ТЕОРЕТИЧНА ЕЛЕКТРОТЕХНІКА *ELECTRICAL ENGINEERING*
2010. Вип. 61. С.195-204 *2010. Is. 61. P.195-204*

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В НАВЧАЛЬНОМУ ПРОЦЕСІ

УДК 681.3.06(075)

ПРО ВИКОРИСТАННЯ ВІЛЬНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З МЕТОЮ СТВОРЕННЯ НАВЧАЛЬНОЇ ОБОЛОНКИ ДЛЯ ВИВЧЕННЯ МОВ ПРОГРАМУВАННЯ

Г. Злобін, В. Скляр, О. Чмихало, В. Шевчик

Львівський національний університет імені Івана Франка
кафедра радіофізики
вул. Ген. Тарнавського, 107, 79017, Львів, Україна

Описано досвід розробки і використання навчальної оболонки для вивчення мов програмування на базі вільного програмного забезпечення.

Ключові слова: програмування, компілятор, вільна програмне забезпечення, Linux.

Майже десятирічний позитивний досвід використання навчальної оболонки Algo для вивчення мови програмування Паскаль на факультеті електроніки ЛНУ імені Івана Франка став поштовхом для створення простої у користуванні оболонки на базі вільного програмного забезпечення.

Оболонка програмування Kuzya не прив’язана до однієї конкретної мови програмування чи до конкретного компілятора. Користувач може обирати одну з підтримуваних мов програмування для роботи за умови, що її компілятор встановлений у системі. Крім того, є можливість вибору компілятора, який буде використовуватись, якщо їх декілька. Це дає змогу працювати з різними компіляторами різних мов програмування, встановленими у системі. Розглянемо детально, як реалізовані ці можливості.

Взаємодія користувача та засобів розробки відбувається через інтерфейс середовища Kuzya. Середовище Kuzya забезпечує можливість компіляції та запуску програми, а також зворотний зв'язок (вивід результатів, повідомлень, помилок, тощо). Для цього Kuzya запускає компілятор як окремий процес з необхідними параметрами та налаштуваннями, виконує зчитування результатів його роботи та виводить ці результати (рис. 1).

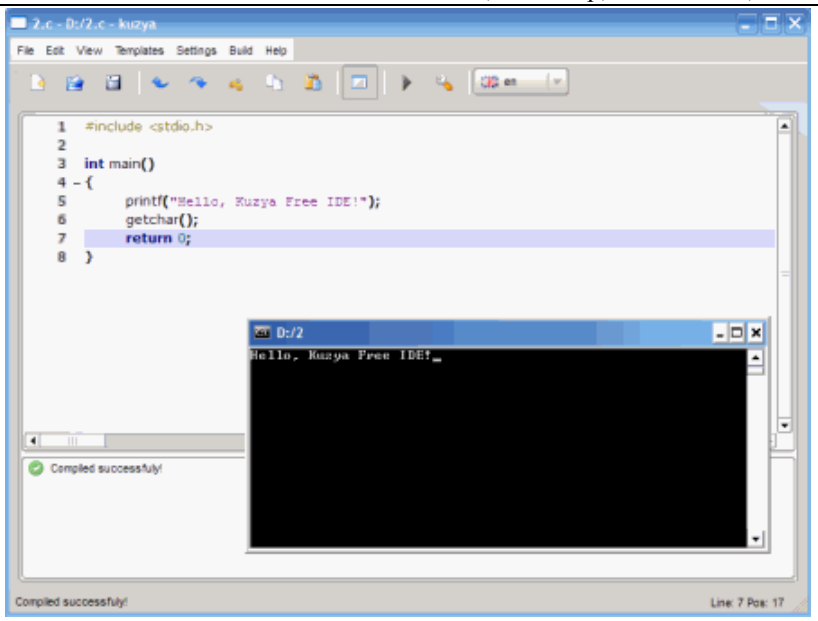

Рис. 1. Приклад компіляції та запуску написаної програми.

