УДК 681.3.06

М.М. Колодницький, к.т.н., доц.
Житомирський інженерно-технологічний інститут

ПРОГРАМНА СИСТЕМА "DSR OPEN LAB 1.0": СУЧАСНИЙ ЗАСІБ АВТОМАТИЗАЦІЇ МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ

Наведено опис нової прикладної програмної системи, яка призначена для автоматизованого моделювання широкого класу математичних структур: "DSR Open Lab 1.0" (Dynamical systems research open laboratory – відкрита лабораторія дослідження динамічних систем). Показано її відмінність від існуючого програмного забезпечення такого класу, наведено приклади її ефективного використання у навчальному процесі та при наукових дослідженнях.

Автоматизація дослідження, проектування, створення та вивчення широкого класу систем (технічних, економічних, фізичних, біологічних, екологічних тощо) призводить до необхідності чисельного моделювання математичних об'єктів того чи іншого виду. Так, наприклад, при моделюванні динамічних систем виникають задачі аналізу математичних структур (моделей, об'єктів) різноманітного типу (наприклад, задача розв'язку систем алгебраїчних рівнянь, знаходження коренів поліномів тощо). Дослідження таких моделей має також самостійне значення як в математиці, так і в прикладних науках – до проблеми аналізу таких математичних структур (не динамічних систем) зводяться задачі моделювання об'єктів у різноманітних галузях.

На даний момент розроблено досить великий арсенал програмних засобів моделювання математичних задач, що надають користувачу суттєвий набір функціональностей [1–2]. Але все-таки, у *наукових дослідженнях* та в *освітянській діяльності* існують потреби у створенні нових програм моделювання і програм, які відповідають новим, всезростаючим вимогам до процесу моделювання.

Існуюче прикладне програмне забезпечення класу CAD/CAM/CAE/PDM-систем в основному орієнтоване на застосування його у *виробничій сфері* [3–4].

Окрім зазначеного класу існує також певний ряд прикладних програм, так би мовити, з широкою сферою застосування, які орієнтовані на їх використання в *наукових дослідженнях* та в *навчальному процесі*. Серед них, наприклад, різноманітні версії MATLAB (MathWorks Inc.), Mathematica® (Wolfram Research Inc.), Mathcad (MathSoft), Derive® (Soft Warehouse Inc.), Maple V (Waterloo Maple Inc.), PV-Wave (Visual Numerics Inc.), Macsyma® (Macsyma Inc.), O-Matrix (Harmonic Software Inc.), TK Solver (Universal Technical Systems Inc.), HiQ (National Instruments™), Scientist® for Windows (MicroMath Research®) тощо [1–2].

В той же час, використання для вищих навчальних закладів (ВНЗ), особливо, в Україні та країнах колишнього СРСР в навчальному процесі існуючих на сучасному ринку програмних засобів моделювання стримується рядом *причин* різноманітного характеру, таких, наприклад, як:

1) юридичні причини:
   - відсутність ліцензійних копій програм;
2) економічні причини:
   - велика вартість програм, особливо для випадку багатокористувацьких мережних ліцензій;
   - програма, що завоювала ринок збуту, не завжди є кращою з точки зору технічного чи наукового вирішення проблеми;
3) технічні причини:
   - різноманітність комп'ютерних платформ (відсутність багатьох із них у ВНЗ, зокрема, в Україні);
   - зміни та розвиток комп'ютерних платформ;
4) педагогічно-методичні причини:
   - пристосування того чи іншого пакета програм до певного навчального курсу є задачею нетривіальною;
   - необхідність неперервного розширення функціональних можливостей програм.

На проблеми, що виникають при використанні існуючих пакетів програм, останнім часом також зверталась увага в обговореннях на деяких наукових конференціях. Наприклад, була

висловлена думка, що "існуючі пакети є обмеженими в їх функціональності та орієнтації на користувача (user-friendliness)", вони є "не гнучкими та не інтерактивними" і "забезпечують лише один вид комунікації з користувачем" [5].

В даній роботі наведено опис нової прикладної програмної системи (ППС), призначеної для автоматизованого моделювання широкого класу математичних структур: "DSR Open Lab 1.0" (Dynamical systems research open laboratory – відкрита лабораторія дослідження динамічних систем). Показано її відмінність від існуючого програмного забезпечення такого класу, наводяться приклади її ефективного використання в навчальному процесі та в наукових дослідженнях.

