УДК 37.018.1(477-65-21)«18/19» DOI https://doi.org/10.26661/2786-5622-2023-3-22

ДОСЛІДЖЕННЯ ПРОБЛЕМ ОЦІНЮВАННЯ ЗНАНЬ ІЗ ПРОГРАМУВАННЯ В УМОВАХ ДИСТАНЦІЙНОГО НАВЧАННЯ НА ПЛАТФОРМІ MOODLE

Славко Г. В.
кандидат технічних наук, доцент,
доцент кафедри інформатики і вищої математики
Кременчуцький національний університет імені Михайла Остроградського
вул. Університетська, 20, Кременчук, Полтавська область, Україна
orcid.org/0000-0001-5821-4587
gvslavko@gmail.com

Сергієнко С. А.
кандидат технічних наук, доцент,
доцент кафедри систем автоматичного управління і електроприводу
Кременчуцький національний університет імені Михайла Остроградського
вул. Університетська, 20, Кременчук, Полтавська область, Україна
orcid.org/0000-0002-3977-5239
serhiy.serhiyenko@gmail.com

Григорова Т. А.
кандидат технічних наук, доцент,
доцент кафедри інформатики і вищої математики
Кременчуцький національний університет імені Михайла Остроградського
вул. Університетська, 20, Кременчук, Полтавська область, Україна
orcid.org/0000-0002-4371-8624
grital0403@gmail.com

Кирилаха Н. Г.
кандидат фізико-математичних наук,
доцент кафедри інформатики і вищої математики
Кременчуцький національний університет імені Михайла Остроградського
вул. Університетська, 20, Кременчук, Полтавська область, Україна
orcid.org/0000-0002-2629-8867
natalykiril582@gmail.com

Ключові слова: *інформатика, викладач інформатики, тестування, класифікація задач з програмування, онлайн-навчання, методики навчання, технологія та методика дистанційного навчання, веб-програмування.*

У статті досліджуються проблеми, пов'язані з оцінюванням знань студентів з програмування під час дистанційного навчання та можливі шляхи їх вирішення. Розглянуті такі проблеми: ускладнення оцінювання у динаміці, шахрайство під час дистанційного оцінювання, обмеженість взаємодії та зворотного зв'язку, нерівні можливості доступу студентів до ресурсів і технологій, ускладнення оцінювання навичок розв'язання практичних завдань. Актуальність роботи обумовлена збільшенням навантаження на викладача під час дистанційного навчання та ускладненням якісного оцінювання учнів через відсутність безпосереднього спілкування. Проведені у роботі дослідження базуються на можливостях платформи Moodle для онлайн перевірки отриманих учнями знань. Аналіз статистики використання різних типів тестових завдань Moodle свідчить, що більшість викладачів віддають перевагу найпростішим типам тестів множинного вибору з можливістю перемішування питань та

варіантів відповідей. Такий вибір не сприяє якісному оцінюванню знань з програмування. У статті розглядаються можливості автоматичного створення тестів та їх ефективність для підвищення якості оцінювання та запобігання шахрайства під час тестування, наведені приклади створення тестових питань. В роботі запропоновано класифікацію завдань та тестових питань для базового курсу програмування за типами. Наведено результати педагогічних експериментів з визначення відсоткової складності різних типів завдань зазначеної класифікації під час викладання програмування у Кременчуцькому національному університеті імені Михайла Остроградського з використанням системи онлайн навчання krnu.org, побудованої на платформі Moodle. У статті обґрунтовується ефективність використання викладачами запропонованої класифікації завдань задля підвищення якості та об'єктивності оцінювання отриманих навичок та компетенцій під час вивчення базового курсу програмування. У висновках статті зазначається, що дослідження удосконалення засобів оцінювання під час дистанційного навчання залишаються актуальними та перспективними на тлі подальшого розвитку цифрової освіти.

INVESTIGATION OF THE PROBLEMS OF PROGRAMMING KNOWLEDGE ASSESSING IN THE CONTEXT OF DISTANCE LEARNING ON THE MOODLE PLATFORM

Slavko G. V.
Candidate of Technical Sciences, Associate Professor,
Associate Professor at the Department of Informatics and Higher Mathematics
Kremenchuk Mykhailo Ostrohradskyi National University
University str., 20, Kremenchuk, Poltava region, Ukraine
orcid.org/0000-0001-5821-4587
gvslavko@gmail.com

Serhiienko S. A.
Candidate of Technical Sciences, Associate Professor,
Associate Professor at the Department of Automatic Control Systems and Electric Drive
Kremenchuk Mykhailo Ostrohradskyi National University
University str., 20, Kremenchuk, Poltava region, Ukraine
orcid.org/0000-0002-3977-5239
serhiy.serhiyenko@gmail.com

Hryhorova T. A.
Candidate of Technical Sciences, Associate Professor,
Associate Professor at the Department of Informatics and Higher Mathematics
Kremenchuk Mykhailo Ostrohradskyi National University
University str., 20, Kremenchuk, Poltava region, Ukraine
orcid.org/0000-0002-4371-8624
grital0403@gmail.com

Kyrylakha N. G.
Candidate of Physico-Mathematical Sciences,
Associate Professor at the Department of Informatics and Higher Mathematics
Kremenchuk Mykhailo Ostrohradskyi National University
University str., 20, Kremenchuk, Poltava region, Ukraine
orcid.org/0000-0002-2629-8867
natalykiril582@gmail.com

