УДК 378.091.21:004-057.21

АНАЛІЗ ЗАКОРДОННИХ ДОСЛІДЖЕНЬ ІЗ ПРОБЛЕМ НАВЧАННЯ МАЙБУТНІХ ІНЖЕНЕРІВ-ПРОГРАМІСТІВ ОБ'ЄКТНО-ОРІЄНТОВАНОМУ ПРОГРАМУВАННЮ

Конюхов С.Л., старший викладач
кафедри інформатики і кібернетики
Мелітопольський державний педагогічний університет імені Богдана Хмельницького

У статті наведені результати аналізу закордонного досвіду подолання проблем, які виникають у процесі формування професійної компетентності майбутніх інженерів-програмістів під час вивчення об'єктно-орієнтованого програмування. Розглядаються роботи, присвячені використанню ігрових технологій, методу проектів і програмних засобів навчального призначення. Сформульовані висновки щодо доцільності використання зазначених підходів і засобів у закладах вищої освіти України.

Ключові слова: *об'єктно-орієнтоване програмування, майбутній інженер-програміст, заклади вищої освіти, закордонний досвід, проектний підхід, візуалізація.*

В статье приведены результаты анализа зарубежного опыта преодоления проблем, возникающих в процессе формирования профессиональной компетентности будущих инженеров-программистов при изучении объектно-ориентированного программирования. Рассматриваются работы, посвященные использованию игровых технологий, метода проектов и программных средств учебного назначения. Сформулированные выводы о целесообразности использования указанных подходов и средств в учреждениях высшего образования Украины.

Ключевые слова: *объектно-ориентированное программирование, будущий инженер-программист, учреждения высшего образования, зарубежный опыт, проектный подход, визуализация.*

Koniukhov S.L. ANALYSIS OF FOREIGN STUDIES ON THE PROBLEMS OF TRAINING FUTURE SOFTWARE ENGINEERS IN OBJECT-ORIENTED PROGRAMMING

In the paper the results of analysis of foreign experience in the field of overcoming problems with forming future software engineers' professional competence in the course of object-oriented programming are presented. The studies, which are devoted to using of game technologies, project approach, and educational software, are examined. The conclusions about expediency of applying these approaches and means in Ukrainian higher educational institutions are formulated.

Key words: *object-oriented programming, future software engineer, institutions of higher education, foreign experience, project approach, visualization.*

Постановка проблеми. Об'єктно-орієнтоване програмування (далі – ООП), незважаючи на достатньо довгий термін існування і зміни, що постійно відбуваються у галузі інформаційних технологій, нині залишається найбільш поширеною методологією програмної розробки. На підтвердження цього факту можна навести рейтинги мов програмування, у яких традиційно лідирують саме об'єктно-орієнтовані мови (C++, C#, Java і деякі інші). Наприклад, за даними компанії TIOBE Software BV (https://www.tiobe.com), які отримуються шляхом систематичного моніторингу галузевих web-сайтів (вакансій працевлаштування, навчальних курсів, допоміжних матеріалів тощо), зазначені мови протягом останніх 10 років постійно входять до першої десятки найбільш популярних мов програмування [1]. Аналогічні результати представлені у рейтингу, складеному на основі опитування українських програмістів у 2018 р. [2]. Отже, використання цих мов як базових для вивчення програмування у процесі професійної підготовки майбутніх інженерів-програмістів у закладах вищої освіти (далі – ЗВО) України і світу є обґрунтованим і доцільним.

Результати прикладних досліджень вітчизняних (І. Бернакевич, В. Бублика, П. Вагіна, П. Кравця, М. Львова, Т. Рака, О. Співаковського й ін.) та іноземних (Т. Бенайа (T. Benaya), М. Бен-Арі (M. Ben-Ari), М. Бергеса (M. Berges), Е. Зур (E. Zur), Н. Ліберман (N. Liberman), Н. Рагоніса (N. Ragonis), П. Хабвейзера (P. Hubwieser) та ін.) науковців свідчать про наявність спільних проблем у процесі формування у студентів здатності до використання об'єктно-орієнтованого підходу під час розробки програм. До таких викликів належать, зокрема, складність опанування понять ООП у процесі роботи у промислових інтегрованих середовищах розробки (англ. Integrated Development Environment, IDE) (оскільки потрібно одночасно вивчати засоби ООП і особливос-

