9. Hamari, J., Koivisto, J., & Pakkanen, T. Do Persuasive Technologies Persuade? A Review of Empirical Studies. Spagnolli, A. et al. (Eds.), Persuasive Technology, LNCS 8462. Springer International Publishing, Switzerland, 2014. P. 118–136.
10. Kapp, K. The Gamification of Learning and Instruction: Game-Based Methods and Strategies for Training and Education. Pfeiffer and ASTD, 2012. 336 p.
11. Social Collaboration and Gamification / Ch. Meske at all, 2016. URL: https://www.researchgate.net/publication/309780791.
12. Salen, K., and Zimmerman, E. Rules of Play: Game Design Fundamentals. Cambridge: MIT Press, 2003. 688 p.

УДК 37.02
DOI 10.25128/2415-3605.23.1.23

СТАНІСЛАВ ТКАЧУК

ORCID: 0000-0001-5077-5865
stanislav660@ukr.net
доктор педагогічних наук, професор
Уманський державний педагогічний університет
імені Павла Тичини
вул. Садова, 2, м. Умань

ОЛЕКСІЙ МЕЛЬНИК

ORCID: 0000-0003-3220-4676
oleksiy.melnyk@udpu.edu.ua
кандидат технічних наук, доцент
Уманський державний педагогічний університет
імені Павла Тичини.
вул. Садова, 2, м. Умань

МЕТОДИКА ВИВЧЕННЯ СЕМАНТИКИ МОВ ПРОГРАМУВАННЯ СТУДЕНТАМИ ЗАКЛАДІВ ПРОФЕСІЙНОЇ ОСВІТИ

Показано, що в закладах професійної (професійно-технічної) освіти включення до змісту навчання мов та методів програмування блоку питань, присвячених семіотиці мов програмування сприятиме успішній реалізації системи їх фундаментальної підготовки в предметній галузі «Інформаційні технології». Методична система навчання, побудована в рамках методичної теорії, сприяє оволодінню студентами знаннями та вміннями з семіотики мов програмування. Викладено основні теоретичні та методологічні положення, на підставі яких побудовано методичну систему навчання семіотики мов програмування та метамодель методики навчання у вигляді методичної теорії, яка визначила модель нашого дослідження. На підставі класифікації моделей синтаксису і семантики мов програмування, проведеної для побудови концептуального базису методичної теорії навчання семіотики мов програмування, нами були виділені наступні розділи для навчання семіотики мов програмування: формальний синтаксис мов програмування; змістовна операційна семантика мов програмування; формальна операційна семантика мов програмування; формальна дедуктивна семантика мов програмування; денотаційна семантика мов програмування. На етапі відбору змісту елементів проєктованої нами моделі методичної системи навчання семіотиці мов програмування центральне місце посідає проблема постановки цілей навчання, оскільки відбір змісту інших елементів спрямовано досягнення поставлених цілей. Семіотичні навчальні моделі включають систему завдань, при яких предметна сфера діяльності розгортається в рамках конкретних методів навчання, що передбачають текстові виклади навчальних проблем або завдань.

***Ключові слова**: освітній процес; семантика мови програмування; методична система; професійна освіта.*

STANISLAV TKACHUK

Doctor in Pedagogical Science, Professor,
Pavlo Tychyna Uman State Pedagogical University
2 Sadova Str., Uman

OLEKSII MELNYK

PhD, Assistant Professor,
Pavlo Tychyna Uman State Pedagogical University
2 Sadova Str., Uman

