УДК 372.3:004

АНАЛІЗ МЕТОДИЧНИХ ПІДХОДІВ ДО ВИКЛАДАННЯ ТЕМИ «АЛГОРИТМИ ТА ВИКОНАВЦІ» У КУРСІ «СХОДИНКИ ДО ІНФОРМАТИКИ»

Стрілецька Наталія Михайлівна,
доцент кафедри дошкільної та початкової освіти Чернігівського національного педагогічного університету імені Т. Г. Шевченка, кандидат педагогічних наук, .smixnat@mail.ru.

Анотація. У статті автор розкриває методичні особливості навчання теми «Алгоритми і виконавці» у курсі «Сходинки до інформатики» на основі аналізу чинних програм, підручників, методичних посібників та досвіду педагогів-практиків.

Ключові слова: Алгоритми і виконавці, способи подання алгоритмів, програмне середовище Scratch, труднощі формування уявлення про алгоритм, методи навчання.

Сучасні зміни, що відбуваються у початковій освіті, характеризуються широким використанням інформаційно-комунікаційних технологій у процесі формування творчої, відповідальної особистості учня, здатної розв'язувати різноманітні життєві і навчальні завдання, у тому числі й засобами ІКТ.

Важливе місце у цьому процесі займає предмет «Сходинки до інформатики», який згідно Державного стандарту початкової загальної освіти (2011 р.) [2] впроваджується з 2013–2014 навчального року, починаючи з 2-го класу. Зазначимо, що на базі варіативної складової навчального плану продовжувалось вивчення пропедевтичного курсу інформатики у 3–4 класах за авторськими програмами, рекомендованими МОН України [1]. Упровадження пропедевтичних курсів інформатики у початкову школу не могло не позначитись і на змісті вищої педагогічної освіти на факультетах початкового навчання педагогічних університетів. З 2012–2013 н. р. за напрямом підготовки 0101 «Педагогічна освіта», спеціальності 6.010102 «Початкова освіта» введено нову навчальну дисципліну «Методика навчання інформатики у початковій школі», одним із головних завдань якої є забезпечення ґрунтовного розуміння студентами методичних ідей початкового курсу інформатики, використання методів і засобів нових інформаційних технологій. У зв'язку з цим набуває актуальності питання дослідження сучасних методичних підходів до реалізації змісту навчання курсу «Сходинки до інформатики».

Різні проблеми пропедевтики навчання інформатики у початковій школі в Україні знайшли своє розв'язання у працях науковців та дослідників, зокрема:

- обґрунтовані авторські концепції методичних комплексів «Сходинки до інформатики», що реалізують вимоги Державного стандарту початкової загальної освіти (Н. Морзе, В. Вембер, Н. Саражинська, О. Коршунова, І. Зарецька, М. Корнієнко, С. Крамаровська, Г. Ломаковська, Г. Проценко, Й. Ривкінд, Ф. Рівкінд);
- розкриті методологічні основи застосування комп'ютера в початковій школі і проведений аналіз чинних підручників «Сходинки до інформатики» на відповідність вимогам до сучасного підручника даної вікової групи (В. Шакотько, О. Кивлюк);
- обґрунтовані деякі шляхи вдосконалення змісту, методів, засобів та організаційних форм навчання інформатики шляхом упровадження пропедевтики змістово-методичної лінії формалізації і моделювання на основі включення спеціальних завдань (М. Глазун, Н. Морзе); методики використання опорних схем, організації розробки міні проектів (О. Коршунова); формувального оцінювання (Н. Морзе, В. Вембер, О. Барна); завдань на розвиток мислення учнів (В. Вембер) та ін.

Проте питання узагальнення методичних підходів опрацювання змістових ліній курсу «Сходинки до інформатики» не стало предметом окремого дослідження. Розв'язання вказаного завдання сприятиме наповненню лекційного курсу навчальної дисципліни «Методика навчання інформатики в початковій школі» на факультетах початкового навчання педагогічних вишів, формуванню у студентів варіативного бачення щодо введення понять, способів дій, використання прийомів і методів навчання, методів і засобів інформаційних технологій.

Мета статті — розкрити методичні особливості навчання теми «Алгоритми і виконавці» у курсі «Сходинки до інформатики» на основі аналізу чинних програми, підручників, методичних посібників і досвіду педагогів-практиків.

Важливість теми «Алгоритми і виконавці» у підготовчому курсі інформатики пов'язана з опануванням алгоритмічних умінь учнями як складової предметної ІКТ-компетентності. Під алгоритмічними уміннями розуміють складання алгоритмів дій із повсякденного життя, з використанням матеріалу навчальних предметів (математики, української мови тощо); уміння аналізувати текст задачі, складати, записувати і виконувати найпростіші алгоритми для виконавців у визначеному середовищі, розрізняти основні алгоритмічні структури [12].

Одним із завдань змістової лінії «Алгоритми й виконавці» є формування міжпредметних компетенцій. Внутрішньопредметні зв'язки теми з іншими темами курсу «Сходинки до інформатики» встановлюються через використання у вивченні останніх алгоритмічного підходу. Тобто діти повинні уміти планувати послідовність дій для виконання завдань, передбачати можливі наслідки розв'язування задачі, для яких відповіддю є не число або твердження, а опис послідовності дій, створення моделі, схеми, графіка тощо [12].

Ознайомлення учнів з алгоритмами і їх виконавцями передбачає розуміння на інтуїтивному рівні понять: виконавця, його середовища, команди, системи команд виконавця, алгоритму, отримання перших уявлень про основні алгоритмічні структури: слідування, розгалуження та повторення, формування навичок виконувати готові алгоритми, а також складати прості алгоритми для виконавців, які працюють у певному зрозумілому для відповідної вікової категорії комп'ютерному середовищі, використовуючи просту систему їхніх команд. Діти отримують уявлення про способи подання алгоритмів: словесний; алгоритмічну мову (як систему команд виконавця комп'ютерного середовища) і графічний (побудова алгоритмів на основі блок-схем).

