Gradus Vol 4, No 1 (2017) 216-222
ISSN 2064-8014

A PROGRAMOZÁSI KÉSZSÉGEK VIZSGÁLATA KÖZÉPISKOLÁBAN ÉS A FELSŐOKTATÁSBAN

ANALYSIS OF PROGRAMMING SKILLS IN SECONDARY SCHOOLS AND COLLEGES

Pap-Szigeti Róbert [12*]

[1] Kecskeméti Bolyai János Gimnázium, Magyarország
[2] Informatikai Tanszék, GAMF Műszaki és Informatikai Kar, Pallasz Athéné Egyetem, Magyarország

Kulcsszavak:
Készségrendszerek
Programozási készségek
Strukturált programozás

Keywords:
System of skills
Programming skills
Structured programming

Cikktörténet:
Beérkezett 2016. szeptember 8.
Átdolgozva 2016. október 31.
Elfogadva 2016. november 5.

Összefoglalás

A tanulmányban a strukturált programozás során alkalmazott ismeretek és készségek azonosítására teszünk kísérletet, majd a feltárt összetevők fejlettségének értékelési lehetőségeit mutatjuk be. A szakirodalmi tanulmány és a tervezett mérőeszközök egy nagyobb vizsgálat első lépéseit jelentik. Ebben a vizsgálatban az emelt szintű érettségi vizsgán elvárt programozási tudás fejlettségét befolyásoló ismeretek, készségek és motívumok rendszerének vizsgálatát tűztük ki célul.

Abstract

In this paper we make an attempt at the identification of knowledge and cognitive skills used in structured programming tasks. After that we show assessing opportunities of the development of identified components. The literature review and the planned tests are first steps of a wide examination, in which we plan to identify the system of knowledge, skills and motives influencing the development of the programming skills expected on the final exam.

1. Bevezetés

Az informatikára épülő technológia eszközei korábban a közvetlenül az informatikával foglalkozó szakemberek munkaeszközeit jelentették. Mára számos munkaterületen, a háztartásban, a tanulásban, a szórakozásban, a közlekedésben, általában az élet sok területén alkalmazzuk őket. A programozás – annak professzionális szintje – továbbra is az informatikusok egy szűkebb rétegének feladatát jelenti. Az új technológiák megértését, az eszközöknek az új problémák megoldásában való hatékony alkalmazását azonban jelentősen segítheti az alapszintű algoritmikus és programozási készségek elsajátítása. [1] A 2013 óta megrendezésre kerülő (Európai) Programozás Hete (EU Code Week) rendezvénye arra hívja fel a figyelmet, hogy „a programozás […] olyan alapvető készséggé vált, melynek – legalább alapszintű – elsajátítására mindannyiunknak szüksége lesz”. [2]

Az informatika hazai közoktatásában az algoritmizálás alapfogalmai és egyszerű készségei, valamint a robotika alapjai az általános iskolai tantervben megjelennek. Ehhez kapcsolódóan számos iskolában alkalmaznak – tanórai vagy szakköri kereketek között – gyerekek számára készített programozási nyelveket (a legismertebbek a *LOGO* különböző változatai és a *Scratch*).

* Kapcsolattartó: Pap-Szigeti Róbert.
E-mail cím: pap-szigeti.robert@gamf.kefo.hu

[3] A mobil robotok programozási lehetőségeit felhasználó szakkörök és versenyek is népszerűvé váltak. [1] A középiskolai kerettanterv elsősorban az algoritmusokra vonatkozó ismeretek és készségek körét bővíti, valamint előírja az egyszerű algoritmusok megvalósítását számítógépen. A tantárgy órakerete az alapóraszámban nem teszi lehetővé egy strukturált programozási nyelv alapjainak elsajátítását, így a programozás tanulására – iskolai keretek között – a szakkörök, informatika tagozatok illetve az emelt szintű érettségire felkészítő kurzusok nyújtanak lehetőséget.