METHODS OF STUDYING THE SEMANTICS OF PROGRAMMING LANGUAGES BY THE STUDENTS OF PROFESSIONAL EDUCATION INSTITUTIONS

The article shows that in institutions of professional (vocational and technical) education, the inclusion of a block of questions devoted to the semiotics of programming languages into the content of learning languages and programming methods will contribute to the successful implementation of the system of their fundamental training in the subject field «Information Technologies». The methodological system of education, built within the framework of the methodologicalical theory, helps students acquire knowledge and skills in the semiotics of programming languages. The main theoretical and methodological provisions are outlined, on the basis of which a methodical system of teaching the semiotics of programming languages and a metamodel of the teaching method itself in the form of a methodical theory have been built. On the basis of the classification of models of syntax and semantics of programming languages, carried out to build a conceptual basis for the methodological theory of teaching semiotics of programming languages, there have been selected the following sections for teaching semiotics of programming languages: formal syntax of programming languages; meaningful operational semantics of programming languages; formal operational semantics of programming languages; formal deductive semantics of programming languages; denotational semantics of programming languages. At the stage of selecting the content of the elements of the model of the methodical system for teaching the semiotics of programming languages, the central place is occupied by the problem of setting learning goals, since the selection of the content of other elements is aimed at achieving the set goals. Semiotic educational models include a system of tasks, in which the subject area of activity is developed within the framework of specific teaching methods, which involve textual statements of educational problems or tasks. The goals and content of learning the semiotics of programming languages, largely influencing each other, determine the choice of teaching methods for the section in question. For example, when considering the last block of learning content, there has been chosen a simulation or social learning model, which provides for the student's assimilation of new information, as well as attempts to get involved in the situation of solving professional tasks based on it.

***Keywords**: educational process; programming language semantics; methodological system; professional education.*

В даний час система української освіти перебуває в стані модернізації, яка зумовлена глибокими структурними змінами, що відбуваються в сучасному світі, які вимагають розвитку нових підходів до побудови загальноосвітньої та професійної підготовки, поєднують у собі одночасно і гуманістичні, і технологічні, і фундаментальні основи, та відповідають одночасно сучасним вимогам інформатизації, гуманітаризації та фундаменталізації освіти. Навіть більше, найважливішою вимогою до сучасної освіти стає не тільки і не стільки надання учням системи знань, скільки озброєння їх продуктивними способами, вміннями набувати, застосовувати на практиці, перетворювати і виробляти самостійно нові наукові знання в будь-якій сфері своєї майбутньої професійної діяльності.

Наукова основа цих напрямів інформатизації та модернізації освіти та розвитку педагогічної науки закладена у роботах К. М. Лавріщевої, А. О. Аронова, В. М. Глушкова, О. В. Федусенко, Р. Портера, Л. Вінслов [1; 2; 4; 7; 10; 11].

На основі аналізу наукових публікацій та практичної діяльності викладачів інформатики, можна констатувати, що в рамках системи освіти (вищої і професійної), як і раніше, спостерігається тенденція поділу завдань навчання інформатиці на три групи – це завдання: 1) вивчення інформатики як технології для обробки інформації (формування комп'ютерної грамотності); 2) вивчення інформатики як фундаментальної науки; 3) формування інформаційної культури, досягнення якої неможливе без перших двох.

Незважаючи на те, що в рамках зазначених завдань, як показує практика, на перший план виходить навчання користувача (технічного) аспекта інформатики, навчання алгоритмізації та програмування, а також теоретичним (фундаментальним) аспектам інформатики все-таки відображається в діючих навчальних програмах як для професійної, так і для вищої освіти.

Знання елементів семантики необхідно для демонстрації того, що програмування виросло з математики і є її галуззю, тим більше що попередником програмування є теорія алгоритмів. Навчання майбутніх випускників закладів професійної (професійно-технічної) освіти

семантики мов програмування важливе і в тому сенсі, що сприяє формуванню їх світогляду, зокрема погляду на програмування як на математичну дисципліну і на програміста як математика, котрий вміє програмувати і доводити правильність алгоритмів та програм.

Аналіз програм та навчальних посібників з інформатики також показав, що в курсі «Інженерія програмного забезпечення» тією чи іншою мірою відображені питання семантики мов програмування (основні оператори та запис основних конструкцій мовою програмування, що вивчається, їх змістовна операційна семантика, синтаксичні та семантичні помилки, правильність програми, використання тверджень для доказу правильності програми, принципи та методи налагодження програм, сфери застосування найбільш уживаних мов програмування). Тож, для успішної реалізації методичної системи фундаментальної підготовки майбутніх випускників закладів професійної (професійно-технічної) освіти потрібна їх підготовка в галузі семантики мов програмування.