У чинних підручниках «Сходинки до інформатики» для 2–3 класів запроваджено різні підходи щодо введення понять та ознайомленням із способом подання алгоритмів. Розглянемо деякі з них (рис. 1, 2).

Рис. 1. Вивчення теми «Алгоритми та виконавці» за О. Коршуновою

Рис. 2. Вивчення теми «Алгоритми та виконавці» за Г. Ломаковською, Г. Проценко, Й. Ривкіндом, Ф. Ривкінд

І підхід (за О. Коршуновою) характеризується послідовним ознайомленням учнів із словесним поданням алгоритму (2–3 класи) і мовою виконавця комп'ютерного середовища **Scratch** (3 клас). Уявлення про ці способи виробляються на прикладах інтуїтивно.

У **ІІ підході** (за Г. Ломаковською та ін.) способи побудови алгоритмів словесний і мовою виконавця **Scratch** вивчаються паралельно. Авторами реалізовано принцип концентричності, що передбачає повторення і вивчення уже знайомих понять у 3 класі на більш високому рівні (повторення, розширення та поглиблення знань). Так, у 3 класі описово дається уявлення про словесний спосіб подання команд, його властивості та інші способи (жести, сигнали); розкривається новий зміст понять команда, алгоритм.

Уявлення про алгоритм і поняття, з ним пов'язані, формуються невіддільно від способів подання алгоритмів. Так, на основі **словесного способу подання алгоритму** діти починають ознайомлення з «алгоритмом» і правилами його створення. Спочатку дається уявлення про «команду»: «Спонукальне речення спонукає до негайної дії. Воно є наказом, командою» [6].

«**Команда** — це речення, що спонукає до дії. Той, хто виконує команди, називається **виконавцем**» [7].

Пізніше (3 клас) за методичним підходом Г. Ломаковської та ін. уявлення про команду розширюється: «**Команда** — наказ, вказівка виконати певні дії» [9]. Звертається увага, що подання команд у формі спонукальних речень — це лише один із способів подання команд, який називається словесним. На основі конкретно-індуктивного методу дітей ознайомлюють з іншими способами подання команд: світловими сигналами (світлофором); звуковими сигналами (дзвінком у школі); жестами (міліціонером-регулювальником); подвійним клацанням лівою клавішею миші на виділеному об'єкті (запуск програм на виконання). За методикою О. Коршунової [13], крім словесного способу подання команд неявно, дається уявлення й про інші способи: мовою стрілок; малюнками; за допомогою скороченого запису команд. За допомогою названих умовних позначень команд учні складають й відповідні алгоритми.

У чинних підручниках поняття «**виконавця**» розглядається залежно від порядку його введення, тобто спирається на поняття, введені раніше. Так, виконавець — це об'єкт (людина, тварина, технічний пристрій), який виконує команди (за Г. Ломаковською та ін.), або об'єкт (людина, технічний пристрій), який виконує алгоритм (за О. Коршуновою).

Вимоги до команд виконавця. Усі команди повинні бути: точними; зрозумілими; виконуваними.

Ці властивості характеризують й поняття «система команд виконавця».

Властивості «зрозумілості» й «виконуваності» діти усвідомлюють інтуїтивно. Наприклад пояснюють, що виконавець водій розуміє (знає) й може виконувати лише команди, передбачені правилами дорожнього руху. У казці «Івасик Телесик» головний герой припливав до берега (виконував команди), коли впізнавав (розумів) голос матусі: «Івасику Телесику, приплинь, приплинь до бережка!» тощо. Для технічних пристроїв властивість «зрозумілості» є умовною (пристрої не можуть розуміти) й означає розпізнання тих команд, які належать до множини виконуваних ним команд (системи команд виконавця).

Властивість «точності» команд можна пояснити на основі аналогії з інструкцією, правилом, рецептом, які людині даються для виконання різних життєвих завдань (наприклад, рецепт приготування чаю, інструкція експлуатації технічного пристрою): чим точніше описані правила, тим швидше людина опанує їх і буде ефективніше застосовувати.

Поняття «алгоритм» є не означуваним, основним поняттям інформатики. Термін «алгоритм» походить від імені давнього філософа і математика із Хорезму — Аль-Хорезмі (IX століття). Він описав правила арифметичних дій над багатоцифровими числами. У сучасних підручниках поняття алгоритму вводиться описово,

при цьому нестрогі означення вказують на ті чи інші необхідні ознаки алгоритму.

Алгоритм — це послідовність точних, зрозумілих *команд* виконавцю, для розв'язання якогось завдання [7].

Усі дії записані у вигляді команд і у певній послідовності, а їх виконання призводить до розв'язання певної задачі. Такий план дій називають **алгоритмом** [8].

Алгоритм — це послідовність команд. До алгоритму можуть входити лише ті команди, які виконавець може виконати, тобто команди із системи команд цього виконавця [9].

Алгоритмом називають порядок команд, що дає змогу виконати певну задачу, а виконавцями — тих, хто їх виконує [6, 5].

Отже, попри уже відомі властивості команд (зрозумілість, виконуваність, точність) додаються нові властивості-ознаки — послідовність команд (команди повинні виконуватись у порядку їх слідування у записі алгоритму, зміна порядку слідування або пропуск команди призведе до неправильного результату); результативність (виконавши алгоритм, ми матимемо певний результат); правильність, що означає досягнення мети (результат повинен бути «розв'язком» задачі). Розумінню названих властивостей сприяє виконання завдань на складання алгоритмів, їх виконання, розміщення даних команд у необхідній послідовності, що призведе до розв'язання задачі (2 клас) [7, 8]; виправлення помилок у послідовності команд, виявлення зайвої або пропущеної команди (за методичним підходом Г. Ломаковської, Г. Проценко, Й.Ривкінда, Ф. Рівкінд такі завдання пропонуються у 3 класі [9]).

