**Bornemissza Csaba**
Debreceni Egyetem Információ Technológia Tanszék
*bornem@inf.unideb.hu*

# LEHETŐSÉGEK AZ AUTOMATIZÁLT PROGRAMOZÁSI FELADATELLENŐRZÉSRE WEBES PLATFORMON

## Bevezetés

A programozási nyelvek oktatása több részfeladatra bontható. A programozási nyelvekkel, programozási problémákkal kapcsolatos elméleti anyag átadására bevált módszerek vannak, és a szakirodalom is bőséges. Ami a gyakorlati oktatást illeti, az egyetemek, főiskolák nehezebb helyzetben vannak. A gyakorlatokon szerzett ismeretek számonkérése az egyik legnehezebb probléma: a papír formájú zárthelyi dolgozatok nehézkesen javíthatók, a hallgatók számára is nehezebb papíron programot írni, mintha számítógépeken tennék.

A programozás gyakoroltatására és a gyakorlati ismeretek számonkérésére elméletileg lehetne olyan automatizált módszereket használni, melyek jelentős emberi munkát takarítanának meg. A C programozási nyelv oktatása során sor került a Debreceni Egyetemen egy kísérleti jellegű, automatizált programhelyesség-ellenőrzésre. Célunk volt, hogy a hallgatóknak lehetőséget biztosítsunk egy ellenőrzött gyakorlásra, mely során az ellenőrzéseket nem ember, hanem programok végzik.

Ez a cikk részben ennek a kíséretnek az eredményeit értékeli, másrészt felvázolja egy olyan webes alkalmazás terveit, mely a Java nyelv gyakorlati oktatásában nyújtana különböző helyzetekben segítséget.

Az internet mint média egyre nagyobb szerepet kap az oktatásban. Egy olyan rendszer lehetőségeit elemezzük, amely nem csak közvetítő szerepet töltene be az oktató és a hallgató között, hanem aktív részvételével bizonyos részfeladatokat automatikusan ellát, ezzel gyorsítva a hallgató gyakorlási lehetőségeit, és segítséget nyújtva az oktató számára a számonkérésben, értékelésben.

## 1. Feladatok a programozás oktatásában

A programozási nyelvek oktatása alapvető feladat a programozó matematikus, programtervező matematikus, informatikatanár szakokon, és minden más olyan szakon, ahol a programozás készség szintű ismerete elengedhetetlen.

Az elmúlt évek tendenciája, hogy a hallgatói létszám a felsőoktatási intézményekben jelentősen megemelkedett, és további növekedés várható. Ezt a folyamatot követni, az oktatás színvonalát fenntartani, sőt emelni – miközben az egyetem oktatóinak létszáma nem biztos, hogy követni tudja arányában a hallgatói létszám növekedését – nagy kihívás.

A programozás oktatásában teremthetünk olyan helyzeteket, ahol az oktatókra háruló feladatok egy része automatizálható, és programok segítségével felgyorsíthatjuk a számonkérési folyamatot, támogathatjuk a hallgatók gyakorlási tevékenységét. Mindebben nagy szerepet játszik az internet rendelkezésre állása egy felsőoktatási intézmény mindennapi életében.

### *1.1. Elméleti oktatás*

A programozási nyelvek elméleti oktatása egyrészt az előadásokon elhangzottakat jelenti. Másrészt a hallgatók szakirodalomra támaszkodva bővíthetik ismereteiket. Szerencsére a magyar nyelvű szakirodalom bőségesnek mondható napjainkban: bármilyen programozási nyelvhez, mely „sikeresnek" bizonyult az ipari alkalmazásokban, több magyar nyelvű tankönyv található a könyvesboltokban.

Nem szabad figyelmen kívül hagyni az interneten megtalálható segédanyagokat: elektronikus kurzusok, példaprogramok, kérdések és válaszok találhatók, csak győzze az ember kiválogatni a neki megfelelőt a találatok tömegéből.

### *1.2. Gyakorlati oktatás*

A programozási nyelvek gyakorlati oktatása kritikus a leendő informatikusok oktatásában. Ez az a kurzus, melynek során a hallgatóknak készség szinten kell működő programokat írniuk az elméleti ismereteiket is alkalmazva. Egyszerre kell tudni alkalmazni a tanult vezérlési szerkezeteket, az adatszerkezeteket stb.

