PROGRAMAREA ORIENTATĂ PE OBIECT: PARTICULARITĂŢI DE PREDARE

***Ala GASNAŞ*, lector superior, UST, alagasnas@yahoo.com**

Rezumat: Gândirea abstractă este o abilitate esenţială atunci când se studiază în domeniul informaticii. Conceptele tehnologiei orientate pe obiect face ca aceste abilităţi să fie mult mai importante. Cu toate acestea, studenţii, chiar şi practicienii cu experienţă în domeniul IT, întâmpină dificultăţi în gândirea în termeni abstracţi, în timp ce practică tehnologiile orientate pe obiect. În acest articol aş sugera o altfel de abordare a metodelor de predare a programării orientate pe obiect pentru sprijinirea studenţilor de a obţine rezultate cât mai înalte la această disciplină şi a aplica aceste cunoştinţe în domeniul economic.

Cuvinte-cheie: informatica, orientare pe obiect.

Abstract: Abstract thinking is an essential skill when studying computer science. Object technology and the concepts it is based upon make this skill even more. However students, as well as experienced practitioners in IT, encounter difficulties in thinking in abstract terms while practicing object oriented development. In this article I suggest a different approach of teaching object-oriented programming to support students to achieve high results in this discipline and to apply this knowledge in economics.

Keywords: informatics, object-oriented programming.

1. Introducere

În ultimii ani Programarea Orientată pe Obiect (POO) a devenit una dintre cele mai influente domenii ale programării. Ea este utilizată pe larg în domeniul educaţiei şi economiei, iar universităţile au introdus cursul de Programare Orientată pe Obiect. În prezent, metodologia POO a devenit una dintre problemele centrale în formarea viitorilor specialişti şi, bineînţeles, există o necesitate în studierea aprofundată a acestei discipline.

Predarea programării orientate pe obiect rămâne, cu toate acestea, destul de dificilă.

De ce este dificilă? Sau, mai precis, de ce predarea programării orientate pe obiect pare a fi mult mai dificilă decât predarea programării structurate?

Cercetările existente arată că sunt o serie de probleme cu care se confruntă cadrele didactice în procesul predării programării orientate pe obiect [6]. Studenţii consideră că conceptele POO, cum ar fi clasele, constructorii, supraîncărcarea constructorilor , funcţiile frend, precum şi alte concepte din programarea orientată pe obiect sunt greu de înţeles [5]. Studenţii, care au practicat mai întâii programarea procedurală, întâlnesc anumite dificultăţi în studierea programării orientate pe obiect şi este nevoie de o anumită perioadă de timp pentru a se familiariza şi a înţelege conceptele programării orientate pe obiect.

Din păcate, multe manuale utilizează programarea procedurală ca o cale spre conceptele obiectuale. Mulți oameni vizualizează Programarea Orientată pe Obiect doar ca un alt limbaj, care poate fi învățat după structurile de control, pointeri și recursii.

Programarea orientată pe obiect este o paradigmă de bază, care modelează gândirea noastră în ceea ce priveşte cum trebuie încadrată o problemă într-un model algoritmic. Ea nu poate fi adăugată la alte construcţii lingvistice, mai degrabă înlocuieşte structura fundamentală a programării procedurale [7].

Deficienţele ce apar la predarea POO sunt cauzate mai degrabă de instrumentele disponibile pentru predarea lor decât de principiile programării orientate pe obiect.

Metoda de studiere centrată pe student este una din cele mai eficiente metode de predare a POO. Învăţarea centrată pe elev constă în facilitarea înţelegerii şi schimbarea conceptuală sau dezvoltarea intelectuală [1]. Ca instrument de predare, această metodă încurajează studenţii de a dezvolta gândirea analitică, precum şi abilităţile de scriere a programelor.

În cazul limbajelor de programare, această abordare se concentrează pe predarea conceptelor prin încurajarea studenţilor să scrie şi să implementeze programe cu scopul de a rezolva problemele propuse.

Cele mai multe dintre dificultăţile pe care studenţii le întâlnesc în faza iniţială de pregătire apar în procesul formării conceptului despre elementele de bază ale programării orientate pe obiect. De aceea, în dependenţă de formarea acestor idei, va depinde studiul ulterior al metodologiei POO.

La etapa iniţială de predare POO, aş recomanda să se folosească prezentările. În aceste prezentări să se utilizeze pe larg animaţiile, să se evidenţieze cu diferite culori liniile de cod şi părţile relevante ale exemplelor de programe, ale elementelor de diagrame, pentru a ajuta studenţii să înţeleagă mai bine conceptele POO [6]. Folosirea prezentărilor la etapa iniţială de predare va permite studenţilor să deprindă abilităţile de scriere a codului de program, orientat pe obiecte mult mai repede şi mai uşor.

