A táblázatkezelés és a programozás didaktikai kapcsolata

Törley Gábor[1], Zsakó László[2]

[1]gabor.torley@inf.elte.hu; ORCID: 0000-0002-0496-936,
[2]zsako@caesar.elte.hu; ORCID: 0000-0002-4614-1509
ELTE Eötvös Loránd Tudományegyetem Informatikai Kar

Absztrakt. Amikor a problémamegoldás készségéről beszélünk, akkor általában a programozás jut eszünkbe, mint tevékenység, ami fejleszti az algoritmikus gondolkodást, illetve az absztrakciós készséget. Ugyanakkor a táblázatkezelésről első gondolatként a szoftveralkalmazás területe juthat eszünkbe, illetve második gondolatként eszünkbe juthat a matematika. Amikor a táblázatkezelés és a programozás egy lapon szerepel, akkor a makrók programozása kerül előtérbe, ami inkább programozás, mint táblázatkezelés. Cikkünkben azt vesszük górcső alá, hogy miként hat egymásra ez a két terület, illetve fel kívánjuk hívni a figyelmet arra, hogy a táblázatkezelés is hatékony eszköz az algoritmikus gondolkodás fejlesztéséhez, sőt, több „áthallás" van a két eszköz között. Példákon keresztül mutatjuk meg továbbá, hogy a programozás és a táblázatkezelés között két irányú kapcsolat létezik, azaz a kölcsönös egymásra építés mindkét témakör számára hasznos lehet.

Kulcsszavak: táblázatkezelés, programozás, problémamegoldás, algoritmikus gondolkodás, tanítási módszerek

1. Bevezetés

A táblázatkezelést, mint tanulási eszközt, nem szokás a problémamegoldás eszközei közé sorolni. Ennek ékes példája a NAT 2012-es kerettanterv [1], amelyben a problémamegoldás informatikai eszközökkel és módszerekkel témakör kizárólag a programozás, algoritmizálás területével foglalkozik, és a táblázatkezelés az alkalmazói ismeretek témakörben kapott helyet. E szerint a táblázatkezelés célja az információk kinyerése.

A 2020-as kerettanterv [2] ebből a szempontból árnyaltabb képet fest, ugyanis itt – helyesen –, a táblázatkezelés már a problémamegoldás fejlesztésének új témaköreként jelenik meg. Ezzel követi az ELTE Informatikai Karán sokkal korábban kidolgozott módszertant. [10, 11]

2. Irodalmi áttekintés

Szalayné szerint [9] a táblázat egy programnak tekinthető adatokkal és előre meghatározott algoritmusokkal. Bár a tanulók egy táblázatot/táblázatkezelőt látnak, mégis meg kell érteniük az általuk írt „programot", azaz a függvényekkel megvalósított megoldásukat. Ilyen módon táblázatkezelő alkalmazásával lehet indirekt módon programozást tanítani.

Bíró és Csernoch szerint a táblázatkezelő szoftver használható a problémamegoldás ezközeként [3], és az általuk leírt Sprego módszer alkalmas a tanulók számítógépes gondolkodásának és algoritmikus készségének fejlesztésére.

Warren szerint, ha először táblázatkezelőt használunk, és utána a programozási nyelvet, akkor gyorsabban el lehet jutni a tananyagban a bonyolultabb algoritmusokig. [7]

3. Változó- és típusfogalom

A táblázatkezeléssel tanítani lehet a skalár, a tömb, a mátrix, valamint az indexelés fogalmát, amelyek alapvető adatszerkezetei a programozásnak, hasonló módon az elemi típusok bevezetését is támogatja a táblázatkezelés témaköre. Haladó lehetőségeivel akár nevet is adhatunk egyes tartományainak, amire névvel hivatkozhatunk (mint egy változóra a programozási nyelvekben).

A típusfogalom kialakításában fontos szerepe van a táblázatkezelésnek, mivel bizonyos műveletek (függvények) csak bizonyos típusú adatokon értelmezhetőek, pl. SZUM, ÁTLAG függvény csak számokon, míg az ÖSSZEFŰZ, FŰZ függvények csak szövegeken értelmezhetők. Ebből a tanuló megértheti, hogy a típus nem csak egy halmazt jelöl, hanem a rajta értelmezett műveleteket is magába foglalja. Ugyanúgy éles különbség van a táblázatkezelésnél a számjegyek (mint szöveg) és a számok között (vö. ”23” és 23 közötti különbség). Az éles különbség a szöveg- és számkonstansok esetében is látszik. (Kitérő: nagy hasonlóságot lehet felfedezni a táblázatkezelők cellái formátumának megadási lehetőségei és a programozási nyelvek formázott kiírási lehetőségei között, pl. hány számjegyre jelenjen meg a szám, hány tizedesjegyre, ...)

