Acta et Commentationes, Sciences of Education, nr. 2(32), 2023 ISSN 1857-0623
p. 27-34 E-ISSN 2587-3636

CZU: 378.016:004.42 DOI: 10.36120/2587-3636.v32i2.27-34

PROIECTAREA ȘI IMPLEMENTAREA CURSULUI DE PROGRAMARE JAVA: STRATEGII, INSTRUMENTE ȘI PROVOCĂRI ÎN PREDARE

Tatiana CHIRIAC, dr., conf. univ.
https://orcid.org/0000-0002-6122-1937
Catedra Informatică și Tehnologii Informaționale
Facultatea Fizică, Matematică și Tehnologii Informaționale
Universitatea Pedagogică de Stat „Ion Creangă”

Rezumat. În articol se specifică aspecte privind proiectarea instrucțională a unității de curs „Programarea Java”. Proiectarea și implementarea cursurilor de programare este, fără îndoială, o sarcină provocatoare. Învățarea limbajelor de programare este un proces continu care constă din interacțiunea dintre învățarea practică și învățarea teoretică a materialului. Elaborarea curriculumului pentru cursul programarea Java se bazează pe analiza metodelor de predare, specifice domeniului de programare, implementarea abordărilor de învățare activă și tehnici de evaluare elaborate în cadrul platformei universitare. Valențele formative ale disciplinei sunt racordate la finalitățile programului de studii universitare de licență.
Cuvinte-cheie: proiectare instrucțională; metode de predare-învățare în programare; învățare activă a unui limbaj de programare.

DESIGN AND IMPLEMENTATION OF JAVA PROGRAMMING COURSE: STRATEGIES, TOOLS, AND TEACHING CHALLENGES

Abstract. The article specifies aspects regarding the instructional design of the "Java Programming" course. Designing and implementing programming courses is undoubtedly a challenging task. Learning programming languages is a continuous process that consists of the interaction between practical and theoretical learning of the material. The curriculum for the Java programming course is based on the analysis of teaching methods specific to the field of programming, the implementation of active learning approaches, and evaluation techniques developed within the university platform. The formative values of the discipline are connected to the aims of the bachelor's degree programme.
Keywords: instructional design; teaching-learning methods in programming; active learning of a programming language.

Introducere

Predarea și învățarea limbajelor de programare este un proces multilateral și este însoțită de mai multe probleme, cum ar fi complexitatea caracteristicilor limbajului de programare, dezvoltarea diferitelor abilități de abstractizare și raționament logic, cât și cunoștințe fragile ale studenților în anumite cazuri. De asemenea, învățarea limbajelor de programare nu poate fi tratată ca o problemă obișnuită pentru programele de studii din domeniul tehnologiilor informaționale și informaticii, deoarece însăși procesul de programare este considerat unul complicat și necesită înțelegerea deplină a conceptelor abstracte din domeniul programării. În acest sens, proiectarea cursurilor de programare pentru studenții de la nivelul licență a fost întotdeauna o sarcină provocatoare.

În această lucrare, analizăm paradigma de proiectare a unității de curs „Programare Java”, care constă din cercetarea teoretică privind designul instrucțional al disciplinei în cadrul programului de studii „Dezvoltarea produselor program și a aplicațiilor”, analiza strategiilor de predare și învățare, controlul calității didactice și proiectarea curriculară a disciplinei. Rezultatele acestei investigații arată că organizarea curriculară a disciplinei, în care teoria și practica sunt intercalate, asociată cu abordarea învățării active bazate pe probleme sau sarcini, oferă avantaje pentru învățarea limbajului de programare Java. Cursul „Programare Java”, se realizează în cadrul catedrei de Informatică și Tehnologii Informaționale, UPSC „Ion Creangă”, care se orientată spre nevoile pieței muncii și pune la dispoziția studenților programe de studii moderne la specialitățile de Informatică ce pregătesc viitorii specialiști pentru spațiul socio-economic autohton.