Pentru a stimula gândirea activă a studenţilor trebuie introduse noi concepte şi principii cu ajutorul unor exemple din viaţa de zi cu zi.

2. Principiile fundamentale ale POO şi gândirea convenţională

În viaţa obişnuită, cele mai folosite metodologii sunt inducţia şi deducţia. Inducţia înseamnă un proces de abstractizare de la special la general. Deducerea înseamnă un proces de la general la special. Vom formula trei declaraţii:

- Totul în lume este un obiect.
- Fiecare sistem este compus din obiecte. Şi un sistem, la rândul lui, este de asemenea un obiect.
- Evoluţia şi dezvoltarea unui sistem este cauzată de interacţiunile dintre obiectele din interiorul sau din afara sistemului.

Prima declaraţie spune că totul în lume este un obiect. In lumea reală, florile, copacii şi animalele sunt obiecte; studenţii şi profesorii sunt obiecte; birourile, scaunele, sălile de clasă şi clădirile sunt obiecte; universităţile, oraşele şi ţările sunt obiecte; chiar lumea şi universul sunt obiecte. În lumea abstractă, un subiect, cum ar fi inginerie, calculator, cibernetică, matematică şi istorie sunt, de asemenea, obiecte.

A doua declaraţie afirmă că fiecare sistem este compus din obiecte. De exemplu, un sistem cultural include istoria, limba, alimentele, costumele, relaţiile, oamenii, etc., care sunt obiecte; un sistem de învăţământ include şcolile, studenţii, profesorii, administratorii, etc., care sunt, de asemenea, obiecte; un sistem informatic include monitorul, tastatura, placa de bază, procesorul, memoria, sistemele de operare şi de software/aplicaţii, care sunt obiecte.

A treia declaraţie descrie dezvoltarea unui sistem cauzată de interacţiuni. De exemplu, o şcoală este un sistem. Dezvoltarea sa este cauzată de interacţiunile dintre elevi, profesori şi personal.

POO suportă ambele metodologii: atât inducţia, cât și deducţia. În faza de analiză, multe obiecte pot fi folosite pentru a forma o clasă. Aceasta este o inducţie. Pe de altă parte, atunci când există deja unele clase, mai multe obiecte sau instanţe pot fi create din aceste clase. Acesta este un proces de deducere.

Atunci când un obiect al unei clase are atribute, care sunt obiecte de alte clase şi aceste obiecte pot avea atribute, care la rândul lor sunt obiecte de alte clase, se spune că acest proces este de decompoziţie.

Atunci când avem mai multe obiecte de clase diferite, putem construi un obiect nou cu aceste obiecte, iar prin acest obiect nou se poate introduce o clasă nouă. Prin urmare, putem avea mai multe clase noi construite din obiecte suplimentare şi acest proces se numeşte compoziţie.

Înainte de a se introduce alte concepte, studenţii pot fi rugaţi să descrie principiile fundamentale ale POO. E de dorit să le descrie în scris sau să le repete oral.

3.Introducerea conceptului de obiect prin noţiunile din lumea reală

Gândirea activă a studenţilor este stimulată la introducerea unor concepte și principii noi, prin utilizarea exemplelor din viaţă.

Este bine de a introduce conceptul de obiect înaintea conceptului de clasă, deoarece conceptul de clasă derivă din captarea obiectelor obişnuite. După ce sunt discutate clasele, conceptul obiect poate fi consolidat pe baza conceptului de clasă.

Dacă pornim de la faptul că totul în lume este un obiect [3], un obiect trebuie să posede următoarele proprietăţi:

- "nume unic", adică orice obiect ar trebui să aibă un nume unic;
- "creat sau distrus";
- "comunicativ", adică un obiect poate face schimb de mesaje cu alte obiecte;
- "imbricat", adică un obiect complex are si alte obiecte în calitate de componente (care la rândul lor pot avea obiecte componente);
- "activ și autonom", adică un obiect nu este controlat direct de către oameni;

Prin urmare, dacă am generaliza şi abstractiza cele spuse mai sus, obiectul ar fi caracterizat de următoarele elemente:

- are nume;
- are o stare sau un corp reprezentat de un set de atribute;
- posedă un set de metode pe care obiectul le poate efectua;
- are o interfaţă care este o submulţime a tuturor metodelor obiectului.

La acest pas, studenţii pot fi rugaţi să descrie cine sunt ei folosind această formulă.

