

Віктор РОМАНЕНКО, Д-р фіз.-мат. наук, Пров. наук. співроб.

ORCID ID: 0000-0001-8554-0214

e-mail: victor.romanenko@gmail.com

Інститут фізики НАН України, Київ, Україна

Наталя КОРНІЛОВСЬКА, Канд. тех. наук, Доц.

ORCID ID: 0000-0002-8331-8027

e-mail: kornilovska.natalya@kntu.net.ua

Херсонський національний технічний університет, Хмельницький, Україна

## REDUCE – КЛАСИКА КОМП'ЮТЕРНОЇ АЛГЕБРИ

**Анотація.** Стаття містить короткий огляд системи комп'ютерної алгебри REDUCE. Незважаючи на значний вік, REDUCE досі активно використовується дослідниками. Метою роботи є надання базових знань про REDUCE, які допоможуть читачеві розпочати роботу з цією системою. Більшість наведених прикладів стосуються математики; їх добір зумовлений прагненням продемонструвати максимально широкий спектр базових можливостей системи в межах короткої публікації.

**Ключові слова:** комп'ютерна алгебра; математичний аналіз; елементарні функції.

### 1. Вступ

Сьогодні практично неможливо виконати науково-дослідницьку чи проєктно-конструкторську роботу без застосування комп'ютерних обчислень чи комп'ютерної обробки отриманих при дослідженнях результатів. Серед сукупності прикладних програм, які при цьому використовуються, є і системи комп'ютерної алгебри (СКА, CAS – Computer Algebra Systems), які значно спрощують аналітичні розрахунки і роблять їх надійнішими. Особливо це важливо у теоретичній фізиці, де раніше подібні розрахунки могли тривати кілька днів без гарантії надійності результату. Водночас ці системи можуть бути корисними і в більш буденних сферах застосування, наприклад, вчителям математики при перевірці правильності наведених у посібниках прикладів або підготовці нових завдань для уроку чи контрольної роботи. Зрозуміло, що зростаючий попит на такі системи призвів до їхньої комерціалізації, і зараз на ринку значне і почесне місце займають такі продукти, як Maple і Mathematica. Навіть у добре відомому середовищі технічних обчислень MATLAB можна виконувати аналітичні розрахунки. Крім комерційних алгебраїчних систем, існують і безкоштовні – REDUCE, Maxima, Axiom. Перша з них створювалася від початку як засіб для розрахунків у теоретичній фізиці, тоді як Maxima і Axiom мають більш універсальне призначення і використовуються як у наукових дослідженнях, так і в навчанні.

Першою великою системою, яка вже вміла працювати з поліномами, виконувати інтегрування та диференціювання, була FORMAC (MacCallum, 2018). Її розробка почалася у 1962 році і була завершена у 1964 році. Ця система базувалася на мові

програмування FORTRAN IV. Наступна СКА, REDUCE, базувалася вже на мові LISP. Власне, вона виникла в результаті спілкування Ентоні Херна (Anthony Hearn) з Джоном Маккарті (John McCarthy), фахівцем з комп'ютерних наук в Стенфордському університеті. Е. Херн прагнув використати комп'ютери для обчислень у квантовій електродинаміці на основі фейнманівських діаграм, і Дж. Маккарті переконав його, що для цього добре підходить мова LISP. Так у 1963 році почалася робота над REDUCE.

Досить швидко Е. Херн зрозумів, що розроблена ним програма може становити загальний інтерес для математичних обчислень, а не бути лише засобом вузької спеціалізації. Опис системи REDUCE було опубліковано у 1968 році (Hearn, 1968). Перша версія для широкого загалу, REDUCE 2, створена однією людиною, з'явилася у 1970 році. Згодом до розробки REDUCE приєдналися інші фахівці та користувачі, що дозволило вже у 1983 році створити дуже продуктивну і добре знайому користувачам версію REDUCE 3 (Hearn, 2005). Загальна кількість спеціалістів, які брали участь у її створенні та розвитку, становила понад сто осіб.