Оскільки правила роботи з окремими компіляторами відрізняються, то необхідно для кожного з них використовувати відповідні параметри командного рядка та дотримуватися визначеного порядку їх чергування. Тому для підтримки певного компілятора потрібно задати правила роботи з ним, тобто вигляд та порядок передавання параметрів.

Для вирішення цієї проблеми використовують спеціальні конфігураційні файли, у яких зберігаються всі налаштування та параметри командного рядка, призначені для роботи з конкретним компілятором. Кожний з цих конфігураційних файлів є INI-файлом. Налаштування всередині них зберігаються у зручному та зрозумілому вигляді і, за потреби, можуть бути замінені або скориговані.

Кожне поле у конфігураційному файлі має вигляд:

назва_параметра = значення_параметру,

де назва_параметра - назва деякої опції, а значення_параметра — текстовий рядок з параметрами або деяка інформація, необхідна для роботи середовища.

Очевидно, що такі налаштування можна доволі легко змінювати. Тому за потреби їх можна відредагувати у звичайному текстовому редакторі. Такий підхід було обрано для полегшення супроводу та обслуговування Kuzya.

Щоб додати підтримку нового компілятора, достатньо створити новий конфігураційний файл, який відображатиме особливості роботи з ним, та помістити його у теку з іншими конфігураційними файлами для заданої мови програмування. Оболонка автоматично знайде, розпізнає та приєднає цей конфігураційний файл. Це дає змогу уникнути зміни вихідних текстів програми та повторної їх компіляції, як було б у випадку, якби вони були би орієнтовані для роботи з конкретним компілятором.

Усі ці можливості реалізовані у класі Compiler, який завантажує налаштування для компіляції з конфігураційних файлів та, за потреби, змінює їх під час виконання. Він сканує наявні конфігураційні файли, приєднує їх та на цій основі створює списки підтримуваних мов і компіляторів. Також у ньому реалізований механізм взаємодії з програмою-компілятором. Такий зворотний зв'язок дає змогу відстежувати хід процесу та виводити результати на екран.

Для роботи з конкретним компілятором за допомогою класу Compiler запускають новий процес з необхідними параметрами (параметри у конфігураційному файлі, які відповідним способом обробляються та передаються) та організовують інтерфейс з ним через потоки уведення/виведення.

Крім того, клас Compiler дає змогу не тільки виконувати процес компіляції програми, а й запуск її на виконання і підтримує можливість читання результатів проведеної компіляції та передавання їх у середовище Kuzya. Це допомагає відстежувати помилки компіляції та виводити результати до відома користувача.

Розглянемо детальніше механізм компіляції. Для компіляції необхідно мати конфігураційний файл з опціями, які специфічні для використовуваного компілятора. Нижче наведено простий приклад профілю компілятора, який демонструє приклад роботи з компілятором GNU C++ Compiler:

```
[info]
compiler=g++
language=c,c++
comment=GNU project C and C++ compiler
[compile]
default=-O2 -o $output$ $options$ $source$
[errors]
error_messages_1=1:3:(\\d+):\\s(.*)error:\\s(.*)
```

Як бачимо, конфігураційний файл розбитий на декілька секцій.

1. Секція [info].

Опція	Призначення
compiler=g++	Задає ім’я компілятора, який буде запускатися в процесі компіляції (тут g++ (є ім'ям виконуваного файлу програми-компілятора).
language=c,c++	Інформація про мову програмування (використовують для інформування користувача про призначення компілятора).
comment=GNU project C and C++ compiler	Інформація про компілятор (використовується для інформування користувача про даний компілятор).

2. Секція [compile].

default=-O2 -o $output$ $options$ $source$ задає параметри командного рядка для компілятора у режимі за замовчуванням. Також є підтримка ще кількох режимів компіляції, що можуть бути обрані користувачем залежно від потреби та кінцевої мети компіляції (наприклад, режими для компіляції бібліотек або створення об'єктних файлів).