A felsőoktatásban a programozás elsősorban az informatikai és műszaki képzések tananyagában jelenik meg. Az empirikus vizsgálatok tanúsága szerint egyrészt a programozási készségek elsajátításának szintjét jelentősen befolyásolja az, hogy a hallgató a közoktatásban foglalkozott-e programozással [4], másrészt a programozáshoz kapcsolódó motívumok fejlődésében szerepet játszik, hogy milyen eszközökkel és módszerekkel kerül sor a programozás oktatására. [5]

A hazai informatikaoktatás azonban annak ellenére kevés neveléstudományi eredménnyel rendelkezik a programozási készségek fejlődésével illetve a fejlettséget befolyásoló készségekkel, motívumokkal kapcsolatban, hogy széles életkori tartományban történik a programozás oktatása. A tanulmányban összegezzük a legfontosabb hazai és külföldi eredményeket, felvázolunk egy a programozás sikerességét leíró lehetséges modellt, valamint megfogalmazzuk a strukturált programozási nyelven történő programozás készségeinek értékelésére alkalmas mérőeszközök követelményeit.

2. A programozási készségek vizsgálatai

Az algoritmizálás és a programozás készségei, ismeretei és motívumai alapvetően a kognitív és szakmai kompetenciák körébe tartoznak [6]. A működtetett készségek egy része azonban – így például a hétköznapi algoritmusok megértése és követése; mások eltérő gondolkodási metódusának megértése – a személyes és szociális kompetenciának is fontos összetevője. A programozás sikeres elsajátításához és alkalmazásához szükséges készségek fejlettségének értékeléséhez fontos, hogy meghatározzuk a készségeket felépítő részkészségeket, rutinokat, valamint a szükséges ismeretek körét. Ennek érdekében a következő fejezetekben röviden leírjuk a készségfejlődés értékelésének módszereit, majd áttekintjük a programozásra irányuló hazai és külföldi vizsgálatok módszereit és eredményeit.

2.1. A kognitív készségek fejlődéséről

A kompetencia fogalmára a hazai szakirodalomban többféle meghatározást is találhatunk. Ezekben a meghatározásokban közös, hogy a kompetenciákat összetett rendszernek tételezik fel, amelyek számos képesség, készség, ismeret és motívum egyidejű aktivációjával működnek. [6] [7] A kompetenciát alkotó összetevők további elemekből – részkészségekből, rutinokból – épülnek fel, amelyek száma és kapcsolatrendszere meghatározza az adott készség, képesség összetettségét. A készségrendszerek, képességek fejlődése ebből adódóan hosszú időt – az összetettségtől függően egy vagy akár több évet – vehet igénybe, ezt az ezredforduló környékén zajlott empirikus vizsgálatok eredményei is alátámasztották. A fejlődés ütemét jelentősen befolyásolják a tanulók motívumai is. [7]

A készségfejlődés általában nem lineáris. [8] Az átlagos fejlődés kezdetben (a rendszer kiépülésének kezdetén) a kognitív készségek többsége esetén lassú, ezt általában egy gyorsuló fejlődési szakasz követi, majd a fejlődés a készség teljes begyakorlása közelében lelassul. Az egyéni fejlődés jellegében hasonló lehet, természetesen nagy egyéni különbségekkel, emellett az egyéni fejlettségben átmeneti visszaesések is bekövetkezhetnek, például a tudásstruktúra egy-egy fogalom mélyebb megértését kísérő átrendeződése során. [7]

Egy adott készség, képesség fejlettségének értékeléséhez fontos ismerni a készség, képesség összetevőit, ezek ismeretében lehetséges a komponensek működését vizsgáló tesztek kidolgozása. A teszteknek – figyelembe véve a fejlődés hosszabb idejét – általában szélesebb életkori tartományban való mérésre kell alkalmasnak lenniük.

2.2. A programozási készségek fejlettségének vizsgálatai

A programozást tanító pedagógusok tapasztalatai szerint a programozási készség kialakulása is időigényes. A lexikai elemek elsajátításától az algoritmizáláson, az algoritmusok formális megadásának és megértésének készségein át az önálló programtervezésig és kivitelezésig számos ismeret és készség megszerzése szükséges a sikeres programozáshoz, ezek rendszerré szerveződéséhez sok időre és gyakorlásra van szükség. [4] Empirikus vizsgálati eredmények ugyan nem ismertek, de az oktatói tapasztalatok azt mutatják, hogy a fejlődés kevéssé gyorsítható a programozás-oktatás intenzitásának növelésével. Az intenzív, rövid időszakon belül megvalósított oktatás elsősorban a tehetséges és motivált tanulók esetén eredményes. A többség számára azonban előnyösebb az időben elosztott, a fogalmak, összefüggések megértésére – nem feltétlenül programozással töltött – időt hagyó oktatás. [4]