Метою статті є побудова методичної системи навчання семантики мов програмування та її конкретної реалізації у вигляді навчальної дисципліни для майбутніх випускників закладів професійної освіти в рамках їхньої фундаментальної підготовки.

Важливо відзначити, що методична теорія в нашому випадку є як метамодель для методики навчання, оскільки в її термінах ми описуємо методику навчання семантики мов програмування.

Семантика освіти становить такий розгляд проблем педагогіки, який на чільне місце ставить зв'язок змісту цілей, засобів, методів освіти зі структурою і функціонуванням знакових систем, співвідносить семіозис з освітнім процесом. Семантика освіти – це такий погляд на педагогічну теорію та практику, в основі якого лежать різноманітні відносини між педагогікою та семантикою; це розгортання семантичної стратегії у педагогіці. Семантика освіти може бути інструментом в оцінці переваг всієї системи навчання, оскільки саме в семантиці культури представлені багато тенденцій розвитку змісту освіти.

Таким чином, у рамках обговорюваного підходу навчання можна визначити як цілеспрямований процес формування функціонуючих семантичних систем в учнів, а також навчання людини виробництву «текстів». У межах зазначеного підходу можна розглянути три аспекти процесу творення: синтаксичний, семантичний та прагматичний [7, с. 87–88].

Спочатку зазначимо, що семантика – це загальна теорія знакових систем. Трьома складовими частинами семантики є: синтаксис (або синтактика) – це теорія, що вивчає правила побудови текстів досліджуваної мови; семантика, об'єктом дослідження якої є зв'язки між (синтаксично правильними) текстами та змістом; прагматика, яка займається відносинами між знаковими системами і тими, хто користується цими системами [8, с. 72].

Наведемо семантичні визначення понять «мова програмування», «програма» та інших, тісно пов'язаних з ними.

Поняття «мова програмування» трактуватимемо в рамках двох наступних визначень: 1) мова програмування – це знакова система для планування поведінки виконавця (зокрема комп'ютера); 2) мова програмування – це формальна мова, що використовується для складання, зберігання та модифікації програм, а також для обміну інформацією про алгоритми як між людиною та машиною (зокрема комп'ютером), так і між людьми.

З поняттям «мова програмування» тісно пов'язане поняття «програма», яке ми трактуємо згідно з наступним поглядом: «Програма (в теоретичному програмуванні) – це об'єкт, виражений формальною мовою, що володіє певною інформаційною та логічною структурою і підлягає виконанню автоматичним пристроєм».

Поняття «синтаксис», «семантика», «прагматика» деталізуємо щодо мови програмування, оскільки вона є знаковою системою. Проаналізувавши визначення поняття «синтаксис мови програмування», ми зупинилися на такому визначенні: «Синтаксис мови програмування – це набір правил, які визначають, які послідовності символів вважаються допустимими виразами (програмами) в мові».

Далі, згідно з [3, с. 286 – 290], семантикою мови програмування називаються правила, що визначають денотати, які відповідають допустимим знакам. Це визначення є суто семантичним.

Проаналізувавши ряд визначень цього поняття, але вже більш змістовного характеру, ми зупинилися на наступному: «Семантика мови програмування – це сукупність правил і угод, встановлюваних описом мови виявлення сенсу текстів цією мовою та його інтерпретації

людиною чи автоматом» [5, с. 20–21]. Таким чином, терміни «синтаксис» і «семантика» щодо мов програмування вживаються відповідно як набір правил побудови правильних текстів на цих мовах і набір способів приписування їм сенсу.

Далі наведемо низку визначень, які мають змістовніший характер, які не можна не враховувати при формуванні погляду на це поняття:

1) Прагматика визначає відносини між знаковою системою (мовою програмування) і тими, хто цю систему використовує (програмісти, з одного боку, і комп'ютер – з іншого), тобто питання, пов'язані з призначенням мов програмування та реалізацією їх на комп'ютері.

