Бобокало А., Юрченко А., Семеніхіна О. Навчання побудови блок-схем для розвитку алгоритмічного мислення майбутніх учителів інформатики. *Освіта. Інноватика. Практика,* 2025. Том 13, № 8. С. 14-19. https://doi.org/10.31110/2616-650X-vol13i8-002.

Bobokalo A., Yurchenko A., Semenikhina O. Navchannia pobudovy blok-skhem dlia rozvytku alhorytmichnoho myslennia maibutnikh uchyteliv informatyky [Teaching flowchart construction for the development of algorithmic thinking in future computer science teachers]. *Osvita. Innovatyka. Praktyka – Education. Innovation. Practice,* 2025. Vol. 13, No 8. S. 14-19. https://doi.org/10.31110/2616-650X-vol13i8-002.

УДК 378:37.091.12:004.421.2
DOI: 10.31110/2616-650X-vol13i8-002

Андрій БОБОКАЛО[1], Артем ЮРЧЕНКО[2], Олена СЕМЕНІХІНА[3]
[1-3] *Сумський державний педагогічний університет імені А. С. Макаренка, Україна*
[1] *https://orcid.org/0009-0003-9207-329X*
a.bobokalo@fizmatsspu.sumy.ua
[2] *https://orcid.org/0000-0002-6770-186X*
a.yurchenko@fizmatsspu.sumy.ua
[3] *https://orcid.org/0000-0002-3896-8151*
e.semenikhina@fizmatsspu.sumy.ua

НАВЧАННЯ ПОБУДОВИ БЛОК-СХЕМ ДЛЯ РОЗВИТКУ АЛГОРИТМІЧНОГО МИСЛЕННЯ МАЙБУТНІХ УЧИТЕЛІВ ІНФОРМАТИКИ

Анотація. *У статті висвітлено результати дослідження, спрямованого на обґрунтування доцільності та ефективності використання блок-схем у процесі вивчення обчислювальних методів як засобу розвитку алгоритмічного мислення майбутніх учителів інформатики. Основна увага приділена специфіці формулювання й опрацювання типових задач з курсу алгоритмізації й програмування (зокрема методів дихотомії, хорд, Ньютона), які мають чітку структуру ітераційного процесу та дозволяють візуалізувати логіку обчислень через блок-схеми. На основі практичної апробації доведено, що включення етапу побудови блок-схеми перед реалізацією коду сприяє глибшому осмисленню кожного етапу обчислення, правильному визначенню умов завершення ітерацій, структуруванню коду та розвитку аналітичних навичок. Розроблено та перевірено ефективність методичного підходу до порівняння альтернативних шляхів розв'язування задачі, що дає змогу студентам зіставляти ефективність методів, кількість ітерацій до збіжності, чутливість до початкових умов, а також складність реалізації й дидактичну доступність для студентів. У межах дослідження було також узагальнено низку методичних рішень щодо інтеграції блок-схем у професійну підготовку майбутніх учителів інформатики. Зокрема, запропоновано поетапну структуру завдань, рефлексивне осмислення побудованих схем, створення навчальних матеріалів на їх основі, а також моделювання уроків із використанням блок-схем як інструмента пояснення. Обґрунтовано, що побудова блок-схем виконує подвійну функцію — когнітивно-візуальну (як інструмент засвоєння алгоритму) та дидактичну (як засіб пояснення й аналізу). Зроблено висновок про необхідність системного включення таких завдань у підготовку майбутніх учителів інформатики як засобу формування в них алгоритмічного мислення, педагогічного передбачення та готовності до викладання інформатики в закладах загальної середньої освіти.*

Ключові слова: *алгоритмічне мислення; блок-схема; візуальне моделювання; обчислювальні методи; професійна підготовка; майбутні вчителі інформатики; дидактичні інструменти; методика навчання програмуванню; цифрові середовища; професійна освіта.*