Правила подання алгоритму словесним способом можна сформулювати так: алгоритм має назву, у якій розкривається його призначення, наприклад «Алгоритм розв'язування рівнянь», «Алгоритм визначення будови слова», «Алгоритм «Банан» тощо. Усі команди нумеруються, що визначає послідовність їх виконання.

З навчальною метою передбачається невелика кількість команд у завданнях і прикладах.

Приклад. Алгоритм відкривання дверей ключем:

1. Дістати ключ.
2. Вставити ключ в отвір дверей.
3. Повернути ключ за годинниковою стрілкою.
4. Вийняти ключ.

Система команд виконавця визначає множину команд (елементарних дій), які виконавець розуміє і здатний їх виконувати. Уявлення про систему команд виконавця за підручником [7] формується конкретно-індуктивним способом. Приклади виконавців та їх систем команд ілюструються за допомогою таблиці. Таке унаочнення сприятиме усвідомленню того, що кожен виконавець може виконати лише команди із власної системи команд, а команди, що не входять до його системи, будуть не зрозумілі.

Інший підхід щодо усвідомлення поняття системи команд здійснений у підручнику [6]. Тут використаний метод порівняльного аналізу прикладів із застосуванням прийому абстрагування:

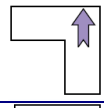
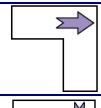
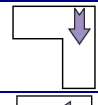
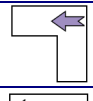
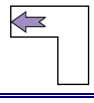
- виконавці абстраговані від дійсності (введення позначень: Виконавець №1, Виконавець №2, таблиці 1, 2);
- образи, на які спираються виконавці, знайомі учням і мають незначну зовнішню відмінність;
- система команд для кожного виконавця може добиратись самостійно і не співпадати із системою команд реального об'єкта, образ якого запозичений.

Це дає змогу підготувати грунт для розуміння комп'ютерного виконавця й системи його команд. Розглянемо приклад із підручника [4].

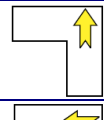
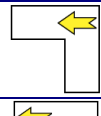
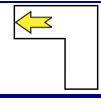
Задача для виконавців — проїхати по маршруту.

Отже, дія «поверни ліворуч» для другого виконавця є «елементарною» — для неї існує відповідна команда, для першого виконавця для цієї дії не існує відповідної команди у системі команд, тому вона виконується трьома командами. На нашу думку, доцільними будуть запитання для учнів: Які команди містить алгоритм виконавця №1? Які команди містить алгоритм виконавця №2? Чому перший алгоритм містить більшу кількість команд? Учні мають усвідомити, що алгоритми для виконавців будуються лише із системи його команд. Виконавець не зможе виконати алгоритм, написаний для іншого виконавця, бо не буде розуміти команд, які не належать до його системи команд. Діти повинні дійти висновку, що алгоритм будується лише з команд, які входять до системи команд виконавця.

Таблиця 1

Виконавець № 	
Уміє виконувати команди: Поверни праворуч Їдь уперед	
Алгоритм для виконавця 1	**Виконання алгоритму**
1. Їдь уперед	
2. Поверни праворуч	
3. Поверни праворуч	
4. Поверни праворуч	
5. Їдь уперед	

Таблиця 2

Виконавець № 	
Уміє виконувати команди: Поверни праворуч. Поверни ліворуч. Їдь уперед	
Алгоритм для виконавця 2	**Виконання алгоритму.**
1. Їдь уперед	
2. Поверни ліворуч	
3. Їдь уперед	

У сучасній методиці навчання інформатики виділяють труднощі, пов'язані з формуванням чіткого уявлення про алгоритм.

В уяві дитини є цілісний образ розв'язання завдання, а вимагається дати чіткий опис цього процесу у формі послідовності спонукальних речень.

У задачах такого типу важко виділити «елементарні» дії, з яких складатиметься алгоритм.

Під час виконання алгоритмів виконавець не повинен демонструвати елементи творчості — лише чітко виконувати команди у вказаному порядку [10, с. 18].

Для їх подолання методисти пропонують використання *рольового методу навчання*. Для цього виділяються ролі *пояснюючого* і *виконавця* алгоритму. Пояснюючий складає алгоритм, зачитує команди, виконавець програє накази. У разі неможливості виконання якоїсь команди, вона уточнюється чи деталізується до тих пір, поки не стане зрозумілою і виконуваною — «елементарною». Щоб виробити навички формального виконання алгоритму (виконавець не повинен розуміти суть проблеми чи завдання, намагатися додати власні дії, яких не має у записі алгоритму), учителі-практики використовують прийом складання алгоритмів для виконавця-робота (програвання ролі робота, згодом перенесення його основних властивостей і на саме поняття виконавця).

Для цього потрібно **актуалізувати знання учнів** про **Робота**. Ти знаєш що таке **Робот**? Спробуй відповісти на такі запитання:

Він живий? **(Ні)** Він з дерева? **(Ні)** Він з металу? **(Так)** Він машина? **(Так)** У нього є мозок? **(Ні)** Він сам може думати? **(Ні)** Хто ж ним керує? **(Людина)** А як же людина керує роботом? **(За допомогою комп'ютера)**.

Робот — це машина, зроблена з металу. Робот допомагає людям виконувати складну роботу. Людина керує роботом. Робот не може думати. Щоб керувати роботом, йому потрібно давати чіткі і зрозумілі **команди**.

Розглянемо гру «Виконай дії» [4]. Метою гри є створення колективного алгоритму для робота з виконання деякого завдання. Правила гри можуть бути такими:

1) Команди алгоритму повинні бути чіткими і зрозумілими.

2) Кожну команду для алгоритму робот повинен програти.

3) Команда може уточнюватись і програватись доти, доки не стане виконуваною.

