ТЕОРІЯ НАВЧАННЯ

УДК 378.147:004

FEATURES OF TRAINING OF FUTURE COMPUTER SCIENCE TEACHERS IN PROGRAMMING

ОСОБЛИВОСТІ НАВЧАННЯ МАЙБУТНІХ ВЧИТЕЛІВ ІНФОРМАТИКИ ПРОГРАМУВАННЮ

Максим Роганов
кандидат технічних наук, доцент
E-mail: maxmar1@ukr.net.
ORCID 0000-0002-2153-6854
«Харківська гуманітарно-педагогічна академія», Україна

Maxim Roganov
PhD in Technical, Associate Professor
E-mail: maxmar1@ukr.net.
ORCID 0000-0002-2153-6854
"Kharkiv Humanitarian-Pedagogical Academy" Ukraine

Максим Роганов
кандидат педагогічних наук, доцент
E-mail: maxipro1987@gmail.com
ORCID 0000-0001-6488-8692
«Харківська гуманітарно-педагогічна академія», Україна

Maxim Roganov
PhD in Pedagogical, Associate Professor
E-mail: maxipro1987@gmail.com
ORCID 0000-0001-6488-8692
"Kharkiv Humanitarian-Pedagogical Academy" Ukraine

Георгій Козак
викладач
E-mail: zhora.kozak2002@gmail.com
ORCID 0009-0008-1106-5180
«Харківська гуманітарно-педагогічна академія», Україна

Heorhii Kozak
Lecturer
E-mail: zhora.kozak2002@gmail.com
ORCID 0009-0008-1106-5180
"Kharkiv Humanitarian-Pedagogical Academy" Ukraine

ABSTRACT

Features of training computer science teachers in programming represent a key aspect of their professional education. The need for such knowledge is due to the rapid development of digital technologies and their widespread introduction into the educational process. Successful training of future teachers involves the formation of not only basic programming skills, but also the ability to apply the knowledge gained in the context of teaching computer science.

Particular attention in the learning process is paid to such aspects as the development of algorithmic thinking, the development of various programming languages, as well as an understanding of the principles of software design and development. An important element of training is the emphasis on the practical application of knowledge, which allows future teachers to effectively use modern tools and technologies in the learning process.

In addition, the training covers methodological approaches to teaching programming. This includes the development of educational materials, the selection of adequate teaching methods and the creation of a supportive educational environment

that promotes the development of students' creative potential. Integration of theoretical and practical aspects of learning allows students to form a deep understanding of the subject and confidence in its teaching.

Thus, the training of future computer science teachers in programming requires an integrated approach that combines the development of professional skills with the formation of pedagogical competence, which contributes to ensuring the high quality of the educational process in the conditions of modern requirements.

Key words: *teacher training, computer science, programming, ICT competence, vocational education, applied software*

Актуальність теми. Підготовка майбутніх учителів інформатики набула особливої актуальності в умовах цифровізації освіти, переходу до компетентнісного навчання та зростання потреби учнів у розвитку навичок алгоритмічного та логічного мислення. Володіння програмуванням для вчителя інформатики є не лише професійною вимогою, а й ключовим інструментом для формування в учнів STEM-компетентностей. Однак стрімкий розвиток ІТ-галузі, поява нових мов програмування та освітніх платформ висувають нові вимоги до професійної підготовки педагогів.

Професійна педагогічна освіта характеризується специфічними особливостями, які вирізняють її серед інших напрямів вищої освіти. На відміну від випускників інших закладів, здобувачі освіти педагогічних закладів вищої освіти повинні не лише опанувати специфічні знання у вибраній предметній галузі, але й бути підготовленими до комплексної організації навчальної, виховної та розвивальної роботи з учнями. Підготовка майбутніх педагогів має відповідати професійно-педагогічній орієнтації, що передбачає органічне поєднання педагогічних принципів із предметними аспектами діяльності. Цей підхід включає засвоєння змісту дисципліни у контексті викладацької практики. Особливе значення у підготовці висококваліфікованих і конкурентоспроможних майбутніх учителів інформатики відіграє їхня спеціалізована підготовка в галузі програмування.

Система навчання програмуванню для майбутніх учителів інформатики залежить від широкого спектра чинників, які необхідно враховувати для забезпечення якісної професійної підготовки випускників педагогічних ЗВО.

В умовах динамічного розвитку інформаційних технологій базовий рівень знань і навичок, пов’язаних із використанням окремих програмних засобів, є недостатнім.