Analizând aspectele generale conexe teoriei cognitive și tehnologiei educaționale elucidate de autorul Jonassen (2003) [1], vom evidenția că învățarea este un proces de transformare a informațiilor și experienței în cunoștințe, abilități, comportamente și atitudini, care face parte atât dintr-un proces continuu pe tot parcursul vieții, dar și parte a educației formale și dezvoltării personale în cadrul sistemului de învățământ.

Concentrându-ne pe partea aplicativă a domeniului informatic, considerăm că designul instrucțional al predării și învățării limbajelor de programare trebuie să urmeze abordarea învățării centrată pe student, în care rolul profesorului este de a promova și de a facilita învățarea studenților și înțelegerea generală a materialului, și de a măsura învățarea atât prin strategii formale, cât și informale. Într-o sala de studiu centrată pe student, predarea și evaluarea sunt conectate, privită ca un proces continuu, metodele cele mai populare de evaluare fiind proiecte individuale, proiecte bazate pe sarcini, sau portofoliile studenților.

Vom specifica că alegerea metodelor de predare depinde de mai multe aspecte: plurivalența unității de curs /disciplinei predate, scopul și obiectivele programului de studiu, demografia grupului de studenți etc. Deci, metodele de predare sunt foarte importante. Dacă profesorul va adopta doar metodologii tradiționale de predare, acesta riscă să piardă din numărul de studenți interesați de învățare. Prin urmare, metodologiile de predare trebuie actualizate și /sau modificate în mod constant.

Metode de predare a limbajelor de programare: analiza literaturii de specialitate

Dobândirea abilităților necesare proiectării și dezvoltării produselor program și aplicațiilor este una dintre principalele provocări cu care se confruntă studenții la informatică. Ca urmare a incapacității de a dezvolta anumite competențe, întâlnim cazuri când studenții abandonează cursurile, și uneori facultatea. Pentru a preveni abandonul,

profesorul trebuie să aplice strategii pentru cultivarea și îmbunătățirea capacității de operare practică a studenților în domeniul programării.

Scopul oricărui domeniu aplicat este de a îmbunătăți practica. Modul în care teoria ghidează practica se reflectă în selectarea algoritmului /acțiunii care va duce la cele mai bune rezultate. După cum menționează autorii Berglund și Eckerdal (2015), „în învățarea programării, există o interacțiune complexă între învățarea practicii (ce să scriem, cum să citim mesajele compilatorului etc.) și învățarea teoriei (la ce sunt buni constructorii, cum „funcționează" o procedură, cum se execută programul etc.). ... și să încurajăm studenții să învețe atât aspectele teoretice ale programării (de exemplu, semnificația unei clauze *if*, ... unde se pune punct și virgulă, cum se acționează la mesajele de eroare, etc.), pentru a obține un rezultat bun de învățare" [2].

Se cunoaște că studenții învață abilități și concepte practice în mare parte prin exersarea unui limbaj de programare. Latura practică a programării devine evidentă atunci când se discută despre dezvoltarea produselor program /software. Dezvoltarea software este un proces de rezolvare a problemelor, împărțit în mod tradițional în mai multe faze: analiza problemelor, proiectarea software-ului, implementarea și testarea, menționează autorul Eckerdal (2009) [3]. Dezvoltatorii de programe lucrează adesea în mod iterativ între aceste faze, și toate fazele conțin atât aspecte practice, cât și teoretice [ib.]. Deci, programarea este mult mai mult decât simpla învățare a sintaxei unui limbaj. Iar după cum menționează autorul Kak (2008), dacă predarea unui limbaj de programare ar însemna doar explicarea sintaxei, munca profesorului ar fi ușoară, dar și extrem de plictisitoare pentru studenți [4]. Cursurile de programare ar trebui să stimuleze și să dezvolte abilitățile și competențele studenților necesare pentru ca aceștia să poată rezolva probleme complexe din lumea reală, punctează autorii Calderon et al. (2022) [5]. În acest context, reiterăm că proiectarea curriculară trebuie să se bazeze pe o legătură strânsă între metoda de predare și rezultatul învățării. În orice limbaj de programare, predat în cadrul programului de studiu universitar de licență „Dezvoltarea produselor program și a aplicațiilor" (C, C++, Java, Python etc.), studenți trebuie să stăpânească nu numai gramatica limbajului și algoritmii de bază, ci și abilitățile de programare. Prin urmare, este necesară îmbunătățirea abilităților de programare și consolidarea cunoștințelor prin experimente și practici de dezvoltare soft.

