150 \cdot 216} = 180$ ezer forint.

Antal több bért kapott februárban, mint Benő.

Megjegyzés

A $\sqrt{150 \cdot 216} = 180$ számot a 150 és a 216 mértani (geometriai) közepének nevezzük. A 150 és a 216 számtani közepé nagyobb, mint a mértani közepük.

2. Egy téglalap két szomszédos oldalának hossza a cm és b cm. Hány cm annak a négyzetnek az oldala, amelynek a téglalapével megegyező a) a kerülete, b) a területe?

Megoldás

a) Legyen a keresett négyzet oldala x cm. A téglalap kerülete $(2a + 2b)$ cm, a négyzet kerülete $4x$ cm. A kerületek egyenlősége $4x = 2a + 2b$. A négyzet oldala tehát $x = \frac{a + b}{2}$ cm, ami az a és b számtani közepe.

b) Legyen a keresett négyzet oldala y cm. A téglalap területe $(a \cdot b)$ cm², a négyzet területe y^2 cm². A területek egyenlősége $y^2 = a \cdot b$. Melyik pozitív számnak a négyzetese az $a \cdot b$? A $\sqrt{a \cdot b}$ -nak. Tehát $y = \sqrt{a \cdot b}$. Ezért a négyzet oldala $\sqrt{a \cdot b}$ cm hosszúságú.

ELMÉLET

A következő példákban három matematikai fogalommal találkoztunk: négyzetgyök, számtani közép, mértani közép.

1. A négyzetgyökök mire használunk?

Egy a pozitív szám négyzetgyökének azt a pozitív számot nevezzük, amelynek a négyzete a . A 0 négyzetgyöke 0, negatív szám négyzetgyökét nem értelmezzük.

Például ha $a \geq 0$, akkor $\sqrt{a}$ létezik, nemnegatív, és $(\sqrt{a})^2 = a$. Ha $a < 0$, akkor $\sqrt{a}$ jel nemrekeszt értelmezés.

Példák

$\sqrt{225} = 15$, mert 225 pozitív szám, 15 is pozitív szám és $15^2 = 225$.
A $\sqrt{-225}$ jel értelmezés, mert a négyzetgyökök alatt negatív szám áll.

62. ábra - Matematika 10 tankönyv 34. oldal karakter azonosításának eredménye

8.4 A LUCA Converter összefüggő szövegelem azonosításának teszt eredményei

109 ÁTLAG, SZÁMTANI KÖZÉP

BEVEZETŐ

A hétköznapi életben több esetben is használjuk az „átlag”, „átlagosság” szavakat. Ha Vali azt mondja, hogy a múlt évben átlagosan havi 145 ezer Ft volt a nettó jövedelme, akkor mindenki arra gondol, hogy 12: 145 = 1740 ezer (1 millió 740 ezer) forint volt az év során kapott nettó jövedelmének összege.

Azért senkinek sem tudnánk meg, hogy valójában mekkora nettó összegre jutott Vali havonta. Lehet, hogy minden hónapban 145 ezer forintot, de az is lehet, hogy 5 hónapig kevesebb havi 103 ezer forintot, majd 7 hónapig kevesebb

havi 175 ezer forintot, 50t, az is lehet, hogy 11 hónapig kevesebb egyetlen 515 ezer nettó jövedelmet sem volt, de volt egy hónap, amelyben éppen 1 millió 740 ezer forint nettó jövedelmet ért el.

Sokfeladatosan alakultak az egyes hónapok nettó jövedelme is, hogy végül a havi átlag 145 ezer Ft legyen. Mindvégül esetben így azonban, hogy ha a havi nettó jövedelmek összeadják, majd az összeget elosztjuk 12-vel, akkor a hátrébb 145 ezer lesz.

PELDA

Annaknak (jarsáékban 150 ezer, februárban 164 ezer, márciusban 154 ezer forint volt a nettó jövedelme. Borizák minden hónapban ugyanannyi volt a nettó jövedelme, az első negyedében átlagosan 150 ezer forintot.

a) Mennyi volt az első negyedében Anna átlagos jövedelme?

b) Mekkora volt Berta havi nettó jövedelme?

Megoldás
a) Anna első negyedében átlagosan 150 + 164 + 154 = 468 ezer forint volt.

FELADAT

- Anna az első negyedében átlagosan havi nettó 150 ezer forint jövedelmet ért el, azaz hat hónapig átlagosan havi 150 ezer forintot keresett. Az utolsó negyedében havi átlagosan 174 ezer forint jövedelme volt.
 - Mekkora volt Anna éves nettó jövedelme?
 - Céla minden hónapban ugyanannyi keresett. Egy év alatt ugyanakkora lett a nettó jövedelme, mint Annának. Mennyi pénzt kapott havonta Céla?
 - Dóri havi nettó jövedelme tavaly a 150 ezer, a 150 ezer és a 174 ezer számtani közepe volt.

- Első havi nettó jövedelme 10 ezer forintot keresett volt, mint Annáé. Mennyi pénzt kapott havonta Berta?
- Egy gépkocsi eladásáért az egyik ártárgyboltban pontosan 24 gramm van értékelve. Az ártárgyboltban ártárgyak eladásánál minden gépkocsi, az ártárgyboltban a kapott ár 50%-át adja. Ha ez az átlag 50%-nál többet tér el a szükséges mennyiségtől,

109 ÁTLAG, SZÁMTANI KÖZÉP

BEVEZETŐ

A hétköznapi életben több esetben is használjuk az „átlag”, „átlagosság” szavakat. Ha Vali azt mondja, hogy a múlt évben átlagosan havi 145 ezer Ft volt a nettó jövedelme, akkor mindenki arra gondol, hogy 12: 145 = 1740 ezer (1 millió 740 ezer) forint volt az év során kapott nettó jövedelmének összege.

Azért senkinek sem tudnánk meg, hogy valójában mekkora nettó összegre jutott Vali havonta. Lehet, hogy minden hónapban 145 ezer forintot, de az is lehet, hogy 5 hónapig kevesebb havi 103 ezer forintot, majd 7 hónapig kevesebb

havi 175 ezer forintot, 50t, az is lehet, hogy 11 hónapig kevesebb egyetlen 515 ezer nettó jövedelmet sem volt, de volt egy hónap, amelyben éppen 1 millió 740 ezer forint nettó jövedelmet ért el.

Sokfeladatosan alakultak az egyes hónapok nettó jövedelme is, hogy végül a havi átlag 145 ezer Ft legyen. Mindvégül esetben így azonban, hogy ha a havi nettó jövedelmek összeadják, majd az összeget elosztjuk 12-vel, akkor a hátrébb 145 ezer lesz.