Здійснену послідовність команд роботом потрібно записати.

На роль робота-виконавця спочатку може обиратися вчитель. Наприклад. «Учитель повідомляє дітям, що він на деякий час стане роботом. Давайте складемо алгоритм виконання завдання «Поливу квітів», які є у нас на підвіконні. Там же стоїть склянка. А вода набирається з крану рукомийника. Отже, якою буде перша команда?» Імовірно, що команди, запропоновані учнями, будуть «Набери у склянку води. Полий квіти». Учитель повинен з гумором реагувати на неправильні команди. Так, до першої команди можна продемонструвати, що робот не може дотягнутися ні до склянки, ні до води. Уточнення потребує команда візьми склянку — робот може показати дію, що хоче взяти склянку, яку йому хтось подасть. Тому потрібно уточнити, звідки він візьме склянку. Якщо пропущена команда «Відкрути кран» — робот може тримати склянку під краном, так і не набравши води. І т. д. Останньою повинна бути команда «Постав склянку на місце», бо без неї робот буде продовжувати тримати склянку. У результаті на дошці має бути записаний алгоритм поливу квітів.

1. Підійди до підвіконня.
2. Візьми склянку з підвіконня.
3. Підійди до рукомийника.
4. Відкрути кран.
5. Набери з під крану у склянку воду.
6. Закрути кран.
7. Підійди до підвіконня.
8. Полий квіти.
9. Постав склянку на її місце.

Важливе місце у системі тем «Алгоритми і виконавці» у чинних підручниках відведено виробленню уявлень про застосування алгоритму у повсякденній діяльності, а також під час виконання завдань з інших навчальних предметів: української мови, математики, трудового навчання тощо. Розширення кругозору учнів полягає в розумінні того, що самі діти, їхні рідні, домашні тварини, технічні пристрої для вирішення власних потреб, задач, послуг та ін. щоразу виконують алгоритми. Формування міжпредметних компетентностей передбачає уміння виконання завдання з іншого предмету на основі алгоритмічного підходу, що ілюструє схема:

Завдання → Алгоритм → Результат

Вироблення названих умінь за методичним підходом Г. Ломаковської та ін. потребує:

- актуалізації теоретичних знань (правила, означення, способу дій) з іншого навчального предмету;
- розгляд прикладу алгоритму (самостійне чи фронтальне його складання) на застосування теоретичних знань до розв'язування завдань;
- виконання алгоритму для 2–3 завдань.

Одним із методів побудови алгоритмів є **алгоритмічна мова**. Алгоритмічна мова — це текстова форма опису алгоритму. Її структурними елементами є алфавіт (символи), слова, команди та сукупність правил для написання алгоритмів. Уявлення про алгоритмічну мову у початковому курсі інформатики формується інтуїтивно, засобом її вивчення є виконавці у комп'ютерному середовищі. Виконавець комп'ютерного середовища повинен задовольняти наступним умовам:

1. Виконавець повинен працювати в певних умовах.
2. Виконавець повинен імітувати процес керування деяким реальним об'єктом.
3. У системі команд виконавця повинні бути усі структурні керування (лінійна, розгалуження, циклічна).
4. Виконавець дозволяє використовувати допоміжні алгоритми (для початкової школи не обов'язковий елемент).

Ще у 1980 році відомий американський учений С. Пайперт розробив навчальну мову програмування ЛОГО для роботи з молодшими школярами. За допомогою виконавця **Черепашки** діти малювали на екрані різні малюнки і так пізнавали основи алгоритмізації. Ці ідеї отримали розвиток, зокрема у російських програмах з виконавцем **Кресляр** (пакет програм для курсу Гейна), **Кенгуренок**, що реалізований фірмою КУДИЦ для IBM PC, **РОБОТ** з пакета «Кумир», **КУКАРАЧА** з пакета програм **«Роботландія»**. З Українських аналогів можна відзначити систему розвивальних завдань програмного комплексу «Сходинки до інформатики», розроблених під керівництвом ОлексіяАндрусича:

- **Ханойська вежа** (3–4 клас);
- Виконавець **«Садівник»** (3–4 клас);
- Виконавець **«Навантажувач»** (3–4 клас);
- Виконавець **«Кенгуру»** (3–4 клас);

• Виконавець **«Восьминіжка»** (3–4 клас).

У чинних підручниках «Сходинки до інформатики» [5, 8, 9] вивчення теми «Алгоритми і виконавці» спирається на роботу учнів у програмному середовищі Scratch, яке призначене для учнів молодшої школи й рекомендоване ЮНЕСКО. Розглянемо основні характеристики програми Scratch.

У перекладі з англійської іменник *scratch* має такі тлумачення: «карлючки», «скрип», «дряпання», «насічка», «мітка», «стартова межа» й т. ін.

Одним із принципів середовища Scratch є ідея щодо *складання програми мишкою з готових блоків-цеглин подібно до того, як діти будують будиночки і машинки з деталей конструктора*. Подібний спосіб складання програм знімає проблему синтаксису, що є істотно важливим для навчання молодших школярів. Значна частина операторів цієї мови призначена для роботи з графікою і звуком, створення анімаційних і відеоефектів. Створену у Scratch програму називають **проектом**. Середовище передбачає і колективну роботу над проектами й обмін результатами через сайт Scratch-товариства (адреса сайту http://scratch.mit.edu.).

Головне програмне вікно середовища (рис. 3) поділено на декілька частин, згрупованих у три стовпчики. **Лівий стовпчик** містить палітру блоків (групи команд, і команди виділеної групи). **Центральний стовпчик** містить піктограму активного спрайту з координатами його розташування на сцені та 3 закладки: **Скрипти**, **Образи**, **Звуки**. Активізація закладки **Скрипти** дозволяє будувати сценарій виконавцю з окремих команд-блоків, перетягуючи їх з лівого стовпчика. Закладка **Образи** призначена для додавання (готового чи створеного) зміненого зображення виконавця. **Правий стовпчик** містить **Сцену** і **Список спрайтів**.