Bár a táblázatkezelés változófogalma sajátos [6], mégis, a függvények működésének mélyebb megértése hozzájárul ahhoz, hogy a tanuló megértse, mi a különbség a skalár és a sokaság között, valamint mit jelent bejárni egy sokaságot (jelen esetben tömböt vagy mátrixot). Az indexelés megértésének egyik legjobb eszköze az INDEX fv. lehet, amely lényegében egy általunk kijelölt tartományon hajtja végre az indexelés műveletét.

4. Vezérlési szerkezetek, sémaalgoritmusok

A táblázatkezelés ugyan függvényszerű gondolkodást igényel (bevezetés a funkcionális programozásba – ennek kifejtése egy másik cikk témája lehet), de a klasszikus függvényei (SZUM, MAX, ...) egyben alapvető programozási algoritmus sémák (programozási tételek) is. Egyes függvények ráadásul úgy érthetők meg, ha elképzeljük a végrehajtási algoritmusukat (pl. FKERES). A függvények paraméterezése, és egymásba ágyazása segítheti a hagyományos nyelvekbeli paraméterezés, paraméterátadás megértését.

A HA függvénnyel, lényegében, az elágazás, mint vezérlési szerkezet működését lehet megérteni, egyben kiváló eszköz a logikai típus, illetve a logikai műveletek (ÉS, VAGY fv.) megértéséhez. A ciklusok fogalma „nyelvi” szinten nem kerül elő a táblázatkezelés során, viszont a tömbképletek használatával és megértésével bevezethető a számlálós ciklus fogalma. A keresési függvények mélyebb vizsgálata juttathat el a feltételes ciklusokig, mivel ezek megértésénél fel lehet tenni a kérdést, hogy végig kell-e futnunk a sokaságon ahhoz, hogy egyértelműen választ adjunk a keresési feladatra.

Haladó táblázatkezelésnél a tömbfüggvényekkel le lehet képezni az összes sémaalgoritmust (programozási tételt), illetve kapcsolatot lehet találni a tömbképletek és a programozási tételek utófeltétele között.

Az alábbi táblázatban foglaljuk össze a táblázatkezelés és a sémaalgoritmusok kapcsolatát:

Sémaalgoritmus	Táblázatkezelés megvalósítás
Sorozatszámítás (feltételes is)	SZUM, SZUMHA, SZUMHATÖBB, ÁTLAG, ÁTLAGHA, ÁTLAGHATÖBB, AB.SZUM, AB.ÁTLAG, FŰZ, ÖSSZEFŰZ
Megszámolás	DARABTELI, DARABHATÖBB, AB.DARAB, AB.DARAB2
Maximumkiválasztás	MAX, MIN
Eldöntés	HA(DARABTELI), HA(DARABHATÖBB)
Kiválasztás	FKERES, VKERES, XKERES, INDEX(HOL.VAN), AB.MEZŐ
Keresés	Eldöntés + Kiválasztás
Másolás	Nincs speciális függvény, valójában a cellahivatkozás másolásával alkalmazható (1. ábra)
Kiválogatás	Irányított szűrés
Feltételes maximum	MAXHA, MINHA
K-adik legnagyobb	NAGY, KICSI
Rendezés	Rendezés (bár csak a feltétel látszik, a módszer nem)

1. táblázat: Táblázatkezelés és sémaalgoritmusok kapcsolata

A fenti táblázatból kiemelendő az eldöntés, a kiválasztás és a keresés algoritmusok táblázatkezelőben való megvalósítása. Egyrészt a táblázatkezelésnél bemutatható, hogy az FKERES és HOL.VAN függvények lényegében a kiválasztást valósítják meg, hiszen nem adnak értelmes választ akkor, ha nincsen meg a keresett elem. Az eldöntést át kell fogalmazni úgy, hogy létezik-e az adott vagy adott tulajdonságú elem (ez a gondolkodásmód már kapcsolatban áll az eldöntés tétel utófeltételével). Ahogy a programozásban a lineáris keresés tételét a kiválasztás és az eldöntés összeépítéséből vezetjük le, ugyanígy látszik, hogy táblázatkezelésnél is a fenti két tétel összeépítésével lehet megvalósítani a lineáris keresést: pl. HA(DARABTELI()>0;FKERES();"Nincs")