PELDA

Annaknak (jarsáékban 150 ezer, februárban 164 ezer, márciusban 154 ezer forint volt a nettó jövedelme. Borizák minden hónapban ugyanannyi volt a nettó jövedelme, az első negyedében átlagosan 150 ezer forintot.

a) Mennyi volt az első negyedében Anna átlagos jövedelme?

b) Mekkora volt Berta havi nettó jövedelme?

Megoldás
a) Anna első negyedében átlagosan 150 + 164 + 154 = 468 ezer forint volt.

FELADAT

- Anna az első negyedében átlagosan havi nettó 150 ezer forint jövedelmet ért el, azaz hat hónapig átlagosan havi 150 ezer forintot keresett. Az utolsó negyedében havi átlagosan 174 ezer forint jövedelme volt.
 - Mekkora volt Anna éves nettó jövedelme?
 - Céla minden hónapban ugyanannyi keresett. Egy év alatt ugyanakkora lett a nettó jövedelme, mint Annának. Mennyi pénzt kapott havonta Céla?
 - Dóri havi nettó jövedelme tavaly a 150 ezer, a 150 ezer és a 174 ezer számtani közepe volt.

- Első havi nettó jövedelme 10 ezer forintot keresett volt, mint Annáé. Mennyi pénzt kapott havonta Berta?
- Egy gépkocsi eladásáért az egyik ártárgyboltban pontosan 24 gramm van értékelve. Az ártárgyboltban ártárgyak eladásánál minden gépkocsi, az ártárgyboltban a kapott ár 50%-át adja. Ha ez az átlag 50%-nál többet tér el a szükséges mennyiségtől,

63. ábra - Matematika 10 tankönyv 32. oldal hasáb és bekezdés azonosításának eredménye

109 ÁTLAG, SZÁMTANI KÖZÉP

BEVEZETŐ

A hétköznapi életben több esetben is használjuk az „átlag”, „átlagosság” szavakat. Ha Vali azt mondja, hogy a múlt évben átlagosan havi 145 ezer Ft volt a nettó jövedelme, akkor mindenki arra gondol, hogy 12: 145 = 1740 ezer (1 millió 740 ezer) forint volt az év során kapott nettó jövedelmének összege.

Azért senkinek sem tudnánk meg, hogy valójában mekkora nettó összegre jutott Vali havonta. Lehet, hogy minden hónapban 145 ezer forintot, de az is lehet, hogy 5 hónapig kevesebb havi 103 ezer forintot, majd 7 hónapig kevesebb

havi 175 ezer forintot, 50t, az is lehet, hogy 11 hónapig kevesebb egyetlen 515 ezer nettó jövedelmet sem volt, de volt egy hónap, amelyben éppen 1 millió 740 ezer forint nettó jövedelmet ért el.

Sokfeladatosan alakultak az egyes hónapok nettó jövedelme is, hogy végül a havi átlag 145 ezer Ft legyen. Mindvégül esetben így azonban, hogy ha a havi nettó jövedelmek összeadják, majd az összeget elosztjuk 12-vel, akkor a hátrébb 145 ezer lesz.

PELDA

Annaknak (jarsáékban 150 ezer, februárban 164 ezer, márciusban 154 ezer forint volt a nettó jövedelme. Borizák minden hónapban ugyanannyi volt a nettó jövedelme, az első negyedében átlagosan 150 ezer forintot.

a) Mennyi volt az első negyedében Anna átlagos jövedelme?

b) Mekkora volt Berta havi nettó jövedelme?

Megoldás
a) Anna első negyedében átlagosan 150 + 164 + 154 = 468 ezer forint volt.

FELADAT

- Anna az első negyedében átlagosan havi nettó 150 ezer forint jövedelmet ért el, azaz hat hónapig átlagosan havi 150 ezer forintot keresett. Az utolsó negyedében havi átlagosan 174 ezer forint jövedelme volt.
 - Mekkora volt Anna éves nettó jövedelme?
 - Céla minden hónapban ugyanannyi keresett. Egy év alatt ugyanakkora lett a nettó jövedelme, mint Annának. Mennyi pénzt kapott havonta Céla?
 - Dóri havi nettó jövedelme tavaly a 150 ezer, a 150 ezer és a 174 ezer számtani közepe volt.

- Első havi nettó jövedelme 10 ezer forintot keresett volt, mint Annáé. Mennyi pénzt kapott havonta Berta?
- Egy gépkocsi eladásáért az egyik ártárgyboltban pontosan 24 gramm van értékelve. Az ártárgyboltban ártárgyak eladásánál minden gépkocsi, az ártárgyboltban a kapott ár 50%-át adja. Ha ez az átlag 50%-nál többet tér el a szükséges mennyiségtől,

109 ÁTLAG, SZÁMTANI KÖZÉP

BEVEZETŐ

A hétköznapi életben több esetben is használjuk az „átlag”, „átlagosság” szavakat. Ha Vali azt mondja, hogy a múlt évben átlagosan havi 145 ezer Ft volt a nettó jövedelme, akkor mindenki arra gondol, hogy 12: 145 = 1740 ezer (1 millió 740 ezer) forint volt az év során kapott nettó jövedelmének összege.

Azért senkinek sem tudnánk meg, hogy valójában mekkora nettó összegre jutott Vali havonta. Lehet, hogy minden hónapban 145 ezer forintot, de az is lehet, hogy 5 hónapig kevesebb havi 103 ezer forintot, majd 7 hónapig kevesebb

havi 175 ezer forintot, 50t, az is lehet, hogy 11 hónapig kevesebb egyetlen 515 ezer nettó jövedelmet sem volt, de volt egy hónap, amelyben éppen 1 millió 740 ezer forint nettó jövedelmet ért el.

Sokfeladatosan alakultak az egyes hónapok nettó jövedelme is, hogy végül a havi átlag 145 ezer Ft legyen. Mindvégül esetben így azonban, hogy ha a havi nettó jövedelmek összeadják, majd az összeget elosztjuk 12-vel, akkor a hátrébb 145 ezer lesz.

PELDA

Annaknak (jarsáékban 150 ezer, februárban 164 ezer, márciusban 154 ezer forint volt a nettó jövedelme. Borizák minden hónapban ugyanannyi volt a nettó jövedelme, az első negyedében átlagosan 150 ezer forintot.

a) Mennyi volt az első negyedében Anna átlagos jövedelme?

b) Mekkora volt Berta havi nettó jövedelme?

