УДК 004.4

ІНСТРУМЕНТАЛЬНІ ЗАСОБИ РОЗРОБКИ ПРОГРАМНОГО КОДУ, НАПИСАНОГО МОВОЮ ПРОГРАМУВАННЯ ВИСОКОГО РІВНЯ

Алфьоров Є.А.
Херсонський державний університет

У статті представлено Середовище демонстрації програм інтегрованого середовища вивчення курсу «Основи алгоритмізації та програмування» (http://weboap.ksu.ks.ua), що дозволяє проводити обчислювальний експеримент для вивчення складності та мажорованості алгоритмів сортування. Описано проектування та розробку нової версії програмного додатку. Велика увага приділяється розробці компонента Редактора коду, що відповідатиме сучасним вимогам до інструментальних засобів написання програм.

Ключові слова: *основи алгоритмізації та програмування, середовище демонстрації програм, обчислювальний експеримент, редактор коду.*

Постановка проблеми. На сьогоднішній день постає проблема створення зручного інтерактивного компонента дослідницького середовища демонстрації та, зокрема, його редактора коду за допомогою сучасних технологій розробки для підвищення ефективності вивчення курсу «Основи алгоритмізації та програмування» у вищих навчальних закладах. Адже цей курс забезпечує фундаментальну підготовку майбутнього спеціаліста.

При традиційному навчанні програмістів велику увагу приділяють вмінню складати алгоритми, вивченню мов програмування та вмінню працювати з транслятором. Для забезпечення сприятливих умов при роботі фахівців у цій галузі, необхідною складовою є Інструментальне ПЗ або системи програмування – системи для автоматизації розробки нових програм.

У загальному випадку для створення програми потрібно мати такі компоненти:

1. Текстовий редактор для створення файлу з вихідним текстом програми.
2. Компілятор або інтерпретатор. Вихідний текст за допомогою програми-компілятора переводиться в проміжний об'єктний код. Вихідний текст великої програми складається з декількох модулів (файлів з вихідними текстами). Кожен модуль компілюється в окремий файл з об'єктним кодом, які потім треба об'єднати в одне ціле.
3. Редактор зв'язків або збирач, який виконує зв'язування об'єктних модулів і формує на виході працездатний додаток – закінчена програма, яку можна запустити на будь-якому комп'ютері, де встановлена операційна система, для якої ця програма створювалася. Як правило, підсумковий файл має розширення .exe або .com.

Сучасні потужні редактори надають розробникам величезні можливості, такі як підсвічування синтаксису, автодоповнення Intellisense, макроси, плагіни, можливості попереднього перегляду і FTP-менеджери. Деякі редактори йдуть ще далі і пропонують повністю інтегроване середовище розробки з численними можливостями і функціями.