A programozás oktatásának eredményességét, a programozás további tanulásának sikerességét alapvetően befolyásolják a programozáshoz fűződő motívumok (pl. a tanuló programozáshoz fűződő énképe). Ezekre a motívumokra az oktatás során alkalmazott módszerek és eszközök is hatással lehetnek [5]. Ugyancsak befolyásolhatják a képességek és a motívumok fejlődését a pedagógus által alkalmazott tanulásszervezési eljárások. [9]

A programozási készségek fejlettségének hazai empirikus vizsgálatai általában egy-egy szűkebb területre irányultak (pl. [5] [10]). Emellett főként oktatás-módszertani tanulmányok és szakdolgozatok jelentek meg, amelyek egyes programozási témák közoktatási feldolgozási lehetőségeit mutatták be (pl. [1] [11] [12]).

A nemzetközi kutatásban a számítógépek iskolai elterjedésével egyidejűleg – az 1980-as évektől – megkezdődtek a programozással kapcsolatos vizsgálatok.

Ezek a korai kutatások részben arra keresték a választ, hogy a programozás oktatása milyen hatással van a tanulók kognitív fejlődésére. [13] A vizsgálatok a konstruktív pedagógia alapgondolatának megfelelően abból a feltételezésből indultak ki, hogy a programozást tanulók tapasztalati úton sokat tanulnak a problémamegoldási eljárásokról, emellett a formális eljárások, változók, transzformációk megértésével lehetőséget kapnak a tervezésre, a problémák részproblémákra bontásának elsajátítására. A hibakeresés, a mások algoritmusainak elemzése pedig növelheti a gondolkodás rugalmasságát. [14, idézi 13] A korai vizsgálatok nem mutattak egyértelmű kapcsolatot, közvetlen transzferhatást a programozás tanulása és a hagyományos problémamegoldó gondolkodás fejlődése között. [13] [15] Gyakran tapasztalták viszont a kutatók a szociális készségek és a tanulási motívumok kedvező változását a kezdő programozóknál. [16] [17]

A múlt század utolsó két évtizedének vizsgálatai másrészt azon készségek, képességek körének azonosítására irányultak, amelyek kapcsolatban állnak a programozási készségek fejlettségével, és amelyek megfelelő fejlettsége szükséges a programozási készségek eredményes elsajátításához. [13] [18] Elsősorban olyan általános képességek kapcsolatát tárták fel a programozási készségekkel, mint az általános matematikai képességek, a feldolgozási kapacitás (a program paramétereinek egyidejű áttekintése), az analógiás gondolkodás, a deduktív gondolkodás, az algoritmikus gondolkodás, az absztrakció képessége. [13] Ezen általános képességek kapcsolata a programozási készségekkel nem vitatható, azonban általában egymással is szoros kapcsolatban állnak, számos közvetített hatást is tartalmaznak.

Az ezredforduló környékén megkezdődött vizsgálatok egy jelentős része már kevésbé az előfeltételekre irányult, sokkal inkább azoknak az összetevőknek az azonosítására, amelyek a programozás ismeret- és készségrendszerét alkotják.

A tanulmányok egy része a szoftverfejlesztés gyakorló szakemberei szemszögéből vizsgálja a programozás készségeit, elsősorban arra helyezve a hangsúlyt, hogy a szakértő programozóknak (akik gyakran csoportban dolgoznak) milyen készségekkel kell rendelkezniük a hatékony fejlesztéshez. Ugyancsak jelentős azonban azoknak a tanulmányoknak a száma is, amelyek pszichológiai-neveléstudományi szempontból, elsősorban a kezdő programozók készségeit vizsgálják. [19] Más felosztásban, a tanulmányok nagy hányada a strukturált programozás készségeire helyezi a hangsúlyt, míg a további tanulmányok az objektumorientált programozáshoz szükséges készségeket (pl. [20]) illetve a két megközelítés közös és eltérő

készségeit vizsgálják. Jelen tanulmányban elsősorban a strukturált programozás pszichológiai-neveléstudományi vizsgálatainak az eredményeit tekintjük át.

