УДК 378.014.13:004
https://doi.org/10.28925/2312-5829/2024.4.1

Олександр РУДИК,
доцент кафедри природничо-математичної освіти і технологій Інституту післядипломної освіти Київського столичного університету імені Бориса Грінченка, кандидат фізико-математичних наук, доцент, м. Київ, Україна
https://orcid.org/0000-0003-3676-0688
o.rudyk@kubg.edu.ua

ПРОБЛЕМИ НАВЧАННЯ ПРОГРАМУВАННЯ ЗДОБУВАЧІВ ОСВІТИ ЗАКЛАДІВ ЗАГАЛЬНОЇ СЕРЕДНЬОЇ ОСВІТИ ТА ВИРІШЕННЯ ЇХ ЗА ДОПОМОГОЮ ВІДКРИТОГО ОСВІТНЬОГО РЕСУРСУ

Глобалізація світової економіки на тлі зростання ролі комп'ютерних технологій ставить перед освітою нові завдання, одне з яких — навчання програмування здобувачів освіти закладів загальної середньої освіти. У статті висвітлено проблеми, пов'язані з таким навчанням. Обґрунтовано важливість при вивченні інформатики робити наголос на вивченні основ створення, а не лише використання, програмних продуктів. У межах компетентнісного підходу проаналізовано проблему відсутності у більшості закладів загальної середньої освіти передумов успішного вивчення природничо-математичних дисциплін взагалі й навчання програмування зокрема. Проаналізовано причину низької мотивації до навчання програмування здобувачів освіти закладів загальної середньої освіти внаслідок відірваності процесу навчання від потреб ринку інформаційних технологій. У тому числі й щодо вибору програмного забезпечення, і щодо можливості перемикання між різними його видами. Пояснено причини неготовності більшості учителів до якісного навчання програмування. Подано результати дослідження ставлення вчителів до розглянутих проблем. Ці дані дають змогу зробити висновок про уявлення вчителів щодо послідовності опанування процедурним та об'єктно орієнтованим програмуванням, можливостей навчати формулювання алгоритмів. Виявлено незадоволення наявним навчально-методичним забезпеченням переважної більшості вчителів. Досліджено ставлення вчителів до опанування мов програмування, актуального для сучасної індустрії програмного забезпечення. Запропоновано комплексне рішення тих проблем, які можна подолати у межах навчання інформатики, що полягає у використанні відкритих навчальних ресурсів з високими вимогами до якості контенту. Подано посилання на приклад завершеного проєкту навчально-методичного забезпечення вивчення інформаційних технологій виключно на основі вільно поширюваного програмного забезпечення та поточного проєкту вивчення основ програмування з використанням різних середовищ програмування. Усі запропоновані підходи не прив'язані жорстко до системи освіти в Україні. Тому цей досвід легко поширити й на інші країни, і (у разі потреби) на інші дисципліни.
Ключові слова: *компетентнісний підхід, мета освіти, алгоритмічне мислення, подійно- та об'єктно орієнтоване програмування, відкриті освітні ресурси.*

Вступ. Затвердження навчальної програми курсу «Інформатика, 5–9 класи загальноосвітніх навчальних закладів (для здобувачів освіти закладів загальної середньої освіти, які вивчали інформатику в 2–4 класах)» наказом Міністерства освіти і науки України від 02.02.2016 № 73 як і її наступні редакції передбачає виділення значної кількості (до 40 %) навчального часу на вивчення програмування. І не процедурного, а подійно- та об'єктно орієнтованого.

Підстави таких змін очевидні. Навчання програмування не лише формує уміння програмувати, але й істотно сприяє загальному розумовому розвитку здобувачів освіти, формуванню вміння концентруватися на вирішенні поставленої задачі, становленню алгоритмічного мислення,

вихованню відповідальності за кожен зроблений крок. Воно дає змогу у майбутньому правильно оцінити доцільність використання програмних засобів та результати їх використання. Звичайно, суспільству не потрібно, щоб усі здобувачі освіти стали програмістами. Але усі майбутні наші програмісти навчаються чи навчатимуться у закладах загальної середньої освіти. Необов'язково у спеціалізованих закладах загальної середньої освіти чи у профільних класах. Від успішності втілення нової навчальної програми істотно залежить конкурентоспроможність випускників на глобалізованому ринку праці.

Намагання запровадити навчання програмування в якомога ранньому віці пов'язане з усвідомленням того, що навчання програмування — процес тривалий. Хоча б тому, що заняття з програмування пов'язане зі зміною не лише мислення, а й фізіологічних властивостей організму. Наприклад, збільшення рівня компетентності щодо аналізу програмного коду зазвичай пов'язане зі збільшенням середньої амплітуди сакади (швидкі, суворо узгоджені й одночасні рухи очей в одному напрямку) та зменшенням середньої тривалості фіксації (Andrzejewska, Kotoniak, 2020).

При навчанні програмування мова йде не лише про майбутніх програмістів. Програмування сприймають як важливу компетентність для розвитку навичок вирішення проблем на додаток до традиційних логічних міркувань. Наприклад, при використанні Scratch п'ятикласниками (Kalelioğlu, Gülbahar, 2014). Але рівень наочного програмування накладає обмеження на складність розв'язуваних завдань і на той рівень навичок вирішення проблем, який можна досягнути. Тому перехід до написання кодів ще в закладі загальної середньої освіти неминучий.

До запровадження вивчення основ об'єктно орієнтованого програмування (надалі ООП) у закладах загальної середньої освіти це було зроблено у закладах вищої освіти. І там проявилися проблеми, яких не уникнути у закладі загальної середньої освіти. Наприклад, зазвичай наголос роблять на синтаксичних особливостях побудови класів та об'єктів конкретною мовою програмування, без пояснення, чому ці класи та об'єкти необхідні. Усі викладачі закладів вищої освіти, опитані у ході дослідження (Donchev, Todorova, 2008), розуміли труднощі та основні причини їх виникнення. Але це не призводило до належних змін у навчальних курсах. Думки опитаних викладачів можна звести до такого (Donchev, Todorova, 2008):

— студентам важко починати з ООП при вивченні програмування;

— концепції ООП краще засвоюють ті, хто має належний досвід процедурного програмування, тому доцільніше викладати ООП лише після курсу процедурного програмування;

— студенти повинні самі прийти до суті парадигми ООП;

— курс повинен наголошувати на ООП та загальному розумінні підходу;

— краще використовувати короткі приклади з насиченим змістом, прості у використанні. Приклади, пов'язані з об'єктами моделювання «з реального світу», придатніші за вирішення суто математичних задач;

— більшість студентів не можуть «прочитати» чужий код, а також розпізнати логічні помилки та написати належні коментарі;

— код програми, написаний здобувачем освіти, має нижчий рівень багаторазового використання.

