Журнал «Перспективи та інновації науки»
(Серія «Педагогіка», Серія «Психологія», Серія «Медицина»)
№ 11(45) 2024

УДК 378.147

https://doi.org/10.52058/2786-4952-2024-11(45)-888-900

Ткачук Галина Володимирівна доктор педагогічних наук, доцент, професор кафедри інформатики і інформаційно-комунікаційних технологій, Уманський державний педагогічний університет імені Павла Тичини, м. Умань, тел.: (04744) 3-45-82, https://orcid.org/0000-0002-6926-1589

МЕТОДИЧНІ ПІДХОДИ ДО ВИКОРИСТАННЯ СЕРЕДОВИЩ ПРОГРАМУВАННЯ У ПРОЦЕСІ ПІДГОТОВКИ МАЙБУТНІХ УЧИТЕЛІВ ІНФОРМАТИКИ

Анотація. Стрімкий розвиток цифрових технологій та зростаюча потреба в інтеграції інформаційних технологій в освітній процес ставлять перед нами актуальну проблему дослідження методичних підходів до використання середовищ програмування у підготовці майбутніх учителів інформатики. Використання таких середовищ допомагає педагогам глибше засвоїти складні концепції програмування та створювати інтерактивні, практико-орієнтовані уроки для учнів. Крім того, це сприяє розвитку творчого мислення, критичного аналізу та здатності швидко адаптуватися до нових технологічних умов — важливих компетенцій для вчителів у сучасному освітньому середовищі.

У статті розглянуто методичний підхід до вивчення програмування як сукупність методів, засобів та форм навчання, які забезпечують ефективний освітній процес. Виявлено, що під час підготовки майбутніх учителів інформатики можна організувати різні форми навчання, зокрема індивідуальну та групову роботу, виконання практичних завдань та дистанційне навчання. В межах кожного виду діяльності доцільно використовувати різні середовища програмування.

Особливу увагу в роботі приділено методам навчання, зокрема пояснювально-ілюстративному, практичному, проєктному та дослідницькому підходам. Кожен з цих методів забезпечує унікальний підхід до навчання, сприяючи розвитку навичок розв'язання проблем, критичного мислення та творчості. Інтеграція теорії з практикою дозволяє студентам глибше зрозуміти матеріал та підготуватися до професійної діяльності. Отже, ці методи не лише покращують навчальний процес, але й формують впевненість студентів у своїх здібностях.

Ключові слова: методичні підходи, середовище програмування, майбутні учителі інформатики, дистанційне навчання, Replit, Thonny.

Tkachuk Halyna Volodymyryvna Doctor of Pedagogical Sciences, Associate Professor, Professor of the Department of Informatics and Information and Communication Technologies, Pavlo Tychyna Uman State Pedagogical University, Uman, tel.: (04744) 3-45-82, https://orcid.org/0000-0002-6926-1589

Журнал «Перспективи та інновації науки»
(Серія «Педагогіка», Серія «Психологія», Серія «Медицина»)
№ 11(45) 2024

METHODOLOGICAL APPROACHES TO THE USE OF PROGRAMMING ENVIRONMENTS IN THE TRAINING OF FUTURE COMPUTER SCIENCE TEACHERS

Abstract. The rapid development of digital technologies and the growing need to integrate information technologies into the educational process present an актуальна problem for researching methodological approaches to the use of programming environments in the training of future computer science teachers. The use of such environments helps future educators better understand complex programming concepts and learn to create interactive, practice-oriented lessons for students. Moreover, it fosters the development of creative thinking, critical analysis, and the ability to quickly adapt to new technological conditions—essential competencies for teachers in the modern educational landscape.

This article examines the methodological approach to programming education as a combination of methods, tools, and forms of learning that ensure an effective educational process. It has been identified that during the training of future computer science teachers, various forms of learning can be organized, including individual and group work, practical assignments, and distance learning. Within each type of activity, it is advisable to use different programming environments.

The work pays special attention to teaching methods, including explanatory-illustrative, practical, project-based, and research approaches. Each of these methods provides a unique approach to learning, fostering the development of problem-solving skills, critical thinking, and creativity. The integration of theory with practice allows students to gain a deeper understanding of the material and prepare for their professional activities. Thus, these methods not only enhance the learning process but also build students' confidence in their abilities.

Keywords: methodological approaches, programming environment, future computer science teachers, distance learning, Replit, Thonny.

