Az Informatika Más…

SZALAYNÉ TAHY ZSUZSANNA
PhD hallgató (és középiskolai informatikatanár)
ELTE IK Doktori Iskola
Az informatika (programozás) oktatásának módszerei
sztzs@caesar.elte.hu

CZIRKOS ZOLTÁN
adjunktus
BME-VIK EET
czirkos@eet.bme.hu

Absztrakt

Az új köznevelési szabályozás alapján az informatika óraszámok csökkennek, eközben az informatika gazdasági, társadalmi jelentősége egyre növekszik. Az ellentmondást az informatika tartalmaknak más tantárgyba integrálásával kívánják áthidalni. Ráadásként, közvélekedés, hogy az informatikát nem kell tanulni.

A programozás alapjainak elsajátításához leginkább a matematikát tartják szükségesnek. Sokan úgy gondolják, hogy a programozás a matematika része.

A matematikatudomány felhasználja az informatika eszközeit. Az informatikatudomány nem létezik a matematika nélkül. Azonban kölcsönös egymásra hatásuk mellett a világról más modellt mutatunk a matematika tanítása során, mint a programozás tanításakor. Mindkét modell fontos, mindkét modellt önmagában meg kell érteni. Külön, saját egységében kell értelmezni. A matematikaórán tanított programozás nem lehet hatékony, mert nem tudja figyelembe venni az informatikai modell sajátosságait. Ugyanígy, az informatikaórán a matematika tanítása (szimulációk, feladatok formájában) nem készít fel a matematika érettségire.

Kutatásunk során a középiskolai követelmények és a BME-VIK mérnökinformatikus szak A programozás alapjai I. tantárgy kapcsolatát vizsgáltuk. A NAT, a kerettanterv és az érettségi követelményeket; a közoktatás gyakorlatát hasonlítottuk össze a tantárgy követelményeivel, tananyagával, oktatási gyakorlatával. Kiválogattuk azon a fogalmakat, amelyeknek tanításában zavart okoz, ha összemosódik a matematikai és informatikai szemlélet. Tipikus problémák:

1. Egyes fogalmak különböző értelmezése abból származik, hogy a matematika deklaratív, az állítások tudománya, az informatika imperatív, a kiszámítás tudománya. Sok esetben az ellentmondást az okozza, hogy a matematika elméleti, logikai alapon bizonyít, míg az informatika gyakorlatias, a kivitelezésen van a hangsúly.

2. A matematika és az informatika tudományának művelői ugyanazokat a kifejezéseket használják, de az adott kifejezéshez kapcsolt fogalmak eltérőek, vagy egyes helyzetekben más jelentőségűek. (Pl. „befogó-tétel”, „keresés tétele”)
3. Nem elhanyagolható, hogy a matematikában értelmesek a „végtelen nagy” és „végtelen kicsi” kifejezések, míg az informatikában nem.

Cikkünkben az egyes fogalmak közoktatásban használt értelmezését tárgyaljuk, úgy értjük, ahogy az érettségiző diák illetve első évfolyamos egyetemi hallgató találkozik vele. Úgy használjuk, ahogy érettségin vagy a különböző tantervekben elvárjuk. Konkrét példákon mutatjuk be a matematikai és informatikai modell egyesítésében rejlő ellentmondásokat. E példák megmutatják, hogy miért nem tudja a matematikatanár beépíteni a programozás tanítását a matematika tananyagba; magyarázatul szolgálnak, miért nem volt sikeres a 20 évvel ezelőtti magyar, illetve sok országban máig tapasztalható módszer, amely a matematikaóra keretében tanította a programozást.

A matematika és az informatika közoktatásban megtanítandó fogalomrendszere és gondolkodási sémája szétvált, ebből következik, hogy a két tudomány alapjainak együtt oktatása mindkét kompetencia elsajátításának rovására megy, vagy a hagyományok miatt erősebb matematika mellett az informatikai kompetenciák nem fejleszthetők.

Kulcsszavak: programozás-oktatás didaktika köznevelés matematika informatika

1. Bevezetés

Az egyik szerző 25 évvel ezelőtt matematika-fizika szakos tanárként kezdte tanári pályáját majd számítástechnika-tanári kiegészítő szak elvégzése és szakvizsgát követően informatikát tanít, a másik szerző villamosmérnöki diplomával a mérnökinformatikus képzés alapozó tárgyának, a programozás alapjainak tantárgyfelelőse. Az egyik szerző 50 éves PhD hallgató, a másik fiatal adjunktus. Kutatásunk során két irányból közelítettük meg azt a kérdést, hogy hogyan lehet hatékonyabbá tenni a középiskolai informatikaoktatást, illetve az egyetemi programozás-oktatást. Általános megoldás az ismeretek egymásra építése, ehhez azonban meg kell ismerni mindkét képzést. Az 527 elsőéves mérnökinformatikus hallgató kérdőíves felmérése mellett komoly tapasztalatot jelentett az oktatás intenzitásának (heti óraszám) a figyelése; az oktatás és értékelés módszereinek, a szervezési és a pedagógiai kérdések megvitatása. A kutatás több, saját magunk és környezetünk napi gyakorlatában használható eredménye mellett általános érvényű megfigyeléseink is vannak.