2) До прагматики програмування відноситься визначення доцільного способу дій, оцінка застосування та засобів мови в контексті розв'язуваної задачі (наприклад, співвідношення об'єктів предметної галузі з реальними характеристиками розв'язуваної задачі, питання часу та вартості ліків, зручність і наочність подання даних і т. д.).

З усього викладеного очевидно, що текст програми, як і будь-який об'єкт, побудований за правилами знакової системи (тобто знак), характеризується трьома сторонами: синтактикою (синтаксисом), семантикою та прагматикою. В цій триєдності визначеною є прагматика, що встановлює мету – відношення програми до її споживача, потім йде семантика, що розкриває сенс тексту програми (одним з численних способів), і, на кінець, синтаксис, необхідний для подання програми у вигляді слова в деякому алфавіті.

Звернемося тепер до поняття «виконавець» і розглянемо його з погляду семантики, тоді виконавець – це людина або автомобіль, що «вміє» сприймати синтаксичні конструкції мови і є носієм семантики. Денотат програми може бути описаний двояко (бо програма має мети двох рівнів): 1) процес, який призводить до результату; 2) саме цей результат.

Зазначимо, що прагматика (яка фактично описує правила використання денотатів для цілей, які виходять за рамки семантики), як правило не включається до рамок знакової системи (наприклад, у мові програмування Java відсутні будь-які правила, що обмежують застосування циклу while для недоступних цілей). Незважаючи на те, що для зручності обговорення семантика та синтаксис зазвичай поділяються, вони тісно пов'язані один з одним. У добре розробленій мові програмування семантика безпосередньо випливає із синтаксису, тобто форма оператора може значною мірою визначати його зміст.

Описувати синтаксис мови програмування набагато легше, ніж семантику, оскільки існують універсальні математичні моделі синтаксису (на жаль, для опису семантики подібних універсальних моделей ще не створено). Перелічимо нижче вказані універсальні моделі опису синтаксису мов програмування.

Граматичні моделі: (регулярна граматика; КС-граматика та її різновиди; атрибутивна (атрибутна) граматика; мова специфікації граматик, що використовується в генераторах компіляторів (типу LALR); VDL (Vienna Definition Language) – це метамова, призначена для опису мов програмування. В цій операційній моделі мови дерево синтаксичного аналізу включає також машинний інтерпретатор. Стан обчислення входить у дерево програми, а також дерево, що описує всі дані для конкретної машини. Черговий оператор переводить дерево в новий стан.

Автоматні моделі: кінцеві автомати, автомати з магазинною пам'яттю (МП-автомати).

Графові моделі: синтаксичні графи.

Алгебраїчні моделі: регулярні вирази.

Зауважимо, що регулярні вирази підтримують мови програмування Perl, AWK, TCL і редактор GNU Emacs.

Семантику будь-якої мови програмування можна описати двома способами: змістовно і формально.

Наведемо типовий приклад змістовного опису семантики: «У команді **введення** через кому вказуються імена величин. При виконанні команди ПК чекає, поки людина введе відповідні значення, і після закінчення введення надає введені значення зазначеним величинам. Ознакою кінця введення служить натискання на спеціальну клавішу (зазвичай на цій клавіші зображена вигнута стрілка « ← »). При введенні кількох чисел вони відокремлюються один від одного комою або пробілом» [8, с. 73].

Такий спосіб опису семантики гарний до того часу, поки ці описи читають люди. Але зустрічаються ситуації, коли опис має бути формальним, тобто «доступним» комп'ютеру, що необхідно, наприклад, при компіляції програми з цієї мови програмування.

Трансляційна семантика мови програмування задається у вигляді визначення правил перекладу кожної синтаксично правильної програми у пропозицію мовою, семантика якого відома.

Другою мовою (семантичною) може бути обраний: а) *псевдокод* як «комбінація» природної мови та мови програмування; б) мова програмування (наприклад, машинна мова або мова програмування високого рівня); в) математична мова (наприклад, мова обчислення або мова першого порядку).

Нижче ми коротко опишемо найпоширеніші способи формального опису семантики, які називатимемо моделями семантики мов програмування.