Галузь програмування характеризується інтенсивними темпами розвитку, що супроводжуються регулярною появою новітніх технологій, мов і середовищ програмування. Таким чином, учитель інформатики повинен орієнтуватися в постійно змінюваному інформаційному просторі, проявляючи здатність оперативно реагувати на його оновлення. Реалізація таких вимог можлива лише за умови фундаментальної підготовки як у сфері програмування, так і в загальній сфері галузі інформатики.

Аналіз останніх досліджень і публікацій. Підготовка майбутніх учителів інформатики з програмування є одним із ключових завдань сучасної педагогічної освіти, що визначається технологізацією суспільства, потребою формування цифрової та алгоритмічної культури школярів, а також зростанням ролі

комп'ютерного мислення в усіх сферах діяльності.

Аналіз наукових досліджень дозволяє виділити три провідні напрями, що формують теоретичні засади та практичні підходи до навчання програмування:

1. Комп'ютерне мислення та алгоритмічна підготовка;
2. Методика навчання програмування і формування професійної компетентності;
3. Психологічні аспекти засвоєння програмування.

Початок сучасної хвилі досліджень пов'язують з роботою (Wing,, 2006), яка ввела поняття computational thinking (комп'ютерне мислення – CT) як універсального способу розв'язування задач, що базується на ідеях інформатики, але призначений для всіх, а не лише програмістів.

У подальшій статті «Computational Thinking: What and Why?» вона уточнює визначення CT як уміння формулювати задачі так, щоб їх могли ефективно розв'язувати як люди, так і машини, підкреслюючи значення декомпозиції, абстракції, оцінки обмежень ресурсів і автоматизації (Wing,, 2010).

На цю концепцію спираються огляди (Barr, Harrison, Conery, 2011), які трактують CT як «цифрову навичку для всіх» і пропонують включати її до базового STEM-ядра шкільної освіти.

(Grover, Pea, 2013) аналізує стан досліджень CT у школі, окреслюючи ключові напрями: визначення CT, інтеграція в шкільні курикулуми, оцінювання та підготовка вчителів. Автори визначають, що ефективне впровадження CT вимагає: системних змін у навчальних планах; розробки інструментів оцінювання CT; спеціалізованої підготовки вчителів, які мають поєднувати знання інформатики зі знаннями педагогіки.

У низці новіших досліджень (Kong et al., 2020), CT пов'язують із міжпредметною інтеграцією (STEM, математика, природничі науки) та розробкою вчительських програм професійного розвитку, орієнтованих на поєднання концептів програмування, педагогіки та методики CT.

Українські дослідження активно інтегрують міжнародну концепцію CT у підготовку вчителів інформатики. (Тихонова, 2018) визначає CT як ядро сучасної інформатичної освіти, (Медведєва, 2021) показує, що формування CT можливе через використання Scratch і Python, а також через діяльнісне навчання.

Українські дослідники (Биков, Яцишин, 2019), (Литвин, Руденко, 2023), (Umanets, Shakhina, Rozputnia, 2025) також розглядають формування комп'ютерного мислення як обов'язковий компонент професійної підготовки майбутніх учителів інформатики. В їхніх роботах акцент робиться на тому, що алгоритмічне мислення має формуватися системно: через задачі різного рівня складності, через роботу з реальними даними, моделювання, розробку програмних продуктів. Вони позиціонують CT як універсальний спосіб розв'язування проблем, який виходить за межі програмування, але формується саме через нього.

У цих дослідженнях підкреслюється, що програмування стає засобом розвитку системного, алгоритмічного й логічного мислення, а не лише технічною навичкою.

Узагальнюючи, можна стверджувати, що вітчизняні й зарубіжні дослідження трактують CT як основу компетентності вчителя інформатики в сфері програмування та як необхідну передумову ефективного навчання школярів.

Методичні аспекти підготовки педагогів з програмування охоплюють

володіння сучасними мовами та середовищами програмування, уміння проектувати навчальні заняття та адаптувати складні технічні концепції до віку учнів.

(Gal-Ezer, Stephenson, 2010) доводять, що якість шкільної інформатичної освіти критично залежить від надійно організованої підготовки вчителів інформатики та їхнього глибокого розуміння предмета.

(Hazzan, Lapidot, Ragonis, 2014) пропонують концептуальну рамку й детальні методичні рекомендації щодо викладання інформатики та програмування, включно з активними методами (CS Unplugged, ігри, візуалізація алгоритмів, концептуальні карти).

