Predarea programării se confruntă cu unele probleme generale de predare, dar se confruntă și cu unele probleme specifice, cum ar fi înțelegerea conceptelor de programare, precum și a algoritmilor pentru rezolvarea sarcinilor de programare.

Experiența noastră de predare cu studenți la nivel de începător universitar a arătat că studenții încearcă adesea să evite să înțeleagă conceptele de programare prin niște „comenzi rapide”, cum ar fi învățarea codului programului prin memorare, copierea programelor de la colegi etc.

Generatorul on-line de exerciții ale elevilor se bazează pe un model de scriptare generator (GSM), introdus în 2005 de Radošević [18] și descris în continuare în 2009 în [19]. Sistemul de generare implementat este prezentat în Figura 1.

Înțelegerea conceptelor de programare și a algoritmilor pentru rezolvarea sarcinilor de programare sunt punctele cheie în realizarea abilităților de programare pentru elev. Însă, studenții încearcă adesea să evite înțelegerea conceptelor de programare prin intermediul unor „comenzi rapide”, cum ar fi învățarea codului de program prin memorare, copierea programelor de la colegi etc. Personalizarea exercițiilor de programare ale elevilor ar trebui să agraveze soluțiile de copiere în rândul studenților care încearcă să evite înțelegerea conceptelor de programare. Generarea automată on-line a exercițiilor de programare ale elevilor oferă o soluție, în care exercițiile sunt legate de ID-urile elevului. Soluția este ușor de utilizat de către elevi, dar în faza actuală de implementare are unele cerințe suplimentare pentru profesori. La început, abordarea oferită necesită mai multă muncă a profesorilor în pregătirea exercițiilor și unele abilități în meta-programarea șablonului. Notarea exercițiilor elevilor ar putea lua ceva timp suplimentar din partea profesorilor, dar acest lucru ar putea fi rezolvat prin unele îmbunătățiri ale sistemului generator, care sunt planificate pentru munca noastră viitoare.

---

În ultimii ani s-a lucrat intens la dezvoltarea de software educațional specializat pentru a ajuta studenții să înțeleagă conceptele de bază de programare și să dezvolte abilități de rezolvare a problemelor. Cei mai cunoscuți reprezentanți ai fiecărui grup menționat vor fi prezentați în acest capitol și vor fi explicate pe scurt scopul și funcționalitățile lor generale. Astăzi există multe instrumente folosite în procesul de predare a programării, variind de la simple instrumente folosite doar la universități și facultăți la care au fost dezvoltate.

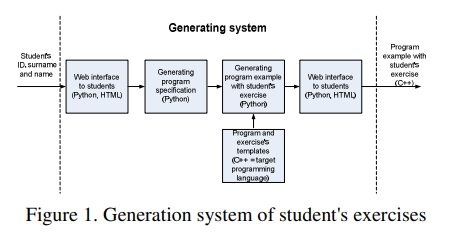
această lucrare prezintă o idee de abordare generativă în rezolvarea acestor probleme. Introducem sistemul de generare automată personalizată a sarcinilor de programare care sunt create după un șablon specific, dar unic pentru fiecare student. Această abordare, aplicată în timpul procesului de predare, reduce dramatic probabilitatea ca soluțiile să fie copiate. iar ulterior are ca rezultat o implicare mai activă a fiecărui elev pentru a crea o soluție unică și proprie. Rezultatul final este cunoștințe mai aplicabile în programare.

Domeniul evaluării automate este imens și există mai multe categorii diferite ale sistemelor existente. Carter şi colab. [6] a împărțit exercițiile în cinci categorii de bază: întrebări cu răspunsuri multiple, sarcini de programare, răspunsuri vizuale, răspunsuri text și evaluare de la egal la egal

Simplitatea întrebărilor cu alegere multiplă le-a făcut o caracteristică foarte populară în sistemele de management al învățării, cum ar fi Moodle și WebCT. Cu toate acestea, în predarea programării, întrebările cu răspunsuri multiple pot fi utile doar pentru adoptarea unor elemente teoretice de bază, dar nu și pentru dobândirea de abilități practice în rezolvarea sarcinilor de programare.

Evaluarea automată a sarcinilor de programare este cel mai obișnuit exemplu de AA în domeniul informaticii.

Cele mai vechi sisteme de evaluare, denumite adesea programe de notare [8][17], s-au bazat pe o metodă foarte simplă de potrivire a rezultatelor: rezultatul creat de un program model pentru profesor a fost comparat cu rezultatul programului elevului. corectitudine, eficiență, stil, Exemple mai recente includ utilizarea arborilor de sintaxă abstractă pentru a determina dacă o trimitere conține constructele de programare necesare studentul este capabil să urmeze convențiile de codare larg acceptate (de exemplu, să folosească comentarii, indentarea codului etc.) și să scrie cod de program ușor de înțeles



Cerințele pentru profesori sunt relativ mari în acest moment, deoarece fiecare exercițiu necesită un set particular de șabloane de cod de program, care ar putea fi moștenit parțial de la exercițiile anterioare și necesită o configurație de generator care poate fi, de asemenea, moștenită parțial de la exercițiile anterioare. Deci, unele abilități ale profesorului în meta-programarea șablonului ar fi binevenite.

O altă problemă este notarea unor astfel de exerciții de programare personalizată, care ar putea dura ceva timp din partea profesorilor.

Ar putea fi rezolvată prin generarea celor două variante polimorfe de cod de programare: una pentru elev și alta pentru profesor, care include tipărirea soluției necesare. Acest lucru ar putea fi automatizat prin compilarea on-line a programului cu rezultat sub formă de pagină web generată dinamic. Deci, sistemul care este planificat pentru munca noastră viitoare ar trebui să introducă această posibilitate pentru profesori, pentru a permite o notare mai ușoară.

Generarea specifica