A közoktatásban tanító informatikatanárokat a tudományegyetemek képezik. Jellemző, hogy a képzés a matematikus, a matematikatanári, fizika- illetve technikatanári képzésből vált önállóvá, az oktatók jellemzően matematikusok. Az informatikatanár szakot átképzésben végzőkre már csak azért is jellemző a matematika vagy természettudományi végzettség, mert más szakosoknak egy évvel hosszabb volt az átképzés ideje. Ezen tényezők eredménye, hogy ma a közoktatásban jellemző a programozás elméleti, matematikai szemlélete. A BME-VIK mérnökinformatikus képzése a villamosmérnöki képzésből vált ki. Az informatikát oktatók jelentős

része eredetileg villamosmérnöki végzettségű. Itt a programozás gyakorlata van előtérben. Eltérő előéletünk eltérő „világképet” eredményezett. Az előadások és gyakorlatok hospitálása során, a több mint 200 oldalas levelezés, 30 órányi beszélgetés, (középiskolai) diákokkal és (egyetemi) hallgatókkal való konzultáció során kiderült, hogy nem kettőnk között, hanem a középiskolai matematika és a műszaki egyetemi programozás fogalomhasználata között van eltérés. A kettő között a középiskolai informatikaoktatásnak (pontosabban programozás-oktatásnak) kellene megteremteni a hidat, de ez a diákok jelentős részénél kimarad.
Tanulmányunkban bemutatjuk a tapasztalt fogalmi anomáliákat, a problémáknak a közoktatási szabályozással, illetve közhiedelemmel való kapcsolatát.

2. Az informatika helye az oktatásban

Az informatika helyét a közoktatásban a Nemzeti Alaptanterv (NAT), az ennek megvalósításáról szóló Kerettanterv (KT) és – 10% mértékben – az iskola specialitásai határozzák meg.
A NAT az egyes műveltségterületek oktatáson belüli százalékos részesedését adja meg, valamint az elvárt minimális tartalmat és kompetenciákat rögzíti műveltségterületenként. A Kerettanterv a NAT-ban foglaltakat hivatott tanévekre, óraszámokra lebontva részletezni. Az iskola profiljától függően a Kerettantervtől 10%-ban lehet eltérni, amit néhol óraszámban, máshol a tanított tananyagra értenek. A NAT-ban rögzített műveltségterületek közötti megoszlást az évfolyam – szintén törvényben rögzített – korosztályonként maximált óraszáma alapján lehet tényleges óraszámokra konvertálni.

1. Informatika óraszámok a rendeletek tükrében

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	***Összes***
NAT ajánlás	2-5%	2-5%	2-5%	2-5%	4-8%	4-8%	4-8%	4-8%	4-8%	4-8%	min 4%	min 4%	
max. heti óraszám	25	25	25	27	28	28	31	31	35	36	35	35	
elvi minimum	0,5	0,5	0,5	0,54	1,12	1,12	1,24	1,24	1,4	1,44	1,4	1,4	***12,4***
elvi maximum	1,25	1,25	1,25	1,35	2,24	2,24	2,48	2,48	2,8	2,88	1,4	1,4	***23,02***
KT előírt (gimnázium)	0	0	0	0	0	1	1	1	1	1	0	0	***5***

Forrás: (NAT, 2012), (KT 1-4, 2012), (KT 5-8, 2012) (KT 9-12G, 2012)

A közoktatási informatika óraszáma (5 óra) és az informatika műveltségterület súlyának óraszámra vetítése (12,4–23 óra) közötti különbség indoka, hogy számos országban más tantárgyba integrálva tanítják – tanították – az informatikát. A tananyag és az óraszámok közötti konfliktus megoldása az, hogy az informatikát non-formálisan (otthonról hozza) illetve informálisan (az egyes szaktárgyakon belül) tanulják a diákok. A modern pedagógia ezt a megoldást támogatja azzal, hogy a készségfejlesztésre, a projektmunkára, a tudományágak integrálására fókuszál. Világszerte jellemző, hogy az informatikai alkalmazások (ICT) oktatása más tárgyakba integrálva jelenik meg. Bár nemzetközi trend (Simon Peyton Jones, et al., 2013), (Hubwieser, et al., 2013) az informatika oktatásának – a programozás külön tantárgyban oktatásának – megjelenése, Magyarországon a közgondolkodás és az oktatáspolitika vezetői szerint az informatikát a matematika, fizika, technika, etika, magyar, földrajz… tárgyak keretében tanulja a diák. A más szaktárgyakba beágyazottság konkrét megvalósíthatóságát most nem vizsgálva, műveltségterület ilyen szétforgácsolása felvet néhány problémát: Az informatika tudománya lekorlátozódik a felhasználói ismeretekre, az informatika tantárgy a műveltségterület szakmai megalapozása helyett a többi tantárgyhoz szükséges eszközkezelési ismeretek begyakorlását szolgálja, azaz az informatikaóra jó közelítéssel az internetezés, a szövegszerkesztés, a prezentációkészítés és a játék keveréke (Zsakó, 2014).
Célunk, hogy megmutassuk, mivel az informatika jelentősen több ennél, az oktatás során is jelentősen több figyelmet érdemel.