ті IDE), слабкий зв'язок між програмними засобами навчального призначення (Blue J, Greenfoot, Raptor та ін.) і професійними IDE [3]. Окрім того, виконуючи навчальні завдання і проекти, студенти зустрічаються з багатьма труднощами реального програмування: від складності ООП до необхідності роботи у команді і дотримання встановлених термінів завершення окремих етапів і повного проекту [4, с. 190]. Таким чином, залишається актуальним завдання щодо пошуку і впровадження нових методів і засобів формування у майбутніх інженерів-програмістів розуміння фундаментальних положень об'єктно-орієнтованої парадигми, а також здатності і готовності до їх практичного застосування. Одним із методів виконання зазначеного завдання є вивчення і аналіз прогресивного закордонного досвіду вивчення ООП у ЗВО.

Аналіз останніх досліджень і публікацій. Професійна підготовка майбутніх інженерів-програмістів у ЗВО є предметом вивчення багатьох вітчизняних науковців, зокрема: А. Власюка, Т. Гончаренко, П. Грицюка, І. Мендзебровського, Т. Морозової, В. Осадчого, В. Седова, С. Семерікова, Я. Сікори, А. Стрюка й ін.

Закордонний досвід навчання програмістів і фахівців з інформаційних технологій, а також можливості його використання у ЗВО України досліджували В.С. Круглик (підготовка майбутніх інженерів-програмістів у провідних іноземних закладах вищої освіти), І. Пододіменко (підготовка бакалаврів комп'ютерних наук в університетах Японії), Р. Шаран (професійна підготовка магістрів інформаційних технологій у США). Зокрема, докладний аналіз програм підготовки ІТ-фахівців в університетах Сполучених Штатів Америки, Канади, Великої Британії, Франції, Німеччини, Китаю виконав В.С. Круглик [5]. На цій основі він сформулював рекомендації щодо удосконалення системи професійної підготовки майбутніх інженерів-програмістів у ЗВО України.

Ґрунтовні дослідження, виконані цими науковцями, характеризують національні системи ІТ-освіти у їхній цілісності з урахуванням освітніх політик, прийнятих в окремих університетах. Водночас аналіз підходів закордонних викладачів до подолання проблем, які виникають у процесі вивчення об'єктно-орієнтованого підходу, практично не виконувався.

Постановка завдання. Метою статті є аналіз іноземних досліджень, присвячених вирішенню проблем, які виникають у процесі формування професійної компетентності майбутніх інженерів-програмістів під час вивчення ООП, а також формулювання висновків щодо можливості впровадження цього досвіду у ЗВО України.

Виклад основного матеріалу дослідження. Вивчення досвіду роботи іноземних викладачів дозволяє більш чітко визначити проблемні місця професійної підготовки майбутніх інженерів-програмістів, оскільки передбачає абстрагування від особливостей вітчизняної системи вищої освіти.

З метою подолання труднощів, які виникають у майбутніх програмістів під час вивчення ООП, у закордонних ЗВО застосовуються різноманітні програмні засоби навчального призначення (засоби статичної та динамічної візуалізації, засоби оцінювання навчальних результатів студентів, засоби програмної підтримки вивчення окремих концепцій ООП, мікросвіти й ін.) [3; 6], ігрові методи (для формування розуміння фундаментальних понять об'єктно-орієнтованого підходу) [7–9], різні варіації методу проектів (для формування у студентів здатності до практичного використання засобів ООП) [4; 10] тощо.

Закордонний досвід використання програмних засобів навчального призначення у процесі вивчення об'єктно-орієнтованого підходу.

Докладний аналіз програмних засобів навчального призначення, які використовуються у процесі вивчення ООП, виконали Y. Hosanee і S. Panchoo [3]. Спираючись на результати вивчення наукових робіт і власний досвід застосування зазначених середовищ, дослідники називають важливі інструменти, які мають входити до складу таких програмних засобів, а саме: засоби побудови наративу (користувач вивчає ООП, слідкуючи за історією, яка розгортається, результатом має бути його здатність створити власну історію); інструменти статичної (відображують внутрішню структуру класів) і динамічної (відображують стан і поведінку класів і об'єктів під час виконання програми) візуалізації; засоби моделювання зв'язків між класами і об'єктами та їхньої поведінки (діаграми класів, діаграми UML, блок-схеми тощо); засоби повідомлення інформації користувачу найбільш зручним для нього способом; надання користувачу доступу лише до тих функцій програми, які відповідають його поточному рівню підготовки; підтримка декількох мов програмування; легкий у використанні інтерфейс [3, с. 2–3]. Автори стверджують, що жоден із розглянутих ними програмних засобів (Alice, BlueJ, JPIE, Greenfoot, Jeeroo, Scratch, Baltie, Jhave Raptor) не надає зазначені інструменти у повному обсязі, і пропонують власний програмний про-

дукт, призначений для надання допомоги студентам у процесі вивчення спадкування, асоціації класів і поліморфізму.