Постановка проблеми. Стрімкий розвиток цифрових технологій та зростаюча потреба в інтеграції інформаційних технологій у освітній процес зумовлює нову актуальну проблему дослідження методичних підходів до використання середовищ програмування у підготовці майбутніх учителів інформатики. Сучасні виклики, такі як дистанційне навчання, персоналізоване навчання та інноваційні підходи до STEM-освіти, вимагають від вчителів не лише знання теорії, а й практичних навичок роботи з різними програмними середовищами. Використання таких середовищ допомагає майбутнім педагогам краще засвоїти складні концепції програмування та створювати інтерактивні і практико-орієнтовані уроки для учнів. Крім того, це сприяє розвитку творчого мислення, критичного аналізу та вміння швидко адаптуватися до нових технологічних умов, що є важливими компетенціями для вчителів у сучасному освітньому просторі.

Журнал «Перспективи та інновації науки»
(Серія «Педагогіка», Серія «Психологія», Серія «Медицина»)
№ 11(45) 2024

Підґрунтям визначеної у статті проблеми вважаємо низку нормативних документів, а саме: Закони України «Про освіту» (2017); «Про вищу освіту» (2014); Стратегія розвитку сфери інноваційної діяльності на період до 2030 року (2019); Стратегія розвитку вищої освіти в Україні на 2022-2032 роки (2022); Концепція розвитку педагогічної освіти (2018); Професійний стандарт «Вчитель закладу загальної середньої освіти» (2024) та інші.

Аналіз останніх досліджень і публікацій. Вивчаючи значний науково-методичний потенціал розробок та праць учених, варто відмітити роботи, які стосуються вивчення програмування та різних аспектів використання середовищ програмування в освітньому процесі. У роботі Т. Вакалюк, В. Болотіної, Є. Байлюк, О. Покотило описано різноманітні ігрові онлайн-сервіси для вивчення мов програмування: Blockly, Code.org, Check iO, FlexBox Froggy, Code Wars, Code Club. Автори роблять висновок про те, що використання таких сервісів на уроках інформатики сприятиме зацікавленості та мотивації студентівдо вивчення мов програмування [2]; Науковці О. Дудка, О. Власій, Н. Магомета розглядають проблему реалізації компетентнісного підходу до вивчення програмування на Scratch. Автори особливу увагу звертають на формування таких компетентностей як: навчальна, соціальна та громадянська, математична, екологічна грамотність, культурна, а також інформаційно-цифрова. Зокрема, проведено аналіз впливу компетентнісних завдань за допомогою Scratch-проектів на розвиток інформаційно-цифрових компетентностей студентів[4]; О. Кривонос, С. Жуковський, М. Кривонос здійснили порівняння середовищ візуального програмування роботів для потреб навчального процесу. Автори наводять критерії порівняння середовищ програмування: візуальна сила засобів програмування; наявність інструментарію, специфіка використання в освітньому процесі [6]; Г. Іванова розглядає середовище блочного програмування Thunkable як логічне продовження вивчення візуальної мови програмування Scratch. Авторкою розглянуто технічні вимоги до роботи із навчальним середовищем, можливі шляхи тестування створеного мобільного додатку, режими роботи середовища Thunkable [5]; В. Величко та ін. розглядають різні засоби для вивчення програмування, в тому числі штучний інтелект. Зокрема, у роботі мова йде про платформу GPT від компанії OpenAI, яка дає змогу створювати вихідні коди та досліджувати їх. При цьому автори зазначають, що генеративний штучний інтелект не здатен повністю замінити програмістів, адже він працює з існуючими алгоритмами і готовими прикладами і не може підходити до виконання завдань творчо [3].

Аналіз наукової літератури дозволив виявити актуальний стан проблеми вивчення програмування у закладах освіти. Незважаючи на значний прогрес у цій сфері, існують суттєві прогалини в методиці навчання програмуванню. Ці прогалини виявляються у недостатньо чітко розроблених підходах до впровадження сучасних технологій, недостатній інтеграції практичних завдань та

Журнал «Перспективи та інновації науки»
(Серія «Педагогіка», Серія «Психологія», Серія «Медицина»)
№ 11(45) 2024

відсутності адаптивних методик для різних рівнів підготовки студентів. Зокрема, питання методичних підходів до використання офлайнових та онлайнових середовищ програмування все ще залишається відкритим, особливо у контексті підготовки майбутніх учителів інформатики. Недостатня увага до цього аспекту може призводити до неповноцінної підготовки педагогів, які повинні не лише володіти технологіями, але й бути здатними ефективно їх використовувати у своїй професійній діяльності. Це потребує подальших досліджень та розробки ефективних методичних рішень.