Вже давно (щонайменше для України) констатують суперечність між обсягом запланованих знань з програмування та недостатністю часу на їх засвоєння (Жалдак, 2010). А це означає, у тому числі, потребу навчальних ресурсів для самостійної роботи. Хоча б для усіх охочих.

Публікації, що стосуються навчання здобувачів освіти програмування й визначення шляхів щодо розв'язання цього питання, переважно констатують проблеми мотивації й акцентують на методиці навчання (Семеніхіна, Руденко, 2018). Інколи роблять наголос на виборі мови та середовища програмування (Базурін, 2017), нехтуючи можливістю використання здобувачами освіти одного класу різних мов програмування.

Відомо, що мова програмування має глибокий вплив на те, як програміст аналізує проблему та формулює її рішення у формі програми. Вибір першої мови програмування, ймовірно, буде дуже важливим (див., наприклад, аналіз опитування щодо використання мов програмування в академічних закладах США (Ben Arfa Rabai, Cohen, Mili, 2015).

З іншого боку, системне дослідження (Taslibeyaz, Kursun, Karaman, 2020) публікацій щодо розвитку навичок роботи у навчальних середовищах вказує на таке: замість того щоб покладатися на привабливість, функціональність, частку ринку освітніх інструментів, основне, що потрібно враховувати, це наявність кваліфікованої навчальної діяльності, зосередженої на вирішенні задач.

Перелічені проблеми не є регіональними. Результати дослідження у Словаччині та Чехії засвідчують як причину слабкості програмування неналежну якість підручників або їх відсутність загалом (Záhorec, Hašková, Munk, 2021). Реформа 2017 року національної навчальної програми Швеції вимагала, щоб усі вчителі математики та технологій обов'язково інтегрували програмування у своє викладання. Більшість вчителів, яких це стосувалося, мали невеликий досвід з програмування або зовсім його не мала. Дослідження (Vinnervik, 2020) думки вчителів, проведене за 4 місяці до впро-

вадження нової навчальної програми, показало стурбованість, зумовлену невизначеністю змісту предмета, нерівними можливості професійного розвитку, браком навчальних матеріалів та періодичними проблемами з ІТ-інфраструктурою закладу загальної середньої освіти.

Проблеми середньої освіти стають проблемами вищої освіти. Результати тестувань чітко вказують на те, що більшість студентів починають навчання з інформатики з недорозвиненими алгоритмічними навичками. У закладах вищої освіти розглядають переважно традиційні середовища програмування й в основному застосовують неефективні методи поверхневого підходу, як це практикують у початковій та середній освіті (Csernoch, Biró, Máth, Abari, 2015). Це пояснює розрив між очікуваннями закладів вищої освіти й реальними здобутками здобувачів освіти.

Актуальним було й залишається вивчення ставлення вчителів до проблем навчання програмування й вирішення їх за допомогою різноманітних навчальних ресурсів. Зазвичай тривогу викликає те, як це вплине на здобувачів освіти з *різними навчальними здібностями* при загальній відсутності обчислювального мислення та навичок вирішення задач у більшості здобувачів освіти (Attard, Busuttil, 2020).

Мета. Виявлення й аналіз наявних проблем навчання програмування здобувачів загальної середньої освіти, бачення цих проблем вчителями інформатики та опис способу вирішення цих проблем за допомогою відкритого освітнього ресурсу.

Методологія. При виконанні дослідження використано такі методи:

— теоретичні — аналіз публікацій, узагальнення джерел з проблеми дослідження, синтез навчально-методичних ідей, досвіду викладацької діяльності, що дали змогу з'ясувати особливості навчання програмування, встановити основні труднощі успішного його здійснення;

— емпіричні — анкетування вчителів з наступною інтерпретацією результатів дослідження, спостереження за освітнім процесом та учасниками конкурсу «Учитель року 2022» у номінації «Вчитель інформатики».

Результати (виклад основного матеріалу).

Перелічимо та проаналізуємо наявні проблеми навчання програмування.

1. ***У більшості закладів загальної середньої освіти відсутні передумови успішного вивчення природничо-математичних дисциплін взагалі й навчання програмування зокрема.*** Стало загальновідомим зауваження Едсгера Вібе Дейкстри про те, що програмування, тобто вироблення алгоритмів, — занадто важка справа, щоб навчити її кожну кухарку. Спроби перетворити програмування на просте, досяжне для кожного заняття приречені на невдачу (Dijkstra, 1983). Пояснення цього, на перший погляд, фаталістичного висловлювання міститься в іншому зауваженні того самого автора: крім математичних здібностей, життєво важливою якістю програміста є досконале володіння рідною мовою. На жаль, на *загальноосвітньому* рівні поки не розроблено й не передбачено створення обов'язкових передумов успішного навчання програмування, а саме:

— досконале опанування здобувачами освіти мовлення, у тому числі ситуативного;

— добротні навички здобувачів освіти щодо використання символьних записів;

— здатність здобувачів освіти унаочнювати різноманітними способами умову й розв'язання задачі.

Протягом тривалого часу з процесу викладання усіх навчальних дисциплін в українських (і не лише українських) закладах загальної середньої освіти вилучали алгоритмічно змістовні завдання. Перехід від вимоги «опиши, поясни» до вимоги «поміть у робочому зошиті» також не сприяв розвитку мовлення. Таким чином, спроби вивчення основ програмування не узгоджені з методикою викладання більшості предметів. А чинні навчальні програми з інформатики не передбачають (і не можуть передбачити), щоб учителі інформатики своєчасно викладали ті розділи математики, які:

— є теоретичною основою понять, реалізованих у мовах програмування;

— дозволяють ставити й розв'язувати алгоритмічно змістовні задачі, що демонструють можливості технології програмування.

Істотним недоліком впровадженої програми є використання лише однієї навчальної години на тиждень у 5–7 класах. Для навчального предмета зі значною кількістю логічних зв'язків та вимогою опанування *технології* роботи це неприпустимо. На думку багатьох вчителів, оволодіння дисципліною, на яке відведено лише одну навчальну годину на тиждень, насправді є лише імітацією навчання. Досягнути поставлену у програмі мету — зробити у 5–7 класах «акцент на набутті навичок практичного <...> застосування» — здебільшого неможливо. Тому вивчення інформатики у 8–9 класах буде ґрунтуватися не на стійких навичках, а на фрагментарних спогадах. Хоча у класі з умотивованими здобувачами освіти, відібраних за рівнем розвитку й мовлення, є можливість досягнути таку мету.