4. Introducerea conceptului de clasă prin abstractizarea mai multor obiecte comune

Prin inducţie, se poate face un concept abstract, cu respectarea lucrurilor concrete comune. Din punctul de vedere al programării orientate pe obiect, o clasă se poate forma prin procesul de abstractizare.

O clasă se defineşte prin următoarele patru elemente:

- un identificator sau un nume de clasă;
- o descriere de spaţiu pentru memorie;
- un set de prototipuri ale metodelor sau implementări;
- o interfaţă unificată a tuturor obiectelor din această clasă.

Se observă că această formulă este similară cu cea a obiectului. Clasa este de fapt o abstracţie a obiectelor. Ea este folosită pentru a reprezenta un grup de obiecte cu proprietăţi identice.

La acest pas, studenţii pot fi rugaţi să descrie clasa la care aparţin ei şi să dea descrierea fiecărui element din această clasă. Prin această ilustrare, ei vor fi capabili să împartă o clasă din C ++ în cele patru elemente ale acestei definiţii.

5. Introducerea noţiunii de Instanţă după ce s-a studiat conceptul de Clasă

Clasa nu este doar un rezultat abstract de obiecte, ci și un şablon pentru a crea obiecte, care mai sunt numite şi instanţe[8]. La această etapă se poate redefini conceptul de obiect bazat pe conceptul de clasă, deoarece a fost predat deja conceptul de clasă.

Pentru instanţele create din clasă şi din punctul de vedere al implementării obiectul poate fi redefinit prin următoarele elemente:

- identificatorul sau denumirea obiectului;
- obiectul clasei identificat prin numele sau identificatorul clasei;
- corpul sau spaţiul, valorile căruia sunt numite atribute, proprietăţi sau stări.

Aici se poate arăta că crearea de obiecte dintr-o clasă în C ++ este un proces similar cu această definiţie.

La acest pas, studenţilor li cere să descrie cine sunt ei prin crearea de la început a clasei, apoi crearea instanţei.

Concluzii:

Majoritatea studenţilor au modul de gândire algoritmic format. Schimbarea modului de gândire al studenţilor apare de obicei atunci când încep să realizeze beneficiile oferite de metodologia POO. Abordarea orientată pe obiect permite de a rezolva problema de construire a sistemelor complexe; de a îmbunătăţi întreţinerea de software şi de a crea codul reutilizabil. Aceste beneficii sunt factorul motivant pentru studierea metodologiei POO. Cunoaşterea decompoziţiei obiectuale este unul dintre factorii decisivi care poate duce la o schimbare în stilul de gândire al studenţilor de la programarea procedurală la programarea orientată pe obiect. Programarea orientată pe obiect în C ++ este foarte importantă în educaţia specialiştilor din domeniul IT şi economie . C ++ este un limbaj de programare foarte bun, cu mai multă flexibilitate. Cu toate acestea, trebuie acordată mai multă atenţie metodelor de predare şi trebuie făcut ca studenţii să cunoască întâi principiile programării orientate pe obiect. Utilizând aceste metode, mecanismele C ++ vor fi predate pentru a ajuta studenţii să folosească aceste mecanisme utile pentru codificare. Numai în acest fel studenţii vor obţine într-adevăr imaginea de ansamblu despre programarea orientată pe obiect în C ++.

BIBLIGRAFIE:

1. Devlin, M., 2006 ” Challenging Accepted Wisdom about the Place of onceptions of Teaching in University Teaching”, Improvement *International Journal of*

Teaching and Learning in Higher Education 18(2), 112-119 http://www.isetl.org/ijtlhe/ ISSN 1812-9129
2. Knudsen, J. L. and O. L. Madsen, "Teaching Object-Oriented Programming is more than teaching Object-Oriented Programming Languages," in *Proceedings of ECOOP '88*, pp. 21-40, Springer-Verlag, Oslo, Norway, 1988.
3. Meyer, B., *Object-Oriented Software Construction*, Prentice Hall, 1988
4. Бертран М. Объектно - ориентиро-ванное конструирование программных систем / Пер. с англ. - М.: Издательско-торговый дом «Русская Редакция», 2005. - 1232 с.
5. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++, 2-е изд. / Пер. с англ. - М.: «Издательство Бином», СПб: «Невский диалект», 1998. - 560 с.
6. http://kar.kent.ac.uk/21839/1/the_problem_of_teaching_object-oriented_kolling.pdf
7. http://www.rae.ru/snt/?section=content&op=show_article&article_id=4802
8. http://www.dsi.fceia.unr.edu.ar/downloads/informatica/info_II/c++.../Practical