Key words: *computer science, computer science teacher, testing, classification of programming tasks, online learning, teaching methods, technology and methods of distance learning, web programming.*

The article investigates the problems related to assessing students' knowledge of programming in distance learning and possible ways to solve them. The following problems are considered: the complexity of assessment in the dynamics, fraud during distance assessment, limited interaction and feedback, unequal opportunities for students to access resources and technologies, and the complexity of assessing practical problem-solving skills. The relevance of the work is due to the increased workload of the teacher in distance learning and the difficulty of qualitative assessment of students due to the lack of direct communication. The research conducted in this paper is based on the capabilities of the Moodle platform for online testing of students' knowledge. The analysis of statistics on the use of different types of Moodle tests shows that most teachers prefer the simplest types of multiple-choice tests with the possibility of mixing questions and answer options. Such a choice does not contribute to the qualitative assessment of programming knowledge. The article investigates the possibilities of automatic test creation and their effectiveness in improving the quality of assessment and preventing fraud during testing, and provides examples of test questions creation. The paper proposes a classification of tasks and test questions for a basic programming course by type. The results of pedagogical experiments of determining the percentage difficulty of different types of tasks of this classification during teaching programming at the Kremenchuk Mykhailo Ostrohradskyi National University using the online learning system krnu.org, built on the Moodle platform, are presented. The article substantiates the effectiveness of using the proposed classification of tasks by teachers to improve the quality and objectivity of assessing the acquired skills and competencies during studying a basic programming course. The article concludes that research on improving assessment tools in distance learning remains relevant and perspective against the background of further development of digital education.

Постановка проблеми. В умовах цифровізації освітнього процесу через виклики сьогодення спостерігається зростання числа онлайн-платформ, які забезпечують взаємодію студента і викладача. Основні функції таких платформ полягають у встановленні зручної комунікації викладача і класу під час лекційних занять (зв'язок: один до багатьох); забезпеченні індивідуального синхронного та асинхронного спілкування (зв'язок: один до одного) між викладачем і учнями, а також між учнями; сприянні зручної роздачі навчального контенту та реалізації перевірки отриманих знань. Основним способом масової перевірки засвоєння навчального матеріалу у таких платформах є тестування. Однак, слід зазначити, що під час вивчення дисциплін, пов'язаних з програмуванням, складання якісних тестових завдань для перевірки отриманих знань потребує значних витрат часу та креативності. Наповнення тестів питаннями чи завданнями має визначати не тільки знання синтаксису мови програмування та теоретичних основ складання програм за алгоритмом, а й практичні навички, набуті учнем чи студентом. Більшість же тестових питань, які зазвичай пропонуються викладачами, спрямована саме на теорію. Тому виникає необхідність дослідити можливості звичайного тестування для визначення рівня практичних навичок учня щодо напи-

сання дієздатних програм мовою програмування для реалізації певного алгоритму.

Аналіз останніх досліджень і публікацій. На тлі дискусій про доцільність упровадження викладання програмування у шкільній програмі з інформатики [3] дослідження у галузі підготовки майбутніх програмістів починаючи зі школи на базовому рівні з продовженням навчання у закладах вищої освіти набувають значущості. Такі дослідження ведуться низкою науковців, зокрема, значну увагу методиці та наповненню програм навчання програмуванню приділяють О. Семеніхіна [9], В. Биков [1], О. Спірін, Ю. Тріус. Якість професійної підготовки програмістів досліджують А. Стрюк, К. Осадча, Ю. Руденко [8], Л. Ібрагімова [2]. Також значна увага науковцями приділяється інструментальному [1; 6; 7] та ресурсному забезпеченню [10] навчання програмуванню, а також освітнім платформам [11; 17; 19] для забезпечення дистанційного навчання. Досліджуються також парадигми [16], методика [5] та теоретичні засади застосування інтегрованих уроків з математичним спрямуванням навчання програмуванню та алгоритмізації [4; 15]. Проте вимушене переведення у дистанційний формат навчання у школах та закладах вищої освіти через виклики сьогодення потребує зміни форматів не тільки у подачі навчального контенту, але й в оцінюванні отриманих знань, що особливо набуває значущості для навчального контенту, який вимагає від учнів та студентів опанування практичних навичок та компетенцій. Ефективність навчання програмуванню передбачає уміння учнем написання дієздатного програмного коду і потребує від викладача не тільки виставлення оцінки, але й виправлення помилок, яких припустився учень під час складання коду. Враховуючи багатоваріантність складання алгоритму до однієї й тієї задачі та її програмної реалізації, перевірка викладачем такого коду та його коригування потребує значних зусиль та витрат часу, що ускладнюється під час дистанційного навчання. Отже, методика перевірки робіт учнів та студентів з програмування та їх оцінювання в умовах дистанційного навчання потребує подальших досліджень та удосконалень.