2. ***Програмування вже не процедурно-орієнтоване, до якого звикли вчителі старшого віку, а подійно- та об'єктно орієнтоване.*** При запровадженні вивчення інформатики у закладах загальної середньої освіти у середині 80-х років і до середини 90-х років XX століття не було й мови про подійно- та об'єктно орієнтоване програмування. Згодом в Україні від процедурно-орієнтованого програмування було здійснено перехід

до курсу користувача інформаційних технологій з виділенням лише 5-ти навчальних годин на тему «Алгоритми й програмування». У ті часи майбутні вчителі та їхні викладачі у більшості випадків не відчували нагальної потреби займатися питаннями програмування, у тому числі впровадженням парадигми подійно- та об'єктно орієнтованого програмування. Останнє вимагає категорійного мислення й системного підходу до розв'язування завдання. *Гарантованого* результату на цій ниві здебільшого досягають вже після 30-ти років і лише після тривалого набуття досвіду в цій сфері.

3. ***Попередня спроба вивчати процедурно-орієнтоване програмування була невдалою у більшості навчальних закладів.*** У другій половині 90-х років XX ст. автор статті займався вивченням рівня знань, умінь і навичок при навчанні інформатики у київських закладах загальної середньої освіти за завданням міського управління освіти і науки. Це був час, коли єдиною темою, яку неможливо було скоротити (наприклад, через відсутність комп'ютерів), була тема «Програмування і алгоритмізація». Дослідження було здійснено на основі аналізу виконання здобувачами освіти 11 кл. протягом 30 хвилин контрольних робіт, що містили такі три завдання:

— або описати словами, або подати блок-схемою, або подати програмою алгоритм розв'язання певної задачі;

— простежити виконання алгоритму для двох наборів вхідних даних;

— доповнити запропонований набір так, щоб він повністю перевіряв алгоритм.

Передбачалося виконання таких завдань.

З'ясувати, яка з двох дат передує іншій.

— Розв'язати рівняння:
- $ax + b = 0$;
- $a/x + b = 0$;
- $ax^2 + bx + c = 0$.

— Визначити вид трикутника:
- за градусною мірою двох внутрішніх кутів;
- за даними квадратами довжин сторін.

Здебільшого (понад 2/3 здобувачів освіти у кожному обстеженому районі Києва) було виявлено:

— хибно складений алгоритм;

— неправильно простежено виконання алгоритму (наприклад, при розв'язуванні рівняння $ax + b = 0$ при $a = 0$ і $b = 1$ при спробі ділити на 0 не вказано на аварійне завершення роботи);

— набір значень неповний або взагалі не записано відповідь.

Але найбільше вражало те, що вчителі здебільшого неправильно оцінювали роботи. Одна з причин цього — наявність алгоритмічних помилок у підручниках і посібниках. Наприклад, в описі розв'язання рівняння $ax^2 + bx + c = 0$ (без застереження, що воно є квадратним) не передбачено виродження лінійного рівняння у тотожно істинне чи у тотожно хибне висловлювання. Чи в інших регіонах України ситуація була кращою, автор напевно не знає. Але підстав вважати, що вона була істотно кращою, немає.

Іншою причиною невдачі природно вважати відсутність чітко окреслених вимог до результатів навчання. Хоча б у вигляді переліку базових задач. І нова програма має той самий недолік. Наприклад, у 9 кл. передбачено вивчення алгоритмів упорядкування масивів. А скільки їх буде обов'язкових для вивчення, наскільки вони будуть ефективними — жодного слова. Цілковита свобода вчителя виправдана там, де тривалий час предмет викладали з дотриманням певного курсу. Для того щоб була змога скористатися результатами зміни курсу інформатики (зокрема, й у закладах вищої освіти), потрібно деталізувати вимоги щодо результатів навчання у самій програмі.

4. ***Багато вчителів не мають не лише досвіду навчання (хоча б процедурно-орієнтованого програмування), але навіть тих знань і навичок, які вимагають від здобувачів вищої чи загальної середньої освіти.*** При вивченні програмування у закладах вищої освіти часто поводяться так, ніби єдина проблема, яку мають здобувачі освіти, — це проблема запису мовою програмування того алгоритму, який вони знають. Проблема такого звуження мети й змісту навчання існує, хоча у педагогічній літературі є чіткі застереження щодо цього (Saeli, Perrenet, Jochems, Zwaneveld, 2011). Внаслідок недоліків отриманої середньої освіти у здобувачів є труднощі навіть з розумінням таких питань як: що таке задача? що означає розв'язати задачу? що значить протестувати задачу? Мова не про завчені означення, а про набуті навички. І не завжди цю проблему долають у закладі вищої освіти. Такого висновку автор дійшов на основі аналізу результатів діагностичних контрольних робіт на курсах підвищення кваліфікації з тими самими завданнями, що пропонували учням 11-го класу. З приблизно такими самими результатами! З 25-ти учасників і туру (по місту Києву) всеукраїнського конкурсу «Учитель року — 2022» у номінації «Вчитель інформатики» лише один педагог продемонстрував прийнятні навички програмування. Зауважимо й таке: внаслідок плинності кадрів у багатьох закладах загальної середньої освіти інформатики дозволяють навчати вчителям інших спеціальностей, яких ніколи не готували до навчання програмування. Це також одна з причин гостроти зазначеної проблеми.

5. ***І для учителів, і для здобувачів освіти інколи (часто? найчастіше?) притаманний низький рівень компетентностей щодо мовлення при записі алгоритмів.*** Більшість шкільних підручників укладають з розрахунку на легке перше знайомство. Тому вчитель має відводити здобувача

освіти «за руку» від такого багатослівного стилю і вести до стислого викладу думки, зручного для роботи. Тобто до того стилю, якого діти не бачили у підручниках і при першому зіткненні з яким відчують шок. Але ж своєю чергою їхні вчителі навчалися за аналогічними книжками, тому мають ті самі хиби. І їм самим часто потрібен «поводир».

6. ***Намагання авторів підручників використовувати переважно навчальні середовища програмування, що істотно знижує мотивацію до навчання.***

Середовища програмування, створені спеціально для навчання, зазвичай:

— приховують те, що може відволікати увагу й що є зручним лише для досвідченого програміста;

— роблять легким і наочним програмування стандартних для навчання дій.

Зрозуміла привабливість таких засобів і для здобувачів освіти, і для учителів, які не хочуть докладати зайвих зусиль. Розраховуючи на більшість, і автори, і видавництва точно не оминуть навчальні середовища програмування. При запровадженні нової програми у 2016 році на семінарі методистів ОІППО згадували лише Lazarus як програмне забезпечення, придатне для покриття *всіх* питань навчальної програми. Використання лише навчальних середовищ істотно знижує мотивацію до навчання програмування через відірваність процесу навчання від потреб ІТ-ринку. І саме для тих, хто у майбутньому спробує використати набуті знання, уміння й навички щодо програмування! Сучасному роботодавцю потрібен спеціаліст, що опанував *актуальне* для індустрії програмного забезпечення середовище програмування і спроможний, у разі потреби, якісно опанувати нову для себе сферу знань і діяльності. *Щоб система освіти не слугувала гальмом для індустрії програмного забезпечення*, потрібно забезпечити усім охочим вільний доступ до навчальних ресурсів щодо використання різноманітних актуальних середовищ програмування хоча б на рівні виконання вимог навчальної програми з інформатики.