Операційна семантика призначена для того, щоб чітко зафіксувати правила поведінки безпосереднього виконавця програми (виразні можливості програміста).

Операційна семантика (імперативних мов програмування) – це спосіб завдання семантики мови за допомогою визначення механізму виконання (як правило, на деякій абстрактній машині) і пояснення потім значення програми, враховуючи те, що обчислюється в результаті застосування до програми цього механізму.

Таким чином, «смислом програми» (з точки зору виконавця програми – людини або комп'ютера) природно вважати послідовність дій виконавця, що вказується цією програмою. Будемо підрозділяти операційну семантику на неформальну (змістовну, інтуїтивну) та формальну.

Неформальною операційною семантикою називатимемо опис семантики мови програмування, семантичною мовою якого є псевдокод («комбінація» природної мови та деякої мови програмування).

Опишемо неформальну операційну семантику оператора присвоєння імперативного програмування Java. «Скромний» оператор присвоювання є дуже складним і виконується таким чином: 1) обчислюється значення виразу у правій частині оператора; 2) обчислюється значення вираження у лівій частині оператора; вираз визначає адресу комірки пам'яті; 3) копіювання значення, обчислене на кроці 1, в комірки пам'яті, починаючи з адреси, отриманої на кроці 2.

Формальною операційною семантикою будемо називати вираз змістовної операційної семантики в термінах деякого *абстрактного* (іноді – р*еального) комп'ютера через* опис наслідків окремих кроків обчислень при виконанні програми. Наприклад, описуються всі зміни, що відбуваються зі змінними програми і під час окремих її команд, з вмістом оперативної пам'яті абстрактного комп'ютера тощо.

Класичними прикладами такого абстрактного виконавця вважаються машини Тьюрінга, кінцеві автомати (автомати Рабіна-Скотта), машина з необмеженими регістрами (машина Катленда) і т. д. [9, с. 60]

Моделі семантики, що описуються нижче, є більш формальними в тому сенсі, що вони спираються не на *обчислювальні моделі теорії алгоритмів,* а на *змістовну математику* (денотаційна семантика) або *математичну логіку* (дедуктивна семантика).

Терміни «математична семантика», «денотаційна семантика», «денотативна семантика», «функціональна семантика» в літературі, присвяченій мовам програмування, вважаються синонімами.

Існує кілька підходів до визначення поняття *«математична семантика».* Основою опису є також модель поняття *«алгоритм»* (як в операційній семантиці), але описується вона не в термінах абстрактної машини, а *змістовною математичною мовою* – в термінах деякої сукупності множин, відносин і відображень. Водночас бажано, щоб ці засоби були *фінітними, тобто* мали б справу лише з об'єктами, що піддаються побудові, і з осяжними, що піддаються опису, множинами таких об'єктів, відображеннями таких множин одне в інше і т. д. Мається на увазі опис природною мовою, але не допускає двох тлумачень.

Денотаційна семантика – це один із способів визначення семантики мов програмування, згідно з яким зміст програми, написаної конкретною мовою, задається деякою оцінною функцією, що надає кожній синтаксичній конструкції мови певне абстрактне значення (наприклад, число, значення істинності чи функцію). Оцінні функції за своєю природою можуть бути композиційними або рекурсивними, при цьому значення програми визначається як функція значень, співвіднесених з окремими синтаксичними елементами. Для роз'яснення

чи ілюстрації сказаного вище зазначимо фрагмент денотаційної семантики оператора присвоєння [6, с. 258–260].

Цей спосіб опису семантики вважається найбільш природним для мов функціонального програмування внаслідок існуючого математичного фундаменту, проте денотаційна семантика може використовуватися і для мов імперативного програмування.

Зазначимо, що одне з ключових питань теорії мов програмування полягає в тому, щоб довести, що операційна семантика мови, заснована на якій ми будуємо реалізації мови, еквівалентна його денотаційної семантики, що є частиною визначення мови.