Számos országban az oktatáspolitikusok is felismerték, hogy az információs társadalomban a társadalmi, gazdasági fejlődés alapja az, hogy az egyén ne csak fogyasztója, hanem értő termelője is legyen az információnak, értse az informatikai rendszerek mibenlétét, működését. A nemzetközi szakirodalomban „computational thinking", magyarul „számítógépes gondolkodás" néven kialakuló fogalom Seymourt Paperttől származik (Pappert, 1996) és Jannette M. Wing értelmezésében terjed világszerte (Wing, 2006), (Wing, 2008). Ma már a legtöbb hazai felsőfokú oktatási intézmény elvárja felvett hallgatóitól a számítógépes gondolkodás képességét (diákok véleménye alapján a színészképzés kivétel). Az ipari, gazdasági szereplőktől, a műszaki, gazdasági és természettudományi képzést folytató intézmények részéről egyetemvezetői nyilatkozatoktól az oktatási gyakorlatig minden szinten megfogalmazódik ez az elvárás; jellemzően a programozás alapjainak ismeretét hiányolva (Kam, 2012), (Információs Tásadalom Parlamentje, 2014).

Mivel a közoktatás – a jelenlegi keretek között – nem biztosítja a programozás alapjainak megismerését, a felsőoktatás az informatika alapjai helyett matematikai alapokat vár el.

Célunk, hogy megmutassuk, az informatika más. Nem helyettesíthetjük matematikai alapozással a programozás alapjait.

Szeretnénk megmutatni, hogy az informatika más… de mégis, milyen? Ki, mit ért programozás alapjain? Megfigyelhető, hogy Pappert (Pappert, 1996) a számítógépes gondolkodás fogalmát a matematikából vezette le, J. M. Wing (Wing, 2006) ezzel szemben más alapokra, főleg példákra hivatkozik a fogalom kifejtésénél. A világ számos országában megértették, hogy a számítógépes gondolkodás nem a matematika számítógépes leképezése, hanem egy másmilyen gondolkodás, egy másik nyelv (Cohen & Haberman, 2010), egy másik tudomány, az informatika eszköze.

Kutatásunk során számos példát láttunk, amelyben felmerült a kétely, hogy a számítógépes gondolkodás – avagy a programozás alapjai – elsajátítható-e a matematikai ismeretek, készségek fejlesztésével. Azt tapasztaltuk, hogy matematikaórán nem lehet programozást tanítani, így megtanulni sem lehet a matematikaórán programozni. A matematika tantárgy kiegészítéseként sem vált be a programozás oktatása (Borsányi & Hack, 1990). A matematika tananyaghoz képest a programozás (az informatika) más tudást jelent.

3. Kutatási tapasztalatok

A BME-VIK mérnökinformatika szakos képzésére az ország számos középiskolájából érkeznek hallgatók. Sikeres érettségit tettek és felvételi pontszámuk is elég magas volt – több mint 360 pont – ahhoz, hogy felvételt nyerjenek. Biztosan tanultak matematikát, legalább közép szinten, hiszen ez kötelező érettségi tárgy. Valószínűleg tanultak valamilyen informatikát. Programozási ismereteik a nullától a nemzetközi versenyeken is kamatoztathatóig terjed. A 2014-2015-ös tanév első félévének mérnökinformatikus hallgatói között kérdőíves felmérést végeztünk. Hetente, A programozás alapjai I. tárgy előadásaihoz kapcsolódó 3–11 feleletválasztós kérdésre önkéntesen válaszoltak a hallgatók. 486 új hallgató 66%-a válaszolt legalább egy kérdésre. Az előismeretekre (hallottál-e már róla korábban), az előadáson elhangzottak megértésére, tanulási szokásokra és szubjektív véleményre is rákérdeztünk. A kérdőíves válaszokat kiegészítették a jelenlét és teljesítmény hivatalosan mért adatai (kis és nagy zárthelyik eredménye…). Valamint az összes előadás és gyakorlat hospitálása. Megfigyeléseinkből három momentumot emelünk most ki:

a) A tantárgy teljesítését nagymértékben előrevetíti az első nagy zárthelyi.

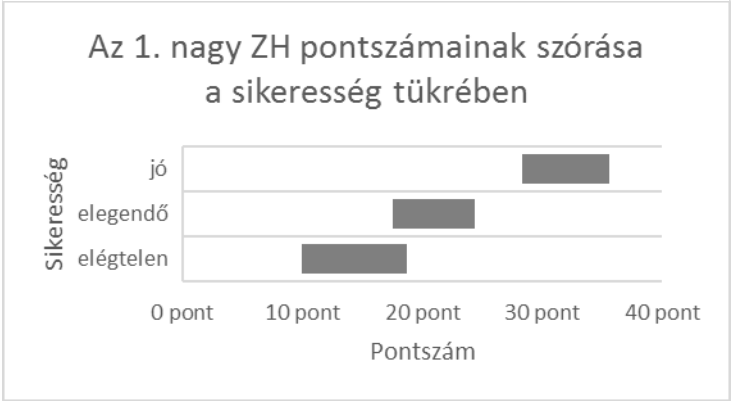
A teljesítés három kategóriáját határoztuk meg: „elégtelen", ha a hallgató nem kapott aláírást (pl. hiányzások miatt) vagy nem érte el a minimális pontszámot; „elegendő", ha jegye 2 vagy 3; „jó", ha jegye 4 vagy 5.

A tantárgy teljesítésének hat feltétele van, amiből az egyik, hogy a két nagy zárthelyi pontszámának összege legalább 40 pont legyen. Az első nagy zárthelyin ennek a fele (20 pont), a szórást is figyelembe véve, előre vetíti a jegyet.