A vizsgálatok – fenti értelemben szűkített köre – a szükséges ismeretek, absztrakt fogalmak felosztását alapvetően hasonló struktúrában adják meg. Ezek leggyakoribb elemei: (1) változótípusok, egyszerű változók és azok tömbjei; (2) a strukturált programnyelvi elemek: ciklusok, elágazások; (3) sztenderd input/output-kezelés; (4) strukturált adattípusok és fájlok; (5) eljárások és/vagy függvények; (6) rekurzió; (7) absztrakt adattípusok; (8) kivételkezelés; (9) függvénykönyvtárak. [21]

Az elsajátítandó készségek köre jóval nagyobb változatosságot mutat a szakirodalomban. *Lahtinen* és mtsai [21] a következő (nem feltétlenül készség jellegű) tevékenységek bonyolultságát értékeltették nagymintás, programozást tanuló egyetemistákat és oktatóikat érintő kérdőíves vizsgálatukban: (1) a fejlesztőkörnyezet használata; (2) a programozási struktúrák megértése; (3) a programnyelv szintaktikájának megértése; (4) a problémához illeszkedő program megtervezése; (5) a tevékenység felosztása eljárásokra; (6) hibakeresés a saját programban. Mind a hallgatók, mind az oktatók az utolsó három tevékenységet érezték nagyobb nehézséget okozónak.

Más tanulmányok felvetik a kész algoritmusok, programok megértésének [19] [22] és az azokban lévő hibák megkeresése készségeinek fontosságát is. Az algoritmusok megértésének fontosságát és nehézségeit jelzi, hogy a kutatók a digitális tanulási környezetek lehetőségeit felhasználva különböző vizualizációs technikák hatékonyságát is vizsgálják a megértési folyamatban. [23]

A program tényleges kódolását megelőző folyamathoz szükséges, számos feltárt összetevő rendszerezését több kutató is egységes modellbe kívánta foglalni. A *Computational thinking* (CT, szó szerinti fordításban kiszámítási gondolkodás, de talán kifejezőbb a *programozói gondolkodás*) fogalmát *Wing* vezette be. [24] Az ő megfogalmazásában a CT olyan kognitív folyamat, amely a problémák formalizálására és a megoldásuk olyan megjelenítésére irányul, amely hatékonyan implementálható valamilyen információ-feldolgozó szoftverben. A modell az absztrakció egymásra épülő szintjeit feltételezi, a „jó gondolkodót” pedig az jellemzi, hogy képes a megfelelő absztrakciók kiválasztására és egyidejűleg több absztrakciós szinten a megfelelő műveletek elvégzésére. A CT absztrakciós osztályai közé sorolja többek között a különböző algoritmusokat, az adatstruktúrákat, a vezérlési szerkezeteket, a kommunikációs megoldásokat, a programnyelveket, a logikai műveleteket, a heurisztikát, az architektúra elemeit. [25] A CT fogalomrendszerére épülő oktatási kísérletekről és módszertani megoldásokról több ezer tanulmány jelent meg az utóbbi évtizedben. A módszertani tanulmányok többsége az egyes absztrakciós szintek különböző fejlesztői környezetben való tanítási lehetőségeit mutatja be. [26]

Az ismeretek és készségek közvetlen körének feltárásán túl számos vizsgálat kiterjedt arra is, hogy milyen tanulási szituációban, milyen tananyagokkal [21] és tanítási módszerekkel (pl. [27] [28]) bizonyul leghatékonyabbnak az oktatás. Jelen tanulmánynak nem célja a fejlesztési lehetőségek áttekintése, célunk – a bevezetővel összhangban – az emelt szintű érettségi vizsgán elvárt programozási ismeretek és készségek, valamint az azok fejlettségét meghatározó legfontosabb tényezők feltárása.

2.3. A programozás ismeret- és készségrendszerének feltételezett elemei

Az előző fejezetben láthattuk, hogy a szakirodalomban a programozáshoz szükséges ismereteket és készségeket többféleképpen közelítik meg a kutatók. A hazai közoktatási dokumentumokhoz és pedagógiai értékelési hagyományokhoz illeszkedően az emelt szintű érettségihez szükséges ismereteket és készségeket az alábbi csoportosításban kívánjuk vizsgálni:

A) Algoritmusok: (1) a megjelenítés lexikai elemei egy vagy két algoritmus-leíró eszközben, érvényes és érvénytelen jelölések; (2) egyszerű algoritmusok (pl. egy- és kétciklusos programozási tételek) megértése; (3) algoritmus alkotása egyszerű problémákhoz; (4) algoritmus helyességének vizsgálata, hibakeresés.