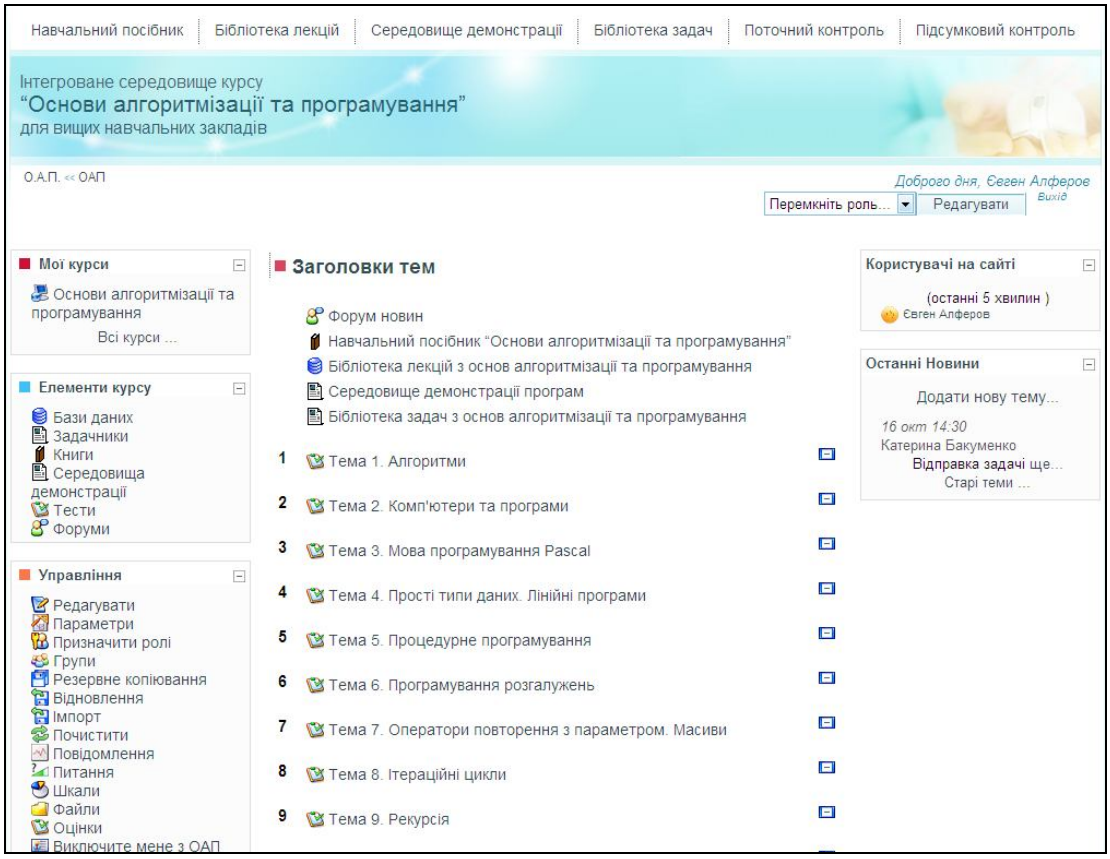
Аналіз останніх досліджень і публікацій. У інтегрованому середовищі вивчення курсу «Основи алгоритмізації та програмування» (мал. 1) пропонується не просто вивчити лексичні конструкції мови програмування, а більш детально зупинитися на способах алгоритмізації та їх широкому застосуванні при розв’язані поставлених задач. Також пропонується разом із вивченням теоретичного матеріалу проводити обчислювальний експеримент для вивчення складності і підвищення ефективності алгоритмів. Такого роду підхід до змісту посилює дослідницьку діяльність студентів, фундаментальну предметну

підготовку майбутніх фахівців, за рахунок формально-логічного відображення причинно-наслідкових зв'язків і, як наслідок, до впливу на мотивацію студентів [1].

Під обчислювальним експериментом розуміється метод вивчення об'єктів та процесів за допомогою математичного моделювання. Експеримент передбачає, що після побудови математичної моделі проводиться її чисельне дослідження, що дозволяє відтворити поведінку досліджуваного об'єкту в різних умовах або в різних модифікаціях.

За допомогою комп'ютерного експерименту студент має можливість зрозуміти особливості певних алгоритмів та усвідомити залежності, що пояснюють їх складність [1].

Обчислювальний експеримент з вивчення ефективності алгоритмів проводиться за допомогою спеціального модуля «Середовище демонстрації» (мал. 2).



Мал. 1.Головна сторінка інтегрованого середовища вивчення курсу «Основи алгоритмізації та програмування»