Прикладна програмна система "DSR Open Lab 1.0" є продовженням робіт, що були започатковані в Житомирському інженерно-технологічному інституті більш ніж 10 років тому [6]. Спочатку це був пакет прикладних програм для розв'язування жорстких систем звичайних диференціальних рівнянь, а пізніше – для дослідження динамічних систем: "Dynamical systems research (DSR)" [7–9].

В процесі свого розвитку програмна система "DSR" поповнювалася новими функціональностями та вийшла за рамки задачі лише аналізу динамічних систем (тим більше звичайних диференціальних рівнянь) [10]. Згодом прийшло розуміння того, що для задоволення вимог до ППС [11], наприклад, вимоги *функціональної відкритості*, доцільно в ППС моделювання динамічних систем об'єднати можливості моделювання різноманітних математичних структур і таких прикладних задач, математичні моделі (ММ) яких входять в деяку множину математичних структур (МС). Це повинно забезпечити краще розуміння взаємозв'язку різних розділів математики, при використанні програми в освітянській діяльності.

Таким чином, у автора виникла ідея створити програмний комплекс, який за своїм функціональним призначенням міг би охоплювати певну множину математичних структур, був би, свого роду, лабораторією для досліджень різноманітних математичних моделей, причому "відкритою" лабораторією [11, 16]. Звідси випливає і назва ППС – "DSR Open Lab 1.0"; в назві нової програмної системи було вирішено залишити (з міркувань спадкоємності) початковий акронім DSR.

При побудові нової програмної системи було проведено аналіз існуючих і розроблено перелік вимог до нових систем. Зокрема, автором було звернуто увагу на те, що у випадку використання програмної системи у навчальному процесі, наприклад, при вивченні тих чи інших розділів математики (випадок розробки так званого математичного програмного забезпечення – "mathsoft"), головна вимога до математичного програмного забезпечення повинна формуватися з боку студента – головного користувача системи. Більш того, користувацький інтерфейс математичного програмного забезпечення повинен бути максимально простим для використання. Це означає, що програмна система повинна враховувати особливості тієї предметної області, яка в ній моделюється в даний момент, і в зв'язку з цим інтерфейс програмної системи повинен *адаптивно* (динамічно) змінюватися відповідним чином. У такому разі користувач буде мати змогу зосередитись саме на вирішені його задачі, а не на розв'язанні питання: "як використовувати програмний інструмент, для того щоб вирішити свою проблему?". В іншому випадку, надлишкова кількість ключових слів складної мови опису, різноманітних функцій, пунктів меню та кнопок може призвести до того, що користувач не знайде правильний шлях вирішення своєї прикладної проблеми. Більш того, математичне програмне забезпечення повинно використовувати інформацію про вид поточної задачі та на основі цього, допомагати роботі користувача, використовуючи певний вид сценарію його роботи. Такого типу архітектура інтерфейсу користувача широко використовується в інформаційних технологіях і відома під назвою "візарди (wizard)". Іншими словами, інтерфейс математичного програмного забезпечення повинен бути суттєво близьким (орієнтованим) до поточної задачі, як це буває у *спеціалізованих* програмних системах.

З іншого боку, програмна система повинна забезпечувати повну функціональність при роботі з даною задачею. Вона повинна мати досить великий набір алгоритмів, тобто повинна бути в певному розумінні *універсальною* системою. Якщо з програмною системою працює студент і вона має зручний інтерфейс та є повною функціональною системою, то він дедалі більше зацікавлюється як її вивченням, так і її використанням для вивчення математики – "апетит приходить під час їжі" – і тому бажає, щоб система була "універсальною". Якщо з програмною системою працюють викладачі та наукові дослідники, то вони очікують можливості використання різноманітних видів аналізу для вирішення тієї чи іншої складної задачі. Для них було би бажано, щоб бібліотека алгоритмів програмної системи поповнювалася новими

алгоритмами. Таким чином, "універсальна" програмна система має бути *відкритою* для подальшої модифікації.

Для вирішення протиріччя між спеціалізацією та універсальністю математичного програмного забезпечення автор запропонував дві основні ідеї:

- *типологію* (*модульність*) математичних моделей [12–14, 20];
- *відкритість* (*openness*) математичного програмного забезпечення.