Таким чином, нами визначено **мету нашого дослідження** – проаналізувати та обґрунтувати методичні підходи до використання середовищ програмування у процесі підготовки майбутніх учителів інформатики; розглянути конкретні приклади використання середовищ програмування під час організації освітнього процесу.

Виклад основного матеріалу. Основу методичного підходу вивчення будь-якої дисципліни складають сукупність методів, засобів та форм навчання, які використовуються для організації ефективного освітнього процесу. Також важливо, які підходи та інструменти застосовуються для перевірки та оцінки знань і вмінь студентів. Розглянемо усі складові методичного підходу використання середовищ програмування у процесі організації відповідної форми навчання.

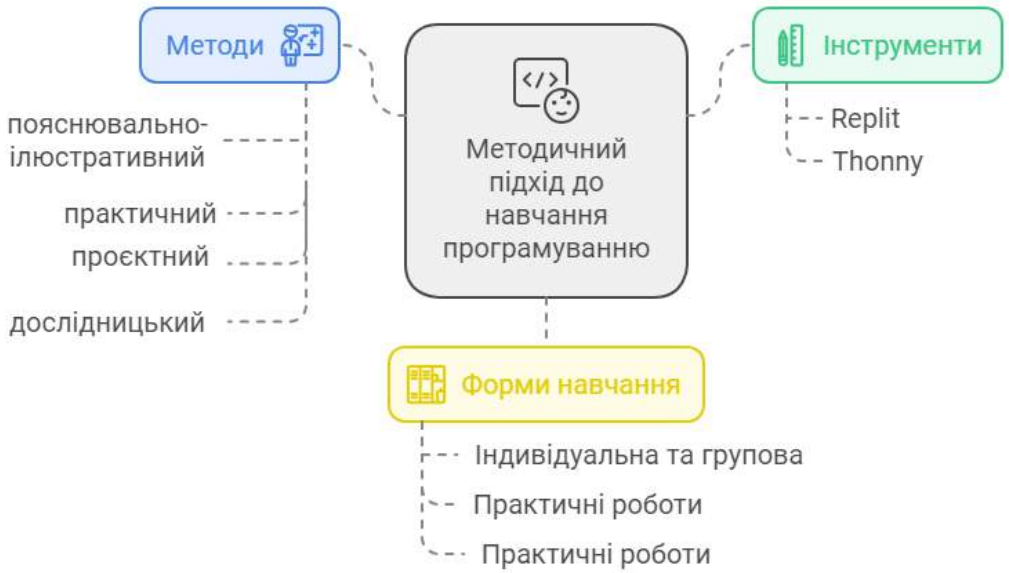


Рис.1. Методичний підхід до навчання програмуванню

У процесі підготовки майбутніх учителів інформатики можна організувати наступні форми навчання під час вивчення програмування: індивідуальна та групова робота, виконання практичних робіт, дистанційна робота. В межах кожного виду робіт можна використовувати те чи інше середовище програмування.

Журнал «Перспективи та інновації науки»
(Серія «Педагогіка», Серія «Психологія», Серія «Медицина»)
№ 11(45) 2024

Під час ***індивідуальної роботи*** студенти працюють самостійно, виконуючи практичні завдання у середовищах програмування. Це допомагає кожному студентові просуватися у власному темпі.

В якості засобу навчання для індивідуальної роботи студентів можна використовувати будь-яке середовище, яке включає мову Python. Якщо викладач обирає середовище Thonny, то перед виконанням індивідуальної роботи студент повинен навчатись встановлювати його на локальний комп'ютер, адже Thonny працює лише як локальне середовище.

Розглянемо приклад організації індивідуальної роботи студентів під час вивчення програмування у середовищі Thonny.

1. Мета: навчити студентів використовувати середовище Thonny для написання та відлагодження програм на Python. Розвинути навички розв'язування алгоритмічних задач, роботу зі змінними та умовними операторами.

2. Опис завдання: створити програму, яка запитує у користувача вік і перевіряє, чи може він отримати водійські права (мінімальний вік – 18 років).

3. План роботи.

Вступна частина

Інструкція: студент відкриває середовище Thonny на своєму комп'ютері та знайомиться з інтерфейсом. Викладач пояснює, як користуватися основними інструментами Thonny: редактором коду, вбудованою консоллю та засобами відлагодження.

Ресурси: студент може переглянути короткий відеоурок або ознайомитися з інструкцією по роботі з Thonny, яку підготував викладач.

Основна частина

Завдання 1. Написання коду.

Студент самостійно пише код програми, використовуючи базові інструкції Python (рис.2).