3. Секція [errors].

Ця секція містить шаблони для синтаксичного аналізу повідомлень компілятора про помилки. Шаблонів може бути необмежена кількість, імена довільні. Їх використовують для попереднього опрацювання повідомлень про помилки перед виведенням результатів компіляції. Також можна додати додаткові секції [warnings] та [msgs] для опрацювання попереджень та інформаційних повідомлень.

Перші дві секції є обов'язковими. Вони забезпечують програму інформацією, яка необхідна для підтримки того чи іншого компілятора. Спеціальні змінні, що виділені символами "$", замінюються на відповідні текстові рядки в процесі аналізу конфігураційного файлу. Вони роблять синтаксис конфігураційних файлів гнучким та дають змогу користувачу задавати додаткові опції без внесення змін у конфігураційний файл. Їх значення вказано нижче:

Змінна	Призначення
$source$	Повне ім'я файлу, який компілюється
$output$	Повне ім'я виконуваного файлу, який отримується внаслідок компіляції
$options$	Додаткові параметри
$compilerdir$	Місцезнаходження програми-компілятора

Для роботи необхідно завантажити конфігураційний файл, зазначивши мову програмування та назву компілятора. На підставі інформації про наявні конфігураційні файли буде знайдено та приєднано до оболонки потрібний компілятор. Якщо все зроблено правильно (конфігураційний файл існує, його формат коректний), то Kuzya виконує компіляцію та виводить її результати.

Всі необхідні для роботи конфігураційні файли зберігаються у теці /profiles. Для забезпечення їх автоматичного приєднання та правильної роботи оболонки вони повинні бути розміщені спеціальним чином. Для кожної з мов програмування передбачена окрема тека з відповідною назвою. У ній розміщені конфігураційні файли для підтримки різноманітних компіляторів, які мають розширення .prf , а також спеціальний конфігураційний файл підтримки мови програмування. Він також має кілька секцій. Обов'язкова секція [info] містить назву мови програмування та список розширень, які мають файли з програмним кодом цією мовою. На рис. 2 зображено вікно налаштування опцій компіляції.

У закладці Compiler розташовані випадаючі списки Programming language та Compiler, за допомогою яких можна вибрати компілятор для налаштування. Ці налаштування є окремими для кожного з компіляторів та зберігаються у загальному файлі з користувацькими опціями середовища програмування. Поле введення Compiler location дає змогу місцезнаходження цієї програми-компілятора. Поле Compiler options містить додаткові параметри, які будуть передані компілятору.

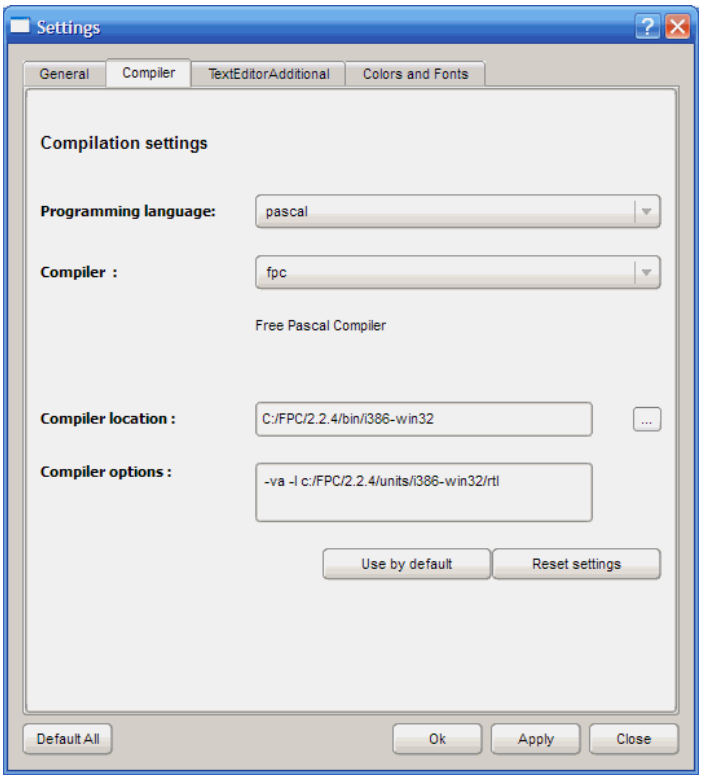

Рис. 2. Налаштування опцій компіляції.