Як правило, великими літерами записують акроніми. REDUCE не є акронімом. Таке написання вибране Е. Херном і частково має жартівливий характер – це натяк на те, що система REDUCE робить вирази простішими (Hearn, 2005). Стосовно REDUCE варто також зазначити, що від самого початку ця СКА була орієнтована на економію ресурсів комп'ютера (Rayna, 1987). Ще однією особливістю REDUCE є модульність – можливість розширення функціональності системи шляхом додавання модулів без зміни її ядра. Крім того, від самого початку REDUCE створювалася як інтерактивна система, що на той час було менш поширеною практикою, ніж нині. За словами Е. Херна, який написав передмову до книги Дж. Райна (Rayna, 1987), REDUCE застосовують у квантовій електродинаміці та квантовій хромодинаміці, небесній механіці, механіці рідин, загальній теорії відносності, чисельному аналізі, фізиці плазми, а також у різноманітних інженерних задачах, таких як аналіз електричних мереж і проектування турбін та корпусів кораблів. Така зрілість системи дає користувачеві впевненість у надійності та зручності використання REDUCE.

Окрім REDUCE, із систем старшого покоління досить успішною виявилася також MACSYMA, робота над якою тривала з 1968 до 1969 року (MacCallum, 2018). Її назва є акронімом від Project MAC SYmbolic MANipulator – проекту Massachusetts Institute of Technology. Зараз ця система існує під назвою Maxima і, як і REDUCE, базується на LISP (діалекти різні; REDUCE – на Standard LISP) та є безкоштовним програмним продуктом. Обидві системи – класика комп'ютерної алгебри. Вони не лише заклали основу для появи комерційних гігантів Maple і Mathematica (компаній Maplesoft та Wolfram Research відповідно), але й досі залишаються актуальними, зокрема як безкоштовні інструменти.

Крім Maxima і REDUCE, до безкоштовних СКА-ветеранів належить також Axiom, розробка якої, як зазначає Axiom Development Team (n.d.), почалася у 1971 році. Не заглиблюючись у деталі порівняння цих трьох систем, зазначимо, що REDUCE від самого початку орієнтувався на фахівців у галузі теоретичної фізики, тоді як Axiom більшою мірою орієнтована на фахівців у галузі математики.

Далі ми розглянемо базові можливості REDUCE і покажемо, як можна швидко почати працювати з цією системою. Зрозуміло, що в короткій статті неможливо дати скільки-небудь повний опис мови REDUCE; наша мета – дати читачеві можливість використовувати цю СКА для простих обчислень. Повний опис REDUCE подано в REDUCE User's Manual (Hearn та Schöpfung, 2025). Цей посібник користувача можна знайти в розділі Documentation на сайті розробника: <https://reduce-algebra.sourceforge.io/>. Крім того, там також наведено список книг з REDUCE та навчальні матеріали (Tutorials).

## 2. Базові поняття

Почнемо з базових понять, знання яких необхідне для коректного запису команд REDUCE та розуміння результатів їх виконання.

Записуючи команди та ідентифікатори REDUCE, перш за все треба зважати на **допустимий набір символів** цієї мови. Це літери англійської абетки (великі і малі літери REDUCE не розрізняє), цифри від 0 до 9 та спеціальні символи: `_ ! " $ % ' ( ) * + , - . / : ; < > = { } і пробіл`.

Числа в REDUCE можуть бути цілими або раціональними (відношенням двох цілих). За необхідності можна змусити REDUCE працювати з наближеними значеннями чисел у форматі з плаваючою точкою (з будь-якою заданою точністю). Такі числа називають дійсними. Приклад запису одного і того ж числа у різних формах: 128, 128.0, 0.128E3, 12.8E1. Положення **десяткової точки** можна регулювати. Не можна писати .5, правильно 0.5.

Будь-який **ідентифікатор** має починатися з літери і може містити цифри, а також символ підкреслення (але не на початку). Наприклад, `this_is_parameter123`, але не `_this_is_parameter`.

Послідовність символів з букв і цифр, яка починається з цифри, REDUCE інтерпретує як добуток. Наприклад, `32a2c` як  $32 \times a2c$ . Виняток – якщо після цифри йде `e`: `32e2c` інтерпретується як  $3200 \times c$ , `32ec` інтерпретується як  $32 \times c$  (якщо немає цифри після `e`, інтерпретується як `e0`). Щоб уникнути несподіванок, краще пишіть **операцію множення** у явному вигляді.