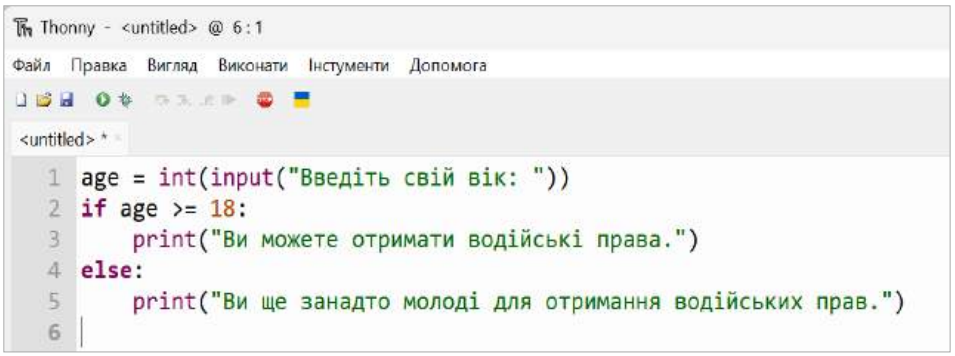


Рис.2. Приклад написаного коду студентом

Студент має протестувати код, вводячи різні значення віку (менше і більше 18 років).

Завдання 2. Відлагодження програми.

Студент використовує інструменти відлагодження в Thonny, щоб покроково виконати програму та спостерігати за тим, як змінюються значення

Журнал «Перспективи та інновації науки»
(Серія «Педагогіка», Серія «Психологія», Серія «Медицина»)
№ 11(45) 2024

змінних. Студент має зрозуміти, як працює оператор if-else, відслідковуючи, коли виконується одна або інша гілка програми.

Завдання 3. Ускладнення завдання.

Викладач пропонує студентам ускладнити програму: додати перевірку на неправильний ввід (наприклад, якщо користувач ввів не число), використовуючи блок try-except (рис.3).

```
try:
    age = int(input("Введіть свій вік: "))
    if age >= 18:
        print("Ви можете отримати водійські права.")
    else:
        print("Ви ще занадто молоді для отримання водійських прав.")
except ValueError:
    print("Будь ласка, введіть коректне число.")
```

Рис.3. Приклад ускладненого завдання

На етапі оцінювання викладач переглядає код студента, оцінює його правильність та стиль написання. Студентові пропонується самостійно проаналізувати, що він вивчив, і які виклики він подолав. Важливо також надавати зворотній зв'язок, вказуючи на сильні сторони та можливі покращення в його програмі.

Після завершення роботи студентові пропонується написати короткий відгук про те, що йому було цікаво і що він хотів би дізнатися далі. Це допомагає усвідомити свої досягнення та заохотити до подальшого вивчення програмування. Такий підхід дозволяє організувати індивідуальну роботу студента у середовищі Thonny, допомагаючи йому самостійно освоїти основи програмування на Python через практичні завдання, при цьому забезпечуючи необхідну підтримку та зворотний зв'язок.

Наступна форма організації освітнього процесу – ***групова робота***, яка передбачає, що студенти працюють у групах, розв'язуючи задачі програмування спільно. Така форма сприяє розвитку командних навичок і кращому засвоєнню матеріалу через обмін досвідом.

Для організації групової роботи студентів щодо вивчення мови Python можна використати онлайнове середовище Replit. Завдяки можливостями, які надає це середовище можна організувати одночасну роботу студентів в межах одного проєкту. Це ідеально підходить для групових завдань, оскільки студенти можуть вносити зміни до коду, бачити правки інших в реальному часі і працювати разом над вирішенням задачі.

Розглянемо приклад організації групової роботи студентів в середовищі Replit.

1. Мета: розвинути навички командної роботи, використовуючи спільне онлайн-програмування в середовищі Replit. Навчити студентіврозв'язувати складніші алгоритмічні задачі, співпрацюючи один з одним у режимі реального часу.

Журнал «Перспективи та інновації науки»
(Серія «Педагогіка», Серія «Психологія», Серія «Медицина»)
№ 11(45) 2024

2. Опис завдання: студенти повинні спільно розробити гру «Вгадай число». Завдання полягає в тому, щоб користувачеві надавали можливість вгадати випадкове число в певному діапазоні (наприклад, від 1 до 100), а програма давала підказки («більше» або «менше»). Завдання розподіляється між учасниками групи для ефективного співробітництва.

3. План роботи.

Вступна частина

Інструкція: викладач пояснює студентам принципи роботи у Replit, демонструючи можливості спільного редагування коду в реальному часі. Студентам показують, як створювати спільний проєкт та працювати в команді через Replit.