Вважаємо, що використання програмних засобів, розглянутих у роботі Y. Hosanee і S. Panchoo, може бути достатньо ефективним на початковому етапі вивчення ООП, оскільки робить цей процес більш цікавим для студентів.

Закордонний досвід використання ігрових навчальних засобів у процесі вивчення об'єктно-орієнтованого підходу.

Закордонні дослідники вважають ігрові методи і засоби ефективним засобом навчання майбутніх програмістів, оскільки вони дозволяють підтримувати інтерес і мотивацію студентів до вивчення програмування і ООП, допомагають наочно пояснити окремі концепції об'єктно-орієнтованої парадигми, сприяють запам'ятовуванню конструкцій мов програмування тощо. У зв'язку з цим в іноземних університетах виконуються роботи з удосконалення наявних і впровадження нових ігрових програмних засобів.

Наприклад, в університеті Вінстон-Сейлем (США) розроблений ігроподібний модуль Java Ninja, який дозволяє продемонструвати особливості реалізації та правильного використання батьківських класів і класів-нащадків, а також перевантаження методів у мові програмування Java. Модуль призначений для надання допомоги студентам під час вивчення такого поняття ООП, як спадкування. Завдання користувача полягає у тому, щоб створити декілька типів ігрового персонажа з різними можливостями і здобути перемогу над супротивником, керуючи цими об'єктами. Більшість задач у модулі виконуються за допомогою механізму «drag and drop», а також з використанням мови UML для опису відносин між батьківськими і похідними класами [7].

Особливостям використання комп'ютерних ігор у процесі професійної підготовки програмістів присвячена робота [8]. Її автори, T. Jordine, Y. Liang й E. Ihler, роблять акцент на вивченні об'єктно-орієнтованої мови програмування Java. Дослідники виконали аналіз стану застосування комп'ютерних ігор у процесі навчання програмування і виділили декілька суттєвих проблем, а саме: відсутність ігор для вивчення Java на мобільних пристроях; недостатня якість системи допомоги (теоретичні відомості, підказки та ін.); недостатня реалізація класичних ігрових елементів (максимальний рахунок, рейтинг гравців, нагороди тощо), що знижує інтерес студентів; більшість ігор із програмування не мають засобів для створення спеціальних демонстраційних фрагментів для лекцій [8, с. 55]. З метою подолання зазначених проблем науковці розробили і впровадили у навчальний процес мобільну гру для вивчення мови Java. У процесі гри користувач повинен складати програмний код для виконання поставленого завдання, яке передбачає створення класів із визначеним переліком властивостей і методів, а також екземплярів цих класів. Серед особливостей гри можна виділити: спеціальна клавіатура для введення коду (користувач вибирає необхідні слова із наведеного переліку), синтаксична перевірка коду, перевірка на відповідність коду поставленому завданню, система допомоги [8, с. 56]. Студенти можуть використовувати гру у будь-який час для тренування навичок програмування мовою Java, а викладачі на заняттях – для демонстрації окремих теоретичних і практичних питань.

Дослідниками з університету Боуі (США) розроблений програмний засіб для вивчення понять «спадкування», «поліморфізм» й «інкапсуляція», який складається з трьох окремих ігор. Для демонстрації спадкування гравцеві пропонується створити клас «транспортний засіб» на основі вихідного батьківського класу. Результат подається у вигляді UML-діаграми. Для демонстрації інкапсуляції гравець повинен правильно визначити модифікатори доступу до полів класу [9].

Вважаємо, що подібні ігри можуть достатньо ефективно використовуватися на початковому етапі вивчення ООП як допоміжні педагогічні засоби. Застосовувати їх слід дозовано, уникаючи надмірного захоплення ігровими ситуаціями і завданнями. Цікавим педагогічним прийомом є також розробка студентами комп'ютерних ігор для вивчення окремих понять об'єктно-орієнтованої парадигми. Його доцільно використовувати на старших курсах з метою закріплення і поглиблення у майбутніх інженерів-програмістів знань з ООП, а також формування у них здатності демонструвати і пояснювати принципи ООП.