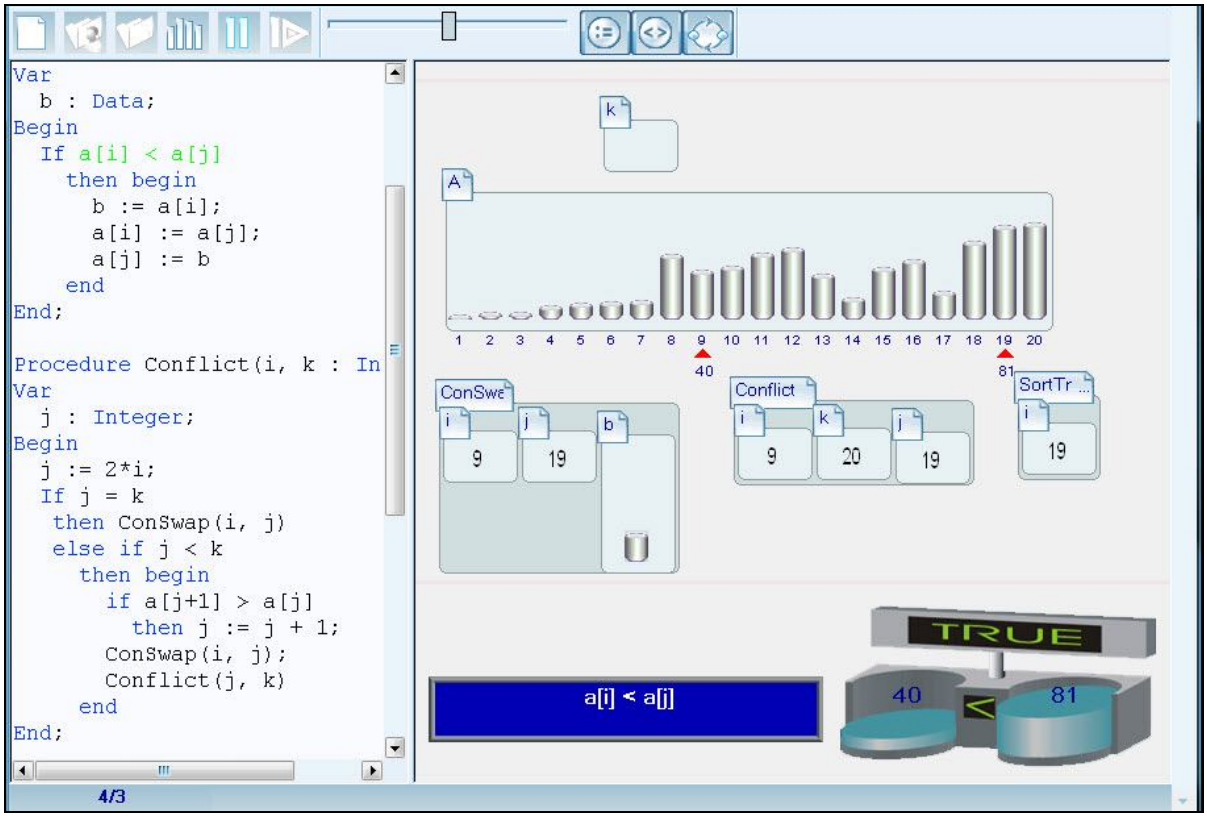
У сьогоднішній версії модуля середовища демонстрації програм інтегровано модифікований інтерпретатор мови Pascal. Ця версія створена за допомогою парсера мов ANTRL – Another Tool Recognition Language. Ця версія виконана у вигляді окремої статичної бібліотеки, яка використовується та компілюється у модуль середовища демонстрації.

Середовище демонстрації створено як ActiveX компонент, тому для виконання алгоритмів необхідно використання браузера Internet Explorer та платформи Windows.

Дана робота є логічним продовженням розвитку педагогічно-орієнтованих систем підтримки практичної діяльності, які розробляються науково-дослідним інститутом інформаційних технологій Херсонського державного університету, і базується на програмно-методичному комплексі «Відеоінтерпретатор алгоритмів пошуку та сортування» [2, 3].

Відмітимо, що у процесі написання програмного коду або скриптів, з метою реалізації певного алгоритму певною мовою програмування, насамперед потрібен зручний і практичний редактор коду. Кодування є частиною програмування, поряд з аналізом, проектуванням, компіляцією, тестуванням.

Формулювання цілей статті (постановка завдання). Розробка програмного коду сьогодні – величезна індустрія. Інтегровані середовища розробки, що спрощують і підтримують процес створення і налагодження вихідних кодів програм, – один з найбільш потрібних на ринку видів програмного забезпечення. Популярні мови програмування найчастіше поширюються у вигляді багатофункціональних інтегрованих середовищ, що надають і спрощують програмісту доступ до можливостей мови. Складність сучасних редакторів, компіляторів, інтерпретаторів вимагає від середовищ розробки все більшої автоматизації, виключення з роботи програміста рутинної технічної роботи.