Також передбачена можливість повернутися до налаштувань за замовчуванням, скориставшись кнопкою Reset settings. Скидаються налаштування тільки вибраного компілятора. Це дає змогу зберегти налаштування для інших компіляторів та уникнути повторного їх введення.

У процесі роботи використовується лише один конкретний компілятор, хоча і підтримуються додатково інші. Для вибору компілятора, з яким буде працювати користувач, слугує кнопка Use by default. Для цього треба лише вибрати компілятор зі списку та натиснути цю кнопку.

Процес роботи оболонки Kuzya відбувається так. Під час відкриття наявного або створення нового файлу за його розширенням визначається мова програмування використана для написання програмного коду всередині цього файлу. Далі визначається компілятор за замовчуванням, що буде використаний під час компіляції (на основі налаштувань користувача у розділі Compiler settings). На основі назви мови програмування та використовуваного компілятора приєднується відповідний конфігураційний файл.

Під час компіляції з нього зчитується рядок з параметрами, спеціальні змінні, визначення яких було наведено вище, замінюються відповідними значеннями (на підставі налаштувань користувача у розділі Compiler settings) або шляхами до файлів. Далі запускається програма-компілятор з цими параметрами та

встановлюється зв'язок з нею через потоки введення/виведення для взаємодії та отримання повідомлень у процесі компіляції. У разі успішного завершення цього процесу програма може бути запущена на виконання.

Якщо в процесі компіляції в програмі знайдено помилки або були виведені інформаційні повідомлення, то вони будуть повернуті головній програмі та показані після відповідної обробки користувачу (рис. 3).

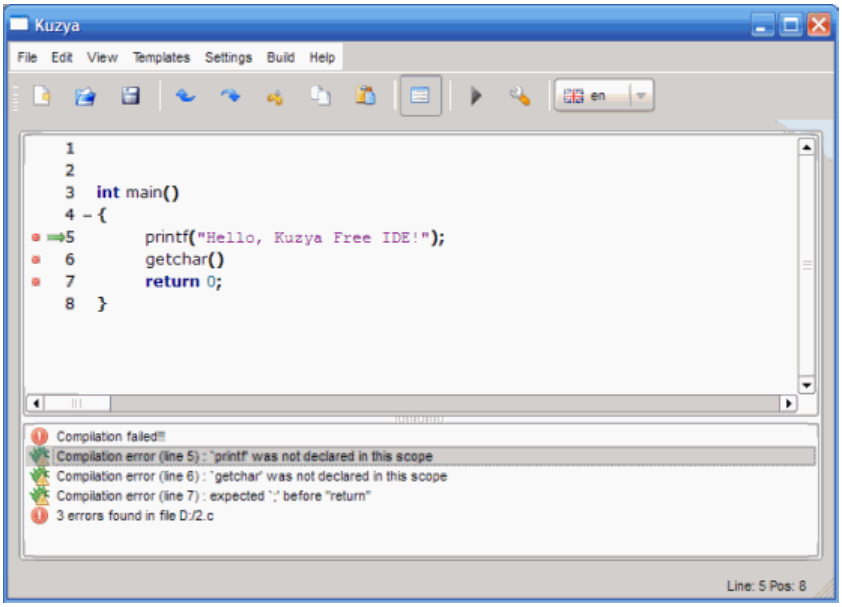

Рис. 3. Повідомлення про помилки

Для виведення цих повідомлень слугує спеціальний список, де відображені результати виконаних середовищем Kuzya операцій та інформаційні повідомлення. Також там відображаються помилки, попередження (з зазначенням рядка програми, де вони виникли) та інформаційні повідомлення компілятора. Підтримується зручна навігація по списку. Рядки програмного коду з помилками та попередженнями позначені маркерами. Якщо натиснути мишею на маркер, то одразу ж виділиться повідомлення у списку, що стосується цього рядка. Вибір повідомлення приводить до зазначення рядка, якого воно стосується з можливістю миттєвого переходу до нього та редагування. Це додає користувачу зручності у виправленні помилок та надає змогу швидко переміщатися по програмі.

