Sipos Marianna
Budapesti Műszaki Főiskola, Neumann János Informatikai Főiskolai Kar
sipos.marianna@nik.bmf.hu

A PROGRAMOZÁSOKTATÁS MEGÚJULÁSA A VISUAL STUDIO .NET KÍNÁLTA LEHETŐSÉGEKKEL

„A diákok többnyire arra emlékeznek a legjobban, amit először tanulnak meg.
Ezért szervezzük a fontossági sorrendnek megfelelően a tananyagot, és keressük a módját, hogyan taníthatjuk a legfontosabb fogalmakat minél előbb!"

Első Fecske Pedagógiai Gyakorlat
Joseph Bergin (1998)

Az informatika az oktatásban

Kezdetben az informatika használatát gyakorlatilag a programozás jelentette. A hardver és szoftver fejlődésével a felhasználási területek is bővültek és egyre inkább felhasználóbarát szoftverek kerültek forgalomba. Az informatikai eszközök használata kezdetben szakemberek kiváltsága volt, ma általánosan elterjedt gyakorlat. Ma már az informatikai eszközök használata nem igényel programozói ismereteket.

Az informatikaoktatásban a programozást kiszorítják a felhasználói ismeretek. Pedig a programozás az informatikában ugyanolyan fontosságú, mint a matematika a mérnöki tudományok területén. Ismerete nélkül nem érthetjük meg megfelelő mélységben a programok működését, nem tudhatjuk milyen problémák megoldására alkalmasak a szoftverek. Ahhoz, hogy egy irodai alkalmazott tudja, hogy az akták digitális tárolása lehetővé teszi a kis helyen tárolást, a gyors megtalálást, a könnyű módosítást, részek áthelyezését egyik dokumentumból a másikba, és könnyű segítségével szép kivitelű munkát letenni az asztalra, ismernie kell a szövegszerkesztők szolgáltatásait. Ahhoz, hogy tudja, hogy az adatainak feldolgozását segíti a táblázatkezelés, vagy még inkább az adatbázis-kezelés, ismernie kell e szoftverek szolgáltatásait. Ha pedig az a kérdés, hogy milyen problémákat lehetne megoldani, vagy milyen tevékenység színvonalát lehetne emelni egy testre szabott munkahelyi szoftverrel, ahhoz a programozás lehetőségeit kell ismernie.

Mindannyian jártunk középiskolába és érettségiztünk olyan tantárgyból, amit aztán munkánk során nem használtunk. De bár nem vállalnánk szakmai környezetben a véleménynyilvánítást, mégis értjük, ha valaki az ízeltlábúak étkezési szokásairól, vagy az egyiptomi fáraókról beszél. Ezzel szemben a nem programozó informatikusok többségének ha az osztály metódusának protected láthatóságáról beszélünk fogalma sincs mire gondolunk.

Pedig a programozásoktatás a nevelés és a gondolkodás fejlesztése terén is hasznot hajtana. Mai oktatási rendszerünk legnagyobb hiányossága, hogy csak kis mértékben teszi lehetővé az önálló gondolkodást az egyéni képességek kibontakoztatását. Ezt igazolják a nemzetközi felmérések.[1] A programozás oktatása az önálló problémamegoldáson kívül logikus gondolkodásra is nevel és az informatikai eszközök használatának egy magasabb szintjét is kifejleszti. Ezért én fontosnak tartanám a programozás valamilyen szintű oktatását már az általános iskolában, de legkésőbb a középfokú oktatásban és elengedhetetlennek tartom az informatikus képzésben.

Programozásoktatás történelmi sorrendben

Az első programozási nyelv létrejötte a Wikipedia digitális enciklopédia[2] szerint 1945-ben készült el Plankalkül[3] néven és egészen 1952-ig, az A-0 megjelenéséig az egyedüli programozási nyelv volt. A programozás tehát kevesebb mint 100 éves tudomány. Fiatal tudományként nagyon dinamikusan fejlődik. Évente több új programozási nyelv készül. 2003-ban például 5 új programozási nyelvet jegyez fel az enciklopédia, de 1993-ban 10 új nyelv került a piacra.[4] A fejlődés hihetetlenül dinamikus.