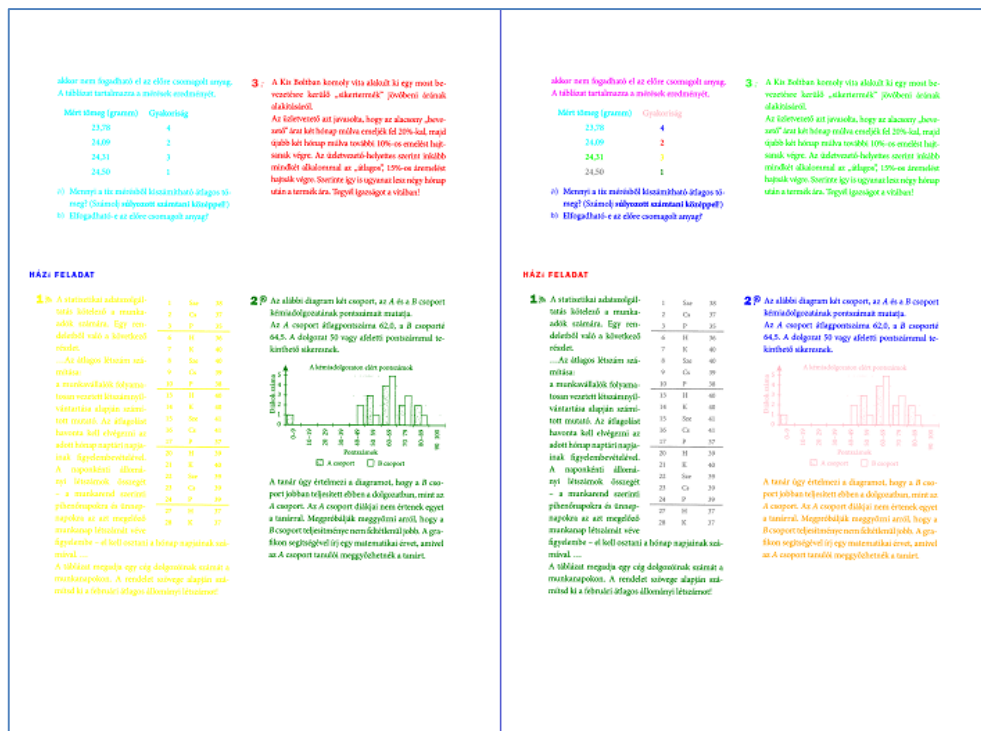
Megoldás
a) Anna első negyedében átlagosan 150 + 164 + 154 = 468 ezer forint volt.

FELADAT

- Anna az első negyedében átlagosan havi nettó 150 ezer forint jövedelmet ért el, azaz hat hónapig átlagosan havi 150 ezer forintot keresett. Az utolsó negyedében havi átlagosan 174 ezer forint jövedelme volt.
 - Mekkora volt Anna éves nettó jövedelme?
 - Céla minden hónapban ugyanannyi keresett. Egy év alatt ugyanakkora lett a nettó jövedelme, mint Annának. Mennyi pénzt kapott havonta Céla?
 - Dóri havi nettó jövedelme tavaly a 150 ezer, a 150 ezer és a 174 ezer számtani közepe volt.

- Első havi nettó jövedelme 10 ezer forintot keresett volt, mint Annáé. Mennyi pénzt kapott havonta Berta?
- Egy gépkocsi eladásáért az egyik ártárgyboltban pontosan 24 gramm van értékelve. Az ártárgyboltban ártárgyak eladásánál minden gépkocsi, az ártárgyboltban a kapott ár 50%-át adja. Ha ez az átlag 50%-nál többet tér el a szükséges mennyiségtől,

64. ábra - Matematika 10 tankönyv 32. oldal sor és szó azonosításának eredménye



65. ábra - Matematika 10 tankönyv 33. oldal hasáb és bekezdés azonosításának eredménye



66. ábra - Matematika 10 tankönyv 33. oldal sor és szó azonosításának eredménye

110 SZÁMTANI KÖZÉP, MÉRTANI KÖZÉP

PELDÁ

1. Janoskának Antal és Bend is 150 ezer forint nettó bért kapott, márciusban pedig mindketten 216 ezer forintot. Antal bért februárban és márciusban is ugyanannyal töltötte, Bend bért pedig mindkét hónapban ugyanannyiszorosára nőtt. Melyikük kapott több bért februárban?

Megoldás
Antal februári bért annyival több a januári bérével, amennyivel a márciusi bért több a februárián. Ha tehát a februári bért x Ft, akkor $x = 150 + 216 - x$. Ebből $2x = 150 + 216$, vagyis Antal februári bért $x = \frac{150+216}{2} = 183$ ezer forint.

Ez éppen a januári és a márciusi bértök a számtani közep.

Bend februári bért annyiszorosa a januári bértnök, akárcsosa a márciusi bért a februárián. Ha a februári bért y Ft, akkor $\frac{y}{150} = \frac{216}{y}$, ebből $y^2 = 150 \cdot 216$, vagyis $y = \sqrt{150 \cdot 216} = 180$ ezer forint.
Antal több bért kapott februárban, mint Bend.

ELMÉLET

A bevezető példában három matematikai fogalommal találkozunk: négyzetgyök, számtani közép, mértani közép.

1. A négyzetgyökök már tanultak:

Egy a pozitív szám négyzetgyökének azt a pozitív számot nevezzük, amelynek a négyzete a .

A 0 négyzetgyöke 0, negatív szám négyzetgyöke nem értelmezett.

Jelölés: ha $a \geq 0$, akkor $\sqrt{a}$ létezik, nemnegatív, és $(\sqrt{a})^2 = a$. Ha $a < 0$, akkor $\sqrt{a}$ jel számunkra értelmezetlen.

Példák

$\sqrt{225} = 15$, mert 225 pozitív szám, 15 a pozitív szám és $15^2 = 225$.

$\sqrt{-225}$ jel értelmezetlen, mert a négyzetgyökök alatt negatív szám áll.

Megjegyzés

A $\sqrt{150 \cdot 216} = 180$ számot a 150 és a 216 mértani (geometriai) közepének nevezzük. A 150 és a 216 számtani közep nagyobb, mint a mértani közép.

2. Egy téglalap két szemköztes oldalának hossza a cm és b cm. Hány cm az az a és b származék az oldal, amelynek a téglalapével megegyező
a) a kerülete, b) a területe?

Megoldás

a) Legyen a keresztirányú oldal x cm. A téglalap kerülete $(2a + 2b)$ cm, a négyzet kerülete $4x$ cm. A kerületek egyenlősé: $4x = 2a + 2b$. A négyzet oldala tehát $x = \frac{a+b}{2}$ cm hosszú, azaz a téglalap két szemköztes oldala hosszának számtani közep.

b) Legyen a keresztirányú oldal y cm. A téglalap terület $(a \cdot b)$ cm², a négyzet területe y^2 cm². A területek egyenlősé: $y^2 = a \cdot b$. Melyik pozitív számunk a négyzet $y = \sqrt{a \cdot b}$ cm hosszú. Ez az a és b származék $\sqrt{a \cdot b}$ cm hosszúságú.