B) Egy programozási nyelv elemei: (1) egyszerű változótípusok ismerete, változók deklarálásának szintaxisa; (2) sztenderd input-output használatának szintaxisa; (3) egydimenziós tömbök deklarálása, tömbelemre való hivatkozás szintaxisa; (4) szám- és logikai típusú kifejezések szintaxisa; (5) vezérlési szerkezetek szintaxisa; (6) függvények létrehozásának és meghívásának

szintaxisa; (7) adatstruktúrák szintaxisa; (8) fájlkezelő függvények szintaxisa; (9) függvény-könyvtárak alkalmazásának szintaxisa.

C) A fejlesztő-környezet technikai kezelése: (1) a szerkesztőprogram funkcióinak alkalmazása; (2) program fordításának és futtatásának készsége; (3) gyakori hibaüzenetek jelentésének ismerete; (4) a nyomkövetés eszközeinek alkalmazása.

D) Algoritmusok implementálása (kódolása): (1) algoritmus és programkód megfeleltetésének készsége; (2) kódolás meglévő algoritmus alapján; (3) a tevékenység függvényekre bontása; (4) konkrét programhibák keresésének készségei.

A fentiekben vázolt tartalmi elemek részletes leírását – így például az egyes tudáselemek elvárt alkalmazási szintjének pontos meghatározását, az egyes készségek részkészségeinek rendszerezését – az előmérést bemutató empirikus tanulmányban adjuk majd meg. A diagnosztikus célú mérésekhez [29] ezen összetevők pontos meghatározása szükséges.

2.4. A programozás sikerességének előfeltételei

Az előző fejezetben leírt programozói ismeretek és készségek – a programozáshoz fűződő motívumokkal együtt – közvetlenül kapcsolódnak az emelt szintű érettségi vizsgán elvárt programozói tudáshoz. Nem minden elem kerül számonkérésre, így például az algoritmizálás készségei vagy a hibakeresés, nyomkövetés eszközeinek alkalmazása nem jelenik meg direkt módon a feladatokban, azonban fejlettségük közvetlen hatást gyakorol a vizsgázó eredményességére.

A sikertelen feladatmegoldást azonban feltehetően nem csak a fenti tudáselemek hiányosságai okozhatják. Számos általános képesség nem megfelelő szintje okozhatja azt, hogy a tanuló nem jut el a sikeres feladatmegoldásig. *Kelemen* kritériumrendszeréhez hasonlóan [30] feltételezhetjük, hogy az alábbi készségek, képességek esetén létezik ezeknek egy olyan fejlettségi szintje, amelynek (a) elérése nélkül a programozói készségek nem tudnak megfelelően funkcionálni; (b) teljesítése felett a kritérium már nem mérvadó, a szintet elérők között egyaránt találhatók a programozási teszten gyengén, közepesen és jól teljesítők is. [30]

Tervezett vizsgálatainkkal igazolni kívánjuk, hogy a fenti értelemben szűrőként működnek a következő készségek: (1) szóolvasás, szakszókincs; (2) szövegértés; (3) a probléma algoritmizálása, modellezése; (4) a kódolás készségei. Azt is feltételezzük továbbá, hogy a kritériumrendszer egyes szűrői lineárisan egymásra épülnek, azaz például azok, akik a szövegértés kritériumszintjét nem érik el, a probléma modellezését sem tudják elvégezni, hiszen nem férnek hozzá a szövegben rejlő információkhoz. [30]

3. Összegzés: a tervezett mérések rendszere

A következő időszakban lefolytatandó vizsgálataink egyrészt a 2.3. bemutatott ismeretek és készségek diagnosztikus értékelésére irányulnak. Ehhez szükséges a felsorolt tudáselemek és alkalmazási követelményeik pontos meghatározása, a megfelelő diagnosztikus mérőeszközök elkészítése. A mérőeszközök a megadott A)–D) területeket nem elkülönülten kívánják mérni. Ahol az alkalmazási kritérium lehetővé teszi, ott elektronikus tesztfeladatokat alkalmazunk. Szükséges lesz azonban ezek kiegészítése hagyományos, papír alapú tesztfeladatokkal illetve tényleges programozási feladattal. Az egyéni különbségek feltárására néhány tudáselem – pl. az A/2, A/4, C/3, D/2 elemek – fejlettségének értékelését szemmozgás-vizsgálattal tervezzük kiegészíteni.