Спрайт — *це об'єкт Scratch, що пов'язаний із зображенням, набором змінних і скриптів, які визначають його поведінку*. Як установлено, використовують спеціального виконавця вказівок — **Рудого кота**. Він може рухатися, говорити, змінювати зовнішній вигляд, взаємодіяти з іншими виконавцями на сцені. Інших виконавців можна долучати: з бібліотеки Scratch; з мережі (локальної чи глобальної); як об'єкти, створені в інших графічних програмах; як об'єкти, створені у графічному редакторі, вбудованому у Scratch.

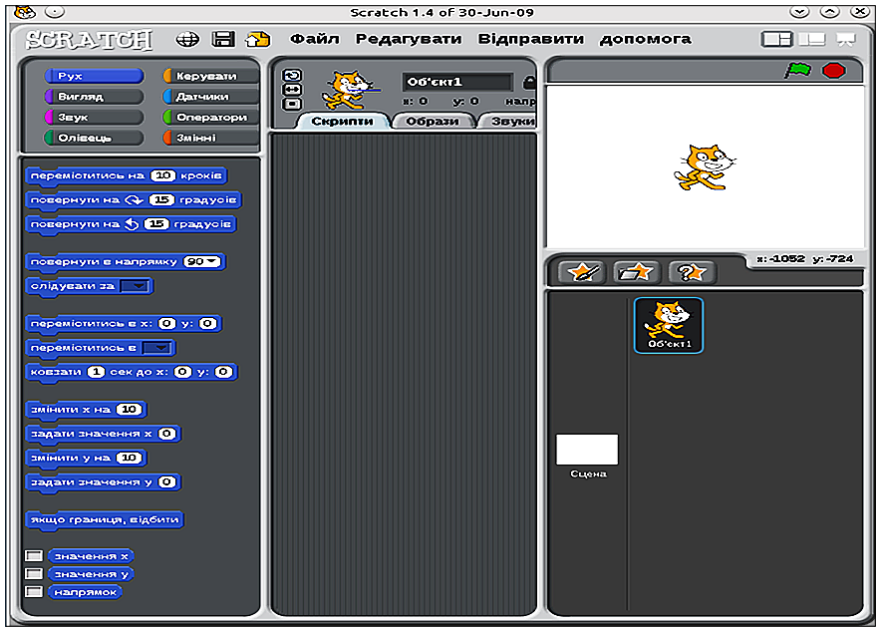


Рис. 3. Вікно програми Scratch

Скрипт (script, сценарій, метод, алгоритм) — *послідовність вказівок, що визначає, які дії і в якому порядку потрібно виконати певному об'єкту*. Скрипти створюють методом сполученням окремих блоків: або послідовно, або розташовуючи блок у визначеному місці іншого блоку (структури, що управляє, функції і т. і.). Один спрайт може мати декілька скриптів, які запускають незалежно дією користувача (натисненням клавіші або кнопки миші), таймером або отриманням повідомлення від іншого спрайту.

Образи (вигляд спрайту) — *сукупність зображень одного й того ж об'єкта (спрайту)*, кожне з яких дещо відрізняється від попередніх.

Звуки — *приєднані звукові ефекти й музика*.

Сцена — *область, у якій діє об'єкт (спрайт) під час виконання програми*.

Наприклад, для того щоб навчити **Спрайта** рухатись під музику, створюють скрипт у такій послідовності (рис. 4, 5).

Запустити на виконання скрипт — підвести до будь-якого блоку вказівник миші і двічі клацнути лівою кнопкою. **Рудий кіт** буде рухатися під звуки барабана.

За методичним підходом Г. Ломаковської та ін. [8, 9], вивчення команд мови виконавця **Scratch** має властивість, яку умовно назвемо дозуванням. За один раз діти мають опанувати 2–3 команди (2 клас) та 2–4 команди (3 клас), що моделюють одну–дві нескладні дії (говорити, змінювати розмір, рух в одно-

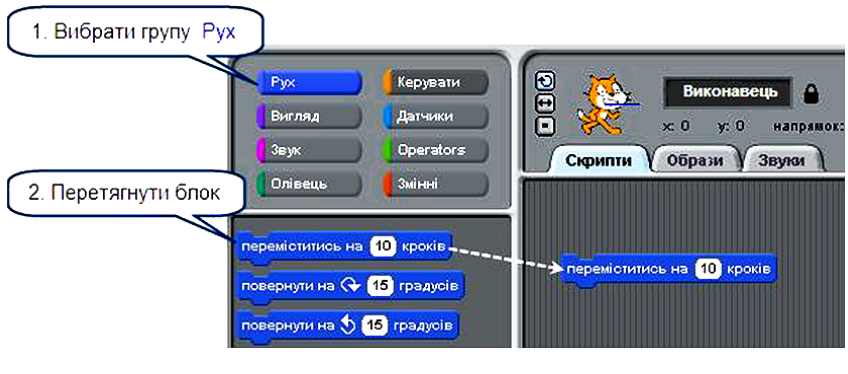


Рис. 4

Рис. 5

му напрямі зі звуком, рисувати лінію одним кольором, додавання виконавця, рух без зупинки, зміна образу, зміна фону тощо). Отже, у 3 класі передбачений лише один складний Проект **Scratch**, що моделює сюжет історії.

Учнів ознайомлюють із **графічним способом** подання алгоритму (4 клас) за допомогою блок-схеми. Цей спосіб реалізовує принцип побудови алгоритму з цеглинок-блоків і служить гарним засобом унаочнення під час вивчення алгоритмічних структур (слідування, розгалуження та повторення).

Потрібно звернути увагу на правила побудови алгоритму:

1) Для запису команд використовуй геометричні фігури прямокутники.