Аналіз тестів, що зазвичай пропонуються викладачами програмування, показує, що більшість завдань у тестах спрямовані на перевірку знання синтаксису мов програмування та є одноманітними, а учень має змогу знайти відповідь в інтернеті якщо тестування проводиться в асинхронному режимі. Тестові завдання, що були запропоновані учням після тестування, досить часто потрапляють у вільний доступ в мережу і стають непридатними для подальшого повторного використання у інших класах. Зазначимо, що учень може вивчити синтаксис мови програмування, але так і не навчитися самостійно складати код програми, яка розв'язує певну задачу та потребує складання алгоритму. Отже, існує проблема якісної перевірки знань учнів під час дистанційного навчання з використанням тестування. Якщо ж учням пропонувати скласти код програми, що розв'язує певну задачу, яку вони потім надають на перевірку викладачу, то викладач змушений витрачати занадто багато часу на аналіз та перевірку наданого коду через багатоваріантність можливих правильних розв'язків та застосованих алгоритмів. Спростити перевірку можна запропонувавши учню пояснити свій код, але в умовах дистанційного навчання такий спосіб не можна вважати ефективним. Як варіант, у роботі [14] доповненням до звичайного тестування пропонується використовувати онлайн-засоби автоматичної перевірки складених учнем програм на додачу до звичайного тестування, але такий спосіб потребує спеціально встановленого програмного серверного забезпечення та значних ресурсів та не є доступним за замовчуванням у типових платформах, що використовуються для забезпечення дистанційного навчання та потребує додаткового фахового налаштування. Отже, існує проблема якісної перевірки знань учнів під час вивчення програмування в умовах дистанційного навчання у випадку обмежених серверних ресурсів та використання типових налаштувань платформ дистанційного навчання, які використовуються закладами освіти задля забезпечення навчального процесу.

Мета статті – дослідити проблеми оцінювання набутих учнями знань, практичного досвіду та компетенцій під час онлайн-навчання програмуванню і розглянути можливості підвищення якості та об'єктивності оцінювання шляхом автоматичного створення тестів з використанням платформи Moodle і застосування класифікації завдань та тестових питань за типами та рівнем складності.

Виклад основного матеріалу дослідження. Розглянемо тут основні проблеми, які виникають під час оцінювання знань учнів чи студентів під час дистанційного вивчення дисциплін з програмування.

Перша проблема – відсутність можливості традиційного практичного оцінювання. Програмування – це не просто запам'ятовування синтаксису та теорії, це – застосування знань для розв'язання реальних проблем. У традиційних класах викладачі можуть легко оцінити навички кодування студентів, даючи їм завдання, коригуючи їх спроби розв'язати ці завдання та спостерігаючи процес розв'язання задачі у динаміці. Дистанційне ж навчання ускладнює ефективне повторення цього

досвіду. Щоб розв'язувати цю проблему, викладачі можуть використовувати онлайн-платформи та інструменти для кодування, які дозволяють студентам писати, тестувати та надсилати код віддалено. Ці платформи часто надають зворотний зв'язок у режимі реального часу і дозволяють викладачам оцінювати вміння студентів розв'язувати проблеми та навички кодування. Але такий підхід потребує індивідуальної роботи з кожним окремим учнем, адже викладач не в змозі взаємодіяти одночасно синхронно з декількома учнями одночасно під час аналізу, перевірки чи спостереження за виконанням завдання. Тому такі дії можливі під час синхронної або асинхронної індивідуальної роботи на виконання якої має бути виділено окремі навчальні години.

Друга проблема – складність запобігання шахрайству. Дистанційне навчання викликає занепокоєння щодо академічної доброчесності, в тому числі щодо шахрайства та плагіату. На курсах програмування у студентів може виникнути спокуса копіювати код з інтернету, отримувати допомогу на спеціалізованих форумах, що ускладнює викладачам точну оцінку їхніх справжніх навичок. Для запобігання списування викладачі можуть розробляти завдання, які вимагають оригінального розв'язання конкретних проблем під безпосереднім контролем викладача. Крім того, викладачі можуть використовувати інструменти виявлення плагіату для виявлення скопійованого коду. Існує також проблема підміни учня чи студента під час тестування іншою людиною. Ідентифікація учасника тестування є непростою задачею, яка потребує додаткових зусиль від викладача [13].

Третя проблема – обмежена взаємодія та зворотний зв'язок. Програмування часто передбачає ітеративне навчання, коли студенти роблять помилки, вчаться на них і вдосконалюють свій код. В умовах дистанційного навчання брак особистої взаємодії та негайного зворотного зв'язку може перешкоджати прогресу студентів. Студентам може бути складно отримати роз'яснення або обговорити свої проблеми з кодуванням з викладачами та колегами. Для розв'язання цієї проблеми викладачі можуть запланувати регулярні віртуальні години персональних консультацій та дискусійні сесії, щоб полегшити взаємодію. Онлайн-форуми та чат-платформи можуть слугувати каналами, де студенти ставлять запитання та діляться своїм кодом з колегами та викладачами для отримання зворотного зв'язку. Надання своєчасного та конструктивного зворотного зв'язку має вирішальне значення для того, щоб допомогти студентам розвивати свої навички програмування. На жаль, планування навчального часу для дисциплін з програмування не передбачає таких додаткових годин у навантаженні. Один із дієвих інструментів платформи Moodle для розв'язання цієї проблеми – створення форуму та надання можливості у такому форумі будь-кому з учнів надавати публічні відповіді на питання тим студентам, хто цього потребує. Одночасно можна оцінювати відповіді одних студентів на питання інших.