Однією з основних проблем для учителя є питання вибору програмного забезпечення. Її вирішення повністю покладено на вчителя. Для усвідомленого вибору вчителю потрібно детально ознайомитися, наприклад, з такими можливими альтернативами: C++, C#, Java, Javascript, PHP, Python, Ruby. Ці мови названо менеджерами середньої ланки на вже згаданому семінарі методистів ОІППО 2016 р. Але потрібно вибрати *кілька* мов програмування, а не лише одну. Здобувачі освіти повинні уміти порівнювати роботу у *різних* середовищах. І це правильно, бо ще у 1975 році Дейкстра вказав, що розвиток програмування як науки з опорою лише на одну вибрану мову програмування неможливий (див. зауваження щодо використання ALGOL 68 у Німеччині (Dijkstra, 1975)): «Німецьке прийняття ALGOL 68 мало аналогічний паралітичний вплив на німців, як російське рішення в кінці шістдесятих років розробити в якості наступної національної серії комп'ютерів біт-сумісну копію IBM 360 — найбільшу американську перемогу в "холодній війні"».

Здобувачі освіти обов'язково повинні мати можливість порівняти динамічно й статично типізовані мови програмування. Сьогодні спостерігаємо зростання зацікавленості українських авторів шкільних підручників до опису мови Python. Це ознака позитивних зрушень. Але проблему для решти найактуальніших мов не вирішено.

1. ***Відсутність україномовних джерел з викладом важливих і важких для сприйняття аспектів вивчення актуальних для індустрії програмного забезпечення мов і середовищ програмування.*** Часто ПЗ виникає як результат роботи інтернаціональних колективів. Зазвичай спочатку створюють англомовний інтерфейс. Створення локалізацій — тривалий процес. Переклад довідкової літератури — ще триваліший.

2. ***Відсутність для багатьох питань щодо програмних засобів, задіяних в індустрії програмного забезпечення, стислого викладу, прийнятного для навчального процесу, навіть у англомовних джерелах.*** При створенні документації ПЗ розробники в першу чергу забезпечують повноту опису й зручність навігації. У процесі навчання (щонайменше у закладі загальної середньої освіти) достатньо ознайомити з найуживанішими прийомами роботи, що дають змогу працювати, маючи невелику кількість навчальних годин. Виклад навчального матеріалу традиційних шкільних дисциплін можуть шліфувати десятиліттями. Щодо інформатики, то всі свідомі того, що зміст поданого може застаріти вже за 10 років. Тому таке шліфування багатьом видається малопривабливим.

3. ***Майже всі джерела щодо актуальних в індустрії програмного забезпечення мов і середовищ програмування розраховані на програмістів з досвідом або хоча б на старшокурсників.*** Становлення програміста — тривалий процес, який здебільшого планують завершити у віці старшокурсника або випускника закладу вищої освіти. Про апробований системний підхід до викладання цих питань здобувачам загальної середньої освіти говорити не доводиться. Копіювання наявних джерел (лише з перекладом) у підручниках нічим не виправдане. Ці матеріали вимагають тривалої і кропіткої роботи щодо пристосування їх до рівня сприйняття здобувачами освіти й насамперед щодо логіки й стислості викладу змісту, зрозумілості.

Нижче наводимо результати дослідження ставлення педагогів до окреслених проблем, проведеного за допомогою анонімного опитування, яке тривало з 30 грудня 2020 р. до 10 квітня

2021 р., шляхом заповнення форми Google. Перед початком опитування на електронні адреси вчителів інформатики міста Києва і методистів ОІППО було надіслано електронного листа з проханням поширити повідомлення про опитування серед педагогів своїх районів. Всього в опитуванні взяло участь 397 вчителів інформатики. Усі дані подано відсотками до цієї кількості. На жаль, вибірка не є репрезентативною для України:

— місто Київ — 60 %;
— Рівненська область — 28,9 %;
— Черкаська область — 5,8 %;
— Київська область — 3,3 %.

Решта регіонів має істотно менші представництва. Істотних відмінностей у розподілі відповідей між киянами та представниками інших регіонів немає.

Розподіл за категоріями населених пунктів такий:

— обласний центр або місто Київ — 63 %;
— районний центр або місто обласного підпорядкування — 10,3 %;
— село або селище міського типу — 26,7 %.

Результати опитування на питання з відповідями «так» і «ні» (частка таких питань була найбільшою) подано у *табл. 1*.

Таблиця 1

РЕЗУЛЬТАТИ АНКЕТУВАННЯ НА ПИТАННЯ З ВІДПОВІДЯМИ «ТАК» І «НІ»

№	Формулювання запитання	Розподіл відповідей	
		так	ні
1	Чи маєте досвід навчання програмування у 5–6 класах?	77,1 %	22,9 %
2	Чи маєте досвід навчання програмування у 7–9 класах?	87,4 %	12,6 %
3	Чи маєте досвід навчання програмування у 10–11 класах?	62,5 %	37,5 %
4	Чи маєте досвід навчання програмування у класах з поглибленим вивченням математики?	20,7 %	79,3 %
5	Чи маєте досвід навчання програмування у класах з поглибленим вивченням гуманітарних дисциплін?	42,8 %	57,2 %
6	Необхідною умовою програмування є опанування здобувачем освіти математики й мовлення. У тому числі ситуативного. Ви з цим погоджуєтеся?	89,7 %	10,3 %
7	Чи вважаєте ви прийнятним рівень опанування мовлення й математики здобувачами освіти вашого навчального закладу для навчання програмування?	49,9 %	50,1 %
9	Чи маєте навички процедурно-орієнтованого програмування?	66,2 %	33,8 %
10	Чи маєте навички подійно- та об'єктно орієнтованого програмування?	75,6 %	24,4 %
11	Чи легко вам розмовною мовою подати алгоритм?	86,9 %	13,1 %
12	Чи ваші здобувачі освіти можуть розмовною мовою подати алгоритм без помилок?	39,5 %	60,5 %
13	Чи є у вашому навчальному закладі підручники, у яких виклад тем програмування повністю вас задовольняє?	21,7 %	78,3 %
14	Чи є у вашому навчальному закладі підручники, у яких програмування ґрунтується на мовах, актуальних для сучасної індустрії програмного забезпечення?	23,7 %	76,3 %
19	Чи маєте бажання вивчити або поглибити знання з мови Pascal?	45,1 %	54,9 %
20	Чи маєте бажання вивчити або поглибити знання з мови PHP?	55,9 %	44,1 %
21	Чи маєте бажання вивчити або поглибити знання з мови Python?	86,4 %	13,6 %
22	Чи маєте бажання вивчити або поглибити знання з мови Java?	67,3 %	32,7 %
23	Чи маєте бажання вивчити або поглибити знання з мови Javascript?	68,8 %	31,2 %
24	Чи маєте бажання вивчити або поглибити знання з мови C++?	61,2 %	38,8 %
25	Чи маєте бажання вивчити або поглибити знання з мови C#?	45,3 %	54,7 %
26	Чи маєте бажання вивчити або поглибити знання з мови Ruby?	32,5 %	67,5 %

Для питань з іншими типами відповіді отримано такі результати (у дужках на початку абзацу вказано номер питання в анкеті).