Для навчання студентів, особливо коли вони тільки ознайомлюються з найпростішими методами побудови комп'ютерних програм та алгоритмів, є корисною можливість використання рідної мови для опису операторів та синтаксичних конструкцій. Це значно спрощує процес написання простих невеликих програм, робить їх зрозумілішими для студентів, дає змогу сконцентруватися на процесі алгоритмізації.

На відміну від деяких середовищ зі схожими можливостями середовище програмування Kuzya має більшу універсальність, адже дає змогу підтримувати

довільну мову перекладу для програм, а також застосувати ці можливості для будь-якої з підтримуваних мов.

Для цього передбачені спеціальні файли перекладу, у яких зберігається список ключових слів, операторів та їхній переклад. Для кожної з мов перекладу передбачений окремий файл, для кожної з мов програмування свій окремий набір файлів перекладу. Файли перекладу мають розширення .tr та розташовані разом з конфігураційними файлами компіляції у теці /profiles. Вони автоматично розпізнаються та приєднуються.

Під час відкриття наявного або створення нового файлу, якщо існують переклади для цієї мови програмування, середовище Kuzya автоматично визначає, чи були застосовані переклади до цього файлу.

Поточний переклад відображається у випадному списку на панелі інструментів. Також він дає змогу миттєво перекладати програму або перемикатися між наявними її перекладами. Для цього необхідно лише вибрати потрібний переклад зі списку, і він миттєво відкриється у вікні оболонки. Перекладені програми також можна компілювати та запускати на виконання. Процеси перекладу залишаються прозорими для кінцевого користувача, компіляція програми нічим не відрізняється від звичайної. На рис. 4 показано вікно середовища Kuzya із записом програми мовою Сі українською мовою.

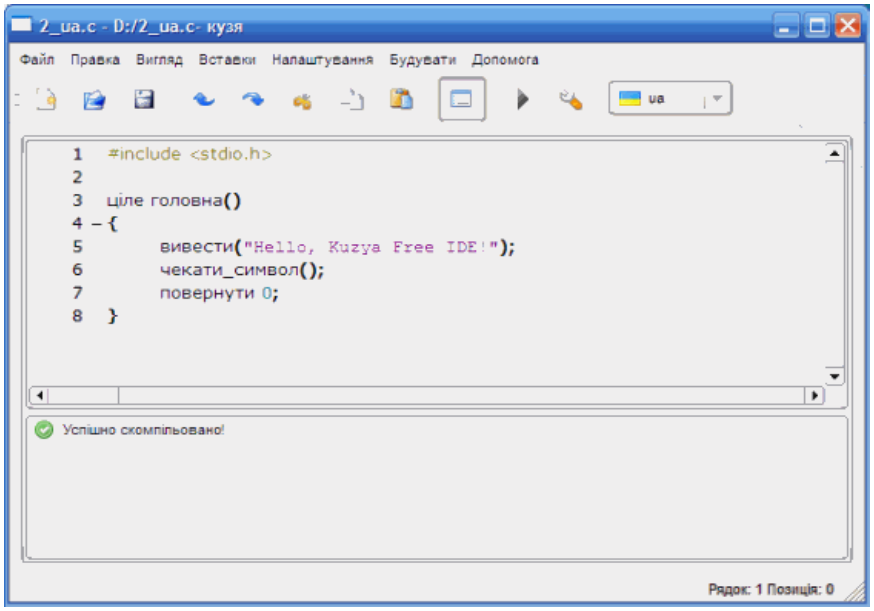

Рис. 4. Переклад програми мовою Сі українською мовою.