A második generációs microcomputer az úgynevezett home computer felhasználóbarát felülettel az 1970-es évek végén a 80-as évek elején jelent meg. Az IBM PC pedig MS-DOS operációs rendszerrel 1981-ben (*1. ábra*).

1. ábra: Az IBM PC zöld monokróm monitorral (5150) és billentyűzettel MS-DOS 5.0-t futtat

[1] OECD PISA Knowledge and Skills for Life – First Results from PISA 2000, Párizs, 2001.
[2] Wikipedia the free encyclopedia http://wikipedia.org
[3] Plankalkül http://en.wikipedia.org/wiki/Plankalk%FCl
[4] A programozási nyelvek időrendi sorrendben a Wikipedia honlapján. 2004. szeptember 24. http://en.wikipedia.org/wiki/Timeline_of_programming_languages

Ez tette lehetővé a számítógépek elterjedését, bevonását az oktatásba. Hazánkban a ma általánosan tanított kezdő programozási nyelv– a Pascal – 1971-ben született. A konzolon futó programok világa mintegy 30 éves. Az akkori számítógépeknek mindenki egyetértésével múzeumban van a helyük. A programozásoktatást mégis így kezdjük. A felsőoktatásban az alapozó képzésnél még mondhatjuk, hogy erős az alapozás. De a programozást nem szakmájukként tanulók talán nem kívánnának ilyen erős alapozást. Az általános és középiskolás diákok pedig még kevésbé.

Lányom, aki az informatika iránt nem nagyon mutatott érdeklődést, és szülői unszolásra – belátva szükségességét – különórán tette le az ECDL vizsgát a gimnáziumi évek alatt, harmadéves főiskolai hallgatóként vett részt első programozás óráján, így vélekedett: „Be kellett írni egy csomó érthetetlen dolgot a gépbe, és amikor készen voltunk, megjelent egy szörnyű fekete ablak, s a bal fölső sarkában a Hello! Tisztára mint egy horrorfilmben!" Ha egy fiatal felnőtt így gondolkozik, mit mondhat akkor a kisdiák?

Márpedig, ha a történelmi utat járjuk végig, az így kezdődött. Ha nem akarjuk a programozást a misztikus tudományok körébe száműzni, akkor nem feltétlenül a történelmi utat kell bejárni az oktatása során.

A pedagógiai kutatások eredményei

Mivel az informatika egy nagyon dinamikusan változó tudomány, fejlődésével az oktatási módszereknek is változniuk kell. Amíg a DOS parancsokat táblára lehet írni szépen sorban és otthon meg lehet tanulni a füzetből, addig a grafikus felületet projektorral kell bemutatni és nincs meghatározott sorrend, nincs rögzített anyag a bemutató végén. Tanulni is a gép előtt kell. Az oktatott újdonságok az oktatással szemben is más követelményeket támasztanak. Például projektmunkában megvalósított feladatok alkalmazását.

Az informatikaoktatás kutatása a tervezési minták analógiájára pedagógiai mintákat készít, hogy segítse a tanárokat az új módszerek megtalálásában. A cikk elején idézett 'Első fecske' (Early Bird) nevű pedagógiai gyakorlat is e kutatások eredményeként született. E gyakorlat szerint – és azt hiszem ezt mindenki saját tapasztalatából is tudja –, amit a tananyag elején tanulunk meg, az jobban rögzül mint a később tanultak. Ha kevésbé fontos fogalmakkal kezdünk azok rögzülnek. Ezért az említett gyakorlat azt javasolja, hogy a fontos fogalmakat vegyük a tananyag elejére, még akkor is, ha nem tudjuk teljes mélységében bemutatni őket. Ily módon lehetővé válik gyakori ismétlésük, miközben a tananyag más részeinél hivatkozunk rájuk. Ezzel elősegítjük, hogy mindenki megértse mik a fontos fogalmak, és a sokoldalú megvilágításnak köszönhetően tiszta kép alakul ki róluk a tanulókban. Időhiány miatt a tananyag végére kerülő anyagrészeken gyakran csak átszaladunk. Ezek kevésbé rögzülnek. Ha a korai tanítás lehetetlennek tűnik, tanítsuk a fontos anyagrészeket olyan korán, amilyen korán csak lehet.