În continuare sunt enumerate și prezentate cele mai răspândite metode de predare a limbajelor de programare descrise de autorii Papp-Varga, Szlávi, și Zsakó (2008) [6]:

- *orientare spre enunț* – limbajul de programare este văzut ca un set de enunțuri, iar elementele individuale ale setului sunt predate într-o anumită ordine;
- *aplicarea în calitate de instrument* – în cazul când se predă programarea și gestionarea bazelor de date, aspectele de predare a bazelor de date sunt considerate ca fiind de primă preocupare și, astfel, limbajul de programare este aplicat în calitate de instrument în măsura necesară;

- *orientare spre tehnologia software* – metoda de predare a limbajului de programare este adaptată la metodologia și tehnologia de dezvoltare software, în care metodologia motivează alegerea unui limbaj de programare (de exemplu, programarea orientată pe obiect (OOP) și implementarea acesteia în limbajele de programare Pascal și C++);
- *orientare spre tipul de sarcină* – metoda dată introduce noi cunoștințe despre un limbaj de programare într-un mod în care sunt necesare în procesul de rezolvare a unor probleme;
- *orientare spre limbajul de programare* – metoda de predare în care un limbaj de programare este văzut ca o unitate structurală, aducând în față logica limbajului și introducând elementele concrete ale acestuia în măsura necesară;
- *orientare spre acțiune* – metoda de predare în care se înțelege modul în care funcționează declarațiile, adică de a face studenții capabili să vizualizeze ce se întâmplă atunci când instrucțiunile sunt executate;
- *orientare spre exemple* – metoda de predare care prezintă un limbaj de programare printr-un șir de exemple.

În prezent, metodologiile de învățare activă sunt cele mai discutate în didactica contemporană, deoarece sunt interpretate ca o combinație între participarea activă a studenților, învățarea experimentală și învățarea prin acțiune. Autorii Gaspar și Langevin (2007) specifică că „aceste practici sunt inspirate de teoriile constructiviste din psihologia educației și implică de obicei activități precum studiul de caz, reflecțiile, discuțiile de grup, experimentele practice etc. Acestea, de asemenea, promovează, într-o anumită măsură, o abordare a învățării prin descoperire sub îndrumarea profesorilor, (re-)descoperă în mod autonom unitățile de cunoștințe care le sunt predate” [7]. De asemenea, se pun accente pe două metode active de predare - „programarea live” și „dezvoltarea bazată pe teste” (test-driven development) [ib.] Prima metodă se referă la strategia de predare prin care profesorul expune procesul de programare în fața grupului de studenți folosind un proiector multimedia. Procesul de programare în cazul dat este condus de profesor. Deși procesul de predare a unui limbaj de programare se îmbunătățește semnificativ, totuși este o strategie de învățare pasivă. Metoda dată poate fi folosită într-un format de tip lecție-video, ca o sugestie /recomandare suplimentară a materialului predat, ceea ce va permite studenților să urmărească materialul în afara orelor de curs. Tehnicile de dezvoltare bazate pe teste sunt menite să implice studenții în învățarea competitivă, fiecare student lucrând individual asupra codului. Tehnica de testare se aplică pentru a specifica și valida ceea ce va face codul, și poate fi utilizată aplicând metoda programării în pereche prin schimbul reciproc (studenții fac schimb între ei pentru a verifica și analiza codul scris de colegi). Astfel studenții vor face comparație între programele elaborate de colegi, identificând cel mai reușit program din punctul de vedere al execuției.