110 SZÁMTANI KÖZÉP, MÉRTANI KÖZÉP

PELDÁ

1. Janoskának Antal és Bend is 150 ezer forint nettó bért kapott, márciusban pedig mindketten 216 ezer forintot. Antal bért februárban és márciusban is ugyanannyal töltötte, Bend bért pedig mindkét hónapban ugyanannyiszorosára nőtt. Melyikük kapott több bért februárban?

Megoldás
Antal februári bért annyival több a januári bérével, amennyivel a márciusi bért több a februárián. Ha tehát a februári bért x Ft, akkor $x = 150 + 216 - x$. Ebből $2x = 150 + 216$, vagyis Antal februári bért $x = \frac{150+216}{2} = 183$ ezer forint.

Ez éppen a januári és a márciusi bértök a számtani közep.

Bend februári bért annyiszorosa a januári bértnök, akárcsosa a márciusi bért a februárián. Ha a februári bért y Ft, akkor $\frac{y}{150} = \frac{216}{y}$, ebből $y^2 = 150 \cdot 216$, vagyis $y = \sqrt{150 \cdot 216} = 180$ ezer forint.
Antal több bért kapott februárban, mint Bend.

Megjegyzés

A $\sqrt{150 \cdot 216} = 180$ számot a 150 és a 216 mértani (geometriai) közepének nevezzük. A 150 és a 216 számtani közep nagyobb, mint a mértani közép.

2. Egy téglalap két szemköztes oldalának hossza a cm és b cm. Hány cm az az a és b származék az oldal, amelynek a téglalapével megegyező
a) a kerülete, b) a területe?

Megoldás

a) Legyen a keresztirányú oldal x cm. A téglalap kerülete $(2a + 2b)$ cm, a négyzet kerülete $4x$ cm. A kerületek egyenlősé: $4x = 2a + 2b$. A négyzet oldala tehát $x = \frac{a+b}{2}$ cm hosszú, azaz a téglalap két szemköztes oldala hosszának számtani közep.

b) Legyen a keresztirányú oldal y cm. A téglalap terület $(a \cdot b)$ cm², a négyzet területe y^2 cm². A területek egyenlősé: $y^2 = a \cdot b$. Melyik pozitív számunk a négyzet $y = \sqrt{a \cdot b}$ cm hosszú. Ez az a és b származék $\sqrt{a \cdot b}$ cm hosszúságú.

ELMÉLET

A bevezető példában három matematikai fogalommal találkozunk: négyzetgyök, számtani közép, mértani közép.

1. A négyzetgyökök már tanultak:

Egy a pozitív szám négyzetgyökének azt a pozitív számot nevezzük, amelynek a négyzete a .

A 0 négyzetgyöke 0, negatív szám négyzetgyöke nem értelmezett.

Jelölés: ha $a \geq 0$, akkor $\sqrt{a}$ létezik, nemnegatív, és $(\sqrt{a})^2 = a$. Ha $a < 0$, akkor $\sqrt{a}$ jel számunkra értelmezetlen.

Példák

$\sqrt{225} = 15$, mert 225 pozitív szám, 15 a pozitív szám és $15^2 = 225$.

$\sqrt{-225}$ jel értelmezetlen, mert a négyzetgyökök alatt negatív szám áll.

67. ábra - Matematika 10 tankönyv 34. oldal hasáb és bekezdés azonosításának eredménye

110 SZÁMTANI KÖZÉP, MÉRTANI KÖZÉP

PELDÁ

1. Janoskának Antal és Bend is 150 ezer forint nettó bért kapott, márciusban pedig mindketten 216 ezer forintot. Antal bért februárban és márciusban is ugyanannyal töltötte, Bend bért pedig mindkét hónapban ugyanannyiszorosára nőtt. Melyikük kapott több bért februárban?

Megoldás
Antal februári bért annyival több a januári bérével, amennyivel a márciusi bért több a februárián. Ha tehát a februári bért x Ft, akkor $x = 150 + 216 - x$. Ebből $2x = 150 + 216$, vagyis Antal februári bért $x = \frac{150+216}{2} = 183$ ezer forint.

Ez éppen a januári és a márciusi bértök a számtani közep.

Bend februári bért annyiszorosa a januári bértnök, akárcsosa a márciusi bért a februárián. Ha a februári bért y Ft, akkor $\frac{y}{150} = \frac{216}{y}$, ebből $y^2 = 150 \cdot 216$, vagyis $y = \sqrt{150 \cdot 216} = 180$ ezer forint.
Antal több bért kapott februárban, mint Bend.

ELMÉLET

A bevezető példában három matematikai fogalommal találkozunk: négyzetgyök, számtani közép, mértani közép.

1. A négyzetgyökök már tanultak:

Egy a pozitív szám négyzetgyökének azt a pozitív számot nevezzük, amelynek a négyzete a .

A 0 négyzetgyöke 0, negatív szám négyzetgyöke nem értelmezett.

Jelölés: ha $a \geq 0$, akkor $\sqrt{a}$ létezik, nemnegatív, és $(\sqrt{a})^2 = a$. Ha $a < 0$, akkor $\sqrt{a}$ jel számunkra értelmezetlen.

Példák

$\sqrt{225} = 15$, mert 225 pozitív szám, 15 a pozitív szám és $15^2 = 225$.

$\sqrt{-225}$ jel értelmezetlen, mert a négyzetgyökök alatt negatív szám áll.

Megjegyzés

A $\sqrt{150 \cdot 216} = 180$ számot a 150 és a 216 mértani (geometriai) közepének nevezzük. A 150 és a 216 számtani közep nagyobb, mint a mértani közép.

2. Egy téglalap két szemköztes oldalának hossza a cm és b cm. Hány cm az az a és b származék az oldal, amelynek a téglalapával megegyező
a) a kerülete, b) a területe?

Megoldás

a) Legyen a keresztirányú oldal x cm. A téglalap kerülete $(2a + 2b)$ cm, a négyzet kerülete $4x$ cm. A kerületek egyenlősé: $4x = 2a + 2b$. A négyzet oldala tehát $x = \frac{a+b}{2}$ cm hosszú, azaz a téglalap két szemköztes oldala hosszának számtani közep.

b) Legyen a keresztirányú oldal y cm. A téglalap terület $(a \cdot b)$ cm², a négyzet területe y^2 cm². A területek egyenlősé: $y^2 = a \cdot b$. Melyik pozitív számunk a négyzet $y = \sqrt{a \cdot b}$ cm hosszú. Ez az a és b származék $\sqrt{a \cdot b}$ cm hosszúságú.