Ez a pedagógiai gyakorlat a programozásoktatásban különösen fontos, mert a dinamikus fejlődésnek köszönhetően a fogalmak fontossága is átértékelődik. Az osztály szintű metódust a hagyományos objektumorientált programozásoktatásban (pl. C++) mint speciális lehetőséget tárgyaltuk, de a tiszta objektumorientált nyelvekben

(Java, C#) használata felértékelődött. Már az első órán a programot indító Main függvénynél beszélnünk kell a statikus metódusról. Más példa: a mai nem programozó informatikust elsősorban az érdekli, miért nem fut le az alkalmazás egy bizonyos dll nélkül, miért nem támogatja a Windows fejlesztő a MySql adatbázist, mikor a Unixos esküszik rá, vagy miért nem mutatja az egyik böngésző a másikban jól látható adatokat. Nem pedig az, hogy mi az összetett kifejezések kiértékelési elve. Ezzel természetesen nem akartam ez utóbbi fontosságát megkérdőjelezni, csupán azt állítom távol áll az informatikai rendszereket használók napi gyakorlatától.

Az 'Átstrukturálás' (restructure)[5] oktatásmódszertani minta szerint az új fogalmak jelentős része vagy gyorsan eltűnik, vagy alapfogalommá válik. A kezdetben a tananyag végén tanított új elveket – ha használatuk rendszeres gyakorlattá válik – egyre korábban kell tanítanunk ahhoz, hogy a rájuk épülő még újabb fogalmak is bekerülhessenek a tananyagba. Ez azonban nem csak a sorrend megcserélését jelenti, hisz a korábban tanított fogalomra építettük fel a későbbi tananyagot. Tehát az egész tematikát, a feladatokat át kell dogozni az alapvetővé váló fogalmak mielőbbi oktatása érdekében ahhoz, hogy naprakész ismereteket tudjunk átadni hallgatóinknak.

Új lehetőségek a programozásoktatásban

A programozás fejlődése egyre bonyolultabb szoftverek előállítását várja a fejlesztőtől. Az igények és a lehetőségek párhuzamosan fejlődnek. Vizuális szerkesztők, kódgenerátorok és osztálykönyvtárak támogatják a gyors szoftverkészítést. Az új fejlesztőeszközök lehetővé teszik, hogy egyszerűen és gyorsan készítsünk olyan alkalmazásokat, amelyek mögött nagyon bonyolult fogalmi rendszer és összetett kódstruktúra áll. A kérdés csak az, hogy e bonyolult szerkezet megismeréséhez a történelmi utat kell-e választanunk, vagy létezik egy érdekesebb, inspirálóbb megoldás is.

Világszerte próbálkoznak a programozást az objektumorientált filozófiával bevezetni. A konferenciákon már évek óta ennek tapasztalatait vitatják meg. Az új tiszta objektumorientált nyelvek nem is teszik lehetővé a probléma megkerülését, azonban sok helyütt az oktatás mégis *objektumorientált köntösbe bújtatott strukturált filozófiát jelent.*

Véleményem szerint a kérdést nem egyszerűsíthetjük le a 'mivel kezdjük' problémájára. Az új vizuális fejlesztőeszközök más tanulási szemléletet várnak el. Egy összetett fejlesztői környezetben látványosan és gyorsan működő kódot generálnak, mely mögött a funkciókat még nem ismeri az első órákon a hallgató. Meg kell tanulni úgy eligazodni e környezetben, hogy nem akarunk egyszerre mindent megérteni, csak azt, amire az adott tanulási szinten szükségünk van. Ez a hagyományosan minden szót elmagyarázó technikától gyökeresen eltér.