(8) Чи була успішною попередня спроба (до орієнтації на курс користувача) вивчати процедурно-орієнтоване програмування?

— «так» — 29,5 %;

— «ні» — 25,9 %;

— «не знаю» — 44,6 %.

(15) Чи стикались ви з адаптованими до рівня здобувачів загальної середньої освіти україномовними джерелами з логічно-послідовним викладом усіх (у межах змісту навчальних програм) питань вивчення актуальних для індустрії програмного забезпечення мов і середовищ програмування?

— «так» — 6 %;

— «ні» — 20,7 %;

— лише з такими, що частково покривають, і здебільшого найлегші теми — 73,3 %.

(16) Наявні у глобальній мережі іншомовні навчальні матеріали вимагають тривалої і кропіткої роботи щодо пристосування їх до рівня здобувачів загальної середньої освіти. У першу чергу щодо логіки й стислості викладу змісту, зрозумілості здобувачами освіти. Ви з цим погоджуєтеся?

— «так» — 55,4 %;

— «ні» — 2,8 %;

— «деякі потребують лише якісного перекладу» — 41,8 %.

(17) Вкажіть адресу найкращого на вашу думку джерела глобальної мережі, придатного для навчання програмування у закладі загальної середньої освіти.

Було подано адреси 47 ресурсів, декілька було вказано неодноразово, проте 76,8 % респондентів не вказали жодного джерела.

(18) Як ви ставитеся до розробок уроків, посилання на які опубліковано за адресою: http://www.kievoit.ippo.kubg.edu.ua/kievoit/2016.html?

— «не знаю про існування цих матеріалів» — 34,3 %;

— «знаю, але не використовую» — 35 %;

— «використовую» — 30,7 %.

(27) Яку мову програмування, крім C++, C#, Java, Javascript, Pascal, PHP, Python, Ruby), маєте бажання вивчити?

5 % респондентів назвали одну з таких мов: Асемблер, Basic, Dart, Go (Golang), Haskell, Hi, Kotlin, NodeJS, Processing, Swift, TypeScript, VisualBasic, 95 % не назвали жодної.

(28) Яка мова програмування зі строгою типізацією змінних найдоцільніша для вивчення у закладі загальної середньої освіти?

— ще не визначився — 43,6 %;

— Pascal — 27,5 %;

— C++ — 16,9 %;

— Java — 9,8 %;

— C# — 1,5 %;

— інші відповіді — 1 %.

(29) Яка мова програмування з динамічною типізацією змінних найдоцільніша для вивчення у закладі загальної середньої освіти?

— Python — 49,4 %;

— ще не визначився (не визначилася) — 37 %;

— Javascript — 10,1 %;

— PHP — 1,8 %;

— Ruby — 0,7 %;

— інші відповіді —1 %.

Проаналізуємо отримані результати опитування за кожним питанням (у дужках на початку абзацу, як і вище, вказано номер питання в анкеті).

(1–3) Для кожної з перелічених вікових категорій (5–6, 7–9 і 10–11 класи) більшість опитуваних має досвід навчання програмування.

(4–5) Більшість учителів має досвід навчання лише на загальноосвітньому рівні, орієнтованих на класи «гуманітаріїв» більше, ніж орієнтованих на «математиків».

(6) Несподівано високий відсоток відповіді: «ні» на риторичне питання. Зрозуміло, кожна людина має право на власну думку. Однак наразі немає прикладів успішного навчання програмування людей з розсіяною увагою, а саме тих, хто:

— слухає, але часто не розуміє почуте;

— говорить, але часто не розуміє сказане ним самим;

— читає, але часто не розуміє прочитане;

— пише, але часто не розуміє, що сам написав.

Згадаймо й те, що першим кроком створення програми, є створення математичної моделі. Результати опитування свідчать про наявність у деяких учителів хибних уявлень. І нехтувати наявністю таких учителів неприпустимо.

(7) На думку щонайменше половини респондентів не створено необхідні умови для успішного опанування програмування. Відповідно у них не може бути впевненості в успіху навчання.

(8) Майже половина респондентів (44,6 %) не знають про ті проблеми навчання програмування, які існували в минулому, й чи були вони взагалі. Тож можуть повторити ті помилки, які вже було зроблено їхніми попередниками або колегами. Майже половина решти респондентів скептично оцінює результати навчання процедурно-орієнтованого програмування. Так само, як і автор статті, ґрунтуючись на результатах вивчення рівня знань, умінь і навичок при вивченні інформатики, проведеного за завданням головного управління освіти й науки міста Києва у 1995–2000 рр. Інакше кажучи, надійної упевненості в успіху на основі минулого досвіду немає.

(9–10) Аналіз первинних даних показав, що 72 респонденти (18,1 %) відповіли, що не мають навичок процедурно-орієнтованого навчання, але мають навички об'єктно- та подійно-орієнто-

ваного програмування. Маємо високий відсоток вчителів інформатики з хибним уявленням про програмування і прийнятний порядок вивчення тем. У столиці й за її межами цей відсоток приблизно однаковий.

(11–12) Більшість учителів вважає, що здатні розмовною мовою подати алгоритм. З іншого боку, переважна більшість не може навчити здобувачів освіти того, що самі уміють. Можна сумніватися у їхній компетентності, але, можливо, причина в іншому: не створено необхідні передумови для успішного опанування програмування.

(13–14) Переважна більшість вчителів незадоволена викладом програмування у наявних підручниках.

(15–16) Переважна більшість вчителів усвідомлює, що наявні ресурси глобальної мережі не можна безпосередньо використати у повсякденній роботі.

(17–18) Учителі постійно шукають серед ресурсів глобальної мережі найпридатніші для своєї роботи, проте більшість не знаходить того, що їх цілком задовольняло б.

(19–26) Серед перелічених мов менше 50 % прихильників мають лише Pascal, C# і Ruby. Для навчальної мови Pascal і мови спочатку платформно обмеженої (з орієнтацією на продукцію Microsoft) це природно. А от дані щодо Ruby свідчать про наявність прогалин у знаннях стосовно актуальних мов програмування. Адже ця мова є аналогом загальновизнаної (навіть серед українських авторів шкільних підручників) мови Python, але значно потужнішої щодо вебпрограмування. Інакше кажучи, мотиви для вивчення Ruby є більш вагомими. 5,3 % респондентів не бажають вивчати жодної з перелічених мов програмування, актуальних для індустрії програмного забезпечення, 3,8 % — жодної з перелічених мов, включаючи Pascal.

