УДК 378.147

Дядюшкін Роман
здобувач третього (освітньо наукового) рівня вищої освіти
науково-навчального інституту педагогіки та управління
Житомирський державний університет
імені Івана Франка
м. Житомир, Україна

ІНТЕГРАЦІЯ ТЕОРЕТИЧНИХ ТА ПРАКТИЧНИХ КОМПОНЕНТІВ У ПІДГОТОВКУ ІТ-ФАХІВЦІВ В ЕПОХУ ГЕНЕРАТИВНОГО ШІ

Анотація. Інтеграція генеративного штучного інтелекту (ШІ) у підготовку фахівців з інформаційних технологій (ІТ) являє собою фундаментальний зсув від традиційного програмування до агентного кодування, де розробники оркеструють автономними ШІ агентами за допомогою намірів, виражених природною мовою [3, с. 1-2]. Джерела з аналізом впливу цих технологій та розробки обґрунтованих рекомендацій надають комплексну педагогічну та професійно орієнтовану перспективу. Метою дослідження є аналіз впливу генеративного штучного інтелекту на розвиток практичних навичок майбутніх розробників та формування сучасних дидактичних основ і обґрунтованих рекомендацій для їхньої професійної підготовки.

Ключові слова: генеративний ШІ, агентне кодування, підготовка ІТ-фахівців, когнітивне розвантаження.

Перехід до кодування за допомогою ШІ приносить значні педагогічні ризики поряд із підвищенням ефективності. Зокрема ілюзія компетентності та когнітивне розвантаження проявляються тоді, коли ШІ бере на себе складні когнітивні завдання. Це може призвести до зміщеного мислення або нейронного режиму очікування, коли студенти відчувають зниження активності мозку та

гірше запам'ятовування [4, с. 2-3]. Подібна ситуація створює ілюзію компетентності, що корениться в ефекті Даннінга Крюгера, коли здобувачі освіти помилково приймають здатність генерувати функціональний код за здатність насправді розуміти та змінювати його [1, с. 4]. Студенти часто надмірно покладаються на когнітивне розвантаження, досягаючи поверхневої майстерності, але зазнають невдачі у разі припинення допомоги з боку ШІ [1, с. 4].

Ця ситуація підкреслює ризик формування ілюзії компетентності, коли студенти, надмірно покладаючись на ШІ, досягають поверхневої майстерності, але втрачають глибоке розуміння коду.

Парадокс розвитку навичок полягає у тому, що інструменти штучного інтелекту загрожують традиційному шляху набуття досвіду молодшими розробниками. Історично інженери отримували досвід шляхом впровадження невеликих компонентів на рівні функцій перед переходом до архітектури на рівні системи [3, с. 9]. Оскільки ШІ тепер автоматизує ці базові завдання, програмісти початківці втрачають свою основну базу експериментального навчання. Це спричиняє ерозію навичок та викликає занепокоєння щодо методології підготовки майбутніх експертів [3, с. 9-10].

Таким чином, автоматизація базових завдань програмування інструментами ШІ створює парадокс, коли початківці розробники втрачають критично важливу базу експериментального навчання, необхідну для набуття досвіду.

Щоб узгодити освіту з професійними реаліями, педагогіка має відображати те, як насправді використовують ШІ досвідчені професіонали. Професійні розробники програмного забезпечення не займаються вайб кодуванням, тобто практикою некритичної генерації за допомогою підказок, коли користувач ігнорує базовий код [3, с. 1-2]. Натомість професіонали суворо контролюють ШІ агентів. Досвідчені розробники ретельно планують роботу перед впровадженням, перевіряють усі результати агентів і покладаються на свій глибокий досвід розробки програмного забезпечення для скерування ШІ [5, с.

10-14]. Таким чином підготовка майбутніх розробників має перейти від навчання запам'ятовування синтаксису до формування професійних навичок вищого порядку. До таких навичок належать архітектурне мислення, формулювання проблем, перевірка коду та налагодження, які стають основними вузькими місцями та найважливішими навичками у нових умовах [3, с. 9-12].

Освітній процес має відображати професійну практику, переорієнтовуючи підготовку з вивчення синтаксису на формування навичок вищого порядку, таких як архітектурне мислення та суворий контроль результатів ШІ.