Ідея типології математичних моделей є подальшим розвитком концепції модульності у застосуванні її в області математичного моделювання. Ця ідея грунтується на класичній концепції математичної структури.

Ідея відкритості не є новою у системному програмуванні, але і не є поширеною при розробці архітектури математичного програмного забезпечення. У будь-якому випадку, сучасне прикладне і, зокрема, математичне програмне забезпечення не використовують широко цю ідею.

Реалізація цих двох ідей в архітектурі програмної системи спричинила такі особливості нашого математичного програмного забезпечення, як:

- *3D-архітектура* математичного програмного забезпечення [10];
- *адаптивний інтерфейс користувача*, який *не* є таким, що повністю грунтується на мові опису (domain specific language) [16–19];
- *інструментальний характер* математичного програмного забезпечення [21].

Отже, програмна система "DSR Open Lab 1.0" за функціональним призначенням має охоплювати певну множину математичних моделей, а саме:

- скінченні множини;
- числові системи (від натуральних до комплексних чисел);
- функціональні відношення (числові функції на відповідних числових множинах);
- алгебраїчні системи з однією бінарною операцією – групи;
- алгебраїчні системи з двома бінарними операціями – поліноми, раціональні дроби;
- векторний простір (задачі матричної алгебри, системи лінійних алгебраїчних рівнянь тощо);
- системи нелінійних алгебраїчних рівнянь, векторні поля;
- тензори (тензорна алгебра та аналіз);
- дискретні логічні моделі (булеві функції, автомати, рекурентні булеві функції);
- імовірнісні моделі (випадкові числа, марковські ланцюги, випадкові процеси);
- динамічні системи (системи звичайних диференціальних рівнянь);
- рівняння в часткових похідних;
- оптимізаційні моделі (лінійне та нелінійне програмування, динамічне програмування тощо).

Можна сказати, що цей перелік утворює, свого роду, базис математичних моделей. З набору базових математичних структур за допомогою їх поєднання, комбінації, нарощування тощо можуть бути побудовані більш складні структури. Тут можна провести аналогію з мозаїкою, в якій із базових елементів (свого роду "кубиків") складається як завгодно складна n-вимірна структура. Така особливість запропонованої систематизації ("мозаїки") ММ відрізняє її від загальноприйнятих класифікацій, які, в основному, майже завжди зводяться до строгої ієрархії (а не "мозаїки") моделей.

З кожною математичною структурою (математичною моделлю) пов'язується набір обчислювальних задач, зміст якого, фактично, визначається змістом (структурою) математичної структури. (У відомих літературних класифікаціях ММ також цього не спостерігаємо). Структура змісту обчислювальних задач подана в [12–14] у такому вигляді, що можна помітити аналогію між окремими задачами для різних математичних структур, наприклад, між задачами визначення "підмножини" для МС "Скінченні множини" та "простих чисел" для МС "Числові системи", "зведених поліномів" для МС "Поліноми", "базису" для МС "Векторний простір" тощо. Виявлення такої аналогії, на думку автора, може бути корисним не тільки для більш глибокого розуміння теорії певної математичної структури, але й для розробки відповідних чисельних алгоритмів.

У програмній системі "DSR Open Lab 1.0" є можливість оперувати моделями, які утворені довільною комбінацією та нарощенням базових моделей.

Можна провести також певну аналогію між покладеними в основу запропонованої класифікації ("мозаїки") об'єктами (математичними структурами та обчислювальними задачами, "кубиками") та поняттям "класу об'єктів", який використовується у сучасній технології об'єктно-орієнтованого програмування (ООП). Як відомо, однією з особливих рис

ООП є можливість, так би мовити, будувати з готових "кубиків" – програмних об'єктів – складні "мозаїки": об'єктно-орієнтовані програмні комплекси. Причому, характерним є те, що класи об'єктів є поєднанням структур даних та задач їх обробки, тобто саме те, що покладено нами в основу класифікації ММ систем – математичних структур та відповідних їм обчислювальних задач. Автор назвав такий підхід до систематизації математичних моделей *типологією* математичних моделей [12]:

тип ММ = математична структура + ієрархічний набір обчислювальних задач.