A gyakorlati oktatás legfontosabb feladata a programozás gyakoroltatása, a hallgatók segítése, hogy túllépjenek az esetleges megtorpanásokon. Esetenként felmerül az elméleti anyagok elmélyítése, pl. az elosztott adatszerkezetekkel kapcsolatban gyakran merülnek fel „ezt most hogyan csináljuk" jellegű kérdések.

Nem szabad figyelmen kívül hagyni azt sem, hogy a modern programozáshoz hozzátartoznak az integrált fejlesztői környezetek (IDE – Integrated Development Environment). Ezek használatára is meg kell tanítani a hallgatókat. A mai rendszerek, pl. a Java nyelvhez elérhető ingyenes fejlesztőrendszerek (Eclipse, NetBeans) is igen bonyolultak lehetnek egy elsőéves számára.

A gyakorlati ismeretek számonkérése nehéz feladat. Számítógépeken nehéz felügyelni, milyen segédeszközöket vesznek igénybe a hallgatók. A papíron benyújtott dolgozatok nem biztos, hogy helyesen tükrözik a programozás tudást: papírra nehezebb is programot írni, a helyességét ellenőrizni is fárasztóbb feladat.

## 2. Automatizálva vagy manuálisan?

A gyakorlati számonkérés során több dologban nagyon hatékonyan lehet szoftveres támogatást igénybe venni. Ha a programokat elektronikus formában adják le a hallgatók, a programok helyességét eldöntheti a fordítóprogram, és a próbafuttatások a tesztesetekre. A manuális ellenőrzés során viszont különbséget tudunk tenni kisebb elgépelések és elvi hibák között: erre egy fordítóprogram nem képes.

A C programozási nyelv oktatása során tettünk egy konkrét módszertani kísérletet arra, hogy a gyakoroltatásban és az eredmények kiértékelésében automatizált módszereket alkalmazzunk. A módszer a következő volt:

- A weboldalunkon publikáltuk a megoldandó feladatokat.
- A megoldott feladatokat e-mailben kellett a hallgatóknak beküldeni. (A forráskódot csatolt állományként kértük, a levél témája tartalmazta a hallgató azonosítóját és a feladat azonosítóját.)
- A levelekből automatikusan szoftver vette ki a forráskódokat, és hallgatók és feladatok szerint rendszerezve tárolta. (PCRMPOP3)
- A programok fordítását a PCRM szoftver végezte automatikusan.
- A feladatok ellenőrzéséhez szükséges teszteseteket manuálisan állítottuk elő.
- A tesztesetekre ellenőrző futtatásokat és az eredmények kiértékelését a PCRM szoftver végezte automatikusan.
- Az eredményeket a weboldalon publikáltuk.

## 3. Tapasztalatok

Egy szemeszteren keresztül öt alkalommal adtunk ki 4–7 feladatot tartalmazó példasorokat, melyeket a fent leírtak szerint értékeltünk ki. Pozitív és negatív tapasztalatok egyaránt voltak. A pozitívumot leginkább abban láttuk, hogy a hallgatók folyamatosan rá voltak kényszerítve, hogy C programokat írjanak, és ennek elég nagy százalékban eleget is tettek. A fél év végi eredmények érezhetően jobbak lettek, mint az azt megelőző években, de mivel ezt több más tényező is befolyásolhatja, ezért itt nem tartom relevánsnak, hogy számadatokat említsek.

A negatívumok között szerepel a plágiumellenőrzés eredménye. Sajnos egyértelműen kimutathatók voltak olyan hallgatócsoportok, melynek tagjai ugyanazokat a forráskódokat küldték be rendszeresen. Egy JPLAG nevű szoftvercsomaggal végeztünk ilyen jellegű vizsgálatokat. A szoftver nem szöveg alapú összehasonlítást végez, hanem a C program topológiai, felépítési elemzésével állapítja meg az egyezőségeket.

Az alábbi negatívumok viszont olyan technikai jellegű problémák voltak, amelyek épp azt az előnyt csökkentették, miszerint bizonyos részfeladatok automatikusan hajtódnának végre.