Сучасні дослідження систем підготовки вчителів інформатики у різних країнах показують широкий спектр моделей підготовки– від короткострокових курсів до повноцінних магістерських програм, а також різний ступінь інтеграції СТ та програмування у програми педагогічної освіти (Chouhan, Ragavan, Karkare, 2025).

(Kong et al., 2020) описують програму професійного розвитку для вчителів, спрямовану на формування компетентностей у сфері СТ, програмування та відповідної педагогіки; результати показують приріст як знань, так і впевненості вчителів щодо впровадження СТ у класі.

Праці (Биков, Яцишин, 2019), (Литвин, Руденко, 2023) дають підстави стверджувати, що цифровізація освіти суттєво змінила вимоги до вчителя інформатики і формування його професійної компетентності.

У дослідженні (Буйницької, 2021) доведено, що вчитель повинен оволодіти: хмарними сервісами, засобами створення цифрових навчальних матеріалів, цифровими освітніми платформами, інструментами організації дистанційного та змішаного навчання.

Методичні рекомендації МОН України (2020) та модульні програми НУШ підтверджують офіційний перехід до цифрових компетентностей учителя як ключових.

Дослідження психологічних труднощів є фундаментальними для розуміння того, чому засвоєння програмування часто викликає проблеми.

Опанування програмування значною мірою залежить від психологічних факторів, зокрема когнітивної сфери, емоційної регуляції та соціальної взаємодії. Сучасні дослідження демонструють, що здатність студента аналізувати, декомпозувати та осмислювати код безпосередньо корелює з якістю сформованих ментальних моделей (Hermans, 2020). Водночас емоційні стани, зокрема тривожність чи фрустрація, можуть істотно впливати на продуктивність (Bergersen et al., 2020).

Зростає також увага науковців до синдрому самозванця серед студентів CS-напряму (Ramachandran et al., 2022), а також до ролі метакогнітивної підтримки у процесі розв’язання задач (Loksa et al., 2016). Отже, психологічний вимір є ключовим у створенні ефективного навчального середовища.

Під час програмування студенти мають паралельно утримувати у свідомості синтаксис, логіку, структури даних та поточні значення змінних. З огляду на це, когнітивне навантаження є одним із найважливіших факторів, що визначають успішність навчання. Огляд (Zhi, Lishinski, 2021) показує, що надмірне навантаження призводить до поверхневого розуміння коду й численних помилок.

(Becker, Quille, 2019) класифікують типові помилки новачків, пов’язані з

перевантаженням робочої пам'яті й недостатньою структуризацією знань. (Hermans, 2020) підкреслює, що труднощі з утриманням одночасних ментальних операцій є фундаментальною перешкодою для новачків.

Ментальні моделі – це внутрішні репрезентації того, як працює код. (Busjahn, Schulte, 2022) виявили, що новачки часто формують неповні або неправильні моделі, що призводить до повторюваних концептуальних помилок.

(Prather et al., 2018) показали, що особливо складним для студентів є розуміння змінних і принципу присвоювання, що свідчить про глибинні розриви в когнітивних уявленнях. Згідно з (Wang et al., 2019), внутрішня мотивація є основним фактором успішного навчання. Дослідження (Loksa et al., 2016) показує, що метакогнітивні стратегії, такі як планування, моніторинг і рефлексія, значно підвищують якість виконання програмних задач.

Психологічні аспекти навчання програмуванню є ключовими для формування глибокого та стійкого розуміння програмних концепцій. Когнітивні та емоційні фактори, метакогнітивні стратегії, соціальна підтримка й якісний зворотний зв'язок формують комплексний механізм успішності. Сучасні дослідження показують, що інтеграція психологічних підходів у навчальні програми здатна істотно підвищити результативність.

Таким чином, навчання програмуванню майбутніх учителів інформатики стає основою їхньої професійної компетентності, здатності впроваджувати інноваційні технології та формувати цифрову грамотність учнів.

Виокремлення невирішених раніше частин загальної проблеми.

Основна проблема полягає у забезпеченні інваріантності змісту навчання програмуванню для майбутніх вчителів інформатики, незалежно від вибору конкретних засобів інформаційної технології. У результаті такого підходу здобувачі освіти повинні не просто опановувати окремі інструменти чи вивчати певну мову програмування. Головна мета – навчити їх впорядковувати та структурувати дані, визначати оптимальні способи їх представлення, освоювати принципи інформаційного моделювання, використовуючи технології програмування.