Можливість представлення запропонованої типології у вигляді набору "класів об'єктів ООП" показує, що остання має не тільки теоретичне значення, але й практичну цінність – вона дозволяє використати сучасну технологію об'єктно-орієнтованого програмування для реалізації програмної системи моделювання – інструмента дослідження математичних моделей систем.

Для того щоб реалізувати концепцію типології (модульності) математичних моделей та ідею відкритості і розробити архітектуру програмної системи, розробники повинні задовольнити ще ряд вимог – так званих "вимог реалізації" [11]. Аналіз постановки задачі показує, що архітектура запропонованого пакета програм виявляється досить складною. Її умовно можна представити як тривимірний кубик, де осі x, y, z означають:

x – залежність від виду функціональності пакета (від виду прикладної задачі);

y – залежність від комп'ютерної платформи;

z – залежність модулів пакета один від одного.

Інакше кажучи, вісь x відображає ієрархію функціональних модулів, вісь y вказує на прошарки залежності від комп'ютерної платформи, тобто відображає можливість міжплатформного перенесення пакета (portability), і, нарешті, вісь z вказує на клієнт-серверні властивості модулів пакета (чи, в загальному випадку – на властивості багатоярусності програми (multi-tier application), тобто на автономність програмних модулів (незалежність програмних модулів) та програмний інтерфейс (взаємозв'язок) між ними. Важливим є те, що на перетині трьох координат x, y, z ми отримуємо певні компоненти програмної системи. Це дозволяє нам назвати таку архітектуру програмної системи *тривимірною компонентною архітектурою*.

Наступною особливістю нашої програмної системи є *адаптивний інтерфейс користувача*.

Роботу користувача в програмній системі можна представити як послідовність кроків. Кожен крок визначає певний етап в описі задачі за допомогою візуального (графічного) інтерфейсу [16]:

1) користувач вибирає тип математичної моделі (математичної структури);
2) користувач вибирає вид аналізу для даної математичної моделі;
3) користувач вводить математичні вирази, що описують модель для даного виду аналізу;
4) користувач вводить додаткові параметри даного опису (якщо вони мають бути в описі даної задачі);
5) користувач вибирає види вихідних результатів даного аналізу;
6) користувач вибирає метод розв'язання задачі (у разі необхідності);
7) програмна система виконує обчислення та формує результати у заданому вигляді;
8) користувач переглядає результати розрахунків для даного виду аналізу.

Сценарій роботи користувача в даній програмній системі має послідовний, покроковий ("step-by-step") характер, схожий на "візарди" (wizards). Це відрізняє дану програмну систему від подібних відомих пакетів. (Див. приклади екранних форм у додатку). Таким чином, у даному випадку інтерфейс користувача не є таким, що повністю базується на використанні деякої мови опису (свого роду мови програмування), як це має місце у інших пакетах [1]. Опис задачі поділяється на декілька стадій, і лише на одній з них використовується *мова* опису моделі, причому, ця мова є повністю декларативною. Її оператори описують математичні вирази (що входять до складу моделі, яка досліджується) і не використовують ніяких мовних конструкцій для управління обчисленнями чи вибором виду представлення результатів розрахунків [17]. Такі обмеження дозволяють нам суттєво спростити синтаксис мови опису.

При роботі з різними типами моделей наша програмна система функціонує подібно. Після того як користувач вибере новий тип математичної структури, в інтерфейсі програмної системи автоматично відбуваються певні настройки, заповнюються деякі його компоненти та змінюються певні етапи опису моделі. Причому основний каркас інтерфейсу системи суттєво не змінюється. Таким чином, інтерфейс програмної системи, так би мовити, підстроюється під поточний тип математичної моделі. Це надає підгрунтя для того, щоб назвати такий інтерфейс *адаптивним*.

Для опису математичної моделі користувач використовує певну мову. Так само як і інтерфейс користувача є адаптивним, так і *мова опису* є також *адаптивною* в нашій програмній

системі. Її синтаксис залежить від поточного типу моделі, тобто лише певна підмножина мови використовується для опису певного типу математичної структури. Таким чином, множина елементів мови (таких як ключові слова, оператори, константи тощо) є суттєво скороченою ("мінімальною") для опису кожного окремого типу моделі.