Utilizarea abordărilor de învățare activă este o alternativă propusă și de autorii Ribeiro și Bittencourt (2019), pentru a aborda dificultățile în învățarea unui limbaj de programare. Este vorba de cazul învățării bazate pe probleme (Problem Based Learning (PBL)) – o abordare de învățare centrată pe student. În PBL, învățarea este auto-dirijată, bazată pe reflecție și contribuie la dobândirea de abilități precum inițiativa, independența, comunicarea, gândirea critică, rezolvarea problemelor, lucrul în grup, acumularea cunoștințelor și aplicarea acestora în diferite contexte [8].

În prezent atestăm o evoluție semnificativă a conținutului cursurilor de programare ceea ce indică faptul că programarea este percepută ca o activitate care depășește simpla scriere a codului, dar necesită multe abilități. Proiectarea, implementarea, testarea și depanarea sunt toate exemple ale setului de aptitudini ale dezvoltatorilor de soft. Urmărim în literatura de specialitate și expuneri privind avantajele utilizării metodelor active de predare în programele de studii ale cursurilor de licență, în special impactul pozitiv asupra învățării studenților, formării și dezvoltării competențelor și aptitudinilor, reflecției asupra problemelor pe care le soluționează, gândirii critice, cât și reducerea eșecurilor la disciplinele de programare. Studiile realizate de cercetătorii din domeniu arată că metodologiile de învățare activă sunt mai eficiente decât lecțiile tradiționale în promovarea rezultatelor educaționale dezirabile, inclusiv învățarea devine mai semnificativă.

Proiectarea instrucțională

La momentul actual, Java este unul dintre cele mai populare și utilizate limbaje de programare din lume. Limbajul Java este folosit pentru a dezvolta aplicații desktop și mobile, dezvoltare Android, procesare de date mari și altele. Potrivit companiei care deține Java, Oracle (https://www.oracle.com/java/), Java este limbajul de programare și platforma de dezvoltare numărul 1. Java este activ utilizat pentru funcționalitatea sa pe diferite platforme (Windows, Mac, Linux etc.); are o cerere mare pe piața muncii, este open-source și gratuit, cât și, are un suport uriaș al comunității IT datorită posibilității de a crea aplicații independente de platformă, applet-uri web, aplicațiile dezvoltate fiind distribuite între servere, sau pot fi parte a unei pagini web. Java este un limbaj orientat pe obiecte (OOP), care oferă o structură clară programelor, și permite reutilizarea codului, reducând costurile de dezvoltare. De asemenea, Java este aproape după sintaxă de limbajele C++ și C#, ceea ce facilitează trecerea ușoară pentru programatori de la un limbaj la altul.

Curriculumul pentru cursul programarea Java se bazează pe cercetarea metodelor de predare, abordărilor de învățare și a sistemului de evaluare, și este elaborat în funcție de valențele formative ale disciplinei și finalitățile programului de studii universitare de licență „Dezvoltarea produselor program și a aplicațiilor”. Dezvoltarea curriculumului implică, de asemenea, consultări aprofundate cu cerințele pieței muncii, expunerile profesioniștilor în materie, analiza literaturii de specialitate, analiza programelor de studii

similare. Curriculumul pune accent pe interacțiunea socială în activitățile de predare-învățare-evaluare care se desfășoară între profesorul de curs și studenți.

Deciziile esențiale privind conținutului curriculumului pentru programarea Java trebuie luate de la bun început cel puțin pentru următoarele aspecte:

- selectarea unui design general de predare și învățare cu scop de a atinge finalitățile cursului și programului de studii;
- selectarea unuia sau mai multor medii de programare;
- selectarea materialelor didactice, resurselor utile, manualelor și pregătirea materialului educațional;
- selectarea strategiilor de evaluare a calității cursului predat.