A programozásra vonatkozó kritériumrendszer-modell helyességét a 3. és 4. szűrő esetén az előző bekezdésben leírt mérőeszközök megfelelő elemeivel végezzük. Az 1. 2. szűrő esetén pedig részben az országos kompetenciamérés szóolvasás-tesztjét, részben saját fejlesztésű teszteket alkalmazunk. Saját tesztet fejlesztünk az érettségihez szükséges szakszókincs vizsgálatára, emellett egy általános tartalmú és egy informatikai szövegre, valamint diagramok olvasására épülő – lehetőség szerint elektronikus – szövegértés-tesztet készítünk.

A vizsgálatok tervezett mintája két csoportból áll. Az egyik csoportba két gimnázium informatika tagozatos illetve emelt szintű érettségire készülő (fakultációs) tanulók tartoznak, összesen körülbelül 60 fő. A részminta 16-19 év közötti tanulókból áll, akik mindegyike tanulta

azokat a tudáselemeket, amelyek az emelt szintű érettségin elvártak. A másik csoportba olyan főiskolás, mérnökinformatikus hallgatók tartoznak majd – szintén körülbelül 60 fő –, akik az érettségin elvárt programozói tudáselemeket a főiskolai programozás-kurzusokban már tanulták (lehetséges, hogy közülük néhányan középiskolában is tanultak programozni).

A mérések előkészítése, a mérőeszközök elkészítése és a próbamérések megvalósítása a 2.3. fejezetben leírt tudáselemek és a 2.4. fejezet szűrői esetén mintegy hat-nyolc hónapot igényelnek. Ezt követően kerülhet sor a nagymintás mérésekre és az egyéni (szemmozgás-) vizsgálatokra, valamint a mérések körének bővítésére, elsősorban a programozásra irányuló motívumok feltárásával. Empirikus vizsgálataink remélhetőleg pontosabbá teszik a sikeres programozáshoz szükséges ismeretekre, készségekre, motívumokra vonatkozó tudásunkat.