110 SZÁMTANI KÖZÉP, MÉRTANI KÖZÉP

PELDÁ

1. Janoskának Antal és Bend is 150 ezer forint nettó bért kapott, márciusban pedig mindketten 216 ezer forintot. Antal bért februárban és márciusban is ugyanannyal töltötte, Bend bért pedig mindkét hónapban ugyanannyiszorosára nőtt. Melyikük kapott több bért februárban?

Megoldás
Antal februári bért annyival több a januári bérével, amennyivel a márciusi bért több a februárián. Ha tehát a februári bért x Ft, akkor $x = 150 + 216 - x$. Ebből $2x = 150 + 216$, vagyis Antal februári bért $x = \frac{150+216}{2} = 183$ ezer forint.

Ez éppen a januári és a márciusi bértök a számtani közep.

Bend februári bért annyiszorosa a januári bértnök, akárcsosa a márciusi bért a februárián. Ha a februári bért y Ft, akkor $\frac{y}{150} = \frac{216}{y}$, ebből $y^2 = 150 \cdot 216$, vagyis $y = \sqrt{150 \cdot 216} = 180$ ezer forint.
Antal több bért kapott februárban, mint Bend.

Megjegyzés

A $\sqrt{150 \cdot 216} = 180$ számot a 150 és a 216 mértani (geometriai) közepének nevezzük. A 150 és a 216 számtani közep nagyobb, mint a mértani közép.

2. Egy téglalap két szemköztes oldalának hossza a cm és b cm. Hány cm az az a és b származék az oldal, amelynek a téglalapával megegyező
a) a kerülete, b) a területe?

Megoldás

a) Legyen a keresztirányú oldal x cm. A téglalap kerülete $(2a + 2b)$ cm, a négyzet kerülete $4x$ cm. A kerületek egyenlősé: $4x = 2a + 2b$. A négyzet oldala tehát $x = \frac{a+b}{2}$ cm hosszú, azaz a téglalap két szemköztes oldala hosszának számtani közep.

b) Legyen a keresztirányú oldal y cm. A téglalap terület $(a \cdot b)$ cm², a négyzet területe y^2 cm². A területek egyenlősé: $y^2 = a \cdot b$. Melyik pozitív számunk a négyzet $y = \sqrt{a \cdot b}$ cm hosszú. Ez az a és b származék $\sqrt{a \cdot b}$ cm hosszúságú.

ELMÉLET

A bevezető példában három matematikai fogalommal találkozunk: négyzetgyök, számtani közép, mértani közép.

1. A négyzetgyökök már tanultak:

Egy a pozitív szám négyzetgyökének azt a pozitív számot nevezzük, amelynek a négyzete a .

A 0 négyzetgyöke 0, negatív szám négyzetgyöke nem értelmezett.

Jelölés: ha $a \geq 0$, akkor $\sqrt{a}$ létezik, nemnegatív, és $(\sqrt{a})^2 = a$. Ha $a < 0$, akkor $\sqrt{a}$ jel számunkra értelmezetlen.

Példák

$\sqrt{225} = 15$, mert 225 pozitív szám, 15 a pozitív szám és $15^2 = 225$.

$\sqrt{-225}$ jel értelmezetlen, mert a négyzetgyökök alatt negatív szám áll.