Четверта проблема – нерівні можливості доступу студентів до ресурсів і технологій. Не всі студенти мають рівний доступ до технологій і ресурсів, необхідних для програмування під час дистанційного навчання. Відмінності в підключенні до Інтернету, апаратному та програмному забезпеченні можуть створювати значні бар'єри для навчання та оцінювання. Викладачі мають пам'ятати про різні ситуації у студентів в особливих умовах дистанційного навчання і застосовувати альтернативні варіанти оцінювання, які враховують різні рівні доступу до технологій. Це може включати дозвіл студентам виконувати оцінювання в режимі офлайн або використання простіших середовищ кодування, які споживають менше ресурсів.

П'ята проблема – оцінювання навичок розв'язання проблем. Оцінювання навичок розв'язування задач є ключовим для оцінювання рівня володіння студентами програмуванням. Однак дистанційні методи оцінювання можуть неадекватно вимірювати ці важливі навички, оскільки вони часто зосереджуються на кінцевому коді, а не на процесі мислення, що стоїть за ним. Викладачі можуть розробити методи оцінювання, які включають як завдання з кодування, так і завдання, які потребують від учня надання пояснень до коду та розв'язання проблеми. Попросивши студентів пояснити свій підхід, аргументацію та вибір того чи іншого алгоритму, викладачі можуть отримати уявлення про їхні здібності до розв'язання проблем та навички критичного мислення.

Зазначимо, що основним засобом розв'язання проблем оцінювання під час дистанційного навчання в особливих умовах, залишається тестування – як засіб масового швидкого оцінювання знань. Тому, під час складання тестових питань та завдань, викладачу слід враховувати зазначені вище проблеми й намагатися адаптувати тестування задля усунення цих проблем в оцінюванні набутих знань та компетенцій.

Як відомо, базове ядро платформи Moodle [18] надає викладачу декілька типів тестів: вибір одного правильного з поміж декількох; вибір декількох правильних з переліку варіантів відповідей; тестове завдання «вірно-невірно», питання на відповідність; питання з перетягуванням слів чи маркерів для встановлення на графічні об'єкти; питання зі складанням текстової відповіді; питання з числовою відповіддю. Зазвичай,

як показує статистика та досвід використання системи онлайн-навчання krnu.org Кременчуцького національного університету імені Михайла Остроградського [12], більшість викладачів під час складання тестів надає перевагу найпростішим типам множинного вибору з можливістю перемішування питань та варіантів відповідей. Для якісного оцінювання знань з програмування таких типів тестових питань не достатньо. Питання, які містять варіанти відповідей можуть застосовуватися ефективно здебільшого для перевірки знання синтаксису та семантики мови програмування.

Однак навіть застосовуючи тестові питання з множинним вибором відповіді можна усе ж складати тестові питання, які потребують від учня не формального аналізу завдання. Наведемо такий приклад тестового питання (рис. 1). Зазначимо, що наступний і подальші приклади коду наводяться мовою C++ на прикладах з платформи Moodle.

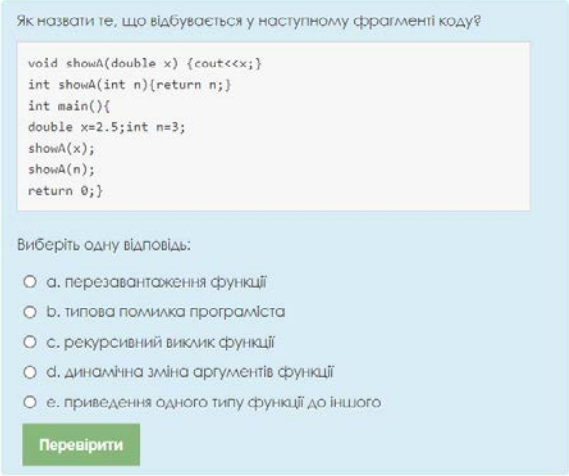

Рис. 1. Завдання тесту потребує аналізу коду C++

Для вибору правильної відповіді учню доведеться принаймні проаналізувати наведений код. Проте проблема залишається: учень має можливість зробити копію тестового завдання, викласти його в мережі чи в чаті класу, чи групи або режимі реального часу під час тестування поставити питання штучному інтелекту ChatGPT.

Завдання можна ускладнити, запропонувавши як питання код програми та створивши тест з числовою відповіддю (рис. 2), яку учень має ввести після аналізування коду програми та виконання відповідних обчислень. Особливість цього завдання полягає у необхідності обчислювати результат з певною точністю.

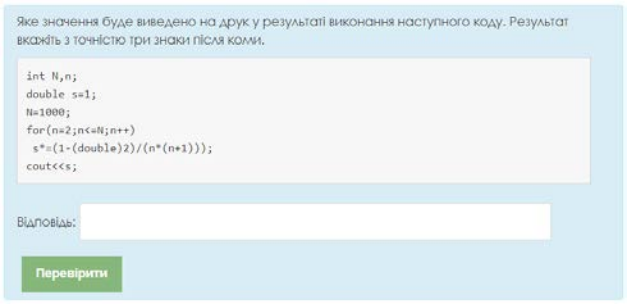

Рис. 2. Тестове завдання з числовою відповіддю

Однак, виникає інша проблема – учень може скопіювати код, запустити компілятор і отримати результат. Певною мірою це не так уже й погано, принаймні учень навчиться виконувати готовий код програми у компіляторі, але бажано аби учень провів обчислення уручну, виконуючи код так якби це робив би компілятор.