Irodalomjegyzék

[1] Kiss Róbert: Robtika a közoktatásban. Gradus, Vol. 2. No. 2. (2015) pp. 81-93.
[2] CodeWeek: Why learn to code? [Online]. Elérhető: http://codeweek.eu [Megtekintés: 30-Aug-2016].
[3] Kiss Gábor: A magyar és a nemzetközi informatikaoktatás összehasonlítása. Doktori értekezés. Debreceni Egyetem, Matematika- és Számítástudományok Doktori Iskola (2012). [Online] Elérhető: https://dea.lib.unideb.hu/dea/bitstream/handle/2437/131732/tezisek3-t.pdf?sequence=9&isAllowed=y [Megtekintés: 28-Aug-2016]
[4] Kiss Róbert és Pásztor Attila: Programozható robotok felhasználása a programozás oktatásban. Előadás. Szakmai Nap, Kecskeméti Főiskola GAMF Kar, Kecskemét (2006).
[5] Pásztor,A., Pap-Szigeti, R. & Török, E.: Effects of Using Model Robots in the Education of Programming. Informatics in Education, Vol. 9, No. 1. (2010) pp. 1–8.
[6] Nagy József: XXI. század és nevelés. Osiris Kiadó, Budapest (2000).
[7] Csapó Benő: A képességek fejlődése és iskolai fejlesztése. Akadémiai Kiadó, Budapest (2003).
[8] Molnár Gyöngyvér és Csapó Benő: A képességek fejlődésének logisztikus modellje. Iskolakultúra, No. 2. (2003) pp. 57-69.
[9] Pap-Szigeti Róbert: Kooperatív módszerek alkalmazása a felsőoktatásban. Iskolakultúra, No. 1. (2007) pp. 56-66.
[10] Johanyák, Zs. Cs., Pap-Szigeti, R. & Alvarez, G. R. P.: Analyzing students' programming failures. A GAMF Közleményei, No. 22. (2008) pp. 115-120.
[11] Törley Gábor: Algoritmus vizualizáció a tanítási gyakorlatban. (2013) [Online] Elérhető: http://real.mtak.hu/31528/1/TG_alg_viz_tan_gyak.pdf
[12] Horváth László András: A Java nyelv tanítása középiskolában. Szakdolgozat, Debreceni Egyetem (2009).
[13] Pea, R. D. & Kurland, D. M.: On the cognitive effects of learning computer programming. New Ideas in Psychology, Vol. 2. No. 2. (1984) pp. 137-168.
[14] Feurzeig, W., Horwitz, P. & Nickerson, R. S.: Microcomputers in education (Report No. 4798). Prepared for: Department of Health, Education, and Welfare; National Institute of Education; and Ministry for the Development of Human Intelligence, Republic of Venezuela. Bolt Beranek & Newman, Cambridge, MA, October (1981).
[15] Lehrer, R., Guckengerg. T. & Sancilio, L.: Influences on LOGO on children's intellectual development. In: Mayer, R. E. (ed.): Teaching and learning computer programming: Multiple research perspectives. Lawrence Erlbaum Associates, Inc., Mahwah, New Jersey, USA (1988). pp. 75-110.
[16] Sheingold, K., Kane, J., Enderweit, M. & Billings, K.: Study of issues related to the implementation of computer technology in schools. Final Report, National Institute of Education (1981).
[17] Webb, N. M. & Lewis, S.: The social context of learning computer programming. In: Mayer, R. E. (ed.): Teaching and learning computer programming: Multiple research perspectives. Lawrence Erlbaum Associates, Inc., Mahwah, New Jersey, USA (1988). pp. 179-206.
[18] Kurland, D. M., Pea, R. D., Clement, C. & Mawby, R.: The Study of the Development of Programming Ability and Thinking Skills in High School Students. In: Soloway, E. & Spohrer, J. C. (eds.): Studying the Novice Programmer. Lawrence Erlbaum Associates, Inc., Hillsdale, New Jersey, USA (1981). pp. 83-112.
[19] Robins, A., Rountree, J. & Rountree, N.: Learning and Teaching Programming: A Review and Discussion. Computer Science Education, Vol. 13. No. 2. (2003). pp. 137-172.
[20] Milne, I. & Rowe, G.: Difficulties in Learning and Teaching Programming – Views of Students and Tutors. Education and Information Technologies, Vol. 7. No. 1. (2002). pp. 55–66.
[21] Lahtinen, E., Ala-Mutka, K. & Järvinen, H.: A study of the difficulties of novice programmers. Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education ITiCSE '05. New York, NY, USA (2005). pp. 14-18.
[22] Hanzalova, P. & Hubalovsky, S.: Algorithm development and programming at elementary education in the Czech Republic. International Journal of Education and Information Technologies, Vol. 9. (2015) pp. 175-179.

[23] Lazaridis, V., Samaras, N. & Sifaleras, A.: An empirical study on factors influencing the effectiveness of algorithm visualization. Computer Applications in Engineering Education, Vol. 21. No. 3. (2013), pp. 410-420.
[24] Wing, J. M.: Computational Thinking. Communications of the ACM, Vol. 49. No. 3. (206), pp. 33-35.
[25] Wing, J. M.: Computational Thinking. OurCS Workshop, Carnegie Mellon University, Pittsburg PA, USA (2011).
[26] Brennan, K., & Resnick, M.: New frameworks for studying and assessing the development of computational thinking. In: Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada (2012). pp. 1-25.
[27] Vihavainen, A., Paksula, M. & Luukkainen, M.: Extreme apprenticeship method in teaching programming for beginners. Proceedings of the 42nd ACM technical symposium on Computer science education SIGCSE '11. New York, NY, USA (2011). pp. 93-98.
[28] Saeli, M., Perrenet, J., Jochems, W. M. G. & Zwaneveld, B.: Teaching Programming in Secondary School: A Pedagogical Content Knowledge Perspective. Informatics in Education, Vol. 10. No. 1. (2011) pp. 73–88.
[29] Vidákovich Tibor: Diagnosztikus pedagógiai értékelés. Akadémia Kiadó, Budapest (1990).
[30] Kelemen Rita: A matematikai szövegesfeladat-megoldó képesség vizsgálata többségi és tanulásban akadályozott 9-13 éves tanulók körében. PhD értekezés, Neveléstudományi Doktori Iskola, Szegedi Tudományegyetem, Szeged (2010).