Andrij BOBOKALO[1], Artem YURCHENKO[2], Olena SEMENIKHINA[3]
[1-3] *Sumy State Pedagogical University named after A.S. Makarenko, Ukraine*
[1] *https://orcid.org/0009-0003-9207-329X*
a.bobokalo@fizmatsspu.sumy.ua
[2] *https://orcid.org/0000-0002-6770-186X*
a.yurchenko@fizmatsspu.sumy.ua
[3] *https://orcid.org/0000-0002-3896-8151*
e.semenikhina@fizmatsspu.sumy.ua

TEACHING FLOWCHART CONSTRUCTION FOR THE DEVELOPMENT OF ALGORITHMIC THINKING IN FUTURE COMPUTER SCIENCE TEACHERS

Abstract. *The article presents the results of a study aimed at substantiating the feasibility and effectiveness of using flowcharts in the study of computational methods as a means of developing algorithmic thinking in future computer science teachers. The main focus is placed on the specifics of formulating and solving typical problems from the course on algorithmization and programming (in particular, the bisection method, the chord method, and Newton's method), which have a clear iterative structure and allow the visualization of the logic of computations through flowcharts. Based on practical testing, it is proven that including the stage of constructing a flowchart before code implementation contributes to a deeper understanding of each step of the computation, the correct determination of iteration termination conditions, code structuring, and the development of analytical skills. A methodological approach was developed and tested to compare alternative algorithms on the same example, enabling students to evaluate the efficiency of the method, the number of iterations required for convergence, sensitivity to initial conditions, as well as implementation complexity and didactic accessibility for learners. The study also summarized a set of methodological solutions for integrating flowcharts into students' professional training. In particular, a step-by-step*

structure of tasks, reflective analysis of constructed diagrams, the creation of educational materials based on them, and the modeling of lessons using flowcharts as an explanatory tool were proposed. It is argued that constructing flowcharts performs a dual function – cognitive-visual (as a tool for mastering an algorithm) and didactic (as a means of explanation and analysis). The conclusion emphasizes the necessity of systematically integrating such tasks into the training of future educators as a means of forming algorithmic thinking, pedagogical foresight, and methodological readiness for teaching computer science in general secondary education institutions.

Keywords: *algorithmic thinking; flowchart; visual modeling; computational methods; professional training; future computer science teachers; didactic tools; methods of teaching programming; digital environments; vocational education.*

Постановка проблеми. У сучасній системі професійної підготовки майбутніх учителів інформатики формування алгоритмічного мислення виступає ключовою складовою, що детермінує ефективність подальшого викладання програмування в загальноосвітніх школах. Особливу значимість у цьому контексті набуває методика візуалізації алгоритмів засобами блок-схем, яка забезпечує не лише засвоєння абстрактних концепцій, а й розвиток системного мислення. Проте аналіз навчальних програм педагогічних закладів вищої освіти свідчить про недостатнє використання цього інструментарію, що призводить до поверхневого засвоєння алгоритмічних принципів та обмежує професійну готовність випускників.

Блок-схеми як графічний спосіб подання алгоритмів виконують кілька ключових функцій у процесі навчання. Вони дозволяють візуалізувати логічну структуру алгоритму, сприяють ефективній декомпозиції складних задач на окремі компоненти, полегшують ідентифікацію помилок у алгоритмічній логіці та формують навички структурованого мислення. Ці аспекти є особливо актуальними при вивченні обчислювальних методів, де точність алгоритмічної реалізації безпосередньо впливає на коректність отриманих результатів.

Для опанування обчислювальних методів, таких як метод дихотомії, метод хорд чи метод Ньютона при розвязуванні нелінійних рінянь, блок-схеми виступають незамінним інструментом. Вони дають змогу наочно відобразити ітераційні процеси, умови завершення обчислень та логічні розгалуження алгоритму. Таке подання не лише сприяє глибшому розумінню математичної сутності методів, а й формує зв'язок між теоретичними знаннями та їх практичною реалізацією в програмному коді. Проте використання блок-схем в освітньому процесі стикається з низкою проблем. Серед них – недостатнє володіння сучасними інструментами візуального моделювання, відсутність чіткої методики інтеграції блок-схем у навчальні курси та фрагментарний характер їх застосування. Ці фактори обмежують потенціал блок-схем як засобу розвитку алгоритмічного мислення та знижують якість професійної підготовки майбутніх педагогів.