Az új szemléletű tanulással – a konzolos környezethez képest – azonnal a hétköznapi életből ismert felületű programot készít a hallgató. Viszonylag kevés háttérismerettel már élvezhető alkalmazások fejleszthetők. A lehetőségeknek csak a fantázia szab határt. Ily módon a tanulás sikerélményt nyújt, önállóságot teremt, és más

[5] Jutta Eckstein, Mary Lynn manns, Helen Sharp, Marianna Sipos: Teching from Differnet Perspectives, EuroPLoP'03, Irsee, 2003. 165-183. old.

megtanulni valamit úgy, hogy felfedezzük azt, vagy egy probléma megoldásához kell az új ismeret.

Az oktatás gyakorlata

A módszer hatására nem lesz a bonyolult egyszerűbb, csak a megismeréséhez vezető út indul a napi gyakorlathoz közelebbről, lehetővé téve, hogy már az első órákon olyan fogalmakkal találkozzék a hallgató, melyekkel felhasználóként, mint ismeretlennel szembesült. Ezek a ma programozásának alapfogalmai s az első fecske (Early Bird) pedagógiai gyakorlat értelmében célszerű ezekkel kezdenünk az oktatást.

A következőkben ismertetem az általam az ELTE-n és a BMF NIK-en többször kipróbált heti 2-4 órás gyakorlatot. A tananyag természetesen lassúbb ütemben is feldolgozható. Az önálló feldolgozáshoz vagy a tanórai munkához nyújt segítséget a 'Programozás élesben' c. könyv – az elmélethez kapcsolódó részletesen kidolgozott, illetve önálló megoldásra szánt feladataival. Az oktatáshoz használt segédanyagok az internetről is elérhetők.[6]

Oktatási cél: A tárgy keretében a hallgatók megismerhetnek egy az eddig tanulttól eltérő gondolkodásmódú tiszta objektumorientált programozási nyelvet a C#-ot és fejlesztői környezetet a Visual Studio .NET fejlesztőeszközt. A C, C++ nyelvcsalád új, kifejezetten az osztott alkalmazások fejlesztése érdekében továbbfejlesztett C# programozási nyelvvel, a különböző programozási nyelveken írt komponensek közti együttműködést támogató Visual Studio fejlesztői környezettel, a többrétegű alkalmazások futtatását támogató .NET környezethez készíthetnek programokat.

Ütemezés:	
Oktatási hét (konzultáció)	**Témakör**
1.	Osztály, objektum. A fejlesztői környezet lehetőségei.
2.	Vezérlők használata, eseménykezelés, vezérlőszerkezetek a C# nyelvben.
3.	Egyszerű alkalmazás készítése, környezet, eseménykezelés, nyomkövetés. Több ablak használata az alkalmazásban. Adatcsere az ablakok között.
4.	A felhasználói felület kezelése, további események kezelése. Átlátszó, áttetsző, tetszőleges alakú ablak.
5.	Öröklés, osztálykönyvtárak, kód-újrahasznosítás. A .NET osztálykönyvtár, névterek és osztályok.
6.-7.	Felügyelt kód, Toolbox komponensek. Process fogalma, új process létrehozása, kezelése az alkalmazásból statikus metódushívással és objektumon keresztül. Az időzítő használata.
8.	Számonkérés, önálló feladatmegoldás gép mellett.
9.	Osztott alkalmazás fogalma, assembly, dll, és használata az alkalmazásokban. Internal láthatóság. Kész komponenst felhasználó alkalmazás (Internet Explorer, Word, Microsoft.Ink…).

[6] A tantárgyak előadásaihoz használt segédletek. www.nik.hu/aii/oktatok/siposmarianna.html.