Всюди нижче терміни «аксіоматична семантика», «дедуктивна семантика», «деривaційна семантика», «логічна семантика» вважатимемо синонімами. Аксіоматична семантика призначена для того, щоб чітко зафіксувати правила поведінки виконавця, що доводить деякі властивості програм (найцікавіше – це властивість отримувати певні результати при певних вхідних даних). Аксіоматична семантика ґрунтується на *системі аксіом,* які постулюють властивості основних конструкцій мови, та *системі правил виведення,* що дозволяють отримувати властивості програм та їх фрагментів за допомогою виведення з аксіом за правилами виведення.

Аксіоматична семантика – це семантика мов програмування, в якій значення мовної конструкції або програми деякою мовою програмування визначається деякою аксіомою.

Для кожного конкретного висловлювання аксіома вказує, що має бути істинним після його реалізації в контексті того, що було істинним до реалізації висловлювання. Діяльність [3, с. 285] вказується, що *«аксіоматична семантика»* має своїм предметом логічне обчислення, використовує поруч із звичайними логічними формулами обчислення висловлювань чи обчислення предикатів (зазвичай першого порядку) формули спеціального виду, містять у собі оператори деякої алгоритмічної мови та описують їх логічні властивості – зв’язок між станами змінних програми до та після виконання цих операторів.

Наведемо аксіоматичну семантику оператора присвоєння x:=e, представлену *аксіомою оператора присвоєння:*

$$wp(x:=e,R)\leftrightarrow\left(R\right)_{e}^{x}$$

Таким чином, аксіоматична семантика абстрагується від численних деталей обчислень, несуттєвих для розуміння сенсу програми, і дає можливість говорити про абстрактні властивості програм. Тим не менш, аксіоматична семантика в основному орієнтована на мови імперативного програмування.

Аксіоматична семантика є потужним інструментом для досліджень у галузі доказів правильності програм (зрозуміло, поняття *«правильність програми»* має бути формалізовано за допомогою методів математичної логіки), вона також створює теоретичну основу для аналізу програм як під час їх створення, так і в процесі експлуатації. Однак її корисність для опису мов програмування дуже обмежена як користувачів мови, так і для розробників компіляторів.

З певним вище поняттям *«аксіоматична семантика»* тісно пов’язаний ще один вид семантики мов програмування – так звана семантика станів. *Семантика станів* – це різновид семантики, в якій робота імперативної програми трактується як послідовність станів, перехід між якими полягає у виконанні деякого оператора.

Трансформаційна семантика є у певному сенсі узагальненням операційного та математичного підходів до опису семантики. Якщо програма починає працювати над деяким вихідним станом пам’яті, то всі оператори, які можуть бути виконані, виконуються, а всі оператори, які не можуть бути виконані (через невизначеність значень змінних) «складаються» в нову програму.

Подібне обчислення, яке називають *змішаним обчисленням, дозволяє* визначати різні перетворення (трансформації) програм. Конкретизуємо сказане, визначивши поняття «змішане обчислення». *Змішані обчислення* – і узагальнений спосіб виконання програм для ЕОМ чи абстрактних обчислювачів, в якому зміні піддаються як дані, оброблювані програмою, а й сама програма. *Коректними змішаними обчисленнями* називаються змішані обчислення, що перетворюють програму та її дані з даними [4, с.22].

Очевидні причини того, що цей спосіб формалізації семантики багатьма авторами книг не вважається самостійним і заслуговує на увагу: 1) часто він є частковим випадком операційної семантики (коли семантична мова, що виражає семантику конструктів мови-об'єкта, збігається з частиною); 2) для опису семантики примітивних команд можуть бути використані інші способи. Однак цей спосіб завжди використовується в таких випадках: при описі семантики макрокоманд у мовах програмування, що допускають *макровизначення:* при описі семантики мови програмування, побудованого *за принципом розширення*, коли спочатку на комп'ютері реалізується якесь ядро мови (набір примітивних команд), а потім всі команди, що вводяться далі, описуються через послідовності примітивних.