Для досягнення цієї мети необхідно визначити інваріантну складову змісту навчання програмуванню і вибрати відповідний підхід до його реалізації. Це особливо важливо з огляду на різний рівень підготовки у галузі програмування серед здобувачів освіти, які навчаються на спеціальності «Інформатика». Така неоднорідність вимагає застосування різнорівневих стратегій, аби вирівняти базовий рівень знань здобувачів освіти без значних часових затрат.

На першому етапі навчання програмуванню варто використовувати прості та доступні інструменти, освоєння яких не передбачає значних зусиль зі сторони здобувачів освіти. Це дозволить створити сприятливі умови для подальшого поглиблення знань і впевненого освоєння більш складних технологій.

Формулювання цілей (мета) статті, постановка завдання. Метою статті є окреслення особливостей підготовки майбутніх учителів інформатики з програмування, аналіз ефективних педагогічних технологій та визначення компетентностей, яких вони мають набути у процесі навчання.

Визначення мети зумовило такі завдання:

– дослідити специфіку навчання майбутніх учителів інформатики програмуванню;

– розглянути труднощі, які виникають у студентів під час опанування

програмування, та шляхи їх подолання;

– визначити структуру та зміст підготовки з програмування майбутніх вчителів інформатики

Виклад основного матеріалу дослідження з повним обґрунтуванням отриманих наукових результатів. Дисципліна «Програмування» відіграє важливу роль у професійній підготовці майбутніх учителів інформатики, забезпечуючи базові знання про процес розробки програмного забезпечення та сприяючи формуванню необхідних компетенцій для успішної педагогічної діяльності. Вивчення цього предмета дозволяє майбутнім педагогам освоїти широкий спектр навичок, серед яких:

– отримання фундаментальних знань у сфері інформаційно-комунікаційних технологій;

– розвиток алгоритмічного та логічного мислення;

– набуття методичних знань для ефективної організації навчального процесу;

– удосконалення універсальних професійних компетенцій;

– підготовка до інтеграції сучасних технологій у освітній процес;

– формування здатності до впровадження інновацій та реалізації проектної діяльності.

В результаті навчання, майбутній учитель інформатики має володіти: мовами програмування (Pascal, Python, Java, JavaScript, C++, Scratch); структурою даних і алгоритмами; принципами об'єктно-орієнтованого програмування та базами даних; основами веб-розробки.

Уміти: використовувати хмарні сервіси (Google Classroom, Moodle); застосовувати віртуальні лабораторії; працювати з навчальними онлайн-платформами (Code.org, Replit, Codecademy, GitHub Classroom тощо).

Розглянемо структуру та зміст підготовки з програмування майбутніх вчителів інформатики.

Професійну підготовку доцільно будувати поетапно.

Перший етап включає теоретико-фундаментальну підготовку яка базується на викладанні освітніх компонентів, таких як: «Вища математика», «Дискретна математика», «Вступ до спеціальності», «Прикладне та системне забезпечення освітнього процесу», «Методи обчислень», «Шкільний курс інформатики та методика його навчання», «Програмування». (https://sites.google.com/view/kaf-inf-hgpa).

Наступний етап це практична підготовка, яка формується через лабораторні роботи з мов програмування; навчальні проєкти (створення ігор, чат-ботів, веб-сторінок); командну роботу над програмними системами.

Третій етап включає методичну підготовку: методику викладання інформатики; дидактичні моделі навчання програмування; педагогічний дизайн навчальних матеріалів; використання інтерактивних сервісів для пояснення алгоритмів.

Четвертий етап це практика. Під час практики студент: проводить уроки з програмування у ЗЗШО та під час літньої практики; створює власні дидактичні ресурси під час інформаційно-технологічної практики; аналізує освітні платформи та методи роботи з учнями; діагностує сформованість ІКТ-компетентностей школярів.

Дисципліна «Програмування» викладається у другому, третьому, четвертому та п’ятому семестрі в Харківській гуманітарно-педагогічної академії (https://sites.google.com/view/kaf-inf-hgpa, 2025).

Протягом вивчення даного курсу здобувачам освіти необхідно систематизувати знання про технології програмування, оволодіти сучасними засобами розробки комп'ютерних програм, сформувати практичні навички роботи з сучасними середовищами програмування.