Для успішної інтеграції міцних теоретичних основ із передовими інструментами ШІ навчальні заклади впроваджують структуровані науково обґрунтовані педагогічні основи. Концепція інтеграції штучного інтелекту в проєктну роботу є сучасною дидактичною концепцією, спеціально розробленою для вбудовування ШІ у навчання на основі проєктів із використанням життєвого циклу розробки [6, с. 317]. Ця концепція розглядає ШІ не як заміну людських здібностей, а як віртуального товариша по команді або колективний інтелект [6, с. 311, 319]. Подібний підхід вимагає від студентів використання оцінювального судження для критичної оцінки згенерованих ШІ результатів на точність, відповідність етичним нормам і релевантність, що перетворює використання ШІ на активну вправу з критичного мислення [6, с. 311, 323].

Впровадження концепції інтеграції ШІ у проєктну роботу є сучасною дидактичною основою, що перетворює використання ШІ на активну вправу з критичного мислення та оцінювального судження.

Щоб оптимізувати когнітивне навантаження, ШІ не слід використовувати однаково на всіх етапах навчання, тому рекомендується поетапний підхід. Протягом перших шести тижнів використання ШІ суворо обмежене для забезпечення набуття студентами фундаментальних схем та подолання початкових труднощів із синтаксисом і семантикою [1, с. 11]. На наступному етапі прискорення з використанням навчальних підпорок ШІ впроваджується для генерації шаблонного коду, але студенти повинні щотижня вручну рефакторити результати ШІ для підтвердження розуміння [1, с. 11]. Під час фази

критичного огляду дозволено повне використання ШІ, але освітній фокус зміщується на виявлення пастки галюцинацій, що навчає студентів працювати як технічні аудитори, здатні ідентифікувати тонкі логічні помилки та вразливості безпеки [1, с. 11-12].

Для оптимізації когнітивного навантаження та запобігання надмірній залежності від ШІ рекомендовано поетапний підхід, який поступово збільшує використання інструментів ШІ, паралельно розвиваючи навички технічного аудиту.

Для боротьби з надмірною самовпевненістю внаслідок швидких відповідей ШІ інструменти мають запровадити педагогічне тертя [4, с. 2]. Спеціальні дидактичні інструменти забезпечують навчання через пояснення своєму майбутньому Я, вимагаючи від студентів пояснювати концепції власною мовою. Також застосовується контрастне навчання, що змушує взаємодіяти з контраргументами та альтернативними точками зору, і керовані підказки, які надають навчальні підпори замість негайних рішень [4, с. 5-7].

Для боротьби з надмірною самовпевненістю, викликаною швидкими відповідями ШІ, необхідно запроваджувати «педагогічне тертя» через механізми навчання через пояснення та контрастне навчання.

Висновки. На основі синтезу дидактичних концепцій та промислових практик сформовано ключові рекомендації щодо інтеграції теоретичних та практичних компонентів. Навчальну програму необхідно переорієнтувати на архітектурне мислення та перевірку коду, зменшивши акцент на генерації сирого синтаксису та збільшивши увагу до системної архітектури і ретельного перегляду результатів. Студенти повинні бути навчені перевіряти, тестувати та налагоджувати результати роботи ШІ так само, як це роблять досвідчені професіонали. Доцільно запровадити раннє навчання на основі проєктів для залучення студентів до роботи зі складними системами, оскільки це змушує практикувати декомпозицію системи, тестування та інтеграцію, тобто ті навички, які не можуть бути повністю автоматизовані. Викладачі мають формувати сприйняття ШІ агентів як інструментів, що вимагають людського

нагляду, суворих операційних меж та безпечних практик кодування. Оцінювання студентів повинно включати такі показники як розрив у поясненні, який вимірює невідповідність між складністю згенерованого коду та фактичним концептуальним розумінням студента.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ:

1. Aiersilan A. The vibe-check protocol: quantifying cognitive offloading in AI programming. URL: https://arxiv.org/abs/2601.02410.
2. Bakal G. Knowledge activation: AI skills as the institutional knowledge primitive for agentic software development. 2026. URL: https://doi.org/10.48550/arXiv.2603.14805.
3. Coding with AI: from a reflection on industrial practices to future computer science and software engineering education / H.-F. Chang et al. 2025. URL: https://doi.org/10.48550/arXiv.2512.23982.
4. Eleftheriou E., Pallis G., Constantinides M. Confidence without competence in ai-assisted knowledge work. URL: https://arxiv.org/abs/2604.09444.
5. Professional software developers don't vibe, they control: AI agent use for coding in 2025 / R. Huang et al. URL: https://arxiv.org/abs/2512.14012.
6. Project-work Artificial Intelligence Integration Framework (PAIIF): Developing a CDIO-based framework for educational integration / S. Nikolic et al. STEM education. 2025. Vol. 5, no. 2. P. 310–332. URL: https://doi.org/10.3934/steme.2025016.