Розподіл ролей: Групи складаються з 3-4 студентів, кожен з яких отримує свою роль у розробці програми. Наприклад:

Програміст 1: Відповідає за генерацію випадкового числа та логіку гри.

Програміст 2: Відповідає за введення даних користувача та перевірку умов.

Програміст 3: Додає функціонал підказок для користувача та повідомлень про перемогу або поразку.

Програміст 4 (якщо є): Відповідає за тестування програми та відлагодження помилок.

Основна частина

Завдання 1. Створення проекту в Replit.

Один із студентів створює спільний проєкт у Replit та ділиться посиланням з іншими членами команди. Всі учасники приєднуються до проекту і можуть одночасно редагувати код у режимі реального часу.

Студенти знайомляться з тим, як Replit відображає зміни, що вносяться всіма учасниками групи, та як користуватися чатом для комунікації в середовищі.

Завдання 2: Спільне написання коду.

```
#Програміст 1
import random
secret_number = random.randint(1, 100)
#Програміст 2
user_guess = int(input("Вгадайте число від 1 до 100: "))
#Програміст 3
if user_guess < secret_number:
  print("Спробуйте більше число.")
elif user_guess > secret_number:
  print("Спробуйте менше число.")
else:
  print("Вітаємо! Ви вгадали число.")
```

Рис.4. Приклад спільного написання коду студентами

Журнал «Перспективи та інновації науки»
(Серія «Педагогіка», Серія «Психологія», Серія «Медицина»)
№ 11(45) 2024

Програміст 1. Пише код для генерації випадкового числа та створює структуру гри. Код для генерації числа може виглядати так:

Програміст 2. Додає логіку введення користувачем свого числа та перевірки правильності:

Програміст 3. Вносить функціонал підказок та виведення повідомлень:

Програміст 4. Тестує код на різних вхідних даних, додає обробку помилок та покращує загальний вигляд програми.

Завдання 3: Відлагодження та тестування.

Після завершення основного коду всі учасники команди працюють над тестуванням програми. Кожен студент перевіряє її на різні сценарії (наприклад, що буде, якщо ввести нечислове значення). Програміст 4 виправляє можливі помилки, використовуючи функціонал відлагодження Replit.

Після успішного тестування студенти презентують свою програму іншим групам, пояснюючи, як вони організували роботу і які виклики долали під час розробки. Викладач запитує студентів про їхній досвід командної роботи, що вони дізналися під час співпраці, і як їм вдалося розв'язати спільне завдання.

Після завершення проєкту кожен студент пише короткий відгук про роботу в команді: що вдалося найкраще, які навички він/вона вдосконалив(-ла) під час спільної роботи, що можна було б покращити у майбутніх проєктах.

Організація групової роботи в середовищі Replit дає змогу студентам спільно вирішувати задачі програмування, розвивати комунікативні та технічні навички, а також навички ефективної співпраці в команді. Replit забезпечує зручні інструменти для онлайн-комунікації та спільного редагування коду, що сприяє успішному виконанню групових проєктів.

Наступна форма організації навчальної діяльності студентів – ***практична робота***. Студенти виконують практичні завдання у комп'ютерному класі під керівництвом викладача, який надає підтримку в реальному часі. Практичні роботи зазвичай проводяться з конкретними завданнями для закріплення знань. Для виконання практичних занять у класі можна використовувати будь-яке середовище програмування з переліку проаналізованих нами.

Більш сучасною формою організації навчального процесу є ***дистанційне навчання***. Така форма навчання передбачає використання онлайнових середовищ програмування (наприклад, Replit), що дають змогу організувати навчання з програмування на відстані. Студенти можуть працювати вдома, маючи доступ до необхідних інструментів через Інтернет. Викладач може просто зайти в середовище Replit і переглянути роботи студентів. В цьому випадку, студент не повинен надсилати свою роботу викладачу, вона одразу стає доступна в середовищі Replit. Для цього студент створює проект у Replit і запрошує викладача до перегляду. Це дозволяє викладачу переглядати роботу в режимі реального часу і спостерігати за змінами, що робить студент.

Журнал «Перспективи та інновації науки»
(Серія «Педагогіка», Серія «Психологія», Серія «Медицина»)
№ 11(45) 2024

Також можна проводити дистанційні заняття, під час якого викладач може транслювати свій екран або запустити Replit і разом з студентами кодувати в реальному часі. Студенти можуть приєднатися до проекту та слідкувати за змінами, водночас спілкуючись через вбудований чат або зовнішній комунікаційний інструмент, наприклад Zoom або Google Meet.