Вибір мови програмування визначається навчальними завданнями освоєння дисципліни «Програмування». Важливо враховувати, що мова програмування – це всього лише інструмент, засіб технології програмування, і вивчення всіх його можливостей не є основною метою навчання програмуванню майбутніх вчителів інформатики. На початковому етапі важлива інваріантність навчання щодо технічних подробиць мови та середовища програмування. Тому обрана мова програмування, з одного боку, повинна мати засоби для адекватного відображення необхідних для засвоєння фундаментальних понять програмування, а з іншого – мати мінімально достатній набір засобів для практичного відпрацювання досліджуваного матеріалу.

Початок навчання програмуванню майбутніх вчителів інформатики доцільно реалізувати на основі використання мови програмування Pascal. Такий вибір ґрунтується на практиці застосування цієї мови саме як навчальної, а також можливості використання безкоштовного середовища програмування Lazarus, що дозволяє створювати програмні продукти комерційного призначення. Сучасні версії мови володіють зручною оболонкою для роботи, мають розширений склад команд, але при цьому консервативна структура програми дозволяє навчити студентів таким поняттям, як змінна, тип змінної, оператори розгалуження, циклу, паралельно вводячи відповідні визначення, чітко описуючи призначення і принцип роботи відповідних операторів. Вивчення складу і можливостей мови програмування Pascal закладає основу для успішного освоєння інших мов програмування.

Крім мови програмування Pascal, в рамках дисципліни «Програмування» здобувачі освіти також починають вивчати мову програмування C++, яка надає значно більше можливостей враховувати архітектуру комп'ютера при розробці програм, що дозволяє поглибити знання програмування, вивчивши основні структури даних та їх реалізацію: стек, черга і т.д. Крім того, з мови програмування C++ слід починати вивчення основ об'єктно-орієнтованого програмування. Однак ці можливості стають доступні і можуть бути усвідомлено використані здобувачами освіти лише при певному рівні володіння основами програмування, що досягається при освоєнні мови програмування Pascal на самому початку вивчення дисципліни. Паралельний розгляд програмних реалізацій типових алгоритмів на мовах програмування Pascal і C++ дозволяє продемонструвати їх відмінності, схожість і обмеження.

Таким чином, успішне засвоєння розділів дисципліни «Програмування» на самому початку навчання за ОПП «Інформатика» закладає міцну основу для продовження навчання програмуванню в рамках інших дисциплін з урахуванням їх особливостей і вивчення таких мов програмування, як Python, Java, VBA, PHP, Scratch та ін. При цьому мови програмування починають розглядатися не тільки як об'єкт вивчення, але і як засіб вивчення цілого ряду дисциплін, включаючи

методику викладання інформатики.

Після вивчення основ програмування доцільно перейти до курсу вивчення програмування на мові Python, який можна використовувати для розгляду основ функціонального програмування, познайомивши здобувачів освіти з цією парадигмою. При цьому Python може активно використовуватися як засіб при вивченні цілого ряду дисциплін, наприклад основ штучного інтелекту. Дана дисципліна прекрасно підходить для розгляду логічної парадигми програмування і логічних мов програмування.

Особливу роль серед мов програмування займає Scratch, яку варто детально опрацювати разом із майбутніми вчителями інформатики. Хоч цей інструмент здебільшого використовується для навчання школярів і молодших учнів основам програмування, він демонструє високу ефективність у роботі з деякими робототехнічними конструкторами. У таких конструкторах часто спостерігаються як розширення для Scratch, так і мови, схожі за принципами.

Також важливо опанувати мови програмування, які використовуються у веб-розробці, таких як PHP чи JavaScript. Спираючись на базові навички здобувачів освіти у роботі з іншими мовами програмування, можна не лише засвоїти ці інструменти, а й глибше вникнути в аспекти веб-програмування. У межах дисципліни «Веб-дизайн та веб-програмування» доцільно розглядати основи використання SQL для роботи із запитами до баз даних.

Окрім того, варто звернути увагу на позааудиторну діяльність і елективні курси, що дозволяють здобувачам освіти самостійно обирати технології та мови програмування для поглибленого вивчення. Такі курси допомагають розширити їхні знання відповідно до індивідуальних інтересів і пріоритетів. Зосереджуючи увагу на практичному підході, елективні курси дають змогу опановувати середовище Visual Studio, працювати з Visual Basic та його модифікацією Visual Basic for Application, а також більш детально займатися вивченням SQL, мови R тощо.