Scopul principal al cursului de programare Java este de a învăța sistematic conținutul de bază al limbajului de programare Java și de a aplica cunoștințe de limbaj și idei de programare orientată pe obiecte în mod flexibil. Învățarea conceptelor conexe cu programarea Java poate duce, de asemenea, la o perspectivă și o înțelegere mai profundă a modului în care funcționează tehnologia din jurul nostru. Cursul programarea Java asigură acumularea noțiunilor teoretice de bază, dar și formarea abilităților practice necesare pentru studenții de la nivelul licență. Competențele specifice de bază ale cursului urmăresc:

- cunoașterea conceptelor de platformă de programare și mașină virtuală;
- cunoașterea tehnologiei de creare a aplicațiilor, utilizând Eclipse;
- însușirea sintaxei limbajului Java;
- cunoașterea destinației și modului de lucru a claselor de bază în Java;
- aprofundarea elementelor standard de programare orientată-obiect (moștenire, polimorfism, încapsulare);
- utilizarea componentelor AWT și formarea deprinderilor de lucru cu biblioteca Swing;
- crearea de aplicații cu interfață grafică cu utilizatorul.

Pentru a asigura accesul studenților la materialele didactice, ne bazăm pe experiența utilizării sistemului de gestiune a conținuturilor de învățare universitar (platforma de învățare https://tic.upsc.md/), cursul putând fi accesat de oriunde și oricând la necesitate. Pentru a garanta calitatea predării, învățării și evaluării, studentul are la dispoziție material didactic plasat pe platformă destinat pentru înțelegerea noului material, realizarea lucrărilor de laborator și sarcinilor pentru lucrul individual, realizarea testelor sumative, cât și resurse bibliografice recomandate. Când se explică materialul teoretic nou, sau unele subiecte, profesorul de curs folosește adesea programe gata făcute sau implementate parțial pentru a le testa direct pe calculator împreună cu studenții. Astfel studenții învață complexitatea codului unui program. Pentru a stimula atenția studenților se recurge și la programe cu greșeli în cod, așa ca, în cadrul unei prelegeri interactive, să fie corectate împreună cu studenții. În așa mod, noile concepte sunt mult mai bine asimilate, iar studenții învață să

răspundă la diferite tipuri de erori. Strategiile de predare menționate sunt aplicate cu scop a stimula interesul studenților pentru disciplina de programare. După fiecare subiect nou studenții realizează sarcini în acord cu fișa de laborator și anexează lucrările realizate la platformă; ulterior profesorul trebuie să revizuiască codul trimis de studenți și să facă comentarii. Sistemul modern de testare al platformei de învățare ne oferă o varietate de tipuri de teste, care a fost aplicat pentru elaborarea evaluărilor sumative.

Învățarea prin practicarea programării, de exemplu în laborator, este o parte importantă a cursului. Studenții, care petrec mai mult timp elaborând programe și exersând abilitățile de rezolvare a problemelor pe calculator, au succese mai mari la evaluări formative și finale. Cercetările și experiențele din domeniul cursurilor de programare, indică cât de importantă este practicarea scrierii codurilor. De asemenea, exersarea practică ajută studenții să conștientizeze lacunele în înțelegerea teoretică a materialului. Ca urmare, s-ar putea declanșa conștientizarea de către studenți a necesității de orientate spre teorie, moment care este uneori ignorat de studenți. Reflecțiile asupra materialului teoretic sunt plictisitoare, dar creează lacune în activitatea practică, dacă sunt omise. În acest context, evidențiem că există o relație strânsă dintre învățarea teoriei și aplicarea cunoștințelor în practică în domeniul programării. De aceea, profesorul trebuie să combine armonios acțiunile orientate spre teorie și acțiunile orientate spre practică, în așa mod conceptele teoretice pot fie mai ușor înțelese, cât și se îmbunătățește autonomia studenților în rezolvarea problemelor. Un rol deosebit în acest context, îl joacă și cursurile online gratuite, care explică principalele concepte de programare și prezintă diverse exemple de utilizare a limbajului Java, plasate pe site-uri specializate precum: *JavaTPoint* (https://www.javatpoint.com/java-tutorial), *Programiz* (https://www.programiz.com/java-programming), *LearnJavaOnline* (https://www.learnjavaonline.org/), *W3schools* (https://www.w3schools.com/java/), și altele.