2) Геометричні фігури з'єднуй лініями і стрілочками.

3) Початок і кінець алгоритму зображуй овалами.

Наприклад, алгоритм обчислення значення виразу 20-(40+24):8 буде мати вигляд (рис. 6).

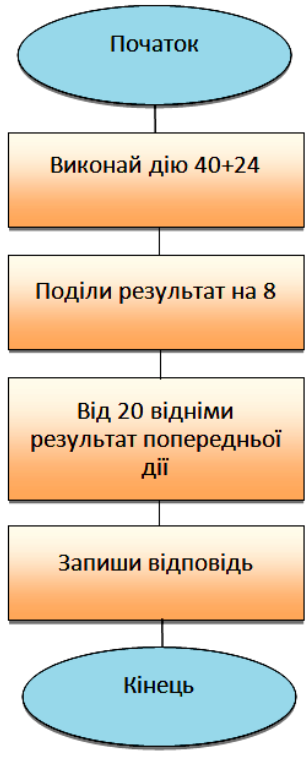


Рис. 6

Алгоритм називається **лінійним**, якщо всі команди виконуються послідовно (слідують) одна за одною. Уявлення про лінійний алгоритм можна закріпити проведеною аналогією з потягом, у якому кожний вагон — це команда [4].

Розглянемо введення поняття розгалуженого алгоритму проблемно-пошуковим способом (приклад автора). Для цього можна запропонувати колективно скласти алгоритм переходу вулиці. Виявлення проблеми — неможливості запису команд у строго лінійному порядку, бо залежно від умови буде виконуватись або одна команда, або інша. Ми приходимо до висновку, що алгоритм буде мати структуру, відмінну від лінійної. Якщо в алгоритмі потрібно дати відповідь на запитання (перевірити умову), і залежно від відповіді (так або ні) на нього, зробити вибір з двох команд, то такий алгоритм називається **розгалуженим.**

Блок-схема розгалуженого алгоритму буде містити новий блок. **Ромб** — це блок, де ми задаємо запитання (перевіряємо мову). Після перевірки ми переходимо до виконання команди 1 (якщо умова виконується — відповідь **Так**) або до команди 2 (якщо умова не виконується — відповідь **Ні**) (рис. 7) . Після цього закінчують побудову алгоритму переходу вулиці (рис. 8).

Уявлення дітей про алгоритм з розгалуженням розширюється на основі розгляду прикладів блок-схем алгоритмів, у яких одна з віток розгалуження містить більше ніж одну команду. Ознайомлюють учнів із блок-схемою неповної форми розгалуження (рис. 9). У сучасних підручниках і методичних посібниках поширено введення поняття алгоритму з циклом конкретно індуктивним способом [1, 10]. Особливістю його використання є пригадування подій, явищ, ситуацій, казок тощо, де б чіткими були повторювані дії. Наприклад, ми повторюємо дії — виконати крок вперед, коли йдемо до школи, помити тарілку — доки є брудний посуд, день змінює ніч, а ніч день. Після цього наводиться приклад алгоритму з повторенням, що розв'язує наочну й зрозумілу задачу (наприклад, роботом-виконавцем необхідно розвісити рисунки на стіні), розкрити правила його виконання і суттєві ознаки нового поняття — алгоритм з циклом.

Розглянемо введення цього поняття частково-пошуковим методом (приклад автора). Для цього запропонуємо учням завдання.

Нехай нам потрібно наповнити відро місткістю 10 літрів те-

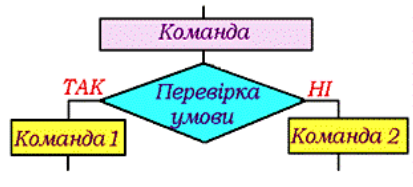


Рис. 7

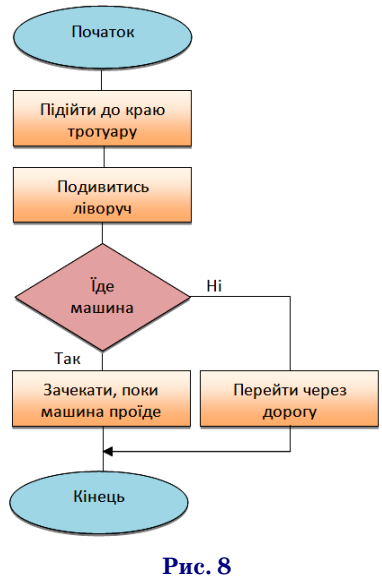


Рис. 8

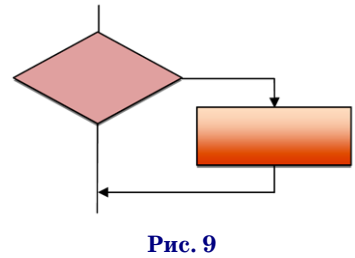


Рис. 9

плою водою. У нас є кварта місткістю 1 літр та крани з гарячою і холодною водою. Щоб отримати теплу воду змішують 1 літр гарячої води і 1 літр холодної води.

Скласти алгоритм наповнення відра водою. Передбачимо, що алгоритм, запропонований учнями може бути таким (рис. 10).

У чому його незручність? (*У тому, що необхідно записувати декілька раз одні й ті ж команди*). Цей алгоритм можна подати в іншій, більш зручній формі, яка називається циклічною. **Цикл** — це група команд в алгоритмі, які записуються один раз, а виконуються багато разів — стільки, скільки потрібно. Які команди ми повторюємо? (*Долити у відро 1 л гарячої води. Долити у відро 1 л холодної води*). До

Рис. 10

яких пір ми будемо повторювати ці команди? (*Доки відро не наповниться*). Отже, виконання команд в алгоритмі відбувається так: виконавши команди циклу, ми щоразу перевіряємо **умову**: «відро наповнене?». Що потрібно робити, якщо умова не виконується? (*Продовжувати наповнювати*). А якщо умова виконується? (*Закінчити наповнювати відро водою*). З яких блоків складатиметься алгоритм? (*два овали — початок і кінець, два прямокутники — команди, що повторюються і ромб — перевірка умови*). Матимемо блок-схему алгоритму, що відповідає нашим поясненням (рис. 11).