При цьому слід пам'ятати, що вчитель інформатики не має ставати професійним програмістом. Відповідно, програма підготовки повинна акцентувати увагу на методичних аспектах викладання алгоритмізації та програмування, формуванні чіткої термінології, освоєнні основних алгоритмічних структур і розвитку навичок розв'язання задач зі шкільного курсу інформатики.

Висновки та перспективи подальших розвідок у даному напрямку. З огляду на зазначені аспекти, програмування як предмет вивчення та як інструмент навчання має залишатися невід'ємною частиною підготовки майбутніх учителів інформатики. Курси, пов'язані з програмуванням, повинні мати чітку професійно-педагогічну орієнтацію, забезпечувати здобувачам освіти можливість набуття практичного досвіду та готувати їх до самостійного вдосконалення знань і навичок у відповідній професійній сфері.

Основні висновки та перспективи досліджень у цьому напрямі зводяться до того, що підготовка вчителів інформатики у галузі програмування має розглядатись як комплексний процес. Він охоплює розвиток професійних, методичних, педагогічних і цифрових компетенцій. Ефективність такого процесу базується на гармонійному поєднанні теоретичних знань із практичною діяльністю, широкому використанні цифрових технологій, застосуванні інноваційних підходів до навчання та акцентуванні на реально важливих освітніх потребах учнів.

Сучасний учитель інформатики повинен бути не тільки кваліфікованим програмістом, а й педагогом-модератором, здатним організовувати навчальний процес, стимулювати пізнавальну активність учнів, розвивати їхню творчість і готувати до викликів цифрового суспільства.

Впровадження підходів до викладання програмування майбутнім учителям інформатики у межах всієї освітньої програми передбачає визначення ключових мов програмування, а також послідовність і поглибленість їх вивчення. Це дозволяє створити основу для фундаментальної підготовки в цій галузі, яка є важливим компонентом інформаційної культури майбутніх педагогів. Такий підхід безпосередньо впливає на якість їхньої професійної діяльності в школах, забезпечуючи ефективну організацію освітнього процесу.

ДЖЕРЕЛА І ЛІТЕРАТУРА

Barr, V., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. Learning & Leading with Technology, 38(6), 20–23.

Becker, B. A., & Quille, K. (2019). A taxonomy of novice programming errors. In Proceedings of the 50th ACM Technical Symposium on Computer Science Education (pp. 311–317). ACM. https://doi.org/10.1145/3287324.3287432

Bergersen, G. R., Arisholm, E., & Sjøberg, D. I. K. (2020). Does emotional state influence programming performance? Empirical Software Engineering, 25, 4922–4955. https://doi.org/10.1007/s10664-019-09794-9

Busjahn, T., & Schulte, C. (2022). How students understand and read code: A systematic review 2015–2022. Education and Information Technologies, 27, 10957–10980. https://doi.org/10.1007/s10639-022-10957-7

Gal-Ezer, J., & Stephenson, C. (2010). Computer science teacher preparation: A global review of existing programs and practices. ACM Inroads, 1(1), 54–61. https://doi.org/10.1145/1721933.1721952

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. Educational Researcher, 42(1), 38–43. https://doi.org/10.3102/0013189X12463051

Hazzan, O., Lapidot, T., & Ragonis, N. (2015). Guide to teaching computer science: An activity-based approach. Springer. https://doi.org/10.1007/978-1-4471-6639-7

Hermans, F. (2020). The programmer's brain: What every programmer needs to know about cognition. Manning Publications. https://www.manning.com/books/the-programmers-brain

Kong, S. C., Lai, M., & Sun, D. (2020). Teacher development in computational thinking: Design and implementation of professional development programs. Computers & Education, 150, 103–113. https://doi.org/10.1016/j.compedu.2020.103833

Loksa, D., Ko, A. J., et al. (2016). Metacognitive support for learning to program. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (pp. 1449–1461). ACM. https://doi.org/10.1145/2858036.2858252

Prather, J., Pettit, R., Liao, S. N., et al. (2018). What students don't understand about variables: An analysis of think-aloud interviews. In Proceedings of the 2018 ACM Conference on International Computing Education Research (pp. 10–18). ACM.

https://doi.org/10.1145/3230977.3230999

Ramachandran, S., Lishinski, A., et al. (2022). Impostor phenomenon in computer science students. ACM Transactions on Computing Education, 22(4), Article 51. https://doi.org/10.1145/3488370

Tykhonova T. , Koshkina H. (2018). Computational thinking як сучасний освітній тренд. Електронне наукове фахове видання "Відкрите освітнє е-середовище сучасного університету", (5), 210-221. https://openedu.kubg.edu.ua/journal/index.php/openedu/article/view/158

Umanets V. O. Training future computer science teachers to use artificial intelligence technologies in the educational process (2024)/ V. O. Umanets, I. Y. Shakhina, B. M. Rozputnia // Сучасні інформаційні технології та інноваційні методики навчання в підготовці фахівців: методологія, теорія, досвід, проблем. Вип. 72. С. 162-170.