1. Táblázat:
az első zárthelyi átlaga és szórása a tárgy sikeressége szerinti bontásban

	Átlag	Szórás
jó	32,0	7,3
elegendő	21,0	6,9
elégtelen	14,3	8,8

1. Diagram



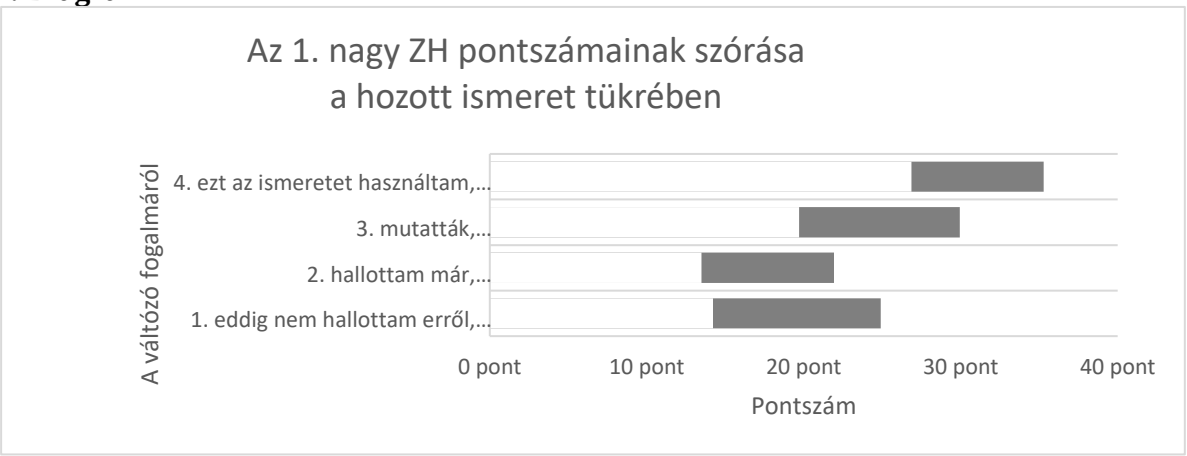
b) Az első zárthelyi dolgozatot összevetve az első héten feltett, korábbi ismeretre vonatkozó kérdésekkel, kimutatható az előképzettség pozitív hatása a tantárgy teljesítésére.

A feltett kérdés: *Hallottál már az első előadást megelőzően a változókról?*
A kérdésre a hallgatók 48%-a válaszolt. Leszűkítve a válaszolókra a 1. Táblázat számításait az eredmény rendkívül szoros kapcsolatot mutat a teljes létszámra vonatkozó adatokkal (r = 0,998).

2. Táblázat:
az első zárthelyi átlaga és szórása a kérdésre adott válaszok szerinti bontásban

	1. eddig nem hallottam erről, teljesen új	**2. hallottam már, de sosem próbáltam**	**3. mutatták, néhányszor próbáltam**	**4. ezt az ismeretet használtam, rutinom van belőle**
Átlag	20	18	25	31
Szórás	10,7	8,4	10,2	8,4

2. Diagram



Azoknak a hallgatóknak, akik az egyetem előtt írtak már néhány programot önállóan, a zárthelyi eredménye lényegében a jó teljesítményű hallgatókéval egyenlő. Akik nem programoztak önállóan, azoknak az egyéni képességeiktől függ, hogy sikerül-e teljesíteni a zárthelyi követelményeit, jelentősen nagyobb a valószínűsége annak, hogy közülük kerülnek ki a kibukók.

c) A hallgatóknak sokszor nehézséget okoz a matematikai ismeretek összehangolása a programozási ismeretekkel.

Érdemes megfigyelni az 5. héten, az asszociativitás fogalmával kapcsolatos kérdésre kapott válaszokat.

42. kérdés: Hogy is van ez? Melyik igaz?

A. Az összeadás művelete asszociatív, az osztás művelete nem asszociatív

B. Az "a/b/c" balról asszociatív, az "a=b=c" jobbról asszociatív

A válaszolók száma a 2. heti kérdésekhez képest csökkent, már csak 15%. Mivel a tárgy teljesítésének a sikere az előzetes ismeretekkel és az első zárthelyivel összefügg, nézzük a válaszokat is ennek megfelelő bontásban:

3. Táblázat: a 42. kérdésre adott válaszok a tárgy teljesítése sikerességének tükrében

Sikeresség	A igaz B hamis	A hamis B igaz	A igaz B igaz	A hamis B hamis
elégtelen	2	0	2	0
elfogadható	8	0	4	2
jó	18	16	18	5

Az „A" válasz az asszociativitás középiskolai matematika tananyagából származik, a „B" válasz a kérdés kiadását megelőző előadáson hangzott el. Mindkettő a saját környezetében igaz. A jókat a többiekkel összevetve az a legfeltűnőbb, hogy körükben a matematikai alapú megfogalmazást többen (16) hamisnak jelölték. Vélhetően a két állítás között ellentmondást véltek felfedezni és erősebb volt az utoljára hallott informatikai szemlélet, mint a matematikai. Azonban összességében a matematikai megfogalmazást jóval többen vélték helyesnek, azaz az érettségire vagy matematikából illetve digitális technikán tanult ismeret elég erősnek bizonyult, a kifejezés előadáson használt értelmezése azonban csak félig tudott érvényesülni.