Так, можна подати код у тестовому завданні як скрин аби ускладнити виконання коду. Але усе ж код учнем може бути набрано уручну для виконання. Тому пропонується код подавати як текст, але використовувати спеціальні символи, які спотворюють код так, що звичайне копіювання [Ctrl+C] та [Ctrl+V] унеможливлює його виконання тому, що код містить приховані помилки та компілятор виводитиме повідомлення про помилки. Візуально код виглядає таким яким він і має бути, але на рівні символів коду він містить неприпустимі для компілятора символи. Учню доведеться в умовах обмеженого часу на завдання аналізувати код самостійно без застосування компілятора. Інший варіант – подавати у завданні лише фрагмент коду так, що учню для компіляції доведеться дописати необхідні рядки і якщо він це зробить, то це уже є свідченням того, що учень має відповідні знання і розуміє, що код потребує певних доповнень.

З усім тим проблема оприлюднення учнем правильної відповіді залишається. Для розв'язання цієї проблеми існує принаймні два способи. Перший – взяти за основу базовий приклад коду і змінюючи у ньому параметри (у нашому прикладі це значення n,N,s) створити комплект однотипних тестових завдань, але з різними початковими даними й відповідями та обирати питання випадково для кожного учня, що проходить тестування. Цей спосіб потребує уважності викладача під час складання комплекту тестів та значних витрат часу. Другий спосіб – скористатися можливостями Moodle, а саме – створити розрахункове питання з варіантами завдань. Цей тип тестів дозволяє вбудувати у текст завдання (код програми) параметри, які будуть вставлятися у завдання на визначені місця випадковим чином, а відповідь буде обчислюватися системою відповідно до отри-

Таблиця 1

Класифікація типів завдань з програмування

№	Тип завдання	Опис завдання	Приклад завдання чи пояснення
1	Пошук синтаксичної помилки у коді	Наводиться фрагмент коду, що містить синтаксичну помилку, яку слід знайти.	Пропонується декілька фрагментів коду, з яких один чи декілька містять помилку.
2	Пошук помилки логіки у алгоритмі	Надається умова задачі, яка має бути розв'язана та фрагмент коду до неї, що містить помилку логіки.	Такі помилки часто виникають у вкладених розгалуженнях або у циклах.
3	Доповнення фрагменту коду	Вказано задачу, яку має розв'язувати код, але код не повний та потребує доповнення.	Доцільно пропускати код з описом типу змінних, їх ініціалізацією, під'єднанням потрібних бібліотек.
4	Визначення задачі, яку розв'язує код	Наведено код і за кодом слід визначити що цей код виводить як результат.	Такі задачі доводиться розв'язувати аналізуючи чужий код.
5	Спрощення коду	Пропонується код програми, який слід спростити.	Пропонується зменшити кількість рядків коду.
6	Оптимізація коду	Дано код, змінити його так, аби він виконувався швидше та зменшив витрати пам'яті.	Може бути встановлено верхню межу на час виконання коду чи пам'ять.
7	Задачі з обмеженнями	Пропонується проста типова задача, але встановлюються певні заборони чи обмеження.	Наприклад, забороняються умовні оператори або цикли, обмежується кількість змінних чи їх типи.
8	Переклад з іншої мови програмування	Є певна задача і готовий код але написаний іншою мовою програмування. Пропонується переписати код мовою, що вивчається.	Є сенс застосовувати коли вивчається друга мова програмування і раніше студенти писали код до тих же самих задач.
9	Зміна парадигми	Наведено код деякої програми для розв'язання певної задачі. Слід переписати код, змінивши стиль програмування.	Рекурсивну реалізацію замінити ітеративною чи навпаки. Застосувати власні функції у коді, чи засоби ООП.
10	Написання власної функції	Пропонується замінити деяку стандартну вбудовану функцію написаною власною функцією.	Напишіть функцію, яка обчислює значення синуса, модуля числа, квадратного кореня.
11	Обчислювальні задачі	Результат виконання коду має бути числовим з визначеною точністю. Числову відповідь має бути складно отримати уручну.	Як приклад, можна розглядати задачу із знаходження нескінченної суми ряда із визначеною кількістю доданків ряду.
12	Складання алгоритму за даним кодом	Студент отримує код програми і має надати текстовий опис реалізованого алгоритму.	Побудова алгоритму потребує розуміння, яку задачу розв'язує наданий код
13	Документування коду	Надайте фрагмент коду чи програму, яка не має належної документації.	Запропонуйте написати керівництво користувача, додати коментарі до коду.
14	Стандарти кодування та рефакторинг	Надайте фрагмент коду і попросіть переписати його за визначеними стандартами кодування.	Запропонуйте виконати рефакторингу коду, щоб підвищити ефективність чи зрозумілість.
15	Тестування коду	Надається готовий код, який розв'язує певну задачу. Слід сформувати кейс тестових допустимих вхідних даних та перевірити роботу коду.	Тестування коду користувачами дозволяє виявити помилки, які не помітив чи не врахував розробник.
16	Продовження чи масштабування коду	Код, що розв'язує певну задачу може бути частиною більш складної задачі та оформлений як функція.	Надайте код, який знаходить НСД двох чисел і запропонуйте поширити його на три числа.
17	Обробка помилок та винятків	Запропонуйте код з відсутніми механізмами обробки помилок і попросіть учнів додати обробку помилок у непередбачуваних ситуаціях.	Найбільша кількість помилок виникає через не правильні дії користувачів. Програма має враховувати можливість таких дій та помилок вводу даних.
18	Контроль версій компіляторів	Відомо, що існує велика кількість компіляторів C++, які мають свої особливості.	Код, для компілятора, наприклад, C++ gcc 4.3.2 переписати під C++ 14
19	Усунення вразливостей коду	Надайте студентам фрагмент коду і попросіть їх визначити та усунути потенційні вразливості безпеки або вразливості, пов'язані з конфіденційністю даних.	Типові вразливості: вихід за межі буфера, не ініціалізовані змінні, втрата значення змінних, помилки у перевірці вхідних даних, витік інформації чи пам'яті
20	Власна задача чи завдання	Складання власної задачі та написання коду її розв'язку чи складання комплекту тестових питання стимулює творчу роботу учнів.	За рівнем складності запропонованої учнем задачі можна визначити рівень засвоєння мови програмування.