Ідентифікатори слугують для позначення змінних, міток, масивів, операторів і процедур. Не слід використовувати зарезервовані ідентифікатори – їхній список наведено у посібнику користувача (Hearn, Schöpfung, 2025).

Якщо **змінній** не надано якоесь значення (числове чи алгебраїчний вираз), її значення дорівнює їй самій.

**Рядок** – це будь-яка кількість символів, взята у подвійні лапки. Якщо подвійні лапки повинні бути частиною рядка, їх набирають двічі.

Для розуміння програми, написаної на REDUCE, пишуться **коментарі**. Коментарі можна записувати так: 1) все, що написано від слова COMMENT (нагадаємо, малі і великі літери REDUCE не розрізняє) до символу `;` (крапка з комою) або `$ REDUCE` сприймає як коментар; 2) частина рядка після знаку `%` теж сприймається як коментар; 3) все, що записано між `/*` і `*/` – коментар.

Оператори в REDUCE можуть бути двох типів – префіксні та інфіксні. **Префіксні оператори** мають аргументи в дужках, розділені комами, якщо їх два або більше. Приклади:  $\sin(x)$  – обчислення синуса  $x$ ,  $df(\cos(x),x)$  – обчислення похідної від косинуса. В деяких випадках дужки можна не писати, але краще цього уникати.

Низку **інфіксних операторів** визначено системою. Серед них:

- арифметичні: + (додавання), - (віднімання), \* (множення), / (ділення), \*\* та ^ (обчислення степеня);
- оператори відношення: = (дорівнює для математичних рівнянь), > (більше), >= (більше або дорівнює), < (менше), <= (менше або дорівнює), eq (дорівнює для логічних перевірок), neq (не дорівнює);
- логічні оператори or (або), and (і) та деякі інші.

Оператор := слугує для присвоєння значення змінній. Символ ; (крапка з комою) в кінці виразу повідомляє REDUCE про необхідність вивести результат на дисплей. Якщо ж в кінці виразу стоїть \$ (результат лише обчислюється і не виводиться).

### 3. Швидкий старт

Відійдемо від систематичного викладу мови REDUCE, щоб показати низку простих, але поширених на практиці операцій, які можна виконувати за допомогою REDUCE. Викладених базових знань про мову програмування REDUCE загалом достатньо для розуміння цих операцій; пояснення нових конструкцій наводитимуться у відповідних місцях.

#### 3.1. Графічний інтерфейс

На сайті розробника REDUCE <https://reduce-algebra.sourceforge.io/> у розділі **Obtain and Run** (меню ліворуч) наведено детальні інструкції щодо встановлення пакета на комп'ютери з операційними системами Microsoft Windows, Linux та macOS.

Для завантаження дистрибутива можна скористатися останнім пунктом меню – **Download REDUCE from SOURCEFORGE**. Крім того, передбачена можливість роботи з онлайн-версією системи: у нижній частині меню слід обрати пункт **Web REDUCE**.

Онлайн-версія має експериментальний характер: не всі функціональні можливості REDUCE у ній реалізовані. Водночас вона є достатньою для ознайомлення з основними принципами роботи системи. У разі використанні мобільних пристроїв передбачено відповідний режим доступу.

Надалі припускати, що користувач працює саме з вебверсією REDUCE.

#### 3.2. Перші спроби

Відкриваємо вебсторінку REDUCE. У віконці внизу (**Input Editor**) записуємо команди і скеровуємо їх на виконання (**Send Input**). Можна також за допомогою нижнього меню переходити між раніше введеними командами, редагувати їх і знову виконувати. Меню вгорі дозволяє полегшити введення команд. Його призначення

інтуїтивно зрозуміле. Ми не будемо ним користуватися: робота з меню може бути корисною для невеликих інтерактивних завдань, тоді як для розв'язування задач у більшості випадків створюють програму у файлі, вміст якого (можливо, частинами) потім можна спрямувати на виконання через вікно **Input Editor**.