- Az e-mailben beküldött forrásfájlok kiolvasásával a következő problémák voltak: eleinte nem kötöttük ki, hogy milyen levelezőprogrammal lehet az állományokat beküldeni. Ez azt eredményezte, hogy bizonyos webes levelezőprogramok extra csatolt állományokat, reklámokat, képeket tettek bele a levélbe, amik megzavarták a PCRMPOP3 szoftver működését, lehetetlenné téve a csatolt C forráskódok automatikus olvasását. Manuálisan kellett sok levélből kinyerni a csatolt forráskódot, hogy tovább tudjunk lépni az értékeléssel. Ezt a problémát úgy kerültük meg, hogy az egyetem hálózatán működő, minden hallgató által elérhető levelezőrendszerre szűkítettük le a használható levelezőprogramok halmazát, így a későbbi fordulókban ilyen probléma már nem jelentkezett.
- Néhány hallgató az ajánlás ellenére nem szabványos (ANSI) könyvtárakat használt a programjában, és ezzel fordítási hibákat okozott. A hallgatók által és az általunk használt C fordítók közötti különbségből is eredt néhány olyan fordítási

hiba, amely nem egyértelműen a hallgatók által elkövetett elvi hibának tekinthető, de ezeket megint csak manuálisan kellett értékelni.

– Bizonyos C programozási feladatoknak az volt a lényege, hogy egy bizonyos nyelvi eszköz használatával oldja meg a hallgató a feladatot. Példa erre, hogy eljárást vagy függvényt használjon, ill. rekurziót vagy ciklust. Ennek az ellenőrzése C nyelvben igen nehézkes. Reguláris kifejezések segítségével lehet bizonyos függvénydefiníciókat keresni, vagy bizonyos struktúrák, nevek meglétét ellenőrizni, de ezek megírása is igen nehézkes, tekintettel a C nyelv rugalmasságára. Erre a problémára nem is sikerült elfogadható megoldást találni. Reguláris kifejezéseket használó szűrőprogramokat használtunk. Ezek nem is voltak tökéletesek, ráadásul ismét sok ráfordítást igényeltek az így végzett ellenőrzések.

A fenti problémák elemzéséből arra a következtetésre jutottunk, hogy ezt a módszert nem lehet hatékonyan alkalmazni. A menet közben felmerült technikai problémák sokkal több emberi beavatkozást, ráfordított időt követeltek meg, mint számítottunk rá.

A továbbiakban egy olyan tervet vázolok, amely kiküszöböli a fent felsorolt, gyakorlatban szerzett negatív tapasztalatokat, problémákat, és lehetőséget nyújt a programozás oktatásának hatékony támogatására.

## 4. Automatikus ellenőrzések webes támogatással

A fent felsorolt problémák miatt a korábbi kísérlethez képest több szempontból másképpen kell a rendszer felépítését elkezdeni:

– E-mail helyett más, szabványos, de elterjedt módszerre van szükségünk, mellyel a forrásállományokat tudják az érintettek beküldeni. Kézenfekvő megoldásnak tűnik a webes űrlap, mellyel böngésző segítségével lehet csatolt állományokat feltölteni a szerverünkre. Az e-mail tárgymezője tartalmazta a hallgató azonosítását és a feladat kódját, ezeket az információkat szintén egyszerűen meg lehet adni azon a webes űrlapon, melyen az állományt feltöltjük.
– A C fordítók közötti különbségek, bizonyos beépített nyelvi támogatások hiánya arra vezetett, hogy a Java nyelv legyen az első nyelv, amire ezt a webes rendszert felépítjük. A Java nyelv előnyeiről később részletesebben kitérünk.
– A PCRM program, a levelek feldolgozása, az eredmények feldolgozása és publikálása folyamatból hiányzott az integráció, túl sok helyen kellett manuálisan beavatkozni. Ez arra a következtetésre vezetett, hogy egyetlen webes alkalmazásba integráljuk a rendszer alapvető funkcióit:
  - a feladatok publikálását,
  - a forráskódok begyűjtését,
  - a programok fordítását, futtatását, eredmények értékelését,
  - az eredmények publikálását,
  - a hibák minél részletesebb indoklását, pl. a program egy tesztesetre milyen hibás kimenetet generált,
  - a helyes kimenet megmutatását.

Az alapötlet tehát a fenti web-alkalmazás elkészítése. Ez folyamatban is van, a rendszer alapos megtervezése és implementálása azonban időigényes folyamat.

## 5. Az internet nyújtotta lehetőségek

Ebben a fejezetben azokat az előnyöket fejtem ki, amelyeket az internet, mint média alkalmazása hozhat a programozás oktatását támogató szoftver alkalmazásakor.