Закордонний досвід реалізації проектного підходу у процесі вивчення ООП.

Дослідники [4; 10] стверджують, і ми погоджуємося з ними, що виконання студентами проектів наближає процес навчання до реального процесу розробки програмного забезпечення.

Автори роботи [4], S. Zschaler, B. Demuth і L. Schmitz, зазначають, що під час вивчення ООП відбувається формування у студентів об'єктно-орієнтованого стилю мислення. Це достатньо тривалий процес, пов'язаний із численними складнощами, оскільки вони повинні перейти від використання алгоритмічного підходу до

об'єктного, навчитися розв'язувати поставлені завдання шляхом створення систем взаємодіючих об'єктів, використовувати наявні класи замість створення нових, організовувати гнучкий процес розробки програм із дотриманням етапів життєвого циклу програм, користуватися шаблонами розробки [4, с. 190]. Для подолання цих труднощів дослідники використовують двоступеневий підхід, коли курс програмування складається з лекційної та проектної частин приблизно однакової тривалості. Виконання проектів дослідники пропонують здійснювати на основі спеціально створеного Java-фреймворку Salespoint (http://www.salespoint-framework.org/), який надає засоби для розробки застосувань для сфери фінансів і торгівлі. Фреймворк містить класи для управління даними і структурою торгових додатків, компоненти для створення інтерфейсу користувача, засоби керування обліковими записами користувачів та ін. Завдяки цьому викладач отримує можливість змоделювати реальний процес промислової розробки програмних проектів, а студенти – взяти участь у ньому як професійні програмісти.

Автори обґрунтовують доцільність такого підходу, зазначаючи, що: 1) на IT-підприємствах початківці здебільшого залучаються до виконання діючих проектів, що дозволяє поєднати виконання функціональних обов'язків із набуттям досвіду застосування певних засобів розробки; 2) викладачі – керівники проектів можуть сформулювати тематику, забезпечуючи різні рівні складності та можливість порівняння отриманих результатів (для подальшого прозорого оцінювання навчальних досягнень); 3) завдання студентів дещо полегшується завдяки можливості використання визначеної архітектури застосувань і готових компонентів; 4) використовуючи наявні шаблони проектування, початківці мають змогу опанувати гарний стиль розробки програм [4, с. 190].

Дещо інший спосіб реалізації проектного підходу пропонують автори роботи [10], а саме: програма вивчення ООП (мова C++) передбачає самостійне виконання студентами декількох послідовних проектів, складність яких поступово зростає. Дослідники підкреслюють, що головна увага у курсі приділяється формуванню у майбутніх програмістів знань базових концепцій об'єктно-орієнтованого підходу, а також навичок розв'язання практичних проблем засобами ООП [10, с. 667]. Під час виконання проекту студенти повинні самостійно обрати засоби розробки, організації групової взаємодії та ін. Результатом є програма для обміну повідомленнями у режимі реального часу (месенджер) з різноманітними додатковими функціями (за вибором розробників), наприклад: передача файлів, можливість скасування / повторення передачі повідомлень, підтримка багатомовності, засоби настроювання інтерфейсу користувача, можливість запису повідомлень та ін. Автори стверджують, що така практика допоможе студентам оволодіти навичками застосування ООП у процесі створення великих програмних проектів [10, с. 670].

Отже, проектний підхід користується значною популярністю як метод професійної підготовки майбутніх програмістів у закордонних університетах: у навчальному процесі використовуються як невеликі проекти, що охоплюють декілька лабораторних занять, так і великі (семестрові і випускні кваліфікаційні) проекти. Аналізуючи наявні дослідження, можна знайти цікаві ідеї для визначення тематики, а також форми виконання таких проектів.