Вводимо

$(a+b)^2$ ;

Натискаємо **Send Input**. Отримуємо:

$$a^2 + 2ab + b^2.$$

Як бачимо, REDUCE розкрив дужки. Результат обчислень зберігається у робочій змінній *ws* (work space). Присвоїмо його змінній *w*:

$w := ws$ ;

Отримаємо

$$w := a^2 + 2ab + b^2.$$

Виконання операцій множення та піднесення до степеня в алгебраїчних виразах залежить від положення перемикача *exp*. За замовчуванням він перебуває в положенні *on* (у наведеному вище прикладі обчислюється квадрат суми змінних *a* і *b*). Переведемо цей перемикач у положення *off* і подивимося, як виглядає *w*:

*off exp*;

*w*;

Отримаємо

$$a^2 + 2ab + b^2.$$

Це природно, адже саме в такому вигляді ми присвоїли змінній *w* її значення операцією  $w := ws$ ; . Але якщо тепер виконати аналогічну операцію для четвертого степеня суми *a* і *b*, дужки не будуть розкриті:

$u := (a+b)^4$ ;

Отримаємо

$$u := (a + b)^4.$$

Як подати вираз для  $w$  в компактному вигляді, як під час початкового введення? Для цього слугує перемикач `factor`, який за замовчуванням вимкнений (`off`, `REDUCE` не виконує факторизацію виразу). Увімкнемо його і подивимося на  $w$ :

```
on factor;
```

```
w;
```

Отримаємо

$$(a + b)^2.$$

Якби ми хотіли зберегти  $w$  саме в такому вигляді, потрібно б було виконати операцію

```
w:=w;
```

Залежно від положення перемикачів `exp` і `factor` система спочатку визначає, чи потрібно розкривати дужки, обчислювати степені, а потім – чи слід факторизувати результат. Може статися, що при різних положеннях перемикачів результат матиме однаковий вигляд: розкриття дужок компенсується факторизацією.

Тепер обчислимо

```
u/w;
```

Отримаємо

$$(a + b)^2.$$

Як бачимо, `REDUCE` скорочує спільні множники. Найпростіші випадки виконуються автоматично, а складніші керуються перемикачем `gcd` (greatest common divisor, за замовченням `off`). Проілюструємо це на прикладі:

```
w1:=a^3-b^3;
```

```
w2:=a^2-b^2;
```

```
w1/w2;
```

```
on gcd;
```

```
w1/w2;
```

Результат:

$$w_1 := a^3 - b^3;$$

$$w_2 := a^2 - b^2;$$

$$\frac{a^3 - b^3}{a^2 - b^2};$$

$$\frac{a^2 + ab + b^2}{a + b}.$$

Бачимо, що скорочення відбулося.

**Зауваження.** Під час одночасного введення кількох команд може з'явитися повідомлення `+++ error: (99 "End-of-file read")`. Це пов'язано з особливостями вебінтерфейсу. У такому разі команди слід вводити послідовно, по одній.

Крім перемикача `gcd`, існує також однойменна функція `gcd`, яка знаходить найбільший спільний дільник двох поліномів або чисел. Наприклад:

```
gcd(a^4-b^4,a^3-b^3);  
gcd(123456,654321);
```

Результат:

$$a - b;$$

3.

Ще один оператор, корисний при роботі з поліномами і цілими числами – `factorize`. Наприклад:

```
factorize(1234567890);
```

Результат:

$$\{\{2,1\},\{3,2\},\{5,1\},\{3607,1\},\{3803,1\}\}.$$

Останнє означає, що

$$1234567890 = 2^1 \times 3^2 \times 5^1 \times 3607^1 \times 3803^1.$$

Інший приклад:

```
factorize(x^4-1);
```

Результат:

$$\{\{x^2 + 1,1\},\{x + 1,1\},\{x - 1,1\}\}.$$

Тобто

$$x^4 - 1 = (x^2 + 1)(x + 1)(x - 1).$$

Увімкнемо перемикач `complex` і повторимо цю операцію:

```
on complex;
factorize(x^4-1);
```

Результат:

$$\{\{x + i, 1\}, \{x - i, 1\}, \{x + 1, 1\}, \{x - 1, 1\}\}.$$

Тобто

$$x^4 - 1 = (x + i)(x - i)(x + 1)(x - 1).$$

**Зауваження.** Експериментуючи з різними перемикачами, можна легко забути їхній поточний стан. Можна або явно задавати їх щоразу, або перезапускати REDUCE, оновивши сторінку браузера чи скориставшись меню **REDUCE**, пункт **Restart Reduce** над вікном **Input/Output Display**. Через це саме меню можна також роздрукувати результати.