Az internet széles körben elérhető a hallgatók számára. Van internet hozzáférés a kollégiumokban, a géptermekben, ahol a gyakorlatokat tartjuk. Egy webes alkalmazás, melyen a hallgatók a gyakorló feladatokat elérhetik, és a tudásukat próbára is tehetik, komoly előnyt jelenthet az eddigi módszerekhez képest.

Erre egy példa pl. a „Google Jam", egy internetes programozói verseny, ami világszerte nyújt lehetőséget a programozási tudásukat próbára tenni vágyóknak, hogy összemérhessék tehetségüket a világ többi informatikusával. Ennek a webes versenynek alapvetően más a megközelítése, mint a jelen cikkben tárgyalt rendszeré, de az elosztottsága szemlélteti, hogy függetlenül attól, hogy hol ül az ember egy számítógép elé, ha internet-hozzáférése van, elérheti a rendszert.

Másik előny, hogy – miután az oktató előkészítette a feladatokat, a teszteseteket a rendszer számára – a feladatok bármikor elérhetőek lesznek, állandóan rendelkezésre álló gyakorlási lehetőséget biztosítva a hallgatóknak.

Egy ilyen rendszer, ami képes visszajelezni a hallgatóknak, sokkal inspirálóbb lehet, mint egy egyszerű példatár, amelyben benne vannak a feladatok, és az eredmények. Kihívás lehet abban, hogy a hallgatónak meg kell győznie az ellenőrző rutinokat, hogy a program helyes. Megfelelően összeválogatott tesztesetekkel a folyamatot érdekesebbé is lehet tenni: előrevesszük azokat a bemeneteket, amelyek szokványosak, és későbbre hagyjuk azokat, melyek a szélsőséges esetek, ill. hibás bemenetek. Így lehet hogy egy program már működik, de a hibás bemeneteket nem ellenőrzi: ekkor már kap pozitív visszajelzést is a hallgató, de a hibáit is érzékeli a rendszer.

A pozitív visszajelzést, a sikerélmény hatását nem szabad lebecsülni: sok elsőéves hagyja ott az egyetemet, mert úgy érzi, hogy folyamatosan kudarcok érik, és nem érzi magát alkalmasnak a pályára.

Az internetes elérés előnyeit nem csak a hallgatók élvezik ebben a rendszerben. Az oktató is könnyebben hozzáférhet az egyes hallgatók által beadott feladatokhoz, egyszerű webes lekérdező űrlapok segítségével. Egyrészt, ha a hallgatónak kérdése van valamelyik programjával kapcsolatban, könnyen elérhető, nem kell könyvtárakban keresgetni. Másrészt, ha csak nézegetni akarja az oktató a beküldött programokat, akkor is könnyebb így elérnie az állományokat.

## 6. A Java nyelv előnyei

A Java nyelv több szempontból előnyös választás. A platformfüggetlensége lehetővé teszi, hogy több típusú szerveren tudjuk majd futtatni a hallgatók által beküldött kódokat. Ezzel a majdani szoftver alkalmazási lehetőségeit nem kell szűkítenünk.

A Java nyelv másik komoly előnye a reflection framework. A *java.lang.reflection* csomagban található eszközökkel olyan programokat írhatunk, melyek a

leadott szoftvereket alkotó Java osztályok struktúráját vizsgálhatják. Ez az eszköz nagyon hiányzott a C nyelvből, ennek segítségével megnézhetjük, hogy pl. a hallgató a feladat szövegében megadott függvénydefiníciót használta-e a probléma megoldására.

A tervezett webes alkalmazást magát is érdemes Java nyelven fejleszteni. A modern fejlesztőeszközök, és a J2EE (Java 2 Enterprise Edition) webes fejlesztést segítő komponenseit használva (pl. Servlet, JSP) viszonylag gyorsan lehet majd az alkalmazást implementálni.

## 7. A .NET (dot NET) keretrendszer lehetőségei

A .NET keretrendszer a Java (J2EE) világ sok tulajdonságát örökölte. Megvan benne például a reflection csomag is (System.Reflection), mely segítségével elemezhetjük a beküldött programokat. A webes alkalmazások támogatása is komoly hangsúlyt kapott a .NET keretrendszer kidolgozásánál, ez sem jelent tehát akadályt, ha a programozás oktatását támogató webes rendszerünket .NET alá akarnánk megvalósítani.