Враховуючи вищезазначене, виникає потреба у системному дослідженні можливостей оптимізації використання блок-схем у процесі вивчення обчислювальних методів. Таке дослідження має охоплювати як аспекти підвищення ефективності засвоєння алгоритмічних принципів, так і розробку методичних підходів до їх подальшого використання у шкільній практиці. Реалізація цих заходів дозволить підвищити якість підготовки майбутніх учителів інформатики та забезпечити формування стійких навичок алгоритмічного мислення, необхідних для сучасної освітньої практики.

Аналіз актуальних досліджень. Формування алгоритмічного мислення у майбутніх учителів інформатики є одним із пріоритетних завдань професійної підготовки, особливо в умовах зростання значущості програмування в сучасній цифровій освіті. Більшість дослідників сходяться на думці, що ефективне навчання програмуванню починається не з технічного опанування синтаксису мови, а з глибокого розуміння логіки побудови алгоритмів, здатності послідовно аналізувати умову задачі та відтворювати її структуру у формі логічно взаємопов’язаних дій. У цьому процесі блок-схеми виступають важливим інструментом – вони допомагають перетворити вербальне формулювання задачі на візуально зрозумілий алгоритм, що полегшує подальше програмне втілення та сприяє розвитку структурного мислення студентів [1; 5; 10].

Особливої значущості набуває вміння будувати блок-схеми при опрацюванні задач, які мають складну ітераційну або умовну структуру, характерну для курсу «Методи обчислень». Такі задачі, як знаходження коренів нелінійних рівнянь методом дихотомії, хорд чи Ньютона, потребують від здобувача не просто навичок алгоритмізації, а глибокого розуміння математичної логіки розв’язання. Без усвідомлення послідовності обчислювальних кроків, ролі початкових умов, критеріїв зупинки та механізмів оновлення значень змінних побудова блок-схеми набуває формального характеру. Отже, успішне візуальне моделювання таких задач можливе лише за умови глибокого аналітичного осмислення алгоритму як об’єкта, що поєднує математичну і програмну складові.

Наукові розвідки [6; 7; 9] актуалізують необхідність методичної підтримки процесу формалізації задач, особливо тих, які потребують гнучкого мислення та розуміння принципів чисельних наближень. Автори підкреслюють, що ключем до ефективної підготовки вчителя є не запам’ятовування формальних конструкцій, а здатність відтворити логіку обчислювального процесу у формі зрозумілої, структурованої схеми. У дослідженні [3] автори акцентують увагу на важливості поєднання математичної та алгоритмічної підготовки, наголошуючи, що графічне представлення алгоритмів сприяє формуванню міждисциплінарних зв’язків і розвитку операційного стилю мислення.

Аналогічно, у статті [4] у результаті локального дослідження показали, що учні, які працювали з візуальними уявленнями алгоритмів, мали вищу успішність у переході до кодування, що підтверджує значення блок-схем як проміжної форми розуміння алгоритму. У роботах [2; 8] візуальні елементи розглядаються не лише як інструмент мотивації, а як засіб покращення розуміння причинно-наслідкових зв'язків у структурі коду. Така перспектива є особливо цінною при роботі з обчислювальними методами, де результат залежить від коректної реалізації умов зупинки та ітераційних переходів, що важко пояснити вербально, але легко подати через блок-схему.

Попри наявність широкого спектра цифрових інструментів для побудови блок-схем, аналіз освітньої практики свідчить про недостатнє їх використання саме в контексті розв'язання типових задач курсу «методи обчислень». Це вказує на потребу системного впровадження методичних рішень, які забезпечують поєднання математичної підготовки, візуального представлення та програмної реалізації.

Таким чином, сучасні дослідження підтверджують, що успішне формування алгоритмічного мислення у процесі опрацювання обчислювальних методів можливе лише за умови поєднання аналітичного розуміння алгоритму з його візуальним моделюванням. Блок-схема в такому випадку перетворюється з допоміжного засобу на центральний компонент навчання, що забезпечує глибоке розуміння, готовність до програмної реалізації й подальше дидактичне пояснення нового матеріалу.