Положення перемикачів за замовчуванням (зразу після запуску REDUCE): on exp, off factor, off gcd, on allfac (регулює винесення спільного множника у виразі).

### 3.3. Вигляд формул у вікні Input/Output Display

Вигляд формул у вікні виведення команд та результатів їх виконання визначається перемикачами fancy (вишуканий) та nat (natural — природний), які за замовчуванням увімкнені. Проілюструємо це на прикладі:

$$w := x^2 / (x + 1);$$

Отримаємо (у стандартному режимі on fancy, on nat):

$$w := \frac{x^2}{x + 1}.$$

Вимкнемо перемикач fancy:

```
off fancy; w;
```

Результат набуває вигляду:

$$\frac{x^2}{x+1}.$$

Тепер вимкнемо також перемикач nat:

```
off nat; w;
```

Результат не виведено.

Знову ввімкнемо fancy:

```
on fancy; w;
```

Отримаємо лінійний запис формули:

```
x**2/(x+1)$
```

(у цьому режимі степінь позначається як \*\*).

Отже, для якісного, «книжкового» відображення формул використовують комбінацію `on fancy`, `on nat` (це режим за замовчуванням). Для однорядкового запису, який зручно копіювати і використовувати під час подальшого введення в **Input Editor**, застосовують `on fancy`, `off nat`. Комбінація `off fancy`, `off nat` не завжди автоматично виводить результат. У разі потреби його можна вивести явно за допомогою команди `write w`;

### 3.4. Обчислення з плаваючою точкою

У REDUCE передбачено можливість виконання обчислень із плаваючою точкою, як і в інших мовах програмування. На відміну від більшості з них, тут обчислення можна проводити з довільною заданою точністю. Перехід до режиму обчислень із плаваючою точкою здійснюємо командою

```
on rounded;
```

Кількість значущих цифр (за замовчуванням – 12), які обчислюються в цьому режимі, задається командою `precision`. Щоб дізнатися поточне значення точності, слід виконати команду

```
precision 0;
```

Розглянемо кілька прикладів.

```
on rounded; pi;
```

Отримаємо:

```
3.14159265359.
```

Тепер змінимо точність:

```
precision 0; precision 50; pi; sqrt(2); precision 0;
```

Результат:

```
12;  
12;  
3.1415926535897932384626433832795028841971693993751;  
1.4142135623730950488016887242096980785696718753769;  
50.
```

Пояснимо отримані результати.

- Після виконання `precision 0`; система виводить поточну точність – 12 значущих цифр.
- Команда `precision 50`; також повертає попереднє значення точності (12), яке діяло на момент її виконання.
- Після цього всі обчислення виконуються з точністю 50 значущих цифр, що видно з поданих значень числа  $\pi$  та  $\sqrt{2}$ .
- Остання команда `precision 0`; показує нове поточне значення точності – 50.

#### 4. Розв'язання рівнянь

Мабуть, розв'язання рівнянь – одна з можливостей СКА, які найчастіше використовуються в різних природничих (і не тільки) науках, в інженерних розрахунках, у задачах економічної теорії. Звичайно, існує багато чисельних методів розв'язування рівнянь, але СКА дають змогу отримувати розв'язки в символьному вигляді, а вже потім, за потреби, обчислювати числові значення для конкретних значень параметрів.

Це можуть бути рівняння статики в інженерних розрахунках, правила Кірхгофа для розрахунку електричних кіл, шкільні задачі про рух тіла під дією сили тяжіння та багато інших прикладів.

##### 4.1. Система лінійних рівнянь

Розглянемо систему трьох лінійних рівнянь. У вікні **Input Editor** вводимо:

```
w1:=x+y+z=6;
w2:=x+y-z=0;
w3:=x-y+z=2;
wsol:=solve({w1,w2,w3},{x,y,z});
```

У вікні **Input/Output Display** отримуємо:

$$\begin{aligned}w_1 &:= x + y + z = 6 \\w_2 &:= x + y - z = 0 \\w_3 &:= x - y + z = 2 \\wsol &:= \{x = 1, y = 2, z = 3\}\end{aligned}$$

Оператор `solve` має два аргументи:

- перший аргумент – список рівнянь (упорядкована множина виразів у фігурних дужках, який може містити вкладені списки),
- другий аргумент – список невідомих.