Для того, щоб ще більше спростити використання операторів мови опису в інтерфейсі користувача, надано спеціальний інструмент для набору математичних виразів ("equation builder"). Цей засіб "прив'язаний" до синтаксису мови, і, таким чином, має також *адаптивну* структуру. Це означає, що для користувача відображається лише мінімальний набір кнопок на панелі (toolbar). Використовуючи інформацію про поточний тип моделі, "equation builder" перебудовує свій інтерфейс динамічно. Таким чином, користувачеві не потрібно пам'ятати весь список функцій та синтаксис речень мови опису кожної моделі.

Для того, щоб реалізувати адаптивний інтерфейс із використанням "майстрів (wizard)", тобто послідовності візуального опису задачі моделювання, ми сконструювали інтерфейс нашої програмної системи таким чином: каркас (frame), навігатор видів аналізу (left navigator), "майстер" послідовного опису задачі (right navigator). Каркас є Active-Document контейнером, (використовуючи термінологію Microsoft Active Technology), навігатор видів аналізу реалізовано як деяку панель (bar), "майстер" – як набір різних серверів (EXE та DLL-modules). Контейнер та всі сервери використовують OLE Automation. У програмній системі широко використовуються також засоби multimedia. Всі дані та структури графічного інтерфейсу програмної системи зберігаються в спеціальній базі даних, яка реалізована з використанням MS Visual FoxPro 6.0. Вся програмна система "DSR Open Lab 1.0" була розроблена за допомогою MS Visual C++ 6.0.

На базі представленого математичного програмного забезпечення було розроблено ряд комп'ютерних практикумів із різних розділів математики, таких як теорія чисел, теорія груп, алгебра, теорія апроксимації функцій [22–25]. Було показано, що дана програмна система є ефективним засобом автоматизації математичного моделювання.

ЛІТЕРАТУРА:

1. *Колодницкий Н.М., Левицкий В.Г.* Обзор основных программных средств для моделирования математических задач // САПР и графика, 1999. – № 10 – С. 56–65.
2. *Колодницкий Н.М., Левицкий В.Г.* Программные средства моделирования систем автоматического управления: проблема выбора // Праці 6-ї Української конференції з автоматичного управління "Автоматика–99", 10–13 травня 1999 р. – Харків, 1999. (в друці).
3. *Колодницький М.М., Чайковський С.С.* Огляд інтегрованих систем автоматизованого проектування для машинобудування. Частина 1. // Вісник ЖІТІ. – 1998. – № 7. – С. 219–229.
4. *Колодницький М.М., Чайковський С.С.* Огляд інтегрованих систем автоматизованого проектування для машинобудування. Частина 2. // Вісник ЖІТІ. – 1998. – № 8. – С. 181–190.
5. Chung-pong Lau, Tat-chung Chau, Tat-yung Poon, Wai-kin Tsui and E. Herbert Li. Interactive and Multimedia Learning in Mathematics. *Proceeding of ATCM'99.* – P. 301–310.
6. *Колодницкий Н.М.* Проблемно-адаптивная организация численного анализа электронных схем // Автореферат на соиск. уч. степ. к.т.н. – Киев: КПИ, 1989. – 18 с.
7. *Колодницкий Н.М., Салитринский В.Е., Назаренко В.С.* ОДЕ/РС – интегрированный пакет программ для решения жестких систем обыкновенных дифференциальных уравнений и исследования свойств численных алгоритмов // Применение вычислительной техники и математических методов в научных и экономических исследованиях: Тезисы докладов. – Киев: КПИ, 1991. – С. 93.
8. *Колодницкий Н.М., Салитринский В.Е.* Пакет программ для изучения динамических систем и методов их анализа // Труды 1-й Украинской научно-методической конференции "Новые информационные технологии в учебных заведениях Украины". – Одесса, 1992. – С. 180–181.
9. *Колодницкий Н.М.* Пакет программ "Dynamical systems research (DSR)" // Праці Житомирського філіалу КПІ. Серія А. Техніка. Вип. 1.– Житомир: ЖФ КПІ, 1993. – С. 81–94.