Використання статичної семантики пов'язані з тим, що є деякі характеристики структури мов програмування, описати складно, інколи просто неможливо. Звичне програмістам, правило, яке вимагає обов'язкового оголошення змінних, не може бути визначене.

Статична семантика мови програмування – це різновид семантики мови, що містить *контекстні умови,* які описують для кожного конструкту програми (імені, ключового слова, оператора і т. д.) той *контекст,* в якому певний конструкт має право бути вжитим (всі можливі поєднання конструктів коректної програми, серед яких може бути даний). Семантика мови називається *«статичною»* тому, що аналіз, необхідний для перевірки контекстних умов, може бути виконаний у процесі компіляції програми.

Отже, статична семантика мови програмування переважно пов'язана з синтаксисом програм. Контекстні умови добре описуються спеціальним класом розширених формальних граматик, які називаються атрибутними граматиками і дуже часто зустрічаються в посібниках з проєктування трансляторів.

У перспективі основним завданням дослідження стає впровадження сучасних мов програмування в широкому застосуванні створення програмних продуктів, при якому педагоги повинні враховувати обов'язкові елементи: людські фактори; зміст (контент); технологічні можливості сучасних персональних комп'ютерів.

ЛІТЕРАТУРА

1. Андон П. І., Лавріщева К. М. Розвиток фабрик програм в інформаційному світі. Вісник НАН України. 2010. № 10. С. 15–41.
2. Аронов А.О., Дзюбенко А.І. Підхід до створення студентської фабрики програм. Проблеми програмування. 2011. № 3. С. 42–49.
3. Базурін В. М. Порівняльний аналіз середовищ програмування мовою Python. Новітні комп'ютерні технології. Кривий Ріг: Видавничий центр ДВНЗ «Криворізький національний університет», 2018. Том XVI. С. 281–292.
4. Глушков В. М. Фундаментальные основы и технология программирования. Программирование. 1980. № 2. С. 13–24.
5. Баранюк О. Пошук шляхів підвищення ефективності вивчення мови асемблера. Наукові записки. Серія: проблеми методики фізико-математичної і технологічної освіти. Кіровоград: РВВ КДПУ ім. В. Винниченка, 2011. Вип. 2. С. 18–26.
6. Лавріщева К. М., Коваль Г. І., Бабенко Л. П. та ін. Нові теоретичні засади технології виробництва сімейств програмних систем у контексті генерувального програмування. Київ: ІПС НАНУ, 2011. 377 с.
7. Федусенко О. В., Федусенко А. О., Доманецька І. М. Концептуальна модель адаптивної інформаційної системи навчання. Інформаційні технології управління. 2017. № 32. С. 86–90.
8. Цюцюра М.І., Єрукаєв А.В. Застосування генетичного алгоритму для формування функції належності нечітких множин. Інформаційні технології управління. 2018. № 36. С. 71–75.
9. Chang C.C. A division algorithm using bisection method in residue number system. International Journal of Computer, Consumer and Control. N 1. 2013. P. 59–66.
10. Porter R. Design Patterns in Learning to Program: A thesis … for the degree of Doctor of Philosophy. Adelaide, 2006.
11. Winslow L. Programming Pedagogy: A Psychological Overview. SIGCSE Bulletin. 1996. Vol. 28. No. 3. P. 17–22.

REFERENCES

1. Andon P. I., Lavrishcheva K. M. Rozvytok fabryk prohram v informatsiinomu sviti [Development of program factories in the information world]. Visnyk NAN Ukrainy. 2010. № 10. S. 15–41.
2. Aronov A. O., Dziubenko A. I. Pidkhid do stvorennia studentskoi fabryky prohram [An approach to creating a student program factory]. Problemy prohramuvannia. 2011. № 3. S. 42–49.