Мал. 2. Середовище демонстрації програм інтегрованого середовища вивчення курсу «Основи алгоритмізації та програмування»

На жаль, розробка та впровадження підтримки нових мов програмування з використанням практично будь-яких з існуючих середовищ – складна і трудомістка задача. Редактор грає головну роль у підвищенні продуктивності праці розробника, саме в його середовищі програміст проводить основний час в процесі створення програмного забезпечення. Інтелектуальні функції редактора дозволяють реалізувати в ньому засоби допомоги та налагодження.

Мета даної роботи: проаналізувати й описати архітектуру і функціональність нової версії середовища демонстрації програм, зокрема і редактор коду як один із його компонентів.

Виклад основного матеріалу дослідження. Якщо Ви розробляєте ПЗ, вам обов'язково буде потрібен текстовий редактор для написання коду або ж просто для редагування конфігураційних файлів. Таких редакторів незліченна кількість, причому всі ведуть себе по-різному, мають свої слабкі і сильні сторони.

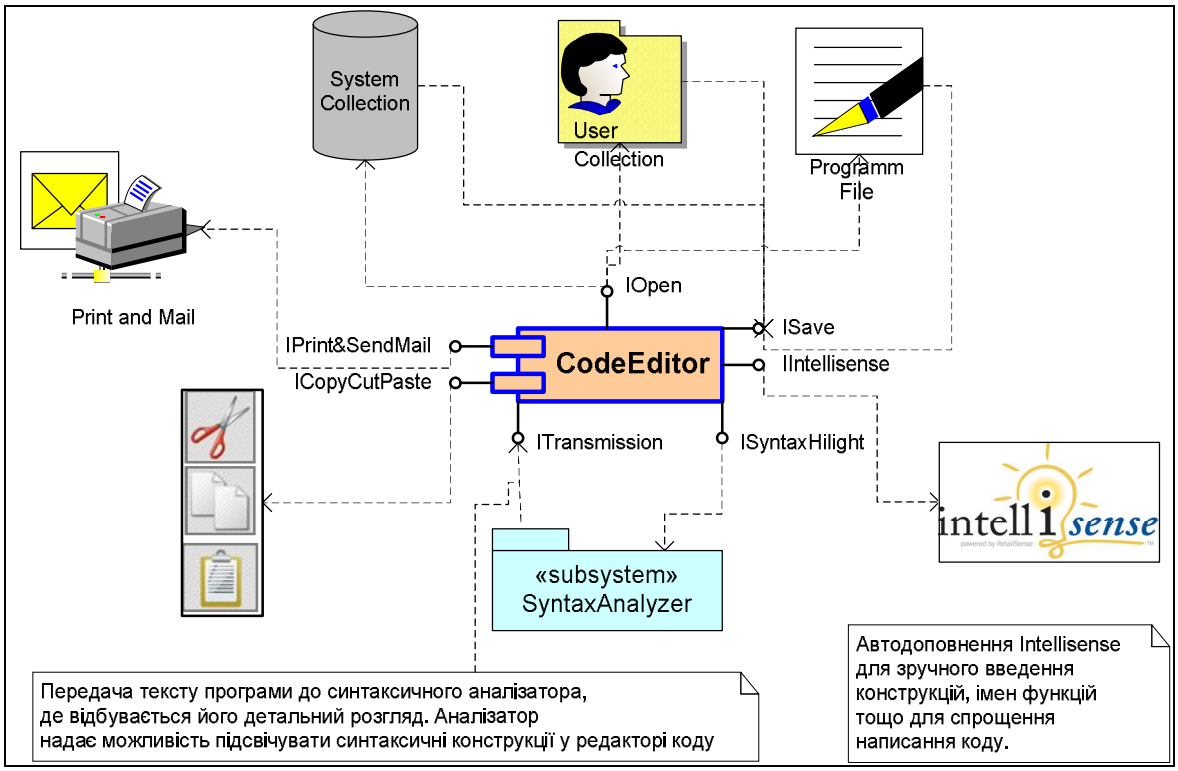
Редактор коду забезпечує ряд можливостей, що допомагають в написанні та редагуванні коду. Конкретні можливості та їх розташування варіюються залежно від мови розробки і поточних параметрів.