Proiectare materialului educațional necesită de obicei mult efort și timp, și necesită o evaluare continuă a eficacității sale didactice în sala de studii. Programarea este un domeniu solicitant din punct de vedere cognitiv și, deși, este în general importantă pentru studenții de la specialitățile de informatică, se observă o scădere a interesului studenților față de programare. Este clar că în mai multe cazuri, motivația studenților depinde de interesul profesional urmărit.

Concluzii

În lucrare se face o încercare de a analiza aspectele metodologice ale designului educațional al unității de curs „Programare Java”, care cuprinde analiza strategiilor de predare și învățare, proiectarea instrucțională a disciplinei, și provocările care apar. Considerăm că designul instrucțional al predării și învățării limbajului Java trebuie să aibă la bază metodologia învățării centrată pe student (metode activ-participative), în care rolul

profesorului este de a facilita învățarea și înțelegerea generală a materialului de către studenți. Analiza cercetărilor din domeniu arată că metodologiile de învățare activă sunt mai eficiente decât lecțiile tradiționale în promovarea rezultatelor educaționale dezirabile. Organizarea curriculară a disciplinei necesită un raport bine gândit de acțiuni orientate spre teorie și acțiuni orientate spre practică, asociată cu învățarea bazată pe probleme sau sarcini. Cursul de programare Java asigură acumularea noțiunilor teoretice de bază, dar și formarea abilităților practice necesare pentru studenții de la nivelul licență.

Bibliografie

1. JONASSEN, D. H. *The Handbook of Research for Educational Communications and Technologies*. 2003. Disponibil de la: http://members.aect.org/edtech/ed1/05/05-05.html [vizitat 15.04.2023].
2. BERGLUND, A.; ECKERDAL, A. *Learning Practice and Theory in Programming Education: Students' Lived Experience*. 2015. Disponibil de la: https://www.researchgate.net/publication/283637460_Learning_Practice_and_Theory_in_Programming_Education_Students'_Lived_Experience [vizitat 17.04.2023].
3. ECKERDAL, A. *Ways of Thinking and Practising in Introductory Programming*. 2009. Disponibil de la: https://www.researchgate.net/publication/229036871_Ways_of_Thinking_and_Practising_in_Introductory_Programming [vizitat 25.04.2023].
4. KAK, A. *Teaching Programming*. 2008. Disponibil de la: https://engineering.purdue.edu/kak/TeachingProgramming.pdf [vizitat 26.04.2023].
5. CALDERON. I.; SILVA, W.; FEITOSA, E. *Active Learning Methodologies for Teaching Programming: a Systematic Mapping Study*. 2022. Disponibil de la: https://www.researchgate.net/publication/371069487_Active_Learning_Methodologies_for_Teaching_Programming_a_Systematic_Mapping_Study [vizitat 10.05.2023].
6. PAPP-VARGA, Z.; SZLÁVI, P.; ZSAKÓ, L. *ICT teaching methods – Programming languages*. 2008. Disponibil de la: http://www.kurims.kyoto-u.ac.jp/EMIS/journals/AMI/2008/ami2008-papp-szlavi-zsako.pdf [vizitat 12.05.2023].
7. GASPAR, A.; LANGEVIN, S. *Active learning in introductory programming courses through Student-led "live coding" and test-driven pair programming*. 2007. Disponibil de la: http://cereal.forest.usf.edu/clue/publications/2007-EISTA.pdf [vizitat 28.05.2023].
8. RIBEIRO, A.L.; BITTENCOURT, R. A. *A Case Study of an Integrated Programming Course Based on PBL*. 2019. Disponibil de la: https://www.researchgate.net/publication/334361449_A_Case_Study_of_an_Integrated_Programming_Course_Based_on_PBL [vizitat 19.05.2023].