A másolás tételére nincs külön függvény, hiszen a tétel szerint ugyanazt a műveletet hajtjuk végre a sokaság összes elemén, és ez eredmény egy sokaság. A táblázatkezelő „képletmásoló" (valójában hivatkozás másoló) funkciója látványosan mutatja be, hogy a másolás során a képletet, tehát a műveletet is „másolom". Így lényegében a függvényleképezést valósítom meg. Ezt mutatja be az 1. ábra.

	A	B	C	
13	1		2	=2*A13
14	2		4	=2*A14
15	3		6	=2*A15
16	4		8	=2*A16
17	5		10	=2*A17
18	6		12	=2*A18
19	7		14	=2*A19
20	8		16	=2*A20
21	9		18	=2*A21
22	10		20	=2*A22

1. ábra: Másolás tételének megvalósítása.

A néhány sémaalgoritmus (más néven programozási tétel) utófeltételének megértésében segíthet a táblázatkezelés. Ehhez a tömbképleteket kell segítségül hívnunk. A táblázatkezelő függvényeivel megvalósított változat sok esetben adja magát (Pl. összegzés, megszámolás, feltételes összegzés, másolás, feltételes másolás). Vegyük például a megszámolás algoritmusát. Ennek az utófeltétele így néz ki:

$$Db := \sum_{\substack{i=1 \\ T(Tömb_i)}}^{N} 1$$

Ahol T jelöli a tulajdonságfüggvényt, N a sokaság (jelen példában tömb) méretét. Jelentése: Ha az adott tömbelem T tulajdonságú, akkor hozzáadok 1-et a Db-hoz. Táblázatkezelőben szó szerint meg lehet valósítani a fenti képletet tömbfüggvénnyel. A nagy szigma jelöli azt, hogy összeadok több elemet, és az alatta levő feltétel azt, hogy mely elemeknél adom hozzá az 1-et az eddigi összeghez. A nagy szigma működését a SZUM, a feltétel-kiértékelést a HA függvénnyel oldja meg a táblázatkezelő, ezért az alábbi képlet megvalósítja a megszámolás tétel utófeltételét: {=SZUM(HA(T(tömb);1;0} Látható, hogy a SZUM egy 0-ákból és 1-esekből álló tömböt fog összeadni, illetve az is, hogy azoknál a tömbelemeknél lesz 1 a SZUM által összeadandó tömbben, amelyik T tulajdonságú volt.

Két korábbi munkánkban bizonyítottuk [5, 8], hogy az összes programozási tétel visszavezethető a sorozatszámítás tételére. Ebben leírtuk, hogy a sorozatszámításra visszavezetett eldöntés algoritmus egy logikai vektor alapján ad helyes megoldást, ahol a logikai vektor i. eleme igaz, amennyiben a tömb i. eleme T tulajdonságú. Az eldöntés algoritmusának két variánsa van: létezik T tulajdonságú elem a tömbben, illetve a tömb minden eleme T tulajdonságú. Könnyen belátható, hogy az alábbi két tömbfüggvény megvalósítja az eldöntés tételét:

- létezik T tulajdonságú elem: {=VAGY(HA(T(tömbelem);IGAZ;HAMIS))}
- minden elem T tulajdonságú: {=ÉS(HA(T(tömbelem);IGAZ;HAMIS))}

Megjegyezzük, hogy az eldöntés tételét úgy is megközelíthetnénk tömbképlettel, mint ahogy azt a „normál” függvényekkel tettük (DARABTELI, DARABHATÖBB), de ez a gondolkodásmód nem vezetne hatékony algoritmushoz, és nem is tudnánk kötni az utófeltételhez.

A tételek összeépítésének megértésénél is segítséget nyújthatnak a tömbfüggvények, erre jó példa a feltételes maximumkiválasztás, ahol az eldöntés tételének utófeltételét kombináljuk a maximumkiválasztásával, azaz, ha létezik T tulajdonságú elem a tömbben, akkor kiszámoljuk ezen tömbelemek

maximumát:
{=HA(VAGY(HA(T(tömbelem);IGAZ;HAMIS));MAX(HA(T(tömbelem);A1:A10;""));"NINCS")}

Sémaalgoritmusok utófeltétele és tömbfüggvények kapcsolatát a 2. ábra és a 2. táblázat mutatja be. A 2. ábrán a T tulajdonság jelentése: a szám páros.