Багато редакторів коду надають функції, які особливо корисні для програмістів. Підсвічування синтаксису і автоматичні відступи – це, мабуть, найважливіші з інструментів

такого роду. Вони дозволяють з одного погляду зрозуміти, що введений код в загальних рисах коректний, що всі блоки правильно вкладені один в один і не містять очевидних помилок.

Розглядаючи різні функціональні можливості багатьох редакторів коду, можна вивести загальний список або перелік основних вимог до сучасного редактора:

- Підсвічування тексту і можливість згортання блоків, згідно синтаксису мови програмування.
- Підтримка великої кількості мов (C, C++, Java, XML, HTML, PHP, Java Script, ASCII, VB/VBS, SQL, CSS, Pascal, Perl, Python, Lua, TCL, Assembler).
- WYSIWYG (друкуєш і отримуєш те, що бачиш на екрані).
- Налаштування користувачем режиму підсвічування синтаксису.
- Авто-завершення слова, що набирається.
- Нумерація рядків.
- Одночасна робота з декількома документами.
- Одночасний перегляд декількох документів.
- Підтримка регулярних виразів Пошук / Заміни.
- Підтримка Drag&Drop.
- Динамічна зміна вікон перегляду.
- Нотатки (написання коментарів).
- Виділення дужок при редагуванні тексту.



Мал. 3.Діаграма взаємозв'язку компонента Редактор Коду (Code Editor) з іншими компонентами середовища демонстрації

На мал. 3 можна побачити взаємодію редактора коду з іншими компонентами у дослідницькому середовищі демонстрації програм. Тут також представлені основні інтерфейси, які забезпечують функціональність системи, що відповідає переліку вимог до сучасного редактора.

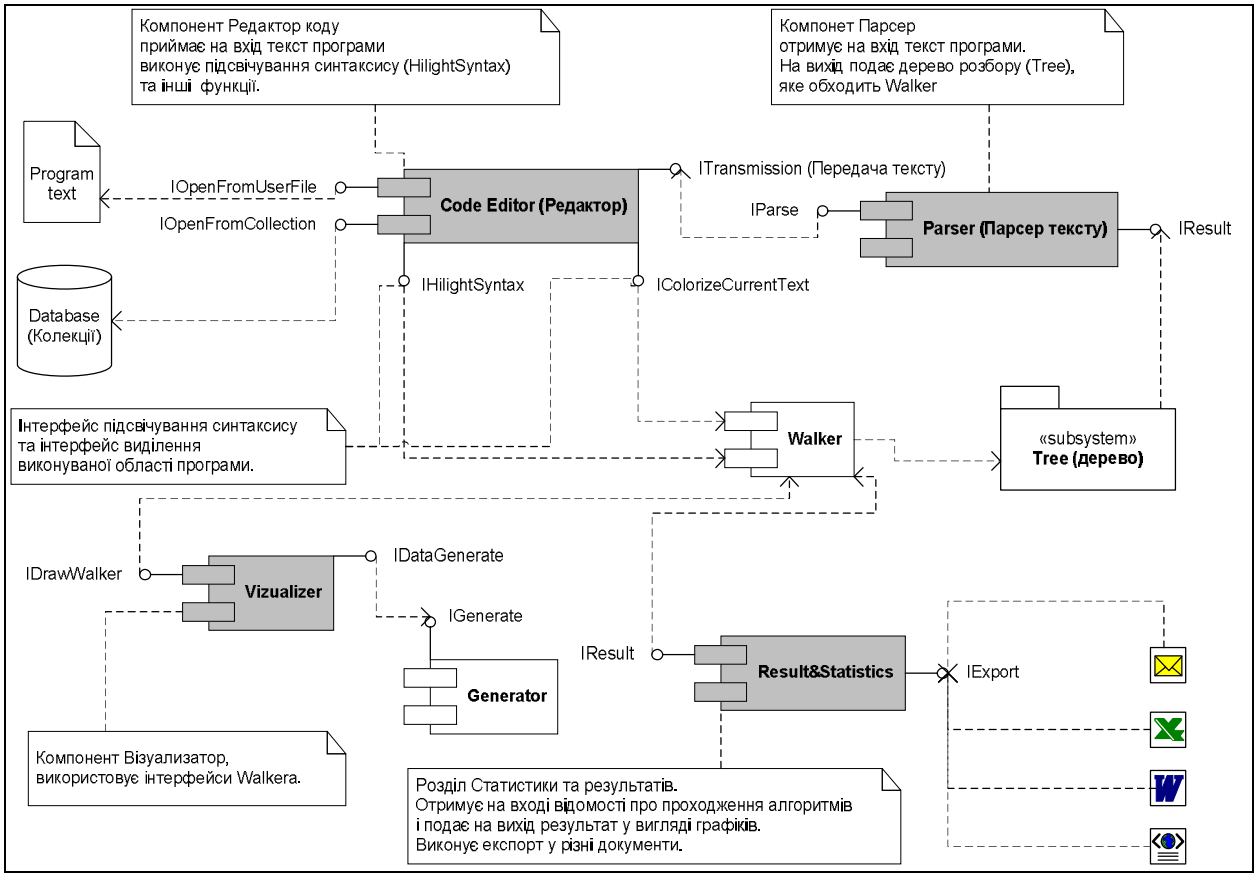
За допомогою новітніх технологічних засобів ведеться робота над вдосконаленням та розширенням функціональних можливостей редактора коду, що спрощують процес редагування.