Ezen és még néhány hasonló példa alapján kezdtük kigyűjteni a hallgatók által korábban nagy valószínűséggel tanult, de az adott helyzetben másképp használt fogalmakat. Tapasztalatunk szerint a legtöbb eltérés abból adódik, hogy az adott fogalom másképp jelenik meg a matematika tanítása során, mint ahogy programozásban használjuk.

Másik oldalról, egy-egy többértelmű fogalommal kapcsolatban a Szent István Gimnázium diákjai körében ellenőriztük a tapasztalatokat.

A következő fejezetekben a programozás tanulását gátló fogalomzavarokból mutatunk be néhány jellemző példát.

3.1. Félrevezető fogalmak

A közoktatásban a hallgatók megtanulták a matematika legfontosabb fogalmait, probléma-megoldási eszközeit. A programozás tanulása során ezek közül nagyon sokat felhasználhatunk. Lássunk néhány példát:

1. *példa*

Definíció:
int a;
Azt mondjuk, hogy az 'a' szám egész szám, ha előállítható két természetes szám különbségeként.

Ez a példa több fogalom eltérő értelmezésére mutat rá. Egyrészt a „definíció" fogalma mást jelent a programozás és mást a matematika terén, másrészt az egész szám értelmezése is eltérő. Matematikai értelemben a definíció a használt fogalmak tudományos meghatározása. Az informatika más… a definícióval egy változónak vagy függvénynek a típusát és nevét határozzuk meg. Egy névhez, különböző programokban, más és más definíciókat adhatunk meg, egyik esetben sem nevezhetjük ezt tudományos meghatározásnak. A példában informatikailag az

„a”-t definiáljuk, viszont a matematikában az „egész szám”-ot. Lehet, hogy egy okozza az eltérést? Hogyan definiáljuk az „egész szám”-ot informatikában, illetve az „a”-t matematikában? A matematikaórán az „egész szám” egy számhalmaz eleme; az informatika az „egész számot” többféle implementációban értelmezi, amelyek a programozási nyelv sajátosságai.
Az informatikai „változó”-t a matematika nem definiálja, hanem bevezet rá egy jelölést. Matematikaórán „változó”-ja a függvényeknek van, amit informatikában leginkább a függvény „argumentuma” vagy „paramétere” néven emlegetünk.
Matematikaórán nem igazán beszélünk argumentumról, a „paraméter” viszont leginkább a „paraméteres egyenletek megoldása” témához kapcsolódik, ahol a paraméter ismertnek tekintett kifejezés, amit az informatikában jellemzően konstansként jelölünk meg. …és ekkor még a függvény fogalmának jelentősen kibővített informatikai értelmezéséről nem mondtunk semmit.
Matematika érettségivel rendelkező, programozást korábban nem tanult hallgató a „Definiáljunk egy egész típusú változót!” mondat szavait egyenként érteni véli, de a feladattal – matematika tudására alapozva – nem tud mit kezdeni.
Elvárható a hallgatótól, hogy kreativitásával élve automatikusan újraértelmezze a „definíció”, az „egész” és a „változó” fogalmát? Azért ez nem jelenthet problémát… De e három fogalom újraértelmezése indukálja számos további fogalom újraértelmezését is. Melyik lesz az a pont, amikor már csak a fogalomzavar marad meg?
Elvárható a matematikatanártól, hogy a kifejezések informatikai értelmezését is megtanítsa? Akkor tanítsa meg az informatikai értelmezését is a fogalomnak, amikor a matematikait vagy máskor? Célszerű egy órán belül többször váltakozva használni az azonos hangzású, de matematikai és informatikai fogalomként eltérő szavakat?
A matematikaórán az egyes fogalmak matematikai értelmezése a feladat. Ez – azaz az, hogy a diák megértse, megtanulja a matematikát, – a matematikatanárnak bőven elegendő feladatot ad, gyakorlatilag nem várható el, hogy matematikaórán kitérjen az informatikai fogalmak tanítására.

3.2. A végtelen végessége

2. *példa*

Az x valós szám: $x \in \mathbb{R}$ vagy double x;

A valós szám fogalmának kialakítása általános iskola végén és a középiskola első éveiben tananyag. Ezzel párhuzamosan kell a diákoknak megérteniük a végtelen, végtelenül kicsi és végtelenül nagy fogalmakat. Évekbe telik a „végtelenségig folytatható sorozat” illetve a „tetszőlegesen kicsi intervallum” megértése.
Ha nagyon sokszor kellene valamit megcsinálni, akkor a matematika az informatikát hívja segítségül. Az informatika azonban csak véges számosságot, véges pontosságot értelmez.
Matematikaórán heteket töltünk azzal, hogy $\sqrt{2} = 2/\sqrt{2}$. Mikor tanulja meg a diák, hogy amikor informatikában valós számról beszélünk, akkor matematikailag véges jeggyel tárolt racionális számokról van szó? Mikor tanulja meg, hogy cos(90°) nem egyenlő nullával, sőt, bármely kétféleképpen kiszámított, nem egész érték matematikai egyenlősége informatikában nem vizsgálható?
Míg a matematikaórán a szám fogalmát a végtelenség és a folytonosság felé tágítjuk, informatikából a szám mindig véges és diszkrét. Matematikában a számhalmazokat az értelmezett műveletek határozzák meg, informatikában a tárolás mérete és módja. Így különböztetünk meg – egyes programozási nyelvekben egyedileg – int, long, short, byte, unsigned int és még számos más „egész” típusú értéket, illetve real, double, single néven lebegőpontos számábrázolást. A számábrázoláshoz definiált tárolási méret és tárolási módszer (abszolútértékes előjeles, kettes komplemens), az előjel és túlcsordulás kezelése, azaz a szám tárolásának megvalósítása határozza meg az értelmezhető és értelmezett műveleteket, a műveletek algoritmusait.