Мета і методи дослідження. Метою цього дослідження є виявити методичні підходи до ефективної інтеграції побудови блок-схем у процес опрацювання типових задач з обчислювальних методів у професійній підготовці майбутніх учителів інформатики, з урахуванням специфіки цих задач та потреб майбутньої професійної діяльності студентів.

У ході дослідження використано метод систематизації, застосований до класифікації типових задач з обчислювальних методів (метод дихотомії, метод хорд, метод Ньютона) за рівнем алгоритмічної складності та потенціалом візуалізації та узагальнення педагогічних практик і досвіду впровадження блок-схем у освітній процес на основі спостереження та якісного аналізу відкритих джерел.

У межах дослідження також було апробовано серію навчальних завдань із побудови блок-схем до обчислювальних методів у курсі алгоритмізації та програмування зі студентами спеціальності «Середня освіта (Інформатика)», що дозволило оцінити ефективність візуального моделювання для розвитку алгоритмічного мислення та педагогічного розуміння структури навчального матеріалу.

Результати дослідження. У процесі дослідження було здійснено аналіз типових задач з обчислювальних методів, які використовуються також у курсах алгоритмізації та програмування при підготовці учителів інформатики. Особливу увагу було зосереджено на тих задачах, що мають чітку покрокову логіку виконання, допускають ітеративні розв'язання, містять умови перевірки наближення до результату й дозволяють ефективну візуалізацію алгоритмічного процесу. Виявлено, що найбільш уживаними є задачі на чисельне знаходження коренів нелінійних рівнянь, зокрема метод дихотомії (бісекції), метод хорд і метод Ньютона. Ці методи мають спільну структуру алгоритму: визначення початкового наближення або інтервалу, перевірка умови завершення обчислень, обчислення нової апроксимації, повернення до початку циклу. Така повторювана логіка дозволяє успішно представити алгоритм у вигляді блок-схеми, де кожен блок виконує чітко окреслену функцію, а структура переходів забезпечує наочне відображення циклічності, умовності та залежності результату від заданої точності. Студенти, працюючи над побудовою блок-схем до таких методів, розвивають уміння виокремлювати етапи обчислювального процесу, формалізувати математичну процедуру, передбачати її перебіг та зупинку.

Аналіз завдань також показав, що їх педагогічна цінність полягає не лише у математичній складовій, а й у можливості застосувати поняття логічних умов, параметрів функції, змінних та циклів, що є базовими елементами будь-якої мови програмування. Таким чином, робота з блок-схемами у межах цих задач дозволяє забезпечити цілісний підхід до навчання: від аналітичного осмислення задачі через її алгоритмічне моделювання до програмної реалізації.

Особливу складність для студентів становить етап формулювання умов завершення обчислень та переходу між ітераціями. Блок-схема дозволяє візуально виокремити ці моменти, спрощуючи розуміння логіки алгоритму й сприяючи його подальшому правильному кодуванню. В процесі аналізу студентських робіт було встановлено, що після побудови блок-схем до обраних методів значно покращилась здатність студентів до самостійного структурування коду, встановлення залежностей між етапами обчислення та контролю коректності результатів. Таким чином, включення задач з обчислювальних методів до освітнього процесу з обов'язковою побудовою блок-схем є доцільним кроком для формування у студентів цілісного уявлення про логіку обчислювальних процедур, розвиток алгоритмічного мислення та підготовки до викладання подібних тем у шкільному курсі інформатики. Це сприяє одночасному опануванню як математичних, так і програмних складових алгоритму через візуально-зрозуміле моделювання.

Було здійснено узагальнення методичних рішень, спрямованих на ефективну інтеграцію побудови блок-схем у навчання студентів педагогічних спеціальностей з урахуванням специфіки їхньої

майбутньої професійної діяльності. Ключовим стало розуміння блок-схем не лише як навчального інструмента для формування алгоритмічного мислення, а як складової методичної підготовки, що дозволяє майбутньому вчителю доступно, наочно і поетапно пояснити учням логіку побудови алгоритмів у шкільному курсі інформатики.