Результат ми присвоїли змінній `wsol`; він має вигляд списку.

Знайдений розв'язок ми можемо підставити, наприклад, у вираз, що містить змінні  $x, y, z$ :

```
w4:=x^2+y^2+z^2; sub(wsol,w4);
```

Отримаємо:

$$w_4 = x^2 + y^2 + z^2.$$

14

Той самий результат дає команда

```
sub({x=1,y=2,z=3},w4);
```

Тут sub – оператор підстановки. Перший аргумент оператора sub – список «змінна = значення», другий – вираз, у який ці значення підставляються. Знайдені значення невідомих  $x$ ,  $y$ ,  $z$  можна зберегти в окремих змінних:

```
xx:=sub(wsol,x); yy:=sub(wsol,y); zz:=sub(wsol,z);
```

Безпосередньо присвоювати знайдені значення змінним  $x$ ,  $y$ ,  $z$  зазвичай небажано. Якщо ж це потрібно, можна виконати:

```
x:=xx; y:=yy; z:=zz;
```

## 4.2. Квадратне рівняння

Якщо розв'язується одне рівняння, його і відповідну невідому в операторі solve можна не оформлювати як списки:

```
sol:=solve(x^2+5x+6=0,x);
```

Отримаємо:

$$\{x = -2, x = -3\}.$$

Знайдено два корені квадратного рівняння.

Щоб присвоїти їх змінним  $x_1$ ,  $x_2$ , скористаємося операторами part, lhs (ліва частина рівності) та rhs (права частина).

Присвоїмо їхні значення змінним  $x_1$ ,  $x_2$ . Для цього нам потрібні оператори part та lhs (left-hand side), rhs (right-hand side).

Спочатку продемонструємо роботу part:

```
length(sol); part(sol,0); part(sol,1); part(sol,2);
```

Результат:

```
2
list
x = -2;
x = -3.
```

За потреби можна звернутися і до складових рівняння. Наприклад:

```
length(part(sol,1)); part(sol,1,0); part(sol,1,1); part(sol,1,2);
```

Тепер присвоїмо значення коренів змінним  $x_1$ ,  $x_2$ :

```
x1:=rhs(part(sol,1)); x2:= rhs(part(sol,2));
```

Отримаємо:

$$x_1 = -2;$$

$$x_2 = -3.$$

Перевіримо, що знайдені корені задовольняють рівняння  $x^2 + 5x + 6 = 0$  :

$$\text{sub}(x=x_1, x^2+5x+6=0); \text{sub}(x=x_2, x^2+5x+6=0);$$

У результаті отримаємо дві рівності вигляду  $0 = 0$ .

### 4.3. Кубічне рівняння

Спробуємо розв'язати кубічне рівняння:

$$\text{equ}:=x^3-x+5=0; \text{solve}(\text{equ},x);$$

Отримаємо:

$$\{x = \text{root\_of}(x^3 - x + 5, x, \text{tag}_1)\}.$$

Якщо розв'язок рівняння громіздкий, він за замовчуванням не розгортається. Щоб побачити його в явному вигляді, слід увімкнути перемикач `fullroots`. Тут для ілюстрації наведемо лише один з трьох розв'язків:

$$\text{on fullroots; wsol:=solve(equ,x)\$ length(wsol); part(wsol,3);}$$

Тут знак `$` використано замість `;` щоб не виводити весь список розв'язків. Потім за допомогою `part` виведено третій корінь. `REDUCE` записав тут розв'язок кубічного рівняння не через радикали, а через гіперболічні функції.

Результат:

$$x = \frac{-2 \cosh\left(\frac{\text{arccosh}\left(\frac{15\sqrt{3}}{2}\right)}{3}\right)}{\sqrt{3}}.$$

Якщо достатньо числового значення кореня, можна перейти до наближених обчислень (за замовчуванням – 12 значущих цифр), навіть не вмикаючи `fullroots`. Вимкнемо `nat`, щоб результат легко копіювався:

$$\text{off fullroots; on rounded; off nat; solve(equ,x);}$$

Отримаємо:

$$\{x=0.952080429567 + 1.31124804408*i, x=0.952080429567 - 1.31124804408*i, x = -1.90416085913\}.$$

## 5. Диференціювання

Диференціювання в REDUCE здійснюється оператором `df`. За допомогою оператора `depend` можна оголосити одну змінну функцією іншої (або кількох інших).