10.	Osztálykönyvtár készítése és felhasználása. Már megírt forráskód felhasználása új alkalmazás fejlesztésekor. Több project kezelése egy Solutionben.
11.	Új Windows vezérlő fejlesztése, felhasználása a Toolbox-ból. Teszt alkalmazás készítése. Tulajdonság fogalma a gyakorlati fejlesztés során, létrehozásuk a vezérlőkben.
12.	Adatbázis-kezelés. Adatbázis elérése a fejlesztői környezetből adatbázisfájl és adatbázisszerver esetén. (Access és SQL Server adatbázis esetén) Az adatbázis egyszerű elérése alkalmazásból. Connection, DataAdapter, DataSet, DataGrid
13.	Web alkalmazás fejlesztés. Szerver oldali web alkalmazás fogalma. IIS, beállítások. WebForm szerkesztése és HTML nézet. Futtatás böngészőben.
14.	Számonkérés, önálló feladatmegoldás gép mellett.

Félévközi követelmények

Oktatási hét (konzultáció)	
8.	Számonkérés, gépes dolgozat
14.	Számonkérés, gépes dolgozat
15.	Pótlások, gépes dolgozat

Pótlás módja: Gépes dolgozat a teljes tananyagból az utolsó gyakorlaton, illetve különeljárási díj ellenében a pótlási időszak alatt.

A *félévzáró érdemjegy* (*J*) kialakításának módszere: A két gépes dolgozat átlaga, az órai munka, az otthon önállóan fejlesztett alkalmazások színvonalának figyelembe vételével.

Irodalom:

Ajánlott:

Magyar nyelvű könyv:

Sipos M.: Programozás élesben, C#, InfoKit, 2004.
Bradley L. Jones: C# mesteri szinten, Kiskapu Kft, 2004.
Albert I., Balássy Gy., Charaf H., Erdélyi T., Horváth Á., Levendovszky T., Péteri Sz., Rajacsics T.: A .NET Framework és programozása, Szak kiadó, 2004.

Magyar nyelvű cikkgyűjtemény:

Visual Studio.NET C# 2001, Software Offline, Animare Software Kft, 2002.
Visual Studio.NET C# 2002, Software Offline, Animare Software Kft, 2003.
www.SoftwareOnline.hu

Angol nyelvű:

Developing Microsoft.NET Applications for Windows Visual C#.NET Microsoft Corporation 2002, Material No: 2555A
Introduction to C# Programming for the Microsoft.NET Platform, Microsoft Corporation 2001, Material No. 2124B

Egyéb segédletek:
Tanórán kiosztott anyagok
A tantárgy honlapján elérhető információk.

Összegzés

A programozásoktatás az informatikai ismeretek de a kulcskvalifikációk fejlesztése érdekében is rendkívüli jelentőséggel bír. Egy ilyen dinamikusan fejlődő területen az oktatásnak is állandó megújulásra van szüksége. Az új elvárások, az új fejlesztőeszközök és az új alapfogalmak az oktatás tartalmi és módszertani változtatását várják el az oktatóktól. Ma már kezünkben vannak azok az eszközök, melyek e változtatást lehetővé teszik, csak élnünk kell velük. Ahhoz, hogy oktatásunk meg tudjon felelni mind a magyar, mind a világpiac elvárásainak, hogy végzős hallgatóink helyt tudjanak állni a nemzetközi versenyben, az oktatás tartalmát és módszereit folyamatosan meg kell újítanunk. A siker záloga a mi kezünkben van.

Felhasznált irodalom:

Joseph Bergin: Early Bird Pedagogical Pattern, Pace University, New York, 1998. http://www-lifia.info.unlp.edu/ar/ppp/pp34.htm

OECD PISA Knowledge and Skills for Life – First Results from PISA 2000, Párizs, 2001.

Wikipedia the free encyclopedia http://wikipedia.org

Plankalkül http://en.wikipedia.org/wiki/Plankalk%FCl

A programozási nyelvek időrendi sorrendben a Wikipedia honlapján. http://en.wikipedia.org/wiki/Timeline_of_programming_languages 2004. 09. 24.

Jutta Eckstein, Mary Lynn Manns, Helen Sharp, Marianna Sipos: Teching from Differnet Perspectives, EuroPLoP'03, Irsee, 2003. 165–183. old.

Sipos Marianna: Programozás élesben, C#, InfoKit, 2004.

A tantárgyak előadásaihoz használt segédletek. www.nik.hu/aii/oktatok/siposmarianna.html.