Wang, T., Shen, C., et al. (2019). Understanding student motivation in CS1 courses using self-determination theory. In Proceedings of the 50th ACM Technical Symposium on Computer Science Education (pp. 1057–1063). ACM. https://doi.org/10.1145/3287324.3287484

Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33–35.

Wing, J. M. (2010). Computational thinking: What and why? The Link Magazine, Carnegie Mellon University.

Zhi, R., & Lishinski, A. (2021). Cognitive load in introductory programming education: A review of research trends and future directions. ACM Transactions on Computing Education, 21(4), Article 45. https://doi.org/10.1145/3446593

Биков, В. Ю., & Яцишин, А. В. (2019). Цифрова трансформація освіти і науки: теорія і практика. Київ: ІІТЗН НАПН України. 250 с.

Литвин А., Руденко Л.(2025). Можливості штучного інтелекту в освітній діяльності вищої школи. Міжнародна науково-практична конференція «Цифрова трансформація освіти: інновації, виклики та можливості» збірник матеріалів. Кропивницький. С. 186-190

Медведєва М.. та ін. (2021). Елементи підготовки майбутніх учителів інформатики до застосування технології формування Computational Thinking. Физико-математическое образование, 1 (27): 67-75.

Методичні рекомендації щодо викладання інформатики у 2020-2021 н.р. згідно Листа МОН України № 1/9-430 від 11.08.2020

Система педагогічного проєктування інформаційно-освітнього середовища для здійснення підготовки майбутніх соціальних педагогів (2021): монографія / О. П. Буйницька. К.: Київ. Ун-т ім. Б. Грінченка, 2021. 568 с.

REFERENCES

Barr, V., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. Learning & Leading with Technology, 38(6), 20–23.

Becker, B. A., & Quille, K. (2019). A taxonomy of novice programming errors. In Proceedings of the 50th ACM Technical Symposium on Computer Science Education (pp. 311–317). ACM. https://doi.org/10.1145/3287324.3287432

Bergersen, G. R., Arisholm, E., & Sjøberg, D. I. K. (2020). Does emotional state influence programming performance? Empirical Software Engineering, 25,

4922–4955. https://doi.org/10.1007/s10664-019-09794-9

Busjahn, T., & Schulte, C. (2022). How students understand and read code: A systematic review 2015–2022. Education and Information Technologies, 27, 10957–10980. https://doi.org/10.1007/s10639-022-10957-7

Gal-Ezer, J., & Stephenson, C. (2010). Computer science teacher preparation: A global review of existing programs and practices. ACM Inroads, 1(1), 54–61. https://doi.org/10.1145/1721933.1721952

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. Educational Researcher, 42(1), 38–43. https://doi.org/10.3102/0013189X12463051

Hazzan, O., Lapidot, T., & Ragonis, N. (2015). Guide to teaching computer science: An activity-based approach. Springer. https://doi.org/10.1007/978-1-4471-6639-7

Hermans, F. (2020). The programmer's brain: What every programmer needs to know about cognition. Manning Publications. https://www.manning.com/books/the-programmers-brain

Kong, S. C., Lai, M., & Sun, D. (2020). Teacher development in computational thinking: Design and implementation of professional development programs. Computers & Education, 150, 103–113. https://doi.org/10.1016/j.compedu.2020.103833

Loksa, D., Ko, A. J., et al. (2016). Metacognitive support for learning to program. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (pp. 1449–1461). ACM. https://doi.org/10.1145/2858036.2858252

Prather, J., Pettit, R., Liao, S. N., et al. (2018). What students don't understand about variables: An analysis of think-aloud interviews. In Proceedings of the 2018 ACM Conference on International Computing Education Research (pp. 10–18). ACM. https://doi.org/10.1145/3230977.3230999