Можна визначити деякі особливості використання середовища Replit в умовах дистанційного навчання.

Перша особливість стосується інтеграції з іншими онлайн-інструментами. Replit добре працює в парі з іншими платформами для відеоконференцій, такими як Zoom, Google Meet або Microsoft Teams, що дозволяє викладачу використовувати його під час онлайнових зустрічей для демонстрації коду або організації живих сесій кодування.

Replit підтримує велику кількість мов програмування, що дозволяє вивчати не тільки Python, але й інші мови, такі як JavaScript, Java, C++, тощо. Це робить його універсальним інструментом для навчання різних аспектів програмування.

Оскільки Replit працює в браузері, студенти можуть використовувати його на будь-якому пристрої (ноутбук, планшет або навіть смартфон), що робить навчання доступнішим, незалежно від технічного оснащення комп’ютера студента.

Завдяки тому, що всі проекти зберігаються у хмарі, студенти можуть почати працювати над завданням на одному пристрої, а продовжити на іншому. Це також дозволяє викладачу легко відстежувати прогрес студента та давати зворотній зв’язок на будь-якому етапі роботи.

Таким чином, середовище Replit забезпечує всі необхідні інструменти для організації дистанційного навчання програмуванню. Його особливості, такі як спільне редагування коду в реальному часі, хмарне зберігання проектів, автоматичні тести та інтеграція з іншими онлайн-інструментами, роблять його ідеальним вибором для навчання студентів в умовах дистанційної освіти.

Варто зауважити, що в умовах дистанційного навчання студенти також можуть встановити програмне середовище на своєму комп’ютері, оскільки визначені нами інструменти є абсолютно безкоштовними і доступні для завантаження в мережі Інтернет. В цьому випадку, виконані завдання студенти повинні завантажити в середовище дистанційного навчання (Moodle, Classroom) для того, щоб викладач міг оцінити їх роботу.

У ході запровадження різних форм навчання важливо використовувати відповідні методи навчання, які визначають, яким чином середовище програмування використовується для досягнення освітніх результатів. Нами виділено такі основні методи навчання програмуванню:

– пояснювально-ілюстративний;
– практичний;

Журнал «Перспективи та інновації науки»
(Серія «Педагогіка», Серія «Психологія», Серія «Медицина»)
№ 11(45) 2024

– проєктний;
– дослідницький.

Пояснювально-ілюстративний метод навчання полягає у тому, що викладач демонструє програмні середовища, показує, як використовувати різні функції, і пояснює основи програмування, а студенти спостерігають та записують інформацію. Пояснювально-ілюстративний метод дає змогу студентам отримати чітку та послідовну інформацію, що полегшує розуміння нових концепцій. Ілюстрації та приклади коду допомагають візуалізувати складні ідеї, що є особливо корисним для новачків. Демонстрація типових помилок і способів їх вирішення допомагає уникати проблем під час програмування. Після кожного теоретичного блоку студенти можуть застосовувати знання на практиці, що сприяє кращому засвоєнню матеріалу.

Практичний метод навчання передбачає, що студенти самостійно або під керівництвом викладача виконують практичні завдання у середовищі програмування. Це основний метод навчання програмуванню, який розвиває навички розв’язання задач.

Основною характеристикою практичного методу є акцент на виконанні студентами програмних завдань, вправ та проектів. Студенти застосовують отримані теоретичні знання для розв’язання реальних проблем і створення програм. Наприклад, після вивчення теми про цикли, студенти одразу ж виконують завдання на написання програми, що використовує цикл для обчислення математичної послідовності.

Практичний метод допомагає студентам розвинути навички вирішення проблем та критичного мислення, оскільки вони постійно стикаються з задачами, що потребують логічного підходу до вирішення. Наприклад, під час роботи над проектом, студенти можуть зіштовхнутися з неочікуваними проблемами, такими як помилки в коді або логічні протиріччя. Вони навчаються аналізувати проблему і знаходити шляхи її вирішення.

Практичні завдання дозволяють викладачеві краще оцінити рівень підготовки кожного студента та надавати індивідуальну підтримку. Деякі студенти можуть швидше засвоїти матеріал, тоді як інші потребують більше часу та уваги для розуміння. Викладач може запропонувати різні рівні складності завдань, щоб кожен студент працював на своєму рівні. Наприклад, одні студенти пишуть базовий код для обчислення площі фігури, тоді як інші додають функціонал для роботи з різними типами фігур.