маної формули числового значення відповіді. Для обчислення відповіді можуть застосовуватися математичні функції, які використовуються мовою php. Цей метод є більш ефективним оскільки він дозволяє надалі швидко змінювати базу випадкових даних (параметрів) та автоматично обчислювати правильні відповіді. Але, одночасно, витрати часу на початкові налаштування та отримання формули відповіді можуть бути значними. Більш того, викладач програмування має володіти математичним апаратом

для створення такого параметризованого завдання. Складання завдань також потребує певного рівня математичної підготовки викладача. Тому такий спосіб можна порадити для використання у разі великої кількості учнів які мають проходити тестування задля створення великої бази завдань.

Для спрощення складання тестових завдань з програмування та підвищення якості оцінювання результатів навчання наведемо далі (таблиця 1) класифікацію завдань з прикладами, які використовуються у системі онлайн-навчання [12] під час викладання дисциплін програмування. Мета такої класифікації – систематизувати тестові завдання задля оцінювання не тільки теоретичних знань, отриманих учнем чи студентом, але й для оцінювання практичного досвіду та набутих компетенцій.

Деякі з наведених тут типів завдань можуть бути використані під час складання тестів. Але такі задачі можна пропонувати й окремо, без тестів. Спростити викладачу перевірку таких завдань можна запропонувавши учням здавати скомпільований код у онлайн-компіляторах та надавати посилання на результати такої компіляції. Для такого здавання можуть буди створені завдання Moodle у межах курсу для зручного оцінювання.

На діаграмі (рис. 3) наведено результати оцінювання двох контрольних груп студентів спеціальності «Середня освіта (Інформатика)», що вивчали базовий курс програмування (мова С++) та проходили тестування чи виконували завдання зазначені у класифікаційній таблиці 1 під час дистанційного навчання в особливих умовах з використанням Moodle-платформ дистанційного навчання Кременчуцького національного університету імені Михайла Остроградського [12].

Рис. 3. Середній відсоток правильно виконаних завдань за типами

Як випливає з отриманих даних, найбільші складнощі виникли із задачами на оптимізацію коду (тип № 6) та задачами, що потребували пошуку помилок логіки (тип № 2). Найпростішими виявились задачі на пошук синтаксичних помилок (тип № 1) та задачі з документування коду (тип № 13).

Висновки. Оцінювання знань студентів з академічної дисципліни програмування під час дистанційного навчання пов'язане з численними проблемами, починаючи від відсутності практичного оцінювання і закінчуючи проблемами шахрайства та обмеженої взаємодії. Однак, використовуючи інноваційні рішення та адаптуючи методи оцінювання до умов дистанційного навчання, викладачі можуть пом'якшити ці проблеми та забезпечити справедливе і точне оцінювання навичок програмування студентів. В умовах дистанційного навчання, що набуває усе більшого застосування та швидко розвивається, гнучкість, адаптивність і прихильність до збереження академічної доброчесності є ключем до ефективного оцінювання знань з програмування.

Розглянуті тут приклади автоматизації створення тестових завдань засобами платформи Moodle підвищують якість тестових завдань з програмування та зменшують можливі прояви недоброчесності з боку учасників тестування. Запропонована у роботі класифікація практичних завдань з програмування сприятиме викладачам у складанні тестів та, відповідно, підвищить якість оцінювання отриманих учнями та студентами знань. Наведені статистичні дані визначення середнього відсотка правильно виконаних завдань з програмування за типами у результаті педагогічних експериментів на платформі системи онлайн-навчання Кременчуцького національного університету імені Михайла Остроградського та аналіз їх складності дозволить викладачам правильно комплектувати тестові завдання, що сприятиме якості та об`єктивності оцінювання практичних умінь.