Висновки з проведеного дослідження. Вивчення досвіду досліджень іноземних науковців з питань удосконалення професійної підготовки майбутніх програмістів дозволяє поглибити розуміння наявних проблем, виявити нові підходи до їх розв'язання, порівняти форми, методи і засоби навчання. У статті нами проаналізовано декілька робіт, присвячених різним аспектам вивчення об'єктно-орієнтованого підходу у закордонних ЗВО, а саме: використанню ігрових технологій, методу проектів і програмних засобів навчального призначення. За результатами аналізу можна зробити такі основні висновки: 1) об'єктно-орієнтований підхід залишається достатньо складним і для студентів (які мають опанувати здатність до реалізації концепцій ООП засобами різних мов програмування), і для викладачів (діяльність яких спрямована на формування у студентів цієї здатності); 2) проблемам вивчення ООП присвячено багато публікацій у закордонних наукових періодичних виданнях і, незважаючи на тривалий період викладання ООП, актуальність досліджень у цьому напрямку не спадає; 3) в іноземних університетах впроваджуються різноманітні програмні засоби, спрямовані на надання допомоги студентам у процесі вивчення фундаментальних понять ООП, зокрема, велика увага приділяється їх візуальному поданню; 4) невід'ємною частиною навчання є виконання проектів, що дозволяє наблизити його до реальної практики розробки програмного забезпечення.

Загалом, зазначені напрями діяльності притаманні і вітчизняній освіті майбутніх інженерів-програмістів, але вважаємо до-

цільним підвищити роль проектного підходу, що дозволить не лише забезпечити формування у студентів здатності до використання ООП, а й професійної компетентності у її комплексі, включаючи здатність до розробки і підтримки програмного забезпечення на усіх етапах життєвого циклу, а також особисті якості, важливі для успішного здійснення діяльності за фахом. Окрім того, для надання допомоги студентам, які мають труднощі у процесі вивчення ООП, на початкових етапах доцільно застосовувати орієнтовані на навчальний процес середовища розробки (наприклад, BlueJ), які є легшими у використанні, ніж промислові IDE.

ЛІТЕРАТУРА:

1. TIOBE Index for May 2018. URL: https://www.tiobe.com/tiobe-index//.

2. Шевченко Р. Рейтинг языков программирования 2018: Go и TypeScript вошли в высшую лигу, Kotlin стоит воспринимать серьезно. 23 января 2018 г. URL: https://dou.ua/lenta/articles/language-rating-jan-2018/.

3. Hosanee Y., Panchoo S. An enhanced software tool to aid novices in learning Object Oriented Programming (OOP). Proceedings of the 2015 International Conference on Computing, Communication and Security (ICCCS). 2015. P. 1–7. DOI: https://doi.org/10.1109/CCCS.2015.7374197.

4. Zschaler, S., Demuth, B., Schmitz, L. (2014). Salespoint: A Java framework for teaching object-oriented software development. Science of Computer Programming. 2014. Vol. 79. P. 189–203. DOI: https://doi.org/10.1016/j.scico.2012.04.005.

5. Круглик В.С. Система підготовки майбутніх інженерів-програмістів до професійної діяльності у вищих навчальних закладах: монографія. Мелітополь: МДПУ ім. Б. Хмельницького, 2017. 384 с.

6. Pears A., Seidman S., Malmi L., Mannila L., Adams E., Bennedsen J., Devlin M., Paterson J. A survey of literature on the teaching of introductory programming. Working group reports on ITiCSE on Innovation and technology in computer science education (ITiCSE-WGR 07). Janet Carter and June Amillo (Eds.). New York: ACM, 2007. P. 204–223. DOI: https://doi.org/10.1145/1345443.1345441.

7. Zhang J., Caldwell E.R., Smith E. Learning the concept of Java inheritance in a game. Proceedings of the 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES 2013). 2013. P. 212–216. DOI: https://doi.org/10.1109/ CGames.2013.6632635.

8. Jordine T., Liang Y., Ihler E. A mobile-device based serious gaming approach for teaching and learning Java programming. International journal of interactive mobile technologies. 2015. Vol. 9. № 1. P. 53–59. DOI: http://dx.doi.org/10.3991/ijim.v9i1.4380.

9. Sharma S., Stigall J., Rajeev S. Game-theme based instructional module for teaching object oriented programming. Proceedings of the 2015 International Conference on Computational Science and Computational Intelligence (CSCI). 2015. P. 252–257. DOI: https://doi.org/10.1109/CSCI.2015.35.

10. Chen Y.-L., Chiang C.-Y., Huang Y.-P., Yuan S.-M. A project-based curriculum for teaching C++ object-oriented programming. Proceedings of the 9th International Conference on Ubiquitous Intelligence and Computing and 9th International Conference on Autonomic and Trusted Computing (UIC-ATC 2012). 2012. P. 667–672. DOI: https://doi.org/10.1109/UIC-ATC.2012.94.