Практична робота часто мотивує студентів, оскільки вони бачать результат своїх зусиль у вигляді працюючих програм. Це сприяє формуванню впевненості у своїх силах та бажанню продовжувати навчання. Наприклад, студенти створюють прості ігри або програми, якими можуть поділитися з друзями або сім’єю, що підвищує їх зацікавленість у програмуванні.

Журнал «Перспективи та інновації науки»
(Серія «Педагогіка», Серія «Психологія», Серія «Медицина»)
№ 11(45) 2024

Практика допомагає студентам закріпити теоретичні знання. Під час написання коду студенти постійно звертаються до вивченого матеріалу, що сприяє кращому запам'ятовуванню і розумінню концепцій програмування. Наприклад, після теоретичного вивчення списків у Python, студенти виконують завдання на роботу з ними, такі як додавання елементів, сортування списку та інші операції, що поглиблює їхнє розуміння теми.

Практичний метод може бути застосований як під час індивідуальної, так і під час групової роботи. Студенти можуть працювати над власними проектами або спільно вирішувати складніші задачі. Наприклад, студенти можуть індивідуально писати невеликі програми, а потім об'єднуватися у групи для створення великих проектів, таких як веб-додатки або гри.

Практичний метод вчить студентів брати відповідальність за результати своєї роботи. Вони отримують миттєвий зворотний зв'язок у вигляді працюючого або непрацюючого коду, що дозволяє їм самостійно оцінити свої успіхи і невдачі. Наприклад, якщо програма не працює належним чином, студенти мають самостійно знайти помилку і виправити її, що вчить їх бути уважними і відповідальними.

Практичний метод при вивченні програмування дозволяє студентам активніше засвоювати матеріал через постійне застосування знань на практиці. Він сприяє розвитку навичок вирішення проблем, підвищує мотивацію та відповідальність студентів, а також допомагає закріпити теоретичні знання. Цей метод є ефективним як для індивідуальної, так і для групової роботи, що робить його універсальним для різних форм навчання програмуванню.

Проєктний метод навчання передбачає, що студенти створюють свої проєкти за допомогою середовищ програмування. Цей метод допомагає застосовувати знання на практиці, розвиває творче мислення та навички управління проектами. Студенти не лише засвоюють теоретичні концепції, але й отримують можливість застосувати їх на практиці, реалізуючи реальні проєкти. Це допомагає глибше зрозуміти, як теоретичні знання працюють у реальних умовах.

Проєктний метод спонукає студентів до самостійного аналізу проблем і пошуку рішень, що розвиває критичне мислення і навички вирішення проблем. Багато проєктів виконуються в групах, що сприяє розвитку комунікативних навичок, уміння працювати в команді і ділитися ідеями. Загалом, проєктний метод навчання сприяє більш глибокому розумінню матеріалу, підвищує мотивацію студентів і дозволяє їм отримати практичний досвід, який буде корисним у їхній подальшій професійній діяльності.

Дослідницький метод навчання програмуванню спрямований на те, щоб студенти самостійно досліджували нові теми, розв'язували проблеми, ставили експерименти та формулювали висновки. Цей підхід розвиває аналітичне мислення, креативність та навички вирішення проблем. У середовищі

програмування дослідницький метод можна реалізувати через створення проєктів, які вимагають від студентів самостійного пошуку рішень та тестування різних підходів.

Загалом, дослідницький метод навчання сприяє глибшому розумінню предмета, розвиває важливі навички для майбутньої професійної діяльності та заохочує студентів до активної участі у навчальному процесі.

Висновки. Аналіз різних форм організації навчальної діяльності студентів виявив, що вибір форми впливає на використання відповідних методів і засобів. Так, усі форми навчання дають змогу використовувати середовища розробки реального часу (Replit), проте обмеження на ресурси в таких середовищах накладає певні обмеження на завдання, які можуть пропонуватись студентам. Тому для більш серйозного проєктування доцільно використовувати локальні засоби.

Особливе значення мають методи навчання, які включають пояснювально-ілюстративний, практичний, проєктний та дослідницький підходи. Використання цих методів навчання програмуванню є критично важливим для досягнення освітніх результатів. Кожен з цих методів забезпечує унікальний підхід до навчання, сприяючи розвитку навичок розв'язання проблем, критичного мислення та творчості. Завдяки інтеграції теорії з практикою студенти отримують можливість глибше зрозуміти матеріал і підготуватися до професійної діяльності. У результаті, ці методи не тільки покращують навчальний процес, але й формують впевненість студентів у своїх здібностях.