Ramachandran, S., Lishinski, A., et al. (2022). Impostor phenomenon in computer science students. ACM Transactions on Computing Education, 22(4), Article 51. https://doi.org/10.1145/3488370

Tykhonova T., Koshkina H. (2018). Computational thinking як сучасний освітній тренд. Електронне наукове фахове видання "Відкрите освітнє е-середовище сучасного університету", (5), 210-221. https://openedu.kubg.edu.ua/journal/index.php/openedu/article/view/158

Umanets V. O. Training future computer science teachers to use artificial intelligence technologies in the educational process (2024)/ V. O. Umanets, I. Y. Shakhina, B. M. Rozputnia // Сучасні інформаційні технології та інноваційні методики навчання в підготовці фахівців: методологія, теорія, досвід, проблем. Вип. 72. С. 162-170.

Wang, T., Shen, C., et al. (2019). Understanding student motivation in CS1 courses using self-determination theory. In Proceedings of the 50th ACM Technical Symposium on Computer Science Education (pp. 1057–1063). ACM. https://doi.org/10.1145/3287324.3287484

Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33–35.

Wing, J. M. (2010). Computational thinking: What and why? The Link

Magazine, Carnegie Mellon University.

Zhi, R., & Lishinski, A. (2021). Cognitive load in introductory programming education: A review of research trends and future directions. ACM Transactions on Computing Education, 21(4), Article 45. https://doi.org/10.1145/3446593

Bykov, V. Yu., & Yatsyshyn, A. V. (2019). Tsyfrova transformatsiia osvity i nauky: teoriia i praktyka. Kyiv: IITZN NAPN Ukrainy. 250 s. [in Ukrainian].

Lytvyn A., Rudenko L.(2025). Mozhlyvosti shtuchnoho intelektu v osvitnii diialnosti vyshchoi shkoly. Mizhnarodna naukovo-praktychna konferentsiia «Tsyfrova transformatsiia osvity: innovatsii, vyklyky ta mozhlyvosti» zbirnyk materialiv. Kropyvnytskyi. S. 186-190 [in Ukrainian].

Medvedieva M.. ta in. (2021). Elementy pidhotovky maibutnikh uchyteliv informatyky do zastosuvannia tekhnolohii formuvannia Computational Thinking. Fyzyko-matematycheskoe obrazovanye, 1 (27): 67-75. [in Ukrainian].

Metodychni rekomendatsii shchodo vykladannia informatyky u 2020-2021 n.r. zghidno Lysta MON Ukrainy № 1/9-430 vid 11.08.2020 [in Ukrainian].

Systema pedahohichnoho proiektuvannia informatsiino-osvitnoho seredovyshcha dlia zdiisnennia pidhotovky maibutnikh sotsialnykh pedahohiv (2021): monohrafiia / O. P. Buinytska. K.: Kyiv. Un-t im. B. Hrinchenka, 2021. 568 s. [in Ukrainian].

АНОТАЦІЯ

Особливості підготовки викладачів інформатики в галузі програмування являють собою ключовий аспект їх професійної освіти. Необхідність таких знань зумовлена стрімким розвитком цифрових технологій та їх широким впровадженням в освітній процес. Успішна підготовка майбутніх педагогів передбачає формування у них не тільки базових навичок програмування, але і здатності застосовувати отримані знання в контексті викладання інформатики.

Особлива увага в процесі навчання приділяється таким аспектам, як розвиток алгоритмічного мислення, освоєння різних мов програмування, а також розуміння принципів проектування і розробки програмного забезпечення. Важливим елементом підготовки є акцент на практичне застосування знань, що дозволяє майбутнім вчителям ефективно використовувати сучасні інструменти і технології в процесі навчання.

Крім того, підготовка охоплює методологічні підходи до викладання програмування. Це включає розробку навчальних матеріалів, вибір адекватних методик навчання і створення підтримуючого освітнього середовища, що сприяє розвитку творчого потенціалу учнів. Інтеграція теоретичних і практичних аспектів навчання дозволяє формувати у студентів глибоке розуміння предмета і впевненість у його викладанні.

Таким чином, підготовка майбутніх вчителів інформатики з програмування вимагає комплексного підходу, що поєднує розвиток професійних навичок з формуванням педагогічної компетентності, що сприяє забезпеченню високої якості освітнього процесу в умовах сучасних вимог.

Ключові слова: *підготовка вчителів, інформатика, програмування, ІКТ-компетентності, професійна освіта прикладне програмне забезпечення*