Для того, щоб реалізувати ці можливості у програмі, створено клас Translator. Він відповідає за роботу з файлами перекладів, за переклад програми та керування перекладами різними мовами. Під час компіляції він обробляє текст програми, замінюючи перекладені конструкції оригінальними, та створює ще один файл зі звичайним текстом програми. Саме цей файл передається компілятору і

компілюється, тому весь процес виглядає для користувача так, ніби він скомпілював перекладену програму.

Використання вільних засобів розробки програм для ОС Linux (наприклад Qt Creator) робить доволі простим процес інтернаціоналізії програми — на цей час оболонка Kuzya забезпечує діалог однією з трьох мов (англійською, українською та білоруською) за вибором користувача. На рис. 5 зображені фрагменти вікна оболонки Kuzya з різними мовами діалогу.

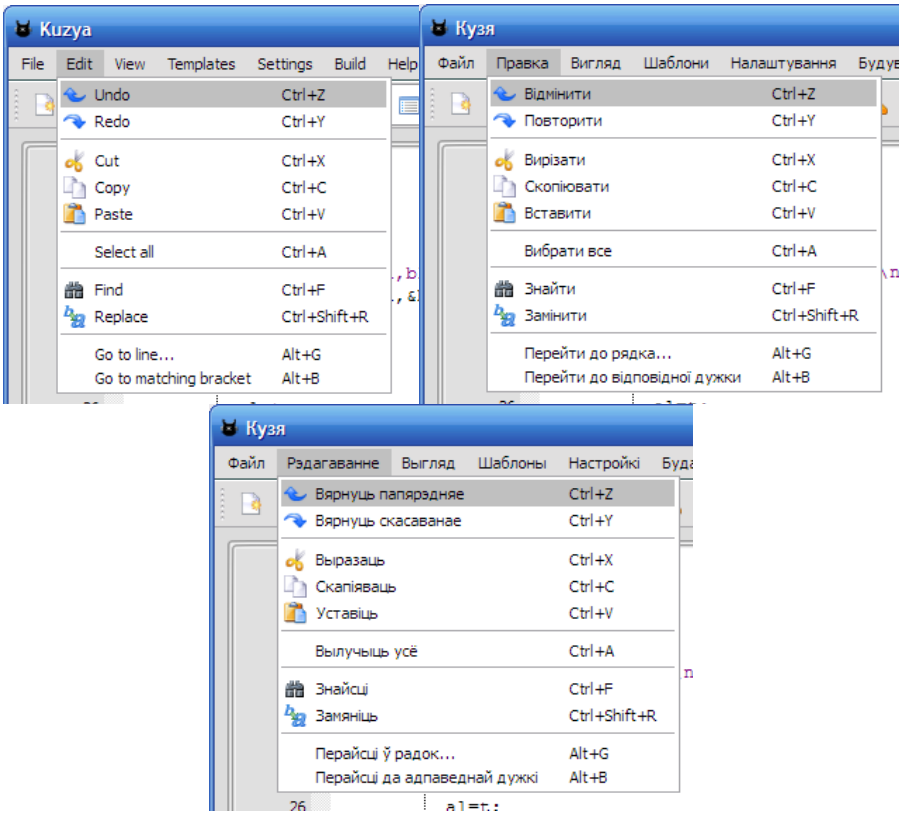

Рис. 5. Фрагменти вікна оболонки Kuzya з різними мовами діалогу

Для виконання графічних побудов написаний графічний двигун, який дає змогу відображати графічні дані у стилі Turbo C. Його основною перевагою є можливість підтримки будь-якої мови програмування. Це досягається завдяки тому, що Kuzya та графічний двигун Kuzyagraph, з'єднані звичайним потоком виведення. Всього реалізовано близько 30, на наш погляд, найбільш необхідних функцій. На рис. 6 зображено результат роботи програми з виведенням у графічне вікно.