(27) Говорити про статистично значимі уподобання неможливо.

(28) 43,6 % респондентів не визначилися щодо мови програмування зі строгою типізацією змінних. Наступна за чисельністю група — прихильники навчальної мови Pascal (27,5 %). Це цілком узгоджено з наявним навчально-методичним забезпеченням. Мова ж переможців олімпіад C++ посіла лише третє місце зі скромним показником 16,9 %. Постає очевидна необхідність проводити роз'яснювальну роботу серед вчителів інформатики, хоча б для того, щоб вони могли швидше визначитися.

(29) Спостерігаємо результат наявності відповідного навчально-методичного забезпечення українською мовою щодо мови Python (49,4 %). Частка тих, хто не визначився (37 %), менша, ніж для мов зі строгою типізацією. Наступними є мови вебпрограмування, і лише потім Ruby. Це черговий доказ непоінформованості учителів щодо цієї мови й необхідності просвітництва з боку інститутів післядипломної освіти.

Отримані відповіді свідчить про те, що думки більшості опитаних вчителів щодо проблем навчання програмування збігаються з поглядами автора. Навіть перелік мов (C++, C#, Java, Javascript, PHP, Python, Ruby) виявився достатнім для вибору тих, які вивчатимуть здобувачі освіти.

Шляхи вирішення проблем, що лежать поза змістом навчальної програми з інформатики, вбачаємо у такому.

1. У навчальних програмах з інших дисциплін (насамперед з математики) потрібно враховувати, що нововведення у навчальній програмі з інформатики вимагають випереджальних у часі інновацій у програмах з інших дисциплін, зокрема алгебри. Наприклад, передбачити розв'язування на уроках математики й задач логічного характеру, й комбінаторних задач, у тому числі з виробленням й дотриманням схеми перебору. Не потрібно навіть нічого створювати. В Україні це було апробовано у межах програми «Росток», яку варто поширити на всі заклади загальної середньої освіти, де вивчають програмування. Цей підхід можна й бажано доповнити досвідом роботи з дошкільнятами (Papy F., Papy G., Incolle D., 1968).

2. Бажано переглянути навчальну програму щодо виділення 2-х навчальних годин на тиждень на оволодіння предметом. Якщо МОН не може відповідним чином змінити навчальний план, то доречно рекомендувати викладати моделювання, основи алгоритмізації й програмування одним блоком, зміщеним у 8–9-ті класи.

Шляхи вирішення проблеми щодо змісту навчальної програми з інформатики вбачаємо в такому.

Наразі вчителі не мають підстав сподіватися на швидку системну допомогу «згори». Так само, як свого часу щодо використання вільно поширюваного програмного забезпечення. Один із можливих способів комплексного розв'язання окреслених проблем — співпраця вчителів у створенні відкритих освітніх ресурсів. Бажано при координації з боку інститутів післядипломної педагогічної освіти. Такими ресурсами спочатку можуть стати збірки розробок уроків у вигляді гіпертекстів, що описують використання різноманітного програмного забезпечення. Вибір формату HTML дає змогу автоматичного форматування тексту під різні розміри (екран монітора, планшет, смартфон). У разі потреби можна забезпечити інтерактивність засобами Javascript. Розробки мають бути настільки детальними, щоб зробити можливим самостійне навчання здобувачів освіти навіть при відсутності системних знань і допомоги вчителя. Усі вперше виконувані дії з інтерфейсом потрібно детально ілюструвати, причому без спотворення зображень внаслідок стискування та з поданням вікон програм повністю. Наразі

у паперових підручниках цього немає. Лише після масової апробації таких (можливо, лише статичних) текстів буде сенс розробляти електронні підручники у повному розумінні цього слова.

Досконала розробка уроку передбачає роботу конкретного вчителя у конкретному класі з розподілом часу на окремі види роботи. Тому спочатку відкриті ресурси міститимуть не розробки уроків, а підбір матеріалів (заготовки) для власне розробок уроків. Спочатку ці матеріали буде використано для освіти чи самоосвіти вчителів, згодом, або навіть одночасно, і для роботи із здобувачами освіти.

Завдяки розробленню аналогічних матеріалів у 2012–2014 рр. на курсах підвищення кваліфікації вчителів інформатики в ІППО Київського університету імені Бориса Грінченка було успішно вирішено проблему навчально-методичного забезпечення викладання шкільного курсу інформатики у 5–9 класах відповідно до *чинної на той час* Державної навчальної програми на основі використання лише вільно поширюваного програмного забезпечення і за рік-два до того, як виходили з друку підручники з грифом МОН. Отже, існує апробована технологія розв'язання складних актуальних проблем змісту освіти.

З вересня 2016 року і до липня 2023 року нами вирішено окреслену вище проблему зміни парадигми викладання інформатики для 8-ми мов програмування (Рудик, 2018, 2020, 2023, 2024). Тож якщо вчитель незадоволений змістом чи формою звичайного підручника або при відсутності останнього, такі розробки уроків можуть стати достойною альтернативою.

При створенні розробок уроків дотримано положень статті (Hadjerrouit, 2009):

— щодо змісту — концептуального педагогічного підходу;

— щодо форми — структурування матеріалу таким чином, щоб для практичної роботи повний, детально проілюстрований опис, від якого можна легко перейти до словесного описання чи навіть стислих інструкцій, виділених у тексті накресленням. Усе це вчитель може легко здійснити за допомогою вилучення файлів зображень чи рядків тексту відповідно до готовності конкретного класу до певного рівня деталізації опису.

Висновки. Виявлено й детально проаналізовано наявні проблеми навчання програмування здобувачів загальної середньої освіти. На основі анонімного опитування встановлено й проаналізовано бачення цих проблем вчителями інформатики. Вказано спосіб вирішення цих проблем за допомогою відкритого освітнього ресурсу, який апробовано для таких мов програмування, як C++, C#, Java, Javascript, Pascal, PHP, Python, Ruby.

Перспективи подальших досліджень. Запропонований підхід не прив'язано жорстко до системи освіти в Україні чи до навчального предмета. Тому цей досвід можна поширити й на інші країни, й на інші дисципліни. Наприклад, для переходу до логічно-послідовного вивчення математики у закладі загальної середньої освіти.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Базурін В. М. Середовища програмування як засіб навчання учнів основ програмування. *Інформаційні технології і засоби навчання.* 2017. Том 59. № 3. С. 13–27. DOI: http://dx.doi.org/10.33407/itlt.v59i3.1601

2. Жалдак М. І. Інформатика — фундаментальна наукова дисципліна. *Комп'ютер у школі та сім'ї.* 2010. № 2. С. 39–43.