	A	B	C	D
1	1	Összegzés	55	{=SZUM(A1:A10)}
2	2	Megszámolás	5	{=SZUM(HA(MARADÉK(A1:A10;2)=0;1;0))}
3	3	Maximum	10	{=SZUM(HA(MAX(A1:A10;A1:A10)=A1:A10;A1:A10;""))}
4	4	Eldöntés (létezik)	IGAZ	{=VAGY(HA(MARADÉK(A1:A10;2)=0;IGAZ;HAMIS))}
5	5	Eldöntés (minden)	HAMIS	{=ÉS(HA(MARADÉK(A1:A10;2)=0;IGAZ;HAMIS))}
6	6	Feltételes összegzés	30	{=SZUM(HA(MARADÉK(A1:A10;2)=0;A1:A10;""))}
7	7	Feltételes max	10	{=HA(VAGY(HA(MARADÉK(A1:A10;2)=0;IGAZ;HAMIS));MAX(HA(MARADÉK(A1:A10;2)=0;A1:A10;""));"NINCS")}
8	8			
9	9			
10	10			
11				

	E	F	G	H	I	J	K	L
1	Kiválogatás		{=HA(MARADÉK(A1:A10;2)=0;A1:A10;"")}	Másolás	2	{=A1:A10*2}		
2		2			4			
3					6			
4		4			8			
5					10			
6		6			12			
7					14			
8		8			16			
9					18			
10		10			20			
11								
12				Feltételes másolás	1	{=HA(MARADÉK(A1:A10;2)=0;A1:A10*2;A1:A10)}		
13					4			
14					3			
15					8			
16					5			
17					12			
18					7			
19					16			
20					9			
21					20			

2. ábra: Táblázatkezelés példa a sémaalgoritmusok és az utófeltétel kapcsolatára.

Látható, hogy a kiválogatás tételénél – a táblázatkezelő sajátosságai miatt – nem lehetett a tömb folytonosságát megőrizni.

Sémaalgoritmus és Utófeltétel	**Megvalósítás tömbfüggvénnyel**
Összegzés (sorozatszámítás) $\sum_{i=1}^{N} Tömb_i$	{=SZUM(tömb)}
Megszámolás $\sum_{\substack{i=1 \\ T(Tömb_i)}}^{N} 1$	{=SZUM(HA(T(tömb);1;0))}
Maximumkiválasztás (egy maximum esetén) ∀i∈[1..N]: max ≥ $Tömb_i$	{=SZUM(HA(MAX(tömb;tömbelem)=tömbelem;tömbelem;""))}
Eldöntés (létezik egy megfelelő) van := ∃i∈[1..N]: T($Tömb_i$)	{=VAGY(HA(T(tömbelem);IGAZ;HAMIS))}
Eldöntés (minden) minden := ∀i∈[1..N]: T($Tömb_i$)	{=ÉS(HA(T(tömbelem);IGAZ;HAMIS))}
Feltételes összegzés $\sum_{\substack{i=1 \\ T(Tömb_i)}}^{N} Tömb_i$	{=SZUM(HA(T(tömbelem);tömbelem;0))}
Feltételes maximum van := ∃i∈[1..N]: T($Tömb_i$) és Van → 1≤MaxI≤N és T($Tömb_{MaxI}$) és ∀i(1≤i≤N): T($Tömb_i$)→$Tömb_{MaxI}$≥$Tömb_i$)	{=HA(VAGY(HA(T(tömbelem);IGAZ;HAMIS)); MAX(HA(T(tömbelem);A1:A10;""));"NINCS")}
Másolás ∀i∈[1..N]: F($Tömb_i$)	{=F(tömbelem)}
Feltételes másolás ∀i∈[1..N]: T($Tömb_i$) esetén: F($Tömb_i$) különben $Tömb_i$	{=HA(T(tömbelem);F(tömbelem);tömbelem)}
Kiválogatás $Db := \sum_{\substack{i=1 \\ T(Tömb_i)}}^{N} 1$ és ∀i∈[1..Db]: T(Y_i)	{=HA(T(tömbelem);tömbelem;""}

2. táblázat: Sémaalgoritmusok utófeltétele és a tömbfüggvények kapcsolata

5. Összegzés

A fentiek alapján egyrészt beláthatjuk, hogy a táblázatkezelést (a táblázatformázás és a diagramformázás kivételével) miért sorolják inkább a számítástudományi ismeretekhez (az algoritmizálással, programozással együtt), mint a digitális írástudáshoz. Látható, hogy jelentős részben ugyanarról a problémamegoldási területről szól, mint az algoritmizálás.