Аналіз навчальних практик і студентських відгуків дозволив виокремити кілька методичних стратегій, які показали високу ефективність у процесі формування професійної готовності студентів. По-перше, доцільним є *застосування поетапного моделювання*, за якого студенти спочатку описують задачу природною мовою, потім будують її логічну структуру у вигляді блок-схеми, а вже після цього переходять до написання програмного коду. Такий підхід сприяє поступовому переходу від інтуїтивного до формалізованого мислення та знижує рівень тривожності, пов'язаний із помилками програмування.

Ефективним виявився *метод порівняльного аналізу альтернативних алгоритмів* через побудову їхніх блок-схем: наприклад, студенти моделювали на одному прикладі два різних обчислювальних методи (дихотомії та хорд), що дозволяло порівнювати складність реалізації, умови зупинки, чутливість до початкових значень. Така діяльність не лише розвивала аналітичні здібності, а й формувала здатність пояснювати переваги певного підходу з педагогічної точки зору. Як приклад, студентам було запропоновано знайти корінь рівняння $f(x) = x^3 - x - 2$ на інтервалі [1, 2] з точністю до $\varepsilon = 0.001$. Побудова блок-схем дозволила візуалізувати відмінності: метод дихотомії передбачав поділ інтервалу навпіл на кожному кроці з обчисленням значення функції в середині, тоді як метод хорд використовував дві крайні точки інтервалу для побудови січної і визначення точки перетину з віссю x. У процесі практичної реалізації було зафіксовано, що метод дихотомії забезпечив збіжність за 11 кроків, тоді як метод хорд - за 7 кроків, але вимагав більш обережного вибору початкових значень через чутливість до форми функції. У блок-схемах це проявлялося у більшій кількості перевірок та перерахунків при використанні хорд, коли відповідні блоки умов мали складнішу структуру, проте виявлялися більш гнучкими у відображенні логіки обчислень. Студенти, зіставляючи ці дві блок-схеми, мали змогу оцінити не лише ефективність збіжності, але й дидактичну складність пояснення кожного методу. Частина з них дійшла висновку, що для шкільного рівня доцільніше починати з методу дихотомії через його концептуальну простоту та стабільність, тоді як метод хорд доцільно вводити пізніше як приклад оптимізації процесу. Такий досвід не лише сприяв глибшому розумінню обчислювальних алгоритмів, але й розвивав педагогічну рефлексію щодо добору методів навчання, дидактичної послідовності і адаптації матеріалу до рівня підготовки учнів.

Було апробовано *включення елементів рефлексії* у завдання: після побудови блок-схеми студенти мали відповісти на запитання щодо її дидактичної цінності, можливих труднощів для учнів, способів адаптації до різного рівня підготовки школярів. Це стимулювало розвиток професійного мислення, орієнтованого на потреби майбутніх учнів. Було визнано ефективною практику *створення інтерактивних дидактичних матеріалів*, зокрема відеоінструкцій, покрокових схем або інтерактивних презентацій, які пояснюють структуру алгоритму через блок-схему. Це сприяло розвитку у студентів навичок цифрової педагогічної творчості та розвитку їх компетентності в розробленні власного навчального контенту.

Загалом узагальнення методичних рішень дозволяє стверджувати, що інтеграція побудови блок-схем у професійну підготовку майбутніх учителів інформатики має ґрунтуватися на принципах поетапності, контекстності, рефлексивності й методичної релевантності. Це забезпечує не лише розвиток власних алгоритмічних компетентностей студентів, а й формує їхню готовність передавати ці знання учням у доступній, наочній і педагогічно виваженій формі.

Обговорення. Результати підтверджують значний методичний потенціал задач з обчислювальних методів для розвитку алгоритмічного мислення студентів. Такі задачі, як пошук коренів рівняння методом дихотомії, хорд або Ньютона, дозволяють моделювати алгоритми з чітко структурованими ітераційними блоками, умовами зупинки й етапами оновлення змінних. Візуалізація цих процесів через блок-схеми відкриває для студентів можливість не просто реалізувати обчислення, а усвідомити логіку кожної дії, структуру повторення та взаємозв'язки між етапами. Це збігається з поглядами Ю. Хворостіни та ін. [3; 11] щодо необхідності поєднання математичної й алгоритмічної підготовки у професійному становленні майбутнього вчителя інформатики.