ЛІТЕРАТУРА

1. Биков В.Ю., Овчарук О.В., Іванюк І.В., Пінчук О.П. Сучасний стан використання цифрових засобів для організації дистанційного навчання в закладах загальної середньої освіти: результати опитування 2022. *Інформаційні технології і засоби навчання.* 2022. Т. 90, № 4. С. 1–18.
2. Ібрагімова Л.А. Аналіз змісту професійної підготовки майбутніх інженерів-програмістів з формування алгоритмічної компетентності. *Педагогічні науки: теорія та практика.* 2021. № 4. С. 136–142.
3. Іщеряков С. Вчити програмування треба в школі чи університеті? Освітня політика. *Портал громадських експертів.* URL: http://osvita.ua/ school/54063/ (дата звернення: 15.09.2023).
4. Славко Г.В. Математика програмістам : навч. підруч. Кременчук : Видавництво ПП Щербатих О.В., 2018, 184 с.
5. Методичні рекомендації «Організація дистанційного навчання в школі» : Міністерство освіти і науки України. (2020. Травень). URL: https://rada.info/upload/users_files/41919831/4c6d4edbb7e057a3ecea236cc72d989d.pdf (дата звернення: 15.08.2023).
6. Олійник В.В., Грабовський П.П., Коновал О. А. Критерії та показники добору цифрової платформи електронного навчання для закладу загальної середньої освіти. *Інформаційні технології і засоби навчання.* 2022. Т. 90, № 4. С. 19–31.
7. Павленко М., Варава Т. Інформаційні засоби навчання основам програ-мування у середній школі. *Ukrainian Journal of Educational Studies and Information Technology.* Том. 5. №. 3. Вересень. 2017. С. 58–73. URL: https://core.ac.uk/download/pdf/233567916.pdf (дата звернення: 10.08.23).
8. Руденко Ю. Професійна адаптація молодих викладачів комп'ю-терних наук. URL: https://fmo-journal.fizmatsspu.sumy.ua/publ/4-1-0-351 (дата звернення: 5.09.2023).
9. Семеніхіна О.В., Руденко Ю.О. Проблеми навчання програмувати учнів старших класів та шляхи їх подолання. *Інформаційні технології і засоби навчання.* 2018. Т.66, № 4. С. 54–64.
10. Семеніхіна О.В., Юрченко А.О., Сбруєва А.А. Відкриті цифрові освітні ресурси у галузі ІТ : кількісний аналіз. *Інформаційні технології і засоби навчання*. 2020. Т. 75, № 1. С. 331–348.
11. Сергієнко С.А., Славко Г.В. Досвід упровадження та інтеграції системи онлайн-навчання на платформі Moodle у освітній простір університету. *Інженерні та освітні технології*. 2019. Т. 7. № 2. С. 126–136.
12. Система онлайн-навчання Кременчуцького національного університету імені Михайла Остроградського. URL: http://krnu.org (дата звернення: 20.09.2023).
13. Славко Г.В. Непрямі методи ідентифікації користувачів Moodle. *Восьма міжнародна науково-практична конференція «MoodleMoot Ukraine 2020». Теорія і практика використання системи управління навчанням Moodle.* Київський національний університет будівництва і архітектури, 2020 р. URL: https://2020.moodlemoot.in.ua/course/view.php?id=30 (дата звернення: 7.09.2023).
14. Славко Г.В. Розробка та впровадження інтерактивної перевірки програм-них кодів у системі онлайн-освіти «Математика.укр». *Інформатика та системні науки. 2017.* URL : http://dspace.puet.edu.ua/bitstream/ 123456789 /5552/1/22%20Славко.pdf (дата звернення: 2.09.2023).
15. Литвин О. Теоретичні та практичні аспекти використання математичних методів та інформаційних технологій в освіті й науці: монографія / за заг. ред. О. Литвин. К. : Київ. ун-т ім. Б. Грінченка, 2021. 332 с.
16. Computing Curricula 2020: Paradigms for Global Computing Education. *New York : Association for Computing Machinery*, 2020. 205 p. URL: https://www.acm.org/binaries/content/assets/education/curricula-recommen-dations/cc2020.pdf (дата звернення: 1.09.2023).
17. Gennadii Slavko, Serhii Serhiienko, Anfisa Shmelova, Kateryna Kovalenko «Problems and organizational measures of implementing and providing the distance electrical engineering education at the university using lms moodle in the conditions of modern challenges in Ukraine». *Proceedings of the International Conference on Modern Electrical and Energy Systems, MEES 2022.*, 20-22 October 2022.
18. LMS Moodle. URL: http://moodle.org (дата звернення: 1.09.2023).
19. Olena Grytsiuk, Larysa Herasymenko, Tetiana Nabok, Tatiana Bryl, Larysa Maksimova. Assessment of learning outcomes in distance learning using the Moodle platform. *Матеріали Міжнародної конференції «MEES 2022. Modern Electrical and Energy System»*, 20–22 October 2022.

REFERENCES

1. Bykov V. Yu., Ovcharuk O. V., Ivanyuk I. V., Pinchuk O. P., Hal′perina V. O. (2022). Suchasnyy stan vykorystannya tsyfrovykh zasobiv dlya orhanizatsiyi dystantsiynoho navchannya v zakladakh zahal′noyi seredn′oyi osvity: rezul′taty opytuvannya 2022 [The current state of the use of digital tools for organization of distance learning in general secondary education institutions: 2022 results]. *Information technologies and teaching tools*. Vol. 90, № 4. P. 1–18.