A két terület egyes témakörei (adatok, típusalgoritmusok) között is nagy hasonlóságot találhatunk, ami előrevetíti, hogy egymás segítségére lehetnek a fogalmaik tanulásában.

Klasszikus tanítási sorrendben előbb tanulnak a diákok táblázatkezelést, mint típusalgoritmusokat (programozási tételeket), azaz a programozástanítást lehetne a táblázatkezelési ismeretekre építeni [1, 2, 9]. Ebben a cikkben azonban megfogalmaztunk olyan lehetőségeket is (ezek a tömbfüggvények), amelyek a klasszikus sorrendben sem előzik meg a hozzájuk illeszkedő programozástanulást, ha egyáltalán megjelennek a tantervekben – tömbfüggvényekkel legfeljebb a tehetséggondozásban foglalkoznak [4].

Irodalom

1. *A 2012-es NAT-hoz illeszkedő Informatika kerettanterv* (2012) https://kerettanterv.oh.gov.hu/05_melleklet_5-12/5.2.21_informat_5-10.doc (utoljára megtekintve: 2020.11.11.)
2. *A 2020-as NAT-hoz illeszkedő tartalmi szabályozók* (2020) https://www.oktatas.hu/kozneveles/kerettantervek/2020_nat (utoljára megtekintve: 2020.11.11.)
3. Biró P., Csernoch M.: *Algoritmusok és/vagy táblázatkezelés?* In: Ujhelyi Adrienn, Lévai Dóra (ed.): VII. Oktatás-Informatikai Konferencia Tanulmánykötet 2015, Budapest (2006) pp. 97–111
4. Molnár K.: Tehetséggondozás az informatikában – Táblázatkezelés. ELTE Informatikai Kar, 2014, http://tehetseg.inf.elte.hu/tananyagok/tablazatkez/index.html (utoljára megtekintve: 2020.11.11.)
5. P. Szlávi, G. Törley, L. Zsakó: *Programming theorems have the same origin* In: Veronika Stoffová, Román Horváth (eds.) New methods and technologies in education and practice: XXX. DIDMATTECH 2017. pp. 52-58 Trnava, 2017. ISBN 978-80-568-0029-4
6. P. Szlávi, G.Törley, L. Zsakó: *The most difficult notion of programming: The variable*, In: Elzbieta Salata; Agata Buda - Education - Technology - Computer Science in Building better future Radom, Lengyelország : Wydawnictwo Uniwersytetu Technologiczno-Humanistycznego w Radomiu, (2018) pp. 108-118. , 11 p.
7. P. Warren: *Learning to program: spreadsheets, scripting and HCI*, in Proceedings of the Sixth Australasian Conference on Computing Education – vol. 30, Darlinghurst, Australia, (2004) 327–333.
8. Szlávi P., Törley G., Zsakó L.: *Az összetett programozási tételek is egy tőről fakadnak* In: Szlávi Péter, Zsakó László (szerk.) INFODIDACT 2017. Konferencia helye, ideje: Zamárdi, Magyarország, 2017.11.23-2017.11.25. Budapest: Webdidaktika Alapítvány, 2017. Paper 21. (ISBN:978-615-80608-1-3)
9. Zs. Szalayné Tahy: *How To Teach Programming Indirectly – Using Spreadsheet Application.* Acta Didactica Napocensia 9 (1), 15-22, 2016. pp. 15-22. ISSN 2065-1430
10. Zsakó L.: Informatika Nemzeti Alaptanterv 2020. In: Szlávi Péter, Zsakó László (szerk.) INFODIDACT 2015. Konferencia helye, ideje: Zamárdi, Magyarország, 2015.11.26-2015.11.27. Budapest: Webdidaktika Alapítvány, 2017. Paper 1. (ISBN: 978-963-12-3892-1)
11. Zsakó L.: Informatikai tantervelmélet? Diszciplínák tanítása – a tanítás diszciplínái 1. Tanulmányok a tudós tanárképzés műhelyeiből. ELTE Eötvös Kiadó, 2015. pp. 92-111. (ISBN 978-963-284-611-8)