Особливістю підходу, що був застосований у дослідженні, стало не лише опрацювання окремих алгоритмів, а й порівняння альтернативних підходів через побудову відповідних блок-схем. Це дозволило студентам перейти від репродуктивної діяльності до аналітичної: вони порівнювали складність реалізації, чутливість до початкових умов, кількість кроків до збіжності. Завдяки цьому зростала не тільки глибина розуміння алгоритму, але й методична здатність обґрунтувати, який метод доцільно використати у тій чи іншій навчальній ситуації. У такий спосіб блок-схема з інструмента для власного розуміння перетворюється на засіб педагогічної адаптації та пояснення.

Виконання четвертого завдання дозволило сформулювати низку методичних рішень, що забезпечують ефективну інтеграцію побудови блок-схем у професійну підготовку майбутніх учителів.

Ключовою серед них є стратегія поетапного навчання, що поєднує усну постановку задачі, побудову блок-схеми та реалізацію коду. Це забезпечує перехід від природномовного опису до програмної формалізації через осмислену візуалізацію. Підтвердженням ефективності такого підходу стали рефлексивні висловлювання студентів, які відзначали, що лише після побудови блок-схеми змогли повністю зрозуміти алгоритм і пояснити його іншим. Також до ефективних методичних прийомів можна віднести моделювання фрагментів уроків з опорою на блок-схеми, аналіз типових помилок учнів на прикладах помилково побудованих схем, а також створення навчального контенту (інтерактивних посібників, відео, презентацій) із поясненням алгоритмів за схемами. Такі практики розвивають у студентів педагогічне передбачення, цифрову творчість та навички роботи з навчальними матеріалами.

Отже, результати третього і четвертого завдань свідчать про те, що блок-схеми в системі професійної підготовки вчителя інформатики мають подвійну функцію – вони одночасно виступають засобом когнітивної візуалізації для студента та інструментом майбутнього педагогічного впливу на учня. Успішне виконання цих завдань можливе лише за умови цілісної методичної інтеграції в освітній процес, що поєднує розвиток логіко-структурного мислення з формуванням дидактичної компетентності.

Висновки. У ході дослідження було встановлено, що задачі з обчислювальних методів, які традиційно включаються до курсів алгоритмізації й програмування, мають значний потенціал для формування алгоритмічного мислення студентів за умови їх подання у формі блок-схем. Методи чисельного розв'язання нелінійних рівнянь є цінними тим, що мають чітку структуру повторюваних дій, умови зупинки та залежності між етапами обчислень, що легко піддаються візуалізації. Побудова блок-схем до таких задач сприяє не лише формалізації обчислювального процесу, а й усвідомленню внутрішньої логіки обраного методу, його переваг, обмежень і чутливості до початкових умов.

Виявлено, що включення етапу побудови блок-схеми перед написанням коду значно підвищує якість програмної реалізації, знижує кількість помилок, а також формує у студентів здатність пояснювати алгоритм з позицій дидактики. При цьому особливо ефективним є порівняння альтернативних методів на прикладі однієї задачі, коли студенти не лише моделюють структуру обчислень, а й оцінюють їхню складність, кількість кроків до збіжності та дидактичну доцільність використання у школі.

Дослідження дозволило узагальнити низку методичних рішень, що забезпечують ефективну інтеграцію блок-схем у навчання студентів з орієнтацією на їхню майбутню професійну діяльність. Доведено, що побудова блок-схем виконує подвійну функцію: вона є одночасно засобом формування власного алгоритмічного мислення та дидактичним інструментом, який студент у майбутньому застосовуватиме в учнівському середовищі. До ефективних методичних рішень було віднесено: поетапне моделювання алгоритму з переходом від словесного опису до коду, рефлексивне осмислення побудованої схеми, порівняння кількох алгоритмів на одному прикладі, а також створення навчального контенту на основі візуалізацій.