Виконання такої роботи передбачає розв'язання ряду задач:

- Дослідити можливості і функції популярних редакторів коду.
- Проаналізувати сучасні технологічні засоби для максимальної відповідності функціональним вимогам.
- Розробити зручний і багатофункціональний редактор коду.
- Домогтися кросплатформеності додатка.
- Інтегрувати редактор в дослідницьке середовище демонстрації.

На даний момент вже ведеться розробка нової версії середовища демонстрації з використанням технології Silverlight. Планується забезпечити максимальну кросплатформеність, інтерактивність і зручність програмного додатку. Майбутнє середовище демонстрації надасть можливість перегляду інтерпретації алгоритмів на таких мовах програмування високого рівня як Pascal, C та Java.

Розробка нового редактора передбачає можливість редагування файлів в одній сесії, створення коду за допомогою автодоповненя Intellisense, звернення блоків коду для більш зручного читання, написання коментарів до коду, підсвічування синтаксису, перевірка правильності розташування дужок, підсвічування коду, що виконується, а також деякі інші додаткові можливості, що спрощують процес написання програмного коду.



Мал. 4. Діаграма всіх компонентів середовища демонстрації та їх взаємодії між собою

Крім того, ведеться робота над компонентом середовища демонстрації, який забезпечує візуалізацію виконання алгоритмів. Заплановано створення ще одного компоненту, який дасть можливість аналізувати складність та ефективність алгоритмів із колекції системи та з власної колекції користувача.

У новій версії передбачається реконструкція інтерфейсу користувача, що буде відповідати основним принципам проектування інтерфейсу.

Система Середовища демонстрації програм передбачає 6 основних компонентів:

- Редактор коду (*Code Editor*);
- Генератор вхідних даних (*Data Generator*);
- Синтаксичний аналізатор (*Syntax Parser*);
- Обхідник (*Walker*);
- Візуалізатор (*Visualizer*);
- Результати та статистика (*Result and Statistics*).

Взаємозв'язок компонентів представлений на діаграмі компонентів (див. мал. 4).

IOpenFromUserFile інтерфейс забезпечує відкриття файлу користувача з текстом програмного коду.