Література:

1. Бондаренко Т.В. Особливості використання програмного засобу Prezi у процесі розробки навчальних презентацій. *Інформаційні технології і засоби навчання*. 2018. Том 63(1). С.1-11.

2. Вакалюк Т. А. та ін. Огляд ігрових онлайн сервісів для вивчення мов програмування. *Інноваційна педагогіка*. 2020. Вип. 22. С. 192–198.

3. Величко В., та ін. Електронне навчання у процесі вивчення програмування: Інформатика та методика її навчання. *Збірник наукових праць фізико-математичного факультету ДДПУ*. 2023. Вип. 13. С. 54–61.

4. Дудка О. М., Власій О. О., Магомета Н. М. Реалізація компетентнісного підходу до вивчення програмування на Scratch. *Відкрите освітнє е-середовище сучасного університету*. 2018. Вип. 5. С. 88–96. DOI: https://doi.org/10.28925/2414-0325.2018.5.8896.

5. Іванова Г. Використання Thunkable для розробки мобільних додатків при вивченні програмування. *Grail of Science*. 2022. Вип. 23. С. 351–354.

6. Кривонос О., Жуковський С., Кривонос М. Порівняння середовищ візуального програмування роботів для потреб навчального процесу. *Наука і техніка сьогодні*. 2022. Вип. 6(6). С. 175–187.

7. Медведєва М. О., Колмакова В. О., Коровнік І. С. Візуалізація навчального матеріалу: аналіз сучасних онлайн-сервісів. *Інноваційна педагогіка*. 2021. Випуск 41. Т.2, С. 128–132.

Журнал «Перспективи та інновації науки»
(Серія «Педагогіка», Серія «Психологія», Серія «Медицина»)
№ 11(45) 2024

References:

1. Bondarenko, T. V. (2018). Osoblyvosti vykorystannia prohramnoho zasobu Prezi u protsesi rozrobky navchalnykh prezentatsii [Peculiarities of Software Prezi Usage for Designing Educational Presentations]. Informatsiini tekhnolohii i zasoby navchannia – *Information Technologies and Learning Tools*, 63(1), 1-11 [in Ukrainian].

2. Vakaliuk, T. A., Bolotina, V. V., Bailyuk, Y. M., & Pokotylo, O. A. (2020). Ohlyad ihrovyh onlajn servisiv dlya vyvchennya mov prohramuvannya [Review of Game Online Services for Learning Programming Languages]. Innovatsiina pedahohika – Innovative Pedagogy, 22, 192–198 [in Ukrainian].

3. Velychko, V., Zhukovskyi, S., & Kryvonos, O. (2023). Elektronne navchannya u protsesi vyvchennya prohramuvannya: Informatyka ta metodyka yiyi navchannya [E-learning in the Process of Learning Programming: Informatics and Methods of Teaching It]. Zbirnyk naukovykh prats fizyko-matematychnoho fakultetu DDPU – Scientific Works of the Faculty of Physics and Mathematics of Donbas State Pedagogical University, 13, 54–61 [in Ukrainian].

4. Dudka, O. M., Vlasii, O. O., & Mahometa, N. M. (2018). Realizatsiya kompetentnisnoho pidkhody do vyvchennya prohramuvannya na Scratch [Implementation of Competence-Based Approach to Learning Programming with Scratch]. Vidkryte osvitnie e-seredovyshche suchasnoho universytetu – Open Educational E-Environment of Modern University, 5, 88–96. https://doi.org/10.28925/2414-0325.2018.5.8896 [in Ukrainian].

5. Ivanova, H. (2022). Vykorystannya Thunkable dlya rozrobky mobil′nykh dodatkiv pry vyvchenni prohramuvannya [Using Thunkable for Developing Mobile Apps in Programming Studies]. Grail of Science, 23, 351–354 [in Ukrainian].

6. Kryvonos, O., Zhukovskyi, S., & Kryvonos, M. (2022). Porivnyannya seredovyshch vizual′noho prohramuvannya robotiv dlya potreb navchal′noho protsesu [Comparison of Visual Programming Environments for Robots for Educational Purposes]. Nauka i tekhnika sohodni – Science and Technology Today, 6(6), 175–187 [in Ukrainian].

7. Medvedieva, M. O., Kolmakova, V. O., & Korovnik, I. S. (2021). Vizualizatsiya navchal′noho materialu: Analyz suchasnykh onlajn-servisiv [Visualization of Educational Materials: Analysis of Modern Online Services]. Innovatsiina pedahohika – Innovative Pedagogy, 41(2), 128–132 [in Ukrainian].