1. ábra: Az ✋ ujjak számának néhány tárolási módja

5 – 4 bájtos egész	|00000000|00000000|00000000|00000101
5.0 – 3+1 bájtos	|00100000|00000000|00000000/00000010|
′5′ – char (53d 35h)	|00110101|
″5.0″ – string (0035,002E,0030)	|00000000|00110101|00000000|00101110|00000000|00110000|
"öt" – string (00F6,0074)	|00000000|11110110|00000000|01001010|
"V" – római szám (0056)	|00000000|01010110|

Például informatikában az 5, az 5.0, az ′5′, az ″5″ ″öt″ és ″V″ egyikes sem azonos semelyik másikkal. Az azonosnak tűnő műveletek eredménye függ a típustól:

- 1 / 5 ⇒ 0 1 div 5 értelemben;
- 1.0 / 5.0 ⇒≈ 0.2 az érték kerekített, mert végtelen szakaszos kettedes tört lenne;
- 1 + 5.0 ⇒ elvégzéséhez 1 + 5 vagy 1.0 + 5.0 algoritmusa nem alkalmas;
- '1'+'5' ⇒ 'f' megjelenítés módjától függően, esetleg 102;
- "1"+"5" ⇒ "15"
- "egy"+"öt" ⇒ "egyöt"
- "I"+"V" ⇒ "IV" azaz négy, nem hat.

A matematika és informatika alapvető fogalmairól van szó. Egyik tananyagba sem lehet konstruktívan beépíteni a másik tananyag fogalmait. Ugyanakkor a két tananyag között fontos lenne meghatározni a fogalmak fejlesztésének ütemét, hogy beépülhessen az oktatásba a két modell különbségének az elemzése. Valószínűleg hatékonyabb lenne először az informatikai szemléletmódot megtanítani (véges pontosság, végig megyek a sorozaton) és a matematikai modellt ennek ismeretében kiterjeszteni.

3.3. Hamiskás állítások

3. példa:

```
if(sorozat[i] == 3 && i < sorozat.Count)
```

Matematikai műveletként az összeadás, a szorzás, az AND, az OR… kommutatív műveletek. Informatika érettségi tétel szól az algebrai műveletek tulajdonságairól, a Boole-algebra azonosságairól. Programozás szempontjából nézve azonban alapvetően a logikai rövidzár kiértékelést használjuk, listák, adattípusok közötti műveleteknél a hosszabb, nagyobb helyfoglalású elem jelenti az alapot, a processzor a művelet eredményét két regiszter között az elsőbe képezi le. Matematikaórán bizonyítjuk a műveletek kommutativitását, amit az informatika a programozás gyakorlatában rendszeresen megcáfol.

A műveleteken túl, a relációk egyike sem kommutatív. A megértést gátolja a feltételek Yoda-feltétel stílusú írása: if (0 == *x*)…

Ehhez társul az értékadó egyenlőség értelmezése. Matematikából értelmes egy új y változó megadására x + 2 = y kifejezés, amivel számos esetben találkozunk kódolás során is, ahol súlyos hiba.

Abban az esetben, ha matematikaórán kell informatikát, programozást tanítani, vajon milyen arányban lesznek a műveletek kommutatívak? Milyen hatással van a matematikai szemlélet kialakítására, ha egy tételt bizonyítunk, majd a következő pillanatban kihasználjuk az állítás tagadását? Összehozható-e egy tananyagba az matematika elméleti felépítése és szigorú struktúrája és az informatikai megvalósítás? A matematika és reáltantárgyak kapcsolatát vizsgálva úgy tűnik, hogy nem. Ahogy nem építhető egybe a matematika- és technika- vagy fizikaoktatás, ugyanígy nem alkalmas a matematikaóra a programozás tanítására sem. Az informatika más…

3.4. Ha… akkor…

Számos egyéni példán tapasztaltuk, hogy a Ha… akkor… mondatszerkezet többértelműsége problémát okoz a diákoknak, hallgatóknak.

4. *példa, hallgató kérdése:*

> *Az A⇒B esetén hogyan programozzuk le, hogy ha A hamis, akkor B bármi lehet? Mit írunk az else-ágba?*

Matematikatanári szemmel nézve, a Ha… akkor… szerkezet egy implikáció, az egyik logikai művelet. Tanítjuk a tagadását és rendszeresen használjuk bizonyításokhoz, feladatok megoldása során az eredmény levezetéséhez. Oktatásának módszertana kutatási téma (Herman, et al., 2012)

Az informatika, a programozás tanítása szempontjából a Ha… akkor… szerkezet egy elemi vezérlési struktúra, az alternáció. Tanítjuk egy-, két- és sokágú változatát; alapvető szerepe van az algoritmusok megfogalmazásában, a folyamatok vezérlésében.

Sokéves tapasztalat, hogy van olyan diák/hallgató, aki matematikai alapokra építve tanulja a programozást és elakad az alternációnál, mert az implikációban a következmény „lesz”, így nem világos számára, hogy miért kell értéket adni. Másik oldalról… egyre gyakrabban találkozhatunk olyan diákokkal, akik sorra írják a programokat (jellemzően játékprogramokat), de matematikából gyengék, nem tudják megoldani a bizonyításos, hosszabb levezetést igénylő feladatokat, nem „érzik” az indirekt bizonyítás bizonyító erejét.