IOpenFromCollection надає можливість користувачу відкрити у середовищі демонстрації одні з фундаментальних алгоритмів, що містяться в колекції системи.

IHighlightSyntax – інтерфес редактора коду, що виконує підсвічування синтаксису в коді програми. Отримує інформацію про відповідну синтаксичну структуру із компонента *Walker*.

ITransmission – інтерфейс редактора коду, яким користується синтаксичний аналізатор для побудови абстрактного дерева (*Tree*) алгоритму.

IDataGenerate забезпечує генерацію вхідних даних для програми.

IColorizeCurrentStatement – інтерфейс, що виділяє ту мовну конструкцію, яка у даний момент виконується.

IParse – інтерфейс синтаксичного аналізатора, який використовує *ITransmission* для отримання тексту програми. Розбирає написаний код на необхідні структури і будує абстрактне дерево.

IDrawWalker – інтерфейс Візуалізатора, що отримує дані від синтаксичного аналізатора і виконує графічну інтерпретацію роботи алгоритму.

IResult – інтерфейс компонента Результати та статистика. Отримує відомості про виконання алгоритму (кількість порівнянь, присвоєнь, час виконання тощо) і на їх основі показує статистичні результати.

Висновки. Середовище демонстрації програм інтегрованого середовища вивчення курсу «Основи алгоритмізації та програмування» надає багато можливостей для ефективного вивчення курсу з основ алгоритмізації та програмування, серед яких:

- демонстрація роботи алгоритмів;
- можливість проведення обчислювального експерименту для вивчення складності та мажорованості алгоритмів сортування;
- отримання зручного інструментарія для формування вмінь та навичок складання алгоритмів, аналізу їх складності під час виконання обчислювального експерименту;
- можливість узагальнити результати аналізу алгоритмів при порівнянні різних методів розв'язання задачі.

Теоретична цінність роботи над вдосконаленням Середовища демонстрації полягає в дослідженні основних відомостей про новітні технології, що забезпечують максимальну кросплатформеність при створенні програмних додатків.

Практична цінність роботи полягає у створенні зручного інтерактивного компонента інтегрованого дослідницького середовища демонстрації програм.

Цей напрямок розробки приділяє увагу підтримці лише деяких базових синтаксисів мов програмування. Сьогодні це здебільшого «mainstream» мови – Pascal, C, Java. Повноцінна ж реалізація середовища для нової мови програмування (або групи мов) – трудомістка і нетривіальна задача навіть при наявності готової інфраструктури і скелета системи.

На сьогоднішній день існує велика кількість редакторів коду з різним ступенем функціональності і якістю підтримки мов програмування. Однією з центральних

можливостей при цьому стає аналіз структури вихідного коду програми і забарвлення його на льоту (в процесі редагування користувачем). Головний показник тут – універсальність. Не багато текстових редакторів можуть похвалитися великим списком мов програмування, що підтримуються. У багатьох випадках кількісні показники ніяк не корелюють з якістю реалізації забарвлення та аналізу вихідного коду.

Виконуючи розробку нової версії Середовища демонстрації, маємо за мету створення програмного додатку, який розширить коло можливостей з ефективного вивчення основ алгоритмізації та програмування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Основи алгоритмізації та програмування. Обчислювальний експеримент. Розв'язання проблем ефективності в алгоритмах пошуку та сортування: Навчальний посібник/А.В. Співаковський, Н.В. Осипова, М.С. Львов, К.В. Бакуменко. – Херсон: Айлант, 2010. – 100 с.: іл..
2. Педагогічні технології та педагогічно орієнтовані програмні системи: предметно-орієнтований підхід /О.В. Співаковський, М.С. Львов, Г.М. Кравцов [та ін.] //Комп'ютер у школі та сім'ї. – №4(22), 2002 – С. 24-28
3. Співаковський О.В. Відеоінтерпретатор алгоритмів інтегрованого середовища вивчення курсу "Основи алгоритмізації та програмування" / Співаковський О.В., Колеснікова Н.В. // Нові інформаційні технології в освіті для всіх: система електронної освіти. – 2008. – № 3. – С. 399-404.