На основі отриманих результатів пропонуються такі практичні рекомендації:

– включати задачі з чисельних методів до змісту практичних занять з обов'язковим етапом побудови блок-схем перед реалізацією коду;

– застосовувати порівняльне моделювання альтернативних алгоритмів як засіб розвитку аналітичного та педагогічного мислення;

– використовувати блок-схеми як інструмент для рефлексивного осмислення не лише обчислювальної, а й дидактичної логіки задачі;

– організовувати виконання студентами методичних завдань, які передбачають створення дидактичних матеріалів для учнів на основі блок-схем (інструкцій, відео, презентацій тощо);

– впроваджувати систематичне обговорення типових помилок у блок-схемах як засобу підготовки до аналізу учнівських робіт у шкільному навчанні.

Отже, задачі з обчислювальних методів у поєднанні з побудовою блок-схем не лише сприяють глибшому розумінню алгоритмічних структур, а й формують у студентів здатність до педагогічного моделювання, аналітичного порівняння підходів і створення наочних, методично виправданих пояснень. У такий спосіб візуальне моделювання перетворюється на основу професійного осмислення алгоритмів і надійний інструмент підготовки до викладання інформатики у школі.

Конфлікт інтересів. Автори підтверджують відсутність фінансових, особистих чи інших інтересів, що можуть розглядатися як потенційний конфлікт інтересів щодо публікації цієї статті.

Фінансування. Робота виконана за відсутності фінансової підтримки з боку будь-яких організацій.

Доступність даних. Це теоретичне дослідження не передбачає використання додаткових наборів даних.

Використання штучного інтелекту. Інструменти штучного інтелекту не використовувались при написанні цієї роботи.

Список використаних джерел

1. Coşkunserçe, O. (2023). Comparing the use of block-based and robot programming in introductory programming education: Effects on perceptions of programming self-efficacy. *Computer Applications in Engineering Education, 31*(5), 1234–1255. https://doi.org/10.1002/cae.22637
2. Espinal, A., Vieira, C., & Guerrero-Bequis, V. (2022). Student ability and difficulties with transfer from a block-based programming language into other programming languages: A case study in Colombia. *Computer Science Education, 33*(4), 567–599. https://doi.org/10.1080/08993408.2022.2079867
3. Khvorostina, Y., Shamonia, V., & Semenikhina, O. (2025). The connection between the study of mathematics and programming through the prism of scientific and pedagogical research. *Вісник науки та освіти, 4*(34), 932–945. https://doi.org/10.52058/2786-6165-2025-4(34)-932-945
4. Rudenko, Y., Drushlyak, M., Osmuk, N., Shvets, O., Kolyshkin, O., & Semenikhina, O. (2022). Problems of teaching pupils of non-specialized classes to program and ways to overcome them: Local study. *International Journal of Computer Science and Network Security, 22*(1), 105–112. https://doi.org/10.22937/IJCSNS.2022.22.1.16
5. Sanusi, I. T., Cudjoe, E. S., Ayanwale, M. A., & Adepoju, B. (2025). Pre-service teachers' perception of programming education. *SAGE Open, 15*(1). https://doi.org/10.1177/21582440251327019
6. Semenikhina, O. V., & Rudenko, Y. O. (2018). Проблеми навчання програмувати учнів старших класів та шляхи їх подолання. *Інформаційні технології і засоби навчання, 66*(4), 54–64.
7. Umezawa, K., Ishida, K., Nakazawa, M., & Hirasawa, S. (2023). Proposal and evaluation of intermediate content for the transition from visual to text-based programming languages. In T. X. Bui (Ed.), *Proceedings of the 56th Hawaii International Conference on System Sciences* (pp. 83–92). https://doi.org/10.24251/HICSS.2023.011
8. Дегтярьова, Н., Петренко, С., & Удовиченко, О. (2023). Робота з графічними віджетами при вивченні мови програмування Python в закладах загальної середньої освіти. *Освіта. Інноватика. Практика, 11*(4), 26–34. https://doi.org/10.31110/2616-650X-vol11i4-004
9. Кобильник, Т., Когут, У., & Жидик, В. (2021). Методичні аспекти вивчення основ алгоритмізації і програмування мовою Python у шкільному курсі інформатики у старших класах. *Фізико-математична освіта, 31*(5), 36–44. https://doi.org/10.31110/2413-1571-2021-031-5-006
10. Пенко, В., & Пенко, О. (2023). Використання візуалізації на різних етапах вивчення дисципліни «Програмування». *Освіта. Інноватика. Практика, 11*(2), 31–39. https://doi.org/10.31110/2616-650X-vol11i2-005
11. Юрченко, А. О., Семеніхіна, О. В., Хворостіна, Ю. В., & Удовиченко, О. М. (2019). Навчання програмувати в старшій школі крізь призму чинних навчальних програм. *Фізико-математична освіта, 2*(20), Ч.2, 47–54.