ALGEBRA ÉS SZÁMELMÉLET – ÖSSZEFOGLALÁS 4. Műveletek racionális kifejezésekkel Egész kifejezések (polinomok) NEVEZETES SZORZATOK: $(a + b)^2 = a^2 + 2ab + b^2,$ $(a - b)^2 = a^2 - 2ab + b^2,$ $(a + b + c)^2 = a^2 + b^2 + c^2 + 2ab + 2ac + 2bc,$ $(a + b)^3 = a^3 + 3a^2b + 3ab^2 + b^3,$ $(a - b)^3 = a^3 - 3a^2b + 3ab^2 - b^3,$ $(a + b) \cdot (a - b) = a^2 - b^2.$ A SZORZATTÁ ALAKÍTÁS MÓDSZEREI: • kiemelés; • kiemelés csoportosítással; • nevezetes azonosságok alkalmazása; • másodfokú egyenlet gyökényezés alakjának felhasználása. 1. példa Bontsuk fel a zárójeleket, és végezzük el a lehetséges összevonásokat: a) $(2a - 3) \cdot (4a + 1) - (3a - 5);$ b) $(4b + 3) \cdot (4b - 3) - (3b - 2) \cdot (3b + 2).$ Megoldás (a) A zárójel felbontása és az összevonás után: $(2a - 3) \cdot (4a + 1) - (3a - 5) = 8a^2 + 2a - 12a - 3 - 3a + 5 = 8a^2 - 13a + 2.$ Megoldás (b) Alkalmazzuk az összeg és különbség szorzására vonatkozó azonosságot: $(4b + 3) \cdot (4b - 3) - (3b - 2) \cdot (3b + 2) = 16b^2 - 9 - 9b^2 + 4 = 7b^2 - 5.$ 2. példa Végezzük el a műveleteket, és a kapott polinomot rendezzük x csökkenő hatványai szerint: a) $(3x + 2)^2 - (2x - 3)^2;$ b) $(2x - 1)^3 - (x + 4)^3;$ c) $(2x + y - 3)^2.$ Megoldás a) $(3x + 2)^2 - (2x - 3)^2 = 9x^2 + 12x + 4 - (4x^2 - 12x + 9) = 9x^2 + 12x + 4 - 4x^2 + 12x - 9 = 5x^2 + 24x - 5;$	ALGEBRA ÉS SZÁMELMÉLET – ÖSSZEFOGLALÁS 4. Műveletek racionális kifejezésekkel Egész kifejezések (polinomok) NEVEZETES SZORZATOK: $(a + b)^2 = a^2 + 2ab + b^2,$ $(a - b)^2 = a^2 - 2ab + b^2,$ $(a + b + c)^2 = a^2 + b^2 + c^2 + 2ab + 2ac + 2bc,$ $(a + b)^3 = a^3 + 3a^2b + 3ab^2 + b^3,$ $(a - b)^3 = a^3 - 3a^2b + 3ab^2 - b^3,$ $(a + b) \cdot (a - b) = a^2 - b^2.$ A SZORZATTÁ ALAKÍTÁS MÓDSZEREI: • kiemelés; • kiemelés csoportosítással; • nevezetes azonosságok alkalmazása; • másodfokú egyenlet gyökényezés alakjának felhasználása. 1. példa Bontsuk fel a zárójeleket, és végezzük el a lehetséges összevonásokat: a) $(2a - 3) \cdot (4a + 1) - (3a - 5);$ b) $(4b + 3) \cdot (4b - 3) - (3b - 2) \cdot (3b + 2).$ Megoldás (a) A zárójel felbontása és az összevonás után: $(2a - 3) \cdot (4a + 1) - (3a - 5) = 8a^2 + 2a - 12a - 3 - 3a + 5 = 8a^2 - 13a + 2.$ Megoldás (b) Alkalmazzuk az összeg és különbség szorzására vonatkozó azonosságot: $(4b + 3) \cdot (4b - 3) - (3b - 2) \cdot (3b + 2) = 16b^2 - 9 - 9b^2 + 4 = 7b^2 - 5.$ 2. példa Végezzük el a műveleteket, és a kapott polinomot rendezzük x csökkenő hatványai szerint: a) $(3x + 2)^2 - (2x - 3)^2;$ b) $(2x - 1)^3 - (x + 4)^3;$ c) $(2x + y - 3)^2.$ Megoldás a) $(3x + 2)^2 - (2x - 3)^2 = 9x^2 + 12x + 4 - (4x^2 - 12x + 9) = 9x^2 + 12x + 4 - 4x^2 + 12x - 9 = 5x^2 + 24x - 5;$	ALGEBRA ÉS SZÁMELMÉLET – ÖSSZEFOGLALÁS 4. Műveletek racionális kifejezésekkel Egész kifejezések (polinomok) NEVEZETES SZORZATOK: $(a + b)^2 = a^2 + 2ab + b^2,$ $(a - b)^2 = a^2 - 2ab + b^2,$ $(a + b + c)^2 = a^2 + b^2 + c^2 + 2ab + 2ac + 2bc,$ $(a + b)^3 = a^3 + 3a^2b + 3ab^2 + b^3,$ $(a - b)^3 = a^3 - 3a^2b + 3ab^2 - b^3,$ $(a + b) \cdot (a - b) = a^2 - b^2.$ A SZORZATTÁ ALAKÍTÁS MÓDSZEREI: • kiemelés; • kiemelés csoportosítással; • nevezetes azonosságok alkalmazása; • másodfokú egyenlet gyökényezés alakjának felhasználása. 1. példa Bontsuk fel a zárójeleket, és végezzük el a lehetséges összevonásokat: a) $(2a - 3) \cdot (4a + 1) - (3a - 5);$ b) $(4b + 3) \cdot (4b - 3) - (3b - 2) \cdot (3b + 2).$ Megoldás (a) A zárójel felbontása és az összevonás után: $(2a - 3) \cdot (4a + 1) - (3a - 5) = 8a^2 + 2a - 12a - 3 - 3a + 5 = 8a^2 - 13a + 2.$ Megoldás (b) Alkalmazzuk az összeg és különbség szorzására vonatkozó azonosságot: $(4b + 3) \cdot (4b - 3) - (3b - 2) \cdot (3b + 2) = 16b^2 - 9 - 9b^2 + 4 = 7b^2 - 5.$ 2. példa Végezzük el a műveleteket, és a kapott polinomot rendezzük x csökkenő hatványai szerint: a) $(3x + 2)^2 - (2x - 3)^2;$ b) $(2x - 1)^3 - (x + 4)^3;$ c) $(2x + y - 3)^2.$ Megoldás a) $(3x + 2)^2 - (2x - 3)^2 = 9x^2 + 12x + 4 - (4x^2 - 12x + 9) = 9x^2 + 12x + 4 - 4x^2 + 12x - 9 = 5x^2 + 24x - 5;$
--	--	--

69. ábra - Sokszínű matematika 12 tankönyv 178. oldal bekezdés, sor és szóazonosításának eredménye