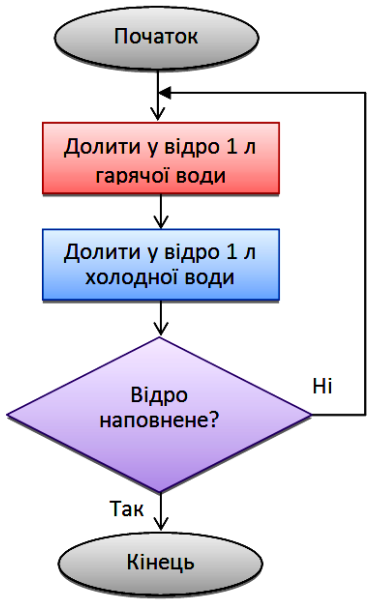


Рис. 11

Якщо в алгоритмі можна виділити послідовність дій, яка виконується підряд декілька разів, то такий алгоритм називають **циклічним**. У діючих підручниках розглядаються алгоритми, у яких умова записана на початку циклу. Пропонуються й більш ускладнені завдання, коли алгоритм містить два цикли, один із яких вкладений і алгоритми, де умова може бути не в запитальній формі (висловлення містить слово **доки**). Наприклад, алгоритм «Збирання ягід» (рис. 12). У тому випадку, коли ми наперед знаємо скільки разів будемо виконувати цикл команд, запис алгоритму набуває вигляду (рис. 13). Методичний підхід авторів підручника [11] щодо вивчення основних алгоритмічних структур полягає в одночасному їх розгляді графічним

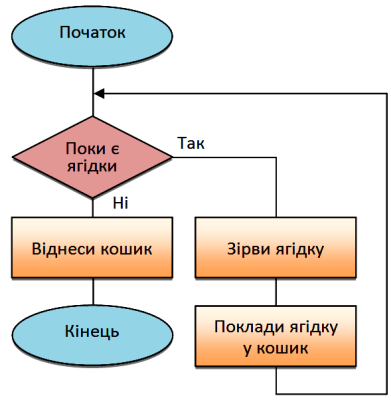


Рис. 12

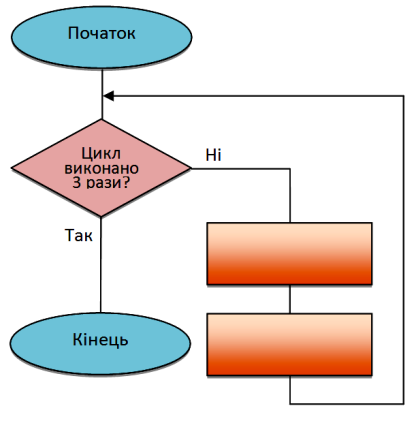


Рис. 13

способом і способом побудови алгоритмів для виконавця комп'ютерного середовища.

Зокрема, під час побудови алгоритмів для виконавців з програмного комплекту «Сходинки до інформатики» використовують команди розгалуження та повторення таких логічних структур:

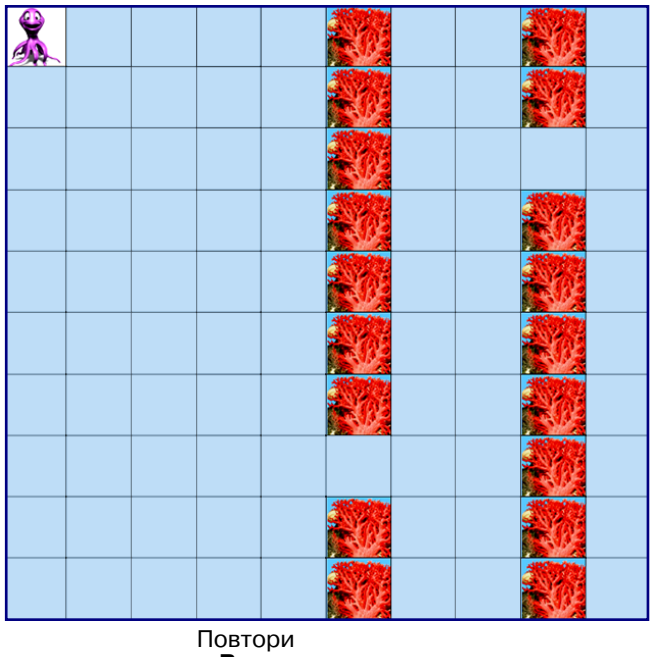
- **Якщо...інакше... все** (у середовищі виконавця **Восьминіжка**).
- **Повтори...3** (4 і т. д.) **рази** (у середовищі виконавців **Садівник** та **Кенгуру**).
- **Повтори...поки...** (у середовищі виконавців **Восьминіжка** та **Кенгуру**).

Уявлення про ці структури команд виробляється інтуїтивно на основі розглядуваних прикладів для конкретного комп'ютерного виконавця. Такі завдання, як правило, складаються з двох частин. Перша частина — це алгоритм-приклад на застосування нової команди у середовищі або опис її структури. Друга частина передбачає:

- змінити поданий алгоритм-зразок для нових умов;
- конструювання алгоритму за вказівками;

- складання алгоритму для ускладненого завдання попередньої практичної робот;
- доповнення поданого алгоритму, який забезпечує виконання лише частини завдання.

Приклад. Справа від **Восьминіжки** корали, що заважають її руху (рис. 14). Їй потрібно знайти прохід. У підручнику приведений алгоритм виконання першої частини завдання — передбачає знаходження отвору у першій справа стінці корала.