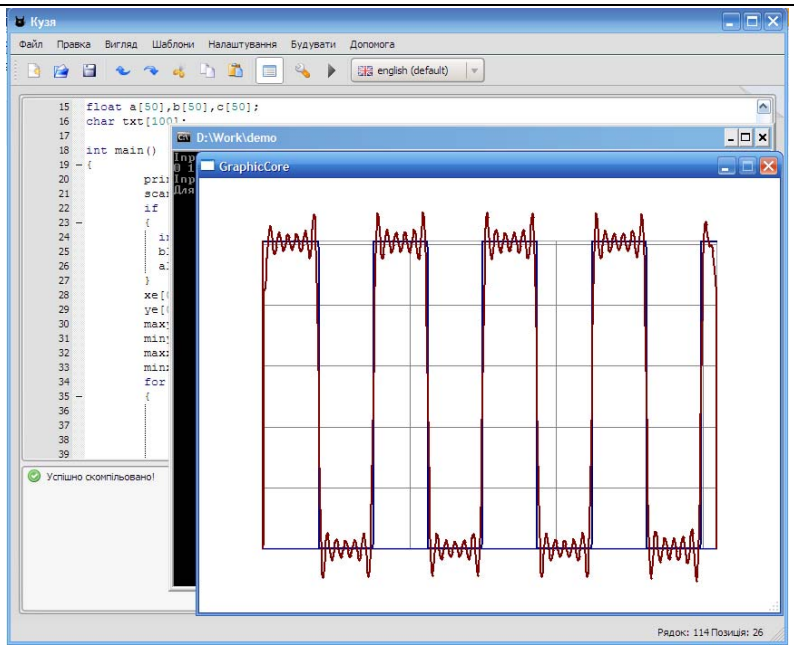

Рис. 6. Результат виконання програми з виведенням у графічне вікно

Середовище програмування Kuzya суміщає в собі функціональну простоту та універсальність. Безоплатність, відкритість програмного коду, кросплатформність та вільний вибір мови програмування і програмних засобів дають користувачам необхідну свободу. Тому Kuzya стає незамінним у навчальних цілях та у випадках, коли потрібне просте та надійне середовище для написання невеликих програм.

Робота середовища перевірена під час технологічної практики студентів другого курсу факультету електроніки влітку 2009 р. та у весняному семестрі 2009/2010 н.р. на лабораторних заняттях з курсу “Обчислювальна техніка і програмування” для студентів першого курсу факультету електроніки ЛНУ ім. Івана Франка.

USE OF FREE SOFTWARE IN ORDER TO CREATE TRAINING SHELL PROGRAMMING LANGUAGE FOR LEARNING

G. Zlobin, V. Skljar, A. Chmyhalo, V. Shevchyk

Ivan Franko National University of L’viv,
Tarnavsky Str. 107, UA–79017 Lviv, Ukraine
zlobin@electronics.wups.lviv.ua

We describe the experience of developing and using of an educational shell aimed at studies for the programming languages, which is based on the freeware.

Key words: programming, compiler, free software, Linux.

ОБ ИСПОЛЬЗОВАНИИ СВОБОДНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ С ЦЕЛЬЮ СОЗДАНИЯ УЧЕБНОЙ ОБОЛОЧКИ ДЛЯ ИЗУЧЕНИЯ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

Г. Злобин, В. Скляр, А. Чмыхало, В. Шевчык

Львовский национальный университет имени Ивана Франко
кафедра радиофизики
ул. Ген. Тарнавского, 107, 79017, Львов, Украина

Описан опыт разработки и использования учебной оболочки для изучения языков программирования на базе свободного программного обеспечения.

Ключевые слова: программирование, компилятор, свободное программное обеспечение, Linux.

Стаття надійшла до редколегії 12.04.2010
Прийнята до друку 26.05.2010