References

1. Coşkunserçe, O. (2023). Comparing the use of block-based and robot programming in introductory programming education: Effects on perceptions of programming self-efficacy. *Computer Applications in Engineering Education, 31*(5), 1234–1255. https://doi.org/10.1002/cae.22637
2. Espinal, A., Vieira, C., & Guerrero-Bequis, V. (2022). Student ability and difficulties with transfer from a block-based programming language into other programming languages: A case study in Colombia. *Computer Science Education, 33*(4), 567–599. https://doi.org/10.1080/08993408.2022.2079867
3. Khvorostina, Y., Shamonia, V., & Semenikhina, O. (2025). The connection between the study of mathematics and programming through the prism of scientific and pedagogical research. *Вісник науки та освіти, 4*(34), 932–945. https://doi.org/10.52058/2786-6165-2025-4(34)-932-945
4. Rudenko, Y., Drushlyak, M., Osmuk, N., Shvets, O., Kolyshkin, O., & Semenikhina, O. (2022). Problems of teaching pupils of non-specialized classes to program and ways to overcome them: Local study. *International Journal of Computer Science and Network Security, 22*(1), 105–112. https://doi.org/10.22937/IJCSNS.2022.22.1.16
5. Sanusi, I. T., Cudjoe, E. S., Ayanwale, M. A., & Adepoju, B. (2025). Pre-service teachers' perception of programming education. *SAGE Open, 15*(1). https://doi.org/10.1177/21582440251327019
6. Semenikhina, O. V., & Rudenko, Y. O. (2018). Проблеми навчання програмувати учнів старших класів та шляхи їх подолання. *Інформаційні технології і засоби навчання, 66*(4), 54–64.
7. Umezawa, K., Ishida, K., Nakazawa, M., & Hirasawa, S. (2023). Proposal and evaluation of intermediate content for the transition from visual to text-based programming languages. In T. X. Bui (Ed.), *Proceedings of the 56th Hawaii International Conference on System Sciences* (pp. 83–92). https://doi.org/10.24251/HICSS.2023.011
8. Dehtiarova . H., Petrenko C., & Udovychenko O. (2023). Working with graphic widgets when learning the Python programming language in institutions of general secondary education. *Education. Innovation. Practice, 11*(4), 26–34. https://doi.org/10.31110/2616-650X-vol11i4-004
9. Kobylnyk, T., Kohut, U., & Zhydyk, V. (2021). Methodical aspects of studying the fundamentals of algorithmization and programming language python school course in informatics in high school. *Physical and Mathematical Education, 31*(5), 36–44. https://doi.org/10.31110/2413-1571-2021-031-5-006
10. Penko B., & Penko O. (2023). Using visualization at different stages of studying the discipline "Programming". *Education. Innovation. Practice, 11*(2), 31–39. https://doi.org/10.31110/2616-650X-vol11i2-005
11. Yurchenko, A. O., Semenikhina, O. V., Khvorostina, Yu. V., & Udovychenko, O. M. (2019). Navchannia prohramuvaty v starshii shkoli kriz pryzmu chynnykh navchalnykh prohram. *Fizyko-matematychna osvita, 2*(20), Ch.2, 47–54.

| Матеріал надійшов до редакції: 28.06.2025 р. | Прийнято до друку: 01.08.2025 р. | Опубліковано: 30.10.2025 р. |