10. *Колодницький М.М., Рожик О.А., Левицький В.Г., Шкаленко С.В., Гладишев А.В.* Програмний комплекс для моделювання динамічних систем "DSR LAB. 1.0" // Сучасні технології в аерокосмічному комплексі. Матеріали III Міжнародної наук.-практ. конференції, 9–11 вересня 1997 р. – Житомир: ЖІТІ, 1997.– С. 97–99.
11. *Колодницький М.М.* Тривимірна компонентна архітектура прикладної програмної системи "DSR Open Lab 1.0" як втілення концепцій реінженерії // Проблемы программирования. – 1998. – № 4. – С. 37–45.
12. *Колодницький М.М.* Типологія математичних моделей технічних систем. Частина 1 // Вісник ЖІТІ. – 1997.– № 6. – С. 130–142.
13. *Колодницький М.М.* Типологія математичних моделей технічних систем. Частина 2 // Вісник ЖІТІ. – 1998. – № 7. – С. 208–218.
14. *M. Kolodnytsky.* The principles of theory of mathematical modelling of systems. *Zhitomir. ZIET*, 2000. (in press).
15. *Kolodnytsky M., Ivanitsky I., Kovalchuk A.,. Kuryata S., Levitsky V.* "DSR Open Lab 1.0" – software system for simulation. *Proceedings of 21st International Conference on Information Technology Interfaces*, Pula, Croatia, 1999. – p. 45.
16. *Колодницький М.М., Ковальчук А.М., Кур'ята С.В.* Типологія архітектури інтерфейсу користувача прикладної програмної системи "DSR Open Lab 1.0". Частина I. Компонентна реалізація // Проблемы программирования. – 1999. – Вып. 3–4 .
17. *Колодницький М.М., Левицький В.Г.* Типологія архітектури інтерфейсу користувача прикладної програмної системи "DSR Open Lab 1.0". Частина II. Лінгвістичне забезпечення // Проблемы программирования. – 2000. – Вып. 3–4.
18. *Колодницький М.М., Левицький В.Г.* Адаптивна організація лінгвістичного забезпечення програмного комплексу "DSR Open Lab 1.0" // Праці 1-ї міжн. наук.-практ. конф. з програмування "УкрПрог'98", 2–4 вересня 1998 р. – Київ, 1998. – С. 145–155.
19. *Колодницкий Н.М., Левицкий В.Г.* Применение формализма порождающих грамматик к анализу родственных языков // Кибернетика и системный анализ. – 2000.
20. *Kolodnytsky M., Kovalchuk A., Kuryata S., Levitsky V.* Software system for mathematics teaching: the approach based on mathematical models typology // Proceedings of the 4th Asian Technology Conference in Mathematics, December 17–21, Guangzhou, China, 1999. – http://www.atcminc.com/ mConferences/ATCM99/
21. *Колодницький М.М., Левицький В.Г.* Сучасні інструментальні засоби автоматизованої побудови мовних процесорів // Проблемы программирования, спец. выпуск. – №1–2. – Материалы 2-й Международной научн.-практ. конф. по программированию "УкрПрог'2000", 23–26 травня 2000 р. – Київ, 2000. – С. 345–350.
22. *Колодницький М.М., Левицький В.Г.* Комп'ютерний лабораторний практикум з обчислювальних задач теорії дискретних груп // Сучасні технології в аерокосмічному комплексі. Матеріали IV Міжнародної наук.-практ. конференції, 7–9 вересня 1999 р. Житомир: ЖІТІ, 1999. – С. 45–48.
23. *Kolodnytsky M., Levitsky V.* The computer laboratory practice for the computing tasks of the group theory // Proceedings of the 4th Asian Technology Conference in Mathematics, December, 17–21, Guangzhou, China, 1999. – http://www.atcminc.com/mConferences/ATCM99/
24. *Колодницький М.М., Левицький В.Г.* Комп'ютерний лабораторний практикум з обчислювальних задач теорії дискретних груп: результати та перспективи // Управляющие системы и машины. – 2000 (в друці).
25. *Kolodnytsky M., Kovalchuk A., Kuryata S., Levitsky V.* The Mathematical Software Implementation for Computational Algebra and Number Theory // Proceedings of the 4th Asian Symposium on Computer Mathematics, December 17–21, Chiang Mai, Thailand, 2000. – (in press).

КОЛОДНИЦЬКИЙ Микола Михайлович – кандидат технічних наук, доцент кафедри ПЗОТ Житомирського інженерно-технологічного інституту.

Наукові інтереси:

- математичне моделювання технічних систем;
- комп'ютерні інформаційні технології.

Подано 27.09.2000