Úgy sejtjük, hogy a különbség nem csak az implikáció és alternáció értelmezése között érvényesül. A gondolkodási mód alapjaiban különbözik a kétféle értelmezésben.

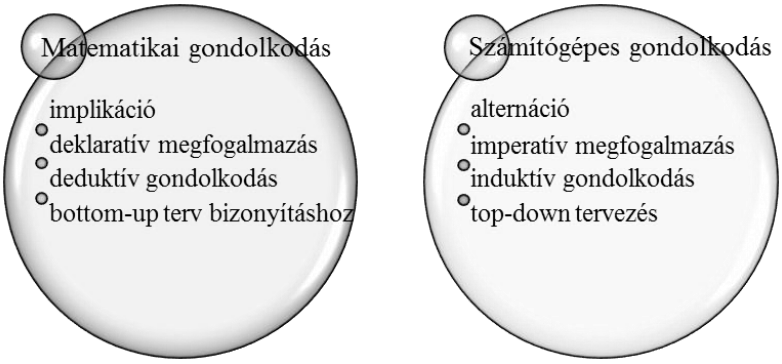
3.5 A gondolkodás tanítása

Bár a közoktatásbeli matematika és a programozás alapjai az implikáció és alternáció közötti különbségre világít rá, az eltérő gondolkodásmód tükröződik a bottom-up- kontra top-down tervezés készségében, a deklaratív kontra imperatív nyelvek preferálásában (Abelson & Sussman, 1996), a programtervező matematikus kontra mérnökinformatikus szemléletben és a deduktív kontra induktív gondolkodásban.

Közoktatásban, illetve kezdő mérnökinformatikus hallgatók körében azt tapasztaljuk, hogy van, aki sok ismeretet egymáshoz tud illeszteni, a megoldandó feladatra ránézve meglátja a megoldás útját és levezeti. Van, aki „egyszerűen nem látja meg”, hogy mit kellene hozzátenni a feladathoz, nem tudja levezetni. Mindkét típusból megtalálhatók olyanok, akiknek nem okoz gondot egy feladat algoritmikus részekre bontása, az egyes ágakon a megvalósítás kidolgozása, a rendszertervezés, másoknak egy keresés algoritmusa is bonyolult, mert nem látja át, hogy az egyes esetekben mit kell tenni.

A matematika tananyagban sokkal nagyobb súllyal szerepel az implikáció, bottom-up tervezés, a deklaráció, a dedukció, levezetés; míg a programozás alapjaira az alternáció, top-down tervezés, utasítások kiadása jellemző.

2. ábra: Domináns gondolkodási elemek



Kutatásunk során tehetséges diákokat és programozást megtanulni akaró hallgatókat vizsgáltunk. Azt tapasztaltuk, hogy problémát jelent a kétféle gondolkodásmód megtanulása, nem fejlődik az egyik terület pusztán a másik fejlesztésétől.

5. *példa, programozás versenyen döntős diákok gondolatai:*

   Matematika versenyeken is döntős, 10, évfolyamos: „A számítógépnek tudnia kell, hogy 3·⅓ az egy. A ⅔-ot úgy kell kiszámítania, hogy 1-⅓, akkor nem fogja elrontani."

   Matematikát nem szerető, 11. évfolyamos: „Brute Force-szal "

A gondolkodásmódbeli különbség a matematika és informatika oktatása során válik jelentőssé. Bár mindkét területen a gondolkodás fejlesztése hangsúlyos, nem lehet a közoktatás szintjén együtt fejleszteni a két készséget. Kétségtelen, hogy egymást kiegészítő, egymásra ható készségekről van szó, de, mint az érem két oldalát, a matematikai és számítógépes gondolkodást is külön-külön kell megismerni. Az információs társadalom elvárásainak megfelelés, a számítógépes gondolkodás fejlesztése, az egyetemeknek a felvételre jelentkező hallgatókkal szembeni elvárása egyaránt a programozás alapjaival kapcsolatos gondolkodási készségeknek, a számítógépes gondolkodásnak a meglétét jelentik.

Összefoglalás

Kutatásunk alapját a BME-VIK mérnökinformatikus képzés A programozás alapjai I. tárgy hatékonyságának vizsgálata adta. A folyamatos monitoring és mérések eredményeiből kiderült, hogy a tárgy teljesítésének problémái a hozott ismeretek hiányosságára, illetve a hozott és a tanított tananyag inkonzisztenciájára vezethető vissza. A problémák jelentős része a közoktatás matematika tananyaga és a programozás alapjai tárgy elsajátításához szükséges informatikai fogalomrendszer és szemléletmód eltéréséből adódik. Tételesen:

- A matematikaórákon tanult és az informatikához szükséges fogalmakra jellemző, hogy azonos szóalakhoz a két tudomány más fogalmat társít.
- A matematika tudomány közoktatásban tanított modelljei, az ezekben megfogalmazott tételeknek ellentmond az informatika tudomány megalapozásához használt modell és megvalósítási módszerek.
- A matematika tantervben rögzített gondolkodási, probléma-megoldási képesség és a programkészítéshez szükséges készségek eltérők, egyik sem helyettesíti a másikat.
- A programkészítéshez szükséges készségek, a számítógépes gondolkodás megléte nem csak a mérnökinformatikusok képzésében, hanem az információs társadalom minden területén elengedhetetlen.
- A jelenlegi közoktatásban a számítógépes gondolkodás fejlesztése nem kap megfelelő teret.
  - az informatikaórán alacsony óraszámban legfeljebb érintőlegesen lehet fejleszteni

- más szaktárgyak – így a legalkalmasabbnak tekintett matematika – a tananyagba ágyazottan ilyen irányú fejlesztésre alkalmatlan.