b) $(2x-1)^3 - (x+4)^3 = 8x^3 - 12x^2 + 6x - 1 - (x^3 + 12x^2 + 48x + 64) = 8x^3 - 12x^2 + 6x - 1 - x^3 - 12x^2 - 48x - 64 = 7x^3 - 24x^2 - 42x - 65;$ c) $(2x+y-3)^2 = 4x^2 + y^2 + 9 + 4xy - 12x - 6y = 4x^2 + 4xy - 12x + y^2 - 6y + 9.$ 3. példa Melyik kéttagú kifejezés négyzete lehet: a) $25x^2 + 10x + 1;$ b) $x^6 - 14x^3 + 49;$ c) $x^4 - 8x^2y + 16y^2?$ Megoldás a) $25x^2 + 10x + 1 = (5x + 1)^2 = (-5x - 1)^2;$ b) $x^6 - 14x^3 + 49 = (x^3 - 7)^2 = (7 - x^3)^2;$ c) $x^4 - 8x^2y + 16y^2 = (x^2 - 4y)^2 = (4y - x^2)^2.$ 4. példa Alakítsuk szorzattá az alábbi kifejezéseket: a) $6x^2 - 4x^3 + 8x^5;$ b) $2x^2 + 4x + xy + 2y;$ c) $9x^2 - 25y^2;$ d) $3x^2 - 12x + 12;$ e) $x^2 + 2x - 15.$ Megoldás (a) Emeljük ki a $2x^2$-et mindegyik tagból: $6x^4 - 4x^3 + 8x^2 = 2x^2 \cdot (3x^2 - 2x + 4).$ A $3x^2 - 2x + 4 = 0$ másodfokú egyenlet diszkriminánsa $D = 4 - 48 = -44$, ezért az egyenletnek nincs megoldása, tehát a $3x^2 - 2x + 4$ másodfokú polinom nem alakítható szorzattá. Megoldás (b) Emeljük ki az első két tagból $2x$-et, az utolsó kettőből pedig y-t, majd az újabb kiemelés után: $2x^2 + 4x + xy + 2y = 2x \cdot (x + 2) + y \cdot (x + 2) = (x + 2) \cdot (2x + y).$ Megoldás (c) Vegyük észre a nevezetes szorzatot: $9x^2 - 25y^2 = (3x + 5y) \cdot (3x - 5y).$ Megoldás (d) A 3 kiemelése után egy kéttagú kifejezés négyzete marad a zárójelben: $3x^2 - 12x + 12x + 12 = 3 \cdot (x^2 - 4x + 4) = 3 \cdot (x - 2)^2.$ 179	b) $(2x-1)^3 - (x+4)^3 = 8x^3 - 12x^2 + 6x - 1 - (x^3 + 12x^2 + 48x + 64) = 8x^3 - 12x^2 + 6x - 1 - x^3 - 12x^2 - 48x - 64 = 7x^3 - 24x^2 - 42x - 65;$ c) $(2x+y-3)^2 = 4x^2 + y^2 + 9 + 4xy - 12x - 6y = 4x^2 + 4xy - 12x + y^2 - 6y + 9.$ 3. példa Melyik kéttagú kifejezés négyzete lehet: a) $25x^2 + 10x + 1;$ b) $x^6 - 14x^3 + 49;$ c) $x^4 - 8x^2y + 16y^2?$ Megoldás a) $25x^2 + 10x + 1 = (5x + 1)^2 = (-5x - 1)^2;$ b) $x^6 - 14x^3 + 49 = (x^3 - 7)^2 = (7 - x^3)^2;$ c) $x^4 - 8x^2y + 16y^2 = (x^2 - 4y)^2 = (4y - x^2)^2.$ 4. példa Alakítsuk szorzattá az alábbi kifejezéseket: a) $6x^2 - 4x^3 + 8x^5;$ b) $2x^2 + 4x + xy + 2y;$ c) $9x^2 - 25y^2;$ d) $3x^2 - 12x + 12;$ e) $x^2 + 2x - 15.$ Megoldás (a) Emeljük ki a $2x^2$-et mindegyik tagból: $6x^4 - 4x^3 + 8x^2 = 2x^2 \cdot (3x^2 - 2x + 4).$ A $3x^2 - 2x + 4 = 0$ másodfokú egyenlet diszkriminánsa $D = 4 - 48 = -44$, ezért az egyenletnek nincs megoldása, tehát a $3x^2 - 2x + 4$ másodfokú polinom nem alakítható szorzattá. Megoldás (b) Emeljük ki az első két tagból $2x$-et, az utolsó kettőből pedig y-t, majd az újabb kiemelés után: $2x^2 + 4x + xy + 2y = 2x \cdot (x + 2) + y \cdot (x + 2) = (x + 2) \cdot (2x + y).$ Megoldás (c) Vegyük észre a nevezetes szorzatot: $9x^2 - 25y^2 = (3x + 5y) \cdot (3x - 5y).$ Megoldás (d) A 3 kiemelése után egy kéttagú kifejezés négyzete marad a zárójelben: $3x^2 - 12x + 12x + 12 = 3 \cdot (x^2 - 4x + 4) = 3 \cdot (x - 2)^2.$ 179	b) $(2x-1)^3 - (x+4)^3 = 8x^3 - 12x^2 + 6x - 1 - (x^3 + 12x^2 + 48x + 64) = 8x^3 - 12x^2 + 6x - 1 - x^3 - 12x^2 - 48x - 64 = 7x^3 - 24x^2 - 42x - 65;$ c) $(2x+y-3)^2 = 4x^2 + y^2 + 9 + 4xy - 12x - 6y = 4x^2 + 4xy - 12x + y^2 - 6y + 9.$ 3. példa Melyik kéttagú kifejezés négyzete lehet: a) $25x^2 + 10x + 1;$ b) $x^6 - 14x^3 + 49;$ c) $x^4 - 8x^2y + 16y^2?$ Megoldás a) $25x^2 + 10x + 1 = (5x + 1)^2 = (-5x - 1)^2;$ b) $x^6 - 14x^3 + 49 = (x^3 - 7)^2 = (7 - x^3)^2;$ c) $x^4 - 8x^2y + 16y^2 = (x^2 - 4y)^2 = (4y - x^2)^2.$ 4. példa Alakítsuk szorzattá az alábbi kifejezéseket: a) $6x^2 - 4x^3 + 8x^5;$ b) $2x^2 + 4x + xy + 2y;$ c) $9x^2 - 25y^2;$ d) $3x^2 - 12x + 12;$ e) $x^2 + 2x - 15.$ Megoldás (a) Emeljük ki a $2x^2$-et mindegyik tagból: $6x^4 - 4x^3 + 8x^2 = 2x^2 \cdot (3x^2 - 2x + 4).$ A $3x^2 - 2x + 4 = 0$ másodfokú egyenlet diszkriminánsa $D = 4 - 48 = -44$, ezért az egyenletnek nincs megoldása, tehát a $3x^2 - 2x + 4$ másodfokú polinom nem alakítható szorzattá. Megoldás (b) Emeljük ki az első két tagból $2x$-et, az utolsó kettőből pedig y-t, majd az újabb kiemelés után: $2x^2 + 4x + xy + 2y = 2x \cdot (x + 2) + y \cdot (x + 2) = (x + 2) \cdot (2x + y).$ Megoldás (c) Vegyük észre a nevezetes szorzatot: $9x^2 - 25y^2 = (3x + 5y) \cdot (3x - 5y).$ Megoldás (d) A 3 kiemelése után egy kéttagú kifejezés négyzete marad a zárójelben: $3x^2 - 12x + 12x + 12 = 3 \cdot (x^2 - 4x + 4) = 3 \cdot (x - 2)^2.$ 179
---	---	---

70. ábra - Sokszínű matematika 12 tankönyv 179. oldal bekezdés, sor és szóazonosításának eredménye