3. Рудик О. Б. Вимоги до навчально-методичного забезпечення дистанційного та змішаного навчання інформатиці. 2020 URL: http://www.kievoit.ippo.kubg.edu.ua/kievoit/course/2020add.html

4. Рудик О. Б. Відкриті освітні ресурси у розв'язанні нагальних проблем трансформації змісту освіти. *Відкрита та дистанційна освіта: від теорії до практики:* зб. м-ів III Всеукр. електрон. наук.-практич. конф. (27 вересня 2018 р., м. Київ). С. 107–109. URL: https://s3-eu-west-1.amazonaws.com/ourboox-media-prod/wp-content/uploads/2018/10/08174512/Зб-матеріалів_Конференція_27_09.pdf

5. Рудик О. Б. Повідомлення для слухачів курсів підвищення кваліфікації. 2024. URL: http://www.kievoit.ippo.kubg.edu.ua/kievoit/course.html

6. Рудик О. Б. Розробки уроків з інформатики для вивчення програмування з використанням мов програмування C++, C#, Java, Javascript, Pascal, PHP, Python, Ruby. 2023. URL: https://www.kievoit.ippo.kubg.edu.ua/kievoit/program.html

7. Семеніхіна О. В., Руденко Ю. О. Проблеми навчання програмувати учнів старших класів та шляхи їх подолання. *Інформаційні технології і засоби навчання.* 2018. Т. 66. № 4. С. 54–64. DOI: http://dx.doi.org/10.33407/itlt.v66i4.2149

8. Andrzejewska M., Kotoniak P. Development of Program Comprehension Skills by Novice Programmers — Longitudinal Eye Tracking Studies. *Informatics in Education.* 2020. Vol. 19 № 4. Pp. 521–541. DOI: https://doi.org/10.15388/infedu.2020.23

9. Attard L., Busuttil L. Teacher Perspectives on Introducing Programming Constructs through Coding Mobile-Based Games to Secondary School Students. *Informatics in Education.* 2020. Vol. 19 № 4. Pp. 543–568. DOI: https://doi.org/10.15388/infedu.2020.24

10. Ben Arfa Rabai L., Cohen B., Mili A. Programming Language Use in US Academia and Industry. *Informatics in Education.* 2015. Vol. 14 № 2. Pp. 143–160. DOI: https://doi.org/10.15388/infedu.2015.09

11. Csernoch M., Biró P., Máth J., Abari K. Testing Algorithmic Skills in Traditional and Non-Traditional Programming Environments, *Informatics in Education.* 2015. Vol. 14 № 2. Pp. 175–197. DOI: https://doi.org/10.15388/infedu.2015.11

12. Dijkstra E. W. Computers and General Education: A position paper. 1983. DOI: https://www.cs.utexas.edu/~EWD/transcriptions/EWD08xx/EWD868.html

13. Dijkstra E. W. Trip Report: NATO Summer School Marktoberdorf 1975. URL: https://www.cs.utexas.edu/users/EWD/transcriptions/EWD05xx/EWD506.html

14. Donchev I., Todorova E. Object-Oriented Programming in Bulgarian Universities, *Informatics and Computer Science Curricula, Informatics in Education.* 2008. Vol. 7 № 2. Pp. 159–172. DOI: https://doi.org/10.15388/infedu.2008.10

15. Hadjerrouit S. Teaching and Learning School Informatics: A Concept-Based Pedagogical Approach. *Informatics in Education*, 2009, Vol. 8 № 2, Pp. 227–250. DOI: https://doi.org/10.15388/infedu.2009.14

16. Kalelioğlu F., Gülbahar Y. The Effects of Teaching Programming via Scratch on Problem Solving Skills: A Discussion from Learners' Perspective. *Informatics in Education.* 2014. Vol. 13 № 1. Pp. 33–50. DOI: https://doi.org/10.15388/infedu.2014.03

17. Papy F., Papy G., Incolle D. Les enfants et les graphes. Didier: Bruxelles — Montréal — Paris, 1968, 189 p.

18. Saeli M., Perrenet J., Jochems W. M. G., Zwaneveld B. Teaching Programming in Secondary School: A Pedagogical Content Knowledge Perspective. *Informatics in Education.* 2011. Vol. 10 № 1, Pp. 73–88. DOI: http://dx.doi.org/10.15388/infedu.2011.06

19. Taslibeyaz E., Kursun E., Karaman S. How to Develop Computational Thinking: A Systematic Review of Empirical Studies, *Informatics in Education.* 2020. Vol. 19. № 4. Pp. 701–719. DOI: http://dx.doi.org/10.15388/infedu.2020.30

20. Vinnervik P. Implementing Programming in School mathematics and Technology: Teachers' intrinsic and extrinsic challenges. *International Journal of Technology and Design Education.* 2020. Vol. 32, Pp. 213–242. DOI: https://doi.org/10.1007/s10798-020-09602-0

21. Záhorec J., Hašková A., Munk M. Assessment of Selected Aspects of Teaching Programming in SK and CZ, *Informatics in Education.* 2021. Vol. 13 № 1. Pp. 157–178. DOI: https://doi.org/10.15388/infedu.2014.09

REFERENCES

Bazurin, V. M. (2017). Seredovyshcha prohramuvannia yak zasib navchannia uchniv osnov prohramuvannia [Programming Environments as a Means of Teaching Pupils to Programming Basics]. *Information Technologies and Learning Tools, 59*(3), 13–27 [in Ukrainian].
http://dx.doi.org/10.33407/itlt.v59i3.1601

Zhaldak, M. I. (2010). Informatyka — fundamentalna naukova dystsyplina [Informatics as Fundamental Scientific Discipline]. *Computer in School and Family, 2*, 39–43 [in Ukrainian].