A .NET keretrendszer kezd tért hódítani az alkalmazásfejlesztés területén, miközben több programozási nyelvet is támogat (pl. Visual Basic, Csharp, C++, Jsharp), melyeket egy közös virtuális kódra (IL – Internal Language) fordít.

A .NET nyelvek oktatása bizonyos intézményekben már benne is van az órarendben, és bevezetését egyre több helyen tervezik.

Mivel a Java és a .NET egyaránt rendelkezik minden olyan előnyös tulajdonsággal, amire szükség van a fent leírt oktatás-támogató rendszer kifejlesztésében, mindkét technológia komoly szerepet tölt be az alkalmazott informatikában, érdemesnek tűnik mind a két rendszerben párhuzamosan kifejleszteni az alkalmazásunkat. Ez nem jelent annyi többletmunkát, mint gondolnánk. Az adatbázis alapokat mindkét rendszer tudná használni, a rendszerterveket egészen az objektum szintű tervezésig csak egyszer kellene elkészíteni, és az implementáció nagy része is egyszerű feladat, hiszen Java nyelvből CSharp-ba, vagy fordítva nem nehéz a programot átírni.

## 8. Lehetséges alkalmazási területek

Több területen lehetne ezt a program-ellenőrző webes rendszert használni. Ezek közül néhányat már említettem és álljanak itt listaszerűen is:

- Programozás gyakorlatokon a gyakorló feladatok publikálására, az órai munka segítésére és értékelésére.
- Programozás gyakorlatok számonkérésére. Ehhez a programnak támogatnia kell az esetleges időhatárokat.
- Programozási versenyek lebonyolítására.
- Távoktatási, e-learning programok megvalósítására.

## 9. Kockázatok

Az automatizált ellenőrzésnek mindig vannak hátulütői is. Hiba és hiba között nagy különbségek lehetnek, és ezt egy fordítóprogram, vagy formai ellenőrző algoritmus nem tudja olyan alapossággal vizsgálni, mint az az oktató, aki konkrét szempontok szerint kéri számon az anyagot.

Egy elgépelés, és a program nincs lefordítva, egy függvényfejléc elírása, és nem felel meg a formai követelményeknek a beadott feladat.

Ezzel szemben előfordulhat, hogy a program működik, de a Java nyelv formai követelményeinek nem felel meg, programozási elveket sért a program felépítése, strukturálatlan a megoldás. Ezek esetleg átcsúszhatnak az automatikus ellenőrzéseken, de oktatási szempontból súlyosabb hibák, mint a fent említettek.

Az emberi tényező semmiképp nem hagyható ki a számonkérési folyamatból.

Ha az oktató úgy használja a rendszert, hogy folyamatosan ellenőrzi a beadott feladatokat, és követi a hallgatók fejlődését, ahogy haladnak az anyag elsajátításában, akkor nem csak munkát vesz le a válláról a rendszer, hanem a hallgatók tudásának tényleges értékelésében is segíthet.

Például a típushibák felismerése nem is történik meg, ha az oktató nem figyeli aktívan a hallgatók munkáját.

Aki arra használna egy ilyen rendszert, hogy teljesen rábízza a hallgatók értékelését, végül akár rosszabb hatékonysággal zárhatja a szemesztert, mintha anélkül dolgozott volna.

## 10. Összegzés

A programozási feladatok automatizált ellenőrzése a programozás oktatásában igen komoly segítséget jelenthet mind az oktatónak, mind a hallgatónak.

Az internet nyújtotta lehetőségek kihasználása egy modernebb, több visszajelzési lehetőséget, motivációt nyújtó környezetet jelenthet az oktatásban.

A cikkben felvázolt rendszer megvalósítása folyamatban van mind Java, mind .NET környezetben. További értékelése legalább egy szemeszteren keresztül történő alkalmazása után lesz lehetséges.

### Referenciák

[1] A PCRM szoftverről: http://www.frenzy.hu/?p=pcrm&l=h
[2] A Jplag szoftver plágium kereső: http://www.ipd.uka.de:2222/
[3] Egyéb plágiumkeresők:
http://www.libraries.psu.edu/mckeesport/mkplagiarism2.html
[4] .NET Keretrendszer technológiai áttekintés:
http://msdn.microsoft.com/netframework/technologyinfo/overview/
[5] J2EE áttekintés: http://java.sun.com/j2ee/overview.html