Повтори
Вправо
Поки Справа вільно
Повтори
Вниз
Поки Справа перешкода
Вправо
Вправо

Рис. 14

Діти записують частину алгоритму у вікні **Програма**. Бажано переконатись, що команди циклу для восьминіжки зрозумілі учням. Які дії зробить восьминіжка, виконуючи 1-шу команду? Відповідь дітей запропонуйте перевірити, запустивши програму на виконання покроково (клавіша **Крок**). Так само перевіряються й наступні команди. При цьому кожний крок на екрані вікна восьминіжки синхронізується з виділенням відповідної команди у вікні **Програма**, де результат перевірки умови позначається словами **Так** або **Ні**. Які дії зробить восьминіжка для подолання другої перешкоди? Якими командами потрібно записати? Запишіть їх. Перевірте правильність алгоритму. Запустіть на виконання.

Отже, аналіз чинних програм, підручників, методичних посібників, підручників з методики навчання інформатики дозволив виділити методичні особливості викладання теми «Алгоритми і виконавці», а саме: розкриті місце і завдання теми у підготовчому курсі інформатики, охарактеризовані суть і правила способів подання алгоритмів, виділені відмінності методичних підходів щодо послідовності, прийомів та методів введення й формування понять, з'ясовані труднощі, що виникають у процесі набуття уміння складати алгоритми та розкриті методичні прийоми щодо їх усунення, в окремих випадках наведені й авторські приклади. Знання особливостей викладання теми «Алгоритми та виконавці» будуть не повні, якщо залишити без уваги аналіз інших підручників, навчально-методичних комплектів, затверджених МОН України. Це завдання відводиться для самостійної роботи студентів і перевіряється на практичних заняттях.

★ ★ ★

Стрелецкая Н.М. Анализ методических подходов преподавания темы «Алгоритмы и исполнители» в курсе «Ступеньки к информатике»

Аннотация. В статье автор раскрывает методические особенности обучения темы «Алгоритмы и исполнители» в курсе «Ступеньки к информатике» на основе анализа действующих программ, учебников, методических пособий и опыта педагогов-практиков.

Ключевые слова: алгоритмы и исполнители, способы представления алгоритмов, программная среда Scratch, трудности формирования представления о алгоритме, методы обучения.

★ ★ ★

Streletska N.M. An analysis of the methodological approaches to teaching of theme «Algorithmes and Doers» at the course «Steps to Information Technology»

Annotation. The article is dedicated to methodological peculiarities of the teaching of theme «Algorithmes and Doers» at the course «Steps to Information Technology» that is based on analysis of the modern valid programmes, textbooks, methodological handbooks and an experience of teachers-experts.

Key words: algorithmes and Doers, methods of presenting algorythmes, programmed medium Scratch, the troubles of formation ideas about the algorithm, the methods of teaching.

Література

1. Інструктивно-методичні рекомендації щодо викладання інформатики у 2013–2014 н. р. [Електронний ресурс]. — Режим доступу: http://mmk.edu.vn.ua/ index.php/ 2011–10–04–12–01–08.
2. Конспекти уроків з інформатики 4 клас [Електронний ресурс]. — Режим доступу: http://www.docme.ru/doc/76866/ konspekti-urok%D1%96v-%D1%96nformatiki-4-klas.
3. Державний стандарт початкової загальної освіти [Електронний ресурс]. — Режим доступу: http://www.mon.gov.ua/ education/average
4. *Земляк І. В.* Методичні рекомендації щодо викладання інформатики в 3-му класі: за програмою «Сходинки до інформатики» / І.В. Земляк — Кам'янець-Подільський, 2009. — 114 с.
5. *Коршунова О. В.* Сходинки до інформатики : підруч. для 3-го кл.загальноосвіт. навч. закл. / О. В. Коршунова. — К. : Генеза, 2014. — 176 с. : іл.
6. *Коршунова О. В.* Методика викладання інформатики у 2-му класі / О. В. Коршунова. — Х.: ФОП Співак В.Л., 2013. — 112 с.
7. *Коршунова О. В.* Сходинки до інформатики: підруч. для 2 кл. загальноосвіт. навч. закл. / О.В. Коршунова. — К.: Генеза, 2012. — 112 с.
8. *Ломаковська Г. В.* Сходинки до інформатики : підруч. для 2 кл. загальноосвіт. навч. закладів / Г.В. Ломаковська, Г.О. Проценко, Й.Я. Ривкінд, Ф.М. Рівкінд. — К.: Видавничий дім «Освіта», 2012. — 160 с.
9. *Ломаковська Г. В.* Сходинки до інформатики: підруч. для 3 кл. загальноосвіт. навч. закладів. / Г.В. Ломаковська, Г.О. Проценко, Й.Я. Ривкінд, Ф.М. Рівкінд. — К.: Видавничий дім «Освіта», 2013. — 160 с.
10. *Морзе Н. В.* Методика навчання інформатики: навч. посібник: у 4 ч./ за ред. акад. М.І. Жалдака / Н.В. Морзе. — К.: Навчальна книга, 2004. Ч. IV: Методика навчання алгоритмізації та програмування. — 368 с.
11. Сходинки до інформатики: підруч. для 4 кл. загальноосвіт. навч. закл. / С.Я. Колесніков, Г.В. Ломаковська, Ф.М. Рівкінд, Й.Я. Ривкінд — 3 вид., перероб. — К.: Світич, 2009 — 69 с.
12. Програма курсу «Сходинки до інформатики»: 2–4 класи загальноосвітніх навчальних закладів // Інформатика та інформаційні тех. в навч. закладах. — 2011. — №4–5. — С. 38–49.
13. *Коршунова О.* Зошит «Сходинки до інформатики» : навч. посіб. для загальноосвіт. навч. закл.: 2-й кл. / О.В. Коршунова. — К.: Генеза, 2013. — 64 с.