Rudyk, O. B. (2020). Vymohy do navchalno-metodychnoho zabezpechennia dystantsiinoho ta zmishanoho navchannia informatytsi [Requirements for Educational and Methodological Support of Distance and Mixed Learning of Informatics] [in Ukrainian].
http://www.kievoit.ippo.kubg.edu.ua/kievoit/course/2020add.html

Rudyk, O. B. (2018). Vidkryti osvitni resursy u rozviazanni nahalnykh problem transformatsii zmistu osvity [Open Educational Resources in Solving Urgent Problems of Transforming the Content of Education]. "Open and Distance Education: From theory to practice." Collection of materials of the III scientific and practical conference, September 27, 107–109 [in Ukrainian].
https://s3-eu-west-1.amazonaws.com/ourboox-media-prod/wp-content/uploads/2018/10/08174512/36-матеріалів_Конференція_27_09.pdf

Rudyk, O. B. (2024). Povidomlennia dlia slukhachiv kursiv pidvyshchennia kvalifikatsii [Notice for Students of Advanced Training Courses]. [in Ukrainian]. http://www.kievoit.ippo.kubg.edu.ua/kievoit/course.html

Rudyk, O. B. (2023). Rozrobky urokiv z informatyky dlia vyvchennia prohramuvannia z vykorystanniam mov prohramuvannia C++, C#, Java, Javascript, Pascal, PHP, Python, Ruby [Development of Computer Science Lessons for Learning Programming Using Programming Languages C++, C#, Java, Javascript, Pascal, PHP, Python, Ruby] [in Ukrainian].
https://www.kievoit.ippo.kubg.edu.ua/kievoit/program.html

Semenykhina, O. V., Rudenko Y. O. (2018). Problemy navchannia prohramuvaty uchniv starshykh klasiv ta shliakhy yikh podolannia [Problems of Educating to Programming of Students and Way of their Overcoming]. *ICT and Learning Tools in Secondary Education, 66*(4), 54–64 [in Ukrainian].
http://dx.doi.org/10.33407/itlt.v66i4.2149

Andrzejewska, M., Kotoniak, P. (2020). Development of Program Comprehension Skills by Novice Programmers – Longitudinal Eye Tracking Studies. *Informatics in Education, 19*(4), 521–541 [in English].
https://doi.org/10.15388/infedu.2020.23

Attard, L., & Busuttil, L. (2020). Teacher Perspectives on Introducing Programming Constructs through Coding Mobile-Based Games to Secondary School Students. *Informatics in Education, 19*(4), 543–568 [in English]. https://doi.org/10.15388/infedu.2020.24

Ben Arfa Rabai, L., Cohen, B., Mili, A. (2015). Programming Language Use in US Academia and Industry. *Informatics in Education, 14*(2), 143–160 [in English]. https://doi.org/10.15388/infedu.2015.09

Csernoch, M., Biró, P., Máth, J., Abari, K. (2015). Testing Algorithmic Skills in Traditional and Non-Traditional Programming Environments, *Informatics in Education, 14*(2), 175–197 [in English]. https://doi.org/10.15388/infedu.2015.11

Dijkstra, E. W. (1983) Computers and General Education: a position paper [in English]. https://www.cs.utexas.edu/~EWD/transcriptions/EWD08xx/EWD868.html

Dijkstra, E. W. (1977). Trip Report: NATO Summer School Marktoberdorf 1975 [in English]. https://www.cs.utexas.edu/users/EWD/transcriptions/EWD05xx/EWD506.html

Donchev, I., Todorova, E. (2008). Object-Oriented Programming in Bulgarian Universities. *Informatics and Computer Science Curricula, Informatics in Education,* 2008, *7*(2), pp. 159–172 [in English]. https://doi.org/10.15388/infedu.2008.10

Hadjerrouit, S. (2009). Teaching and Learning School Informatics: A Concept-Based Pedagogical Approach. *Informatics in Education*, *8*(2), 227–250 [in English]. https://doi.org/10.15388/infedu.2009.14

Kalelioğlu, F., Gülbahar, Y. (2014). The Effects of Teaching Programming via Scratch on Problem Solving Skills: A Discussion from Learners' Perspective. *Informatics in Education, 13*(1), 33–50 [in English]. https://doi.org/10.15388/infedu.2014.03

Papy, F., Papy, G., Incolle, D. (1968). Les enfants et les graphes. [Children and Graphs]. Didier: Bruxelles–Montréal–Paris, 189 p. [in French].

Saeli, M., Perrenet, J., Jochems, W. M. G., Zwaneveld, B. (2011). Teaching Programming in Secondary School: A Pedagogical Content Knowledge Perspective. *Informatics in Education, 10*(1), 73–88 [in English]. http://dx.doi.org/10.15388/infedu.2011.06

Taslibeyaz, E., Kursun, E., Karaman, S. (2020). How to Develop Computational Thinking: A Systematic Review of Empirical Studies, *Informatics in Education, 19*(4), 701–719 [in English]. http://dx.doi.org/10.15388/infedu.2020.30

Vinnervik, P. (2020) Implementing Programming in School Mathematics and Technology: Teachers' intrinsic and extrinsic challenges. *International Journal of Technology and Design Education, 32*, 213–242 [in English]. https://doi.org/10.1007/s10798-020-09602-0

Záhorec, J., Hašková, A., Munk, M. (2021). Assessment of Selected Aspects of Teaching Programming in SK and CZ. *Informatics in Education, 13*(1), 157–178 [in English]. https://doi.org/10.15388/infedu.2014.09

Oleksandr RUDYK,
PhD in Physics and Mathematics, Associate Professor,
Associate Professor at the Department of Natural and Mathematical Education and Technologies,
Institute of In-Service Education,
Borys Grinchenko Kyiv Metropolitan University,
Kyiv, Ukraine

https://orcid.org/0000-0003-3676-0688
o.rudyk@kubg.edu.ua

PROBLEMS IN TEACHING PROGRAMMING TO PUPILS AT GENERAL SECONDARY EDUCATION INSTITUTIONS AND THE WAYS TO SOLVE THEM USING AN OPEN EDUCATIONAL RESOURCE

Globalisation of the world economy, in the context of the increasing significance of computer technology, presents novel challenges for the domain of education, one of which is teaching programming to students of general secondary education institutions. The article deals with the problems of such teaching. The primary focus of the study of the basis of designing as well as using software is justified. The problem of the absence of preconditions for the successful study of natural sciences and mathematics in general and teaching of programming in particular in most schools is analysed within the framework of the competence approach. This study examines the factors that contribute to the low motivation for learning programming which results in isolation of the learning process from the requirements of the software market, including the choice of software and the possibility of switching between different types of software. The study reveals the factors that contribute to the negative attitude of the majority of teachers towards quality programming education. The results of the study of teachers' attitudes to the analysed problems are presented. The data provide an opportunity to draw a conclusion regarding the teachers' ideas about the sequence of mastering procedural and object-oriented programming skills, as well as their ability to teach algorithms formulation. Dissatisfaction with the available teaching and methodological support of the vast majority of teachers is revealed. Teachers' attitudes

towards mastering programming languages, relevant for the modern software industry, is studied. A comprehensive solution to those problems that can be overcome within the framework of computer science education is proposed, which consists in the use of open educational resources with high content quality requirements. The study provides a link to the example of the completed project of educational and methodological support for the study of information technologies only on the basis of freely distributed software, as well as the current project for studying the basics of programming using various programming environments. It is important to note that the proposed approaches are not rigidly implemented in education system in Ukraine, and, therefore, this experience can be easily broadened to other countries and (if necessary) to other disciplines.
Keywords: *competence approach, purpose of education, algorithmic thinking, event-driven and object-oriented programming, open educational resources.*

Стаття надійшла до редакції: 31.05.2024
Прийнято до друку: 26.12.2024