3. Bazurin V. M. Porivnialnyi analiz seredovyshch prohramuvannia movoiu Python [Comparative analysis of Python programming environments]. Novitni kompiuterni tekhnolohii. Kryvyi Rih: Vydavnychyi tsentr DVNZ «Kryvorizkyi natsionalnyi universytet». T. XVI. 2018. S. 281–292.
4. Hlushkov V. M. Fundamentalnie osnovi y tekhnolohyia prohrammyrovanyia [Fundamental principles and technology of programming]. Prohrammyrovanye. 1980. № 2. S. 13–24.
5. Baraniuk O. Poshuk shliakhiv pidvyshchennia efektyvnosti vyvchennia movy asemblera [Finding ways to improve the efficiency of learning assembly language]. Naukovi zapysky. Seriia: problemy metodyky fizyko-matematychnoi i tekhnolohichnoi osvity. Kirovohrad: RVV KDPU im. V. Vynnychenka. 2011. Vyp. 2. S. 18–26.
6. Lavrishcheva K. M., Koval H. I., Babenko L. P. Novi teoretychni zasady tekhnolohii vyrobnytstva simeistv prohramnykh system u konteksti heneruvalnoho prohramuvannia [New theoretical foundations of technology for the production of families of software systems in the context of generative programming]. Kyiv: IPS NANU,2011. 377 s.
7. Fedusenko O. V., Fedusenko A. O., Domanetskaya I. M. Conceptual model of adaptive information system of education [Conceptual model of an adaptive information system of learning]. Informatsiini tekhnolohii upravlinnia. 2017. 32. S. 86–90.
8. Tsyutsyura M. I., Yerukaiev A. V. Application of genetic algorithm for formation of fuzzy set membership function [Application of the genetic algorithm for forming the membership function of fuzzy sets]. Informatsiini tekhnolohii upravlinnia. 2018. 36. S. 71–75.
9. Chang, C.C. & Yang, J.H., (2013). A division algorithm using bisection method in residue number system. International Journal of Computer, Consumer and Control, 1, 59–66.
10. Porter R. (2006). Design Patterns in Learning to Program: A thesis … for the degree of Doctor of Philosophy. Adelaide.
11. Winslow L. (1996). Programming Pedagogy: A Psychological Overview. SIGCSE Bulletin. Vol. 28. No. 3. 17–22.

УДК [378.016:[373.5.011.3–051:004]]:37.091.2
DOI 10.25128/2415-3605.23.1.24

ІГОР ВОЙТОВИЧ

https://orcid.org/0000–0003–2813–5225
ihor.voitovych@rshu.edu.ua
доктор педагогічних наук, професор
Рівненський державний гуманітарний університет
вул. С. Бандери 12, м. Рівне

НАТАЛІЯ ПАВЛОВА

https://orcid.org/0000–0002–7817–6781
nataliia.pavlova@rshu.edu.ua
кандидат педагогічних наук, доцент
Рівненський державний гуманітарного університет
вул. С. Бандери 12, м. Рівне

МЕТОДИКА НАВЧАННЯ ІНФОРМАТИКИ У ДИСКУРСІ ОСВІТНЬО–ПРОФЕСІЙНОЇ ПРОГРАМИ «СЕРЕДНЯ ОСВІТА (ІНФОРМАТИКА)»

Розкрито значення методики навчання інформатики у професійному становленні студентів, які здобувають кваліфікацію «вчитель інформатики» за освітньо-професійною програмою (ОПП) «Середня освіта (Інформатика)». На основі тлумачення понять «методика» і «методика навчання предмету» сформульовано авторське бачення змісту навчальної програми з дисципліни «Методика навчання інформатики». Проаналізовано освітньо-професійні програми закладів вищої освіти (ЗВО) щодо наявності згаданої дисципліни та особливостей її вивчення, зокрема, у Рівненському державному гуманітарному університеті (РДГУ). Сформульовано мету вивчення методики інформатики і конкретизовано завдання (навчальні, пізнавальні, виховні, практичні), вирішення яких сприяє здобуттю студентами обізнаності про організацію освітнього процесу з інформатики у закладах загальної середньої освіти (ЗЗСО), застосування апаратного і програмного забезпечення, розробці власних й вдосконалення існуючих інформаційно-ресурсних, навчально-дидактичних і методичних матеріалів. Зазначено, що розвиток інформаційно-комунікаційних технологій (ІКТ), технічних засобів і парадигм