2. Ibrahimova L. A. (2022). Analiz zmistu profesiynoyi pidhotovky maybutnikh inzheneriv-prohramistiv z formuvannya alhorytmichnoyi kompetent-nosti [Analysis of the content of professional training of future software engineers in the formation of algorithmic competence]. *Pedagogical Sciences: Theory and Practice*, (4), P. 136–142.
3. Ishcheriakov S. Vchyty prohramuvannya treba v shkoli chy universyteti? [Should you learn programming at school or university?] *Educational politics Educational policy. Portal of public experts*. URL: http:// osvita.ua/school/ 54063/ (retrieving date: 15.09.2023).
4. Slavko G. V. (2018). Matematyka prohramistam: navchal′nyy pidruchnyk [Mathematics for programmers: a textbook]. Kremenchuk: Publishing house Shcherbatyh O. V., 184 p.
5. Ministry of Education and Science of Ukraine. (May 2020). Metodichni rekomendaciyi «Organizaciya distancijnogo navchannya v shkoli» [Methodical recommendations «Organization of distance learning at school»]. URL:https://rada.info/upload/users_files/41919831/4c6d4edbb7e057a3ecea236cc72d989d.pdf (retrieving date: 15.08.2023).
6. Oliynyk V. V., Hrabovskyi P. P., Konoval O. A. (2022). Kryteriyi ta pokaznyky doboru tsyfrovoyi platformy elektronnoho navchannya dlya zakladu zahal′noyi seredn′oyi osvity [Criteria and indicators for the selection of a digital platform for e-learning at the secondary school]. *Information technologies and teaching tools*. Vol. 90. № 4. P. 19–31.
7. Pavlenko M., Varava T. (September 2017). Informatsiyni zasoby navchannya osnovam prohramuvannya u seredniy shkoli [Information tools for teaching the basics of programming in high school]. *Ukrainian Journal of Educational Studies and Information Technology*. Vol. 5. №. 3. P. 58–73. URL: https://core. ac.uk/download/pdf/233567916.pdf (retrieving date: 10.08.2023).
8. Rudenko Y. O. (January 28, 2018). Profesiyna adaptatsiya molodykh vykladachiv komp′yuternykh nauk [Professional adaptation of young computer science teachers]. URL: http://fmo-journal.fizmatsspu.sumy. ua/ publ/4-1-0-351 (retrieving date: 5.09.2023).
9. Semenykhina O. V., Rudenko U. O. (2018). Problemi navchannya programuvati uchniv starshih klasiv ta shlyahi yih podolannya [Problems of educating to programming of students and way of their overcoming]. *Information technologies and teaching tools*. Vol. 66, № 4. P. 54–64.
10. Semenikhina O. V., Yurchenko A. O., Sbruieva A. A., Kuzminskyi A. I., Kuchai O. V., Bida O. A. (2020). Vidkryti tsyfrovi osvitni resursy u haluzi IT: kil′kisnyy analiz [The open digital educational resources in it-technologies: quantity analysis]. *Information technologies and teaching tools*. Vol. 75, № 1. P. 331–348.
11. Serhiienko S. A., Slavko G. V. (2019). Dosvid uprovadzhennya ta intehratsiyi systemy onlayn-navchannya na platformi Moodle u osvitniy prostir universytetu [Experience in implementing and integrating the online learning system on the Moodle platform into the educational environement of the university]. *Engineering and educational technologies*. Vol. 7. № 2. P. 126–136.
12. Online Learning System of Kremenchuk Mykhailo Ostorogradskiy National University. URL: http://krnu.org.
13. Slavko G.V. (2020). Nepryami metody identyfikatsiyi korystuvachiv Moodle [Indirect methods of Moodle user identification]. *8th International Scientific and Practical Conference «MoodleMoot Ukraine 2020». Theory and practice of using the learning management system Moodle.* Kyiv National University of Construction and Architecture. URL: https://2020.moodlemoot.in.ua/ course/view.php?id=30 (retrieving date: 7.09.2023).
14. Slavko G. V. (2017). Rozrobka ta vprovadzhennya interaktyvnoyi perevirky prohram-nykh kodiv u systemi onlayn-osvity «Математика.укр» [Development and implementation of interactive verification of software codes in the online education system «Математика.укр»]. *Informatics and system sciences*. URL: http://dspace.puet.edu.ua/bitstream/123456789/5552/ 1/22%20Славко.pdf (re-trieving date: 2.09.2023).
15. Teoretychni ta praktychni aspekty vykorystannya matematychnykh metodiv ta informatsiynykh tekhnolohiy v osviti y nautsi [Theoretical and practical aspects of the use of mathematical methods and information technologies in education and science]: monograph edited by O. Lytvyn. (2021) Kyiv: Borys Grinchenko Kyiv university. 332 p.
16. Computing Curricula (2020): Paradigms for Global Computing Education. *New York: Association for Computing Machinery.* 205 p. URL: https://www. acm.org/binaries/content/assets/education/curricula-recommendations/cc2020 .pdf (retrieving date: 1.09.2023).
17. Slavko G., Serhiienko S., Shmelova A., Kovalenko K. (20–22 October 2022) «Problems and organizational measures of implementing and providing the distance electrical engineering education at the university using lms moodle in the conditions of modern challenges in Ukraine». *Proceedings of the International Conference on «Modern Electrical and Energy Systems» (MEES 2022).*
18. LMS Moodle. URL: http://moodle.org (retrieving date: 1.09.2023).
19. Grytsiuk O., Herasymenko L., Nabok T., Bryl T., Maksimova L. (2022) Assessment of learning outcomes in distance learning using the Moodle platform. *Proceedings of International Conference «Modern electrical and energy system» (MEES 2022).* 20–22 October 2022.