Ugyanazok a szavak mást jelentenek a két szakmában, amelyeknek keverése didaktikai szempontból káros. Ahogy az is káros didaktikai szempontból, ha összemossuk a kétféle gondolkodásmódot. A diákoknak meg kell tanulni logikusan érvelni matematikaórán és modellt alkotni, algoritmizálni, programozni, tesztelni informatikaórán ahhoz, hogy mindkettőt a megfelelő helyen és módon tudják alkalmazni.

Irodalomjegyzék

Törvények, rendeletek:

Kerettanterv az általános iskola 1–4. évfolyamára - Szabadon választható - Informatika. 51/2012. (XII. 21.) számú EMMI rendelet. Letöltve: http://kerettanterv.ofi.hu/01_melleklet_1-4/1.3.3_informat_1-4.doc

Kerettanterv az általános iskola 5–8. évfolyamára - Informatika. 51/2012. (XII. 21.) számú EMMI rendelet Letöltve: http://kerettanterv.ofi.hu/02_melleklet_5-8/2.2.15_informat_5-8.doc

Kerettanterv a gimnáziumok 9-12. évfolyama számára – Informatika. 51/2012. (XII. 21.) számú EMMI rendelet. Letöltve: http://kerettanterv.ofi.hu/03_melleklet_9-12/3.2.16_informat_9-12.doc

Nemzeti Alaptanterv. Magyar Közlöny 2012. évi 66. kiadás, 2012. június 4. a 110/2012. (VI. 4.) Korm. rendelet melléklete. Letöltve: http://net.jogtar.hu/jr/gen/hjegy_doc.cgi?docid=A1200110.KOR

Könyvek

Abelson, H. & Sussman, G. J., (1996): *Structure and interpretation of computer programs*. 2 szerk. London, England: The MIT Press.

Borsányi, K. & Hack, F., (1990): *Matematika Feladatgyűjtemény III.*. 3 szerk. Budapest: Tankönyvkiadó.

Folyóirat cikkek, konferencia közlemények:

Cohen, A. & Haberman, B., (2010): *CHAMSA: Five Languages Citizens of an Increasingly Technological World Should Acquire*. ACM Inroads, 1(4).

Herman, G. L., Loui, M. C., Kaczmarczyk, L. & Zilles, C., (2012): *Describing the What and Why of Students' Difficulties in Boolean Logic*. ACM Transactions on Computing Education, 12(1), pp. 3.1-3.28.

Hofoku, Y., Cho, S., Nishida, T. & Kanemune, S., (2013): *Why Is Programming Difficult? - Proposal For Learning Programming In "Small Steps" And A Prototype Tool For Detecting "Gaps"*. In: Informatics in Schools. Sustainable Informatics Education for Pupils of all Ages. Potsdam: Springer.

Hubwieser, P. és mtsai., (2013): *Pedagogical Content Knowledge For Computer Science In German Teacher Education Curricula*. WiPSE '13 Proceedings of the 8th Workshop in Primary and Secondary Computing Education

Kam, M., (2012): *Rethinking Computation in the Engineering Curriculum*. Philadelphia: Mid Atlantic ECEDHA Meeting.

Pappert, S. (1996): *An Exploration in the Space of Mathematics Educations*. International Journal of Computers for Mathematical Learning, 1(1), pp. 95-123. Letöltve: http://www.papert.org/articles/AnExplorationintheSpaceofMathematicsEducations.html

Simon Peyton Jones, Bill Mitchell & Simon Humphreys (2013): *Computing at school in the UK*. Elérhető: http://research.microsoft.com/en-us/um/people/simonpj/papers/cas/computingatschoolcacm.pdf

Wing, J. M. (2006): *Computational Thinking and CS@CMU*. Carnegie Mellon University. Letöltve: http://www-cgi.cs.cmu.edu/afs/cs/usr/wing/www/CT_at_CMU.pdf

Wing, J. M. (2008): *Computational thinking and thinking about computing*. Philosoohical Transactions of The Royal Society A, 28 10, 366(1881), p. 3717–3725. Letöltve: http://rsta.royalsocietypublishing.org/content/366/1881/3717.short

Zsakó, L. (2014): *Miért elavult informatikából a NAT, a kerettanterv, az érettségi?* Készült az "Országos koordinációval a pedagógusképzés megújításáért" című TÁMOP-4.1.2.B.2-13/1-2013-0007 pályázat keretében. Letöltve: http://infoera.hu/infoera2014/ea/problemak-az-informatika-oktatassal.ppt

Internetes hivatkozások:

Információs Társadalom Parlamentje, 2014. *IT Parlament 2014 - kerekasztal-beszélgetés a köznevelés fejlesztéséről*. Elérhető: http://infoter.eu/video/it_parlament_2014_kerekasztal-beszelgetes_a_kozneveles_fejleszteserol [Hozzáférés dátuma: 2014. október 29.].