ALGEBRA ÉS SZÁMELMÉLET - ÖSSZEFOGLALÁS Megoldás (e) I. megoldás Alakítsunk teljes négyzetté, majd alkalmazzuk a nevezetes szorzatokat: $x^2 + 2x - 15 = x^2 + 2x + 1 - 1 - 15 = (x + 1)^2 - 16 =$ $= (x + 1)^2 - 4^2 = (x + 1 + 4) \cdot (x + 1 - 4) = (x + 5) \cdot (x - 3).$ II. megoldás A másodfokú egyenlet gyöktényezős alakját használjuk fel. Oldjuk meg az $x^2 + 2x - 15 = 0$ egyenletet: $x_{1,2} = \frac{-2 \pm \sqrt{4 + 60}}{2} = \begin{cases} x_1 = 3, \\ x_2 = -5, \end{cases}$ Az $ax^2 + bx + c = 0$ másodfokú egyenlet gyöktényezős alakja, ha $D \geq 0$: $a \cdot (x - x_1) \cdot (x - x_2) = 0$, ahol x_1 és x_2 a másodfokú egyenlet gyökei. A gyöktényezős alaknak megfelelően: $x^2 + 2x - 15 = (x - 3) \cdot (x + 5).$ Az algebrai törtek 5. példa Egyszerűsítsük az alábbi törteteket: a) $\frac{a^2 + 4a + 4}{3a + 6};$ b) $\frac{2b^3 - 12b^2 + 18b}{b^3 - 9b}.$ Megoldás (a) Az első lépés a tört értelmezése: $3a + 6 \neq 0$, tehát $a \neq -2$. A számlálót és a nevezőt is alakítsuk szorzattá, majd az azonos tényezővel egyszerűsítsünk: $\frac{a^2 + 4a + 4}{3a + 6} = \frac{(a + 2)^2}{3 \cdot (a + 2)} = \frac{a + 2}{3}.$ Megoldás (b) A nevező nem lehet nulla: $b^3 - 9b \neq 0$. Szorzattá alakítással: $b^3 - 9b = b \cdot (b^2 - 9) = b \cdot (b + 3) \cdot (b - 3).$ Tehát $b \neq 0$, $b \neq -3$, $b \neq 3$. Kiemeléssel a számláló is szorzattá alakítható: $\frac{2b^3 - 12b^2 + 18b}{b^3 - 9b} = \frac{2b \cdot (b^2 - 6b + 9)}{b \cdot (b - 3) \cdot (b + 3)} =$ $= \frac{2b \cdot (b - 3)^2}{b \cdot (b - 3) \cdot (b + 3)} = \frac{2 \cdot (b - 3)}{b + 3}.$ 180	ALGEBRA ÉS SZÁMELMÉLET - ÖSSZEFOGLALÁS Megoldás (e) I. megoldás Alakítsunk teljes négyzetté, majd alkalmazzuk a nevezetes szorzatokat: $x^2 + 2x - 15 = x^2 + 2x + 1 - 1 - 15 = (x + 1)^2 - 16 =$ $= (x + 1)^2 - 4^2 = (x + 1 + 4) \cdot (x + 1 - 4) = (x + 5) \cdot (x - 3).$ II. megoldás A másodfokú egyenlet gyöktényezős alakját használjuk fel. Oldjuk meg az $x^2 + 2x - 15 = 0$ egyenletet: $x_{1,2} = \frac{-2 \pm \sqrt{4 + 60}}{2} = \begin{cases} x_1 = 3, \\ x_2 = -5, \end{cases}$ Az $ax^2 + bx + c = 0$ másodfokú egyenlet gyöktényezős alakja, ha $D \geq 0$: $a \cdot (x - x_1) \cdot (x - x_2) = 0$, ahol x_1 és x_2 a másodfokú egyenlet gyökei. A gyöktényezős alaknak megfelelően: $x^2 + 2x - 15 = (x - 3) \cdot (x + 5).$ Az algebrai törtek 5. példa Egyszerűsítsük az alábbi törteteket: a) $\frac{a^2 + 4a + 4}{3a + 6};$ b) $\frac{2b^3 - 12b^2 + 18b}{b^3 - 9b}.$ Megoldás (a) Az első lépés a tört értelmezése: $3a + 6 \neq 0$, tehát $a \neq -2$. A számlálót és a nevezőt is alakítsuk szorzattá, majd az azonos tényezővel egyszerűsítsünk: $\frac{a^2 + 4a + 4}{3a + 6} = \frac{(a + 2)^2}{3 \cdot (a + 2)} = \frac{a + 2}{3}.$ Megoldás (b) A nevező nem lehet nulla: $b^3 - 9b \neq 0$. Szorzattá alakítással: $b^3 - 9b = b \cdot (b^2 - 9) = b \cdot (b + 3) \cdot (b - 3).$ Tehát $b \neq 0$, $b \neq -3$, $b \neq 3$. Kiemeléssel a számláló is szorzattá alakítható: $\frac{2b^3 - 12b^2 + 18b}{b^3 - 9b} = \frac{2b \cdot (b^2 - 6b + 9)}{b \cdot (b - 3) \cdot (b + 3)} =$ $= \frac{2b \cdot (b - 3)^2}{b \cdot (b - 3) \cdot (b + 3)} = \frac{2 \cdot (b - 3)}{b + 3}.$ 180	ALGEBRA ÉS SZÁMELMÉLET - ÖSSZEFOGLALÁS Megoldás (e) I. megoldás Alakítsunk teljes négyzetté, majd alkalmazzuk a nevezetes szorzatokat: $x^2 + 2x - 15 = x^2 + 2x + 1 - 1 - 15 = (x + 1)^2 - 16 =$ $= (x + 1)^2 - 4^2 = (x + 1 + 4) \cdot (x + 1 - 4) = (x + 5) \cdot (x - 3).$ II. megoldás A másodfokú egyenlet gyöktényezős alakját használjuk fel. Oldjuk meg az $x^2 + 2x - 15 = 0$ egyenletet: $x_{1,2} = \frac{-2 \pm \sqrt{4 + 60}}{2} = \begin{cases} x_1 = 3, \\ x_2 = -5, \end{cases}$ Az $ax^2 + bx + c = 0$ másodfokú egyenlet gyöktényezős alakja, ha $D \geq 0$: $a \cdot (x - x_1) \cdot (x - x_2) = 0$, ahol x_1 és x_2 a másodfokú egyenlet gyökei. A gyöktényezős alaknak megfelelően: $x^2 + 2x - 15 = (x - 3) \cdot (x + 5).$ Az algebrai törtek 5. példa Egyszerűsítsük az alábbi törteteket: a) $\frac{a^2 + 4a + 4}{3a + 6};$ b) $\frac{2b^3 - 12b^2 + 18b}{b^3 - 9b}.$ Megoldás (a) Az első lépés a tört értelmezése: $3a + 6 \neq 0$, tehát $a \neq -2$. A számlálót és a nevezőt is alakítsuk szorzattá, majd az azonos tényezővel egyszerűsítsünk: $\frac{a^2 + 4a + 4}{3a + 6} = \frac{(a + 2)^2}{3 \cdot (a + 2)} = \frac{a + 2}{3}.$ Megoldás (b) A nevező nem lehet nulla: $b^3 - 9b \neq 0$. Szorzattá alakítással: $b^3 - 9b = b \cdot (b^2 - 9) = b \cdot (b + 3) \cdot (b - 3).$ Tehát $b \neq 0$, $b \neq -3$, $b \neq 3$. Kiemeléssel a számláló is szorzattá alakítható: $\frac{2b^3 - 12b^2 + 18b}{b^3 - 9b} = \frac{2b \cdot (b^2 - 6b + 9)}{b \cdot (b - 3) \cdot (b + 3)} =$ $= \frac{2b \cdot (b - 3)^2}{b \cdot (b - 3) \cdot (b + 3)} = \frac{2 \cdot (b - 3)}{b + 3}.$ 180
---	---	---

71. ábra - Sokszínű matematika 12 tankönyv 180. oldal bekezdés, sor és szóazonosításának eredménye