Розглянемо приклади обчислення похідних:

```
depend u,x; w:=cos(x*u); df(w,x); df(cos(sin(x)),x);
```

Отримаємо:

$$w := \cos(ux)$$

$$- \sin(ux) \left( \frac{\partial u}{\partial x} x + u \right)$$

$$\cos(\sin(x)) \cos^2(x) + \sin(\sin(x)) \sin(x).$$

У першому випадку обчислено похідну за змінною  $x$  від  $w := \cos(ux)$  за умови, що  $u$  є функцією  $x$ . У другому – похідну від  $\cos(\sin(x))$ .

Обчислити другу похідну від  $w$  за  $x$  можна командою `df(w,x,x)` або `df(w,x,2)`.

Засобами REDUCE можна розкласти функцію у **ряд Тейлора**. Наприклад, для функції помилок

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

розклад біля точки  $x = 0$  до членів порядку  $x^5$  включно можна отримати так:

```
w:=erf(x); u:=taylor(w,x,0,5); length(u); part(u,0);
```

Результат:

$$w := \operatorname{erf}(x);$$

$$\frac{2\sqrt{\pi}}{\pi} x - \frac{2\sqrt{\pi}}{3\pi} x^3 + \frac{\sqrt{\pi}}{5\pi} x^5 + O(x^6).$$

У цьому прикладі:

- перший аргумент `taylor` – функція, яку потрібно розкласти в ряд;
- другий – змінна розкладу;
- третій – точка розкладу;
- четвертий – порядок розкладу (включно).

Є можливість розкладу за двома і більше змінними, але на цьому тут не зупиняємося.

Окрім оператора `taylor`, для роботи зі степеневими рядами можна використовувати оператор `ps` (powers series) з пакету TPS (Truncated/Extendible Power Series). Він створює об'єкт типу «степеневий ряд», з яким можна працювати як із формально нескінченним рядом. Оператор `pstruncate` перетворює такий ряд на поліном заданого порядку.

Розглянемо приклад:

```
u:=sin(x);
```

```
ups:=ps(u,x,0);
```

Отримаємо:

$$u := \sin(x);$$

$$\text{ups} := x - \frac{1}{6}x^3 + \frac{1}{120}x^5 + O(x^7).$$

Тут змінній  $u$  присвоєно значення  $\sin(x)$ , після чого виконано розклад у ряд біля точки  $x = 0$ .

Тепер обріжемо ряд до сьомого порядку за  $x$ :

```
up:=pstruncate(ups,7);
```

Отримаємо:

$$\text{up} := \frac{x(-x^6 + 42x^4 - 840x^2 + 5040)}{5040}.$$

Вираз трохи незвичний, оскільки REDUCE звів дробу до спільного знаменника і виніс спільний множник за дужку.

Щоб отримати почленне ділення, увімкнемо перемикач `div`, а щоб система не виносила спільний множник – вимкнемо `allfac`

```
on div; off allfac;
```

Тепер введемо:

```
up;
```

У результаті маємо:

$$-\frac{1}{5040}x^7 + \frac{1}{120}x^5 - \frac{1}{6}x^3 + x.$$

## 6. Інтегрування

Для інтегрування в REDUCE використовується оператор `int`.

Приклад невизначеного інтеграла:

```
int(exp(-x)*x,x);
```

Результат:

$$\frac{-(x+1)}{e^x}.$$

Приклад визначеного інтеграла (від 0 до нескінченності):

```
int(exp(-x^2),x,0,infinity);
```

Результат:

$$\frac{\sqrt{\pi}}{2}.$$

Якщо інтеграл не обчислюється аналітично, можна скористатися чисельним інтегруванням за допомогою оператора `num_int`. Наприклад,

```
num_int(sin(sin(x)),x,0,1);
```

Результат:

0.430606103121.

## 7. Диференціальні рівняння

Для розв'язування диференціальних рівнянь у REDUCE використовується оператор `odesolve`.

Розглянемо приклад:

```
depend y, x; eqy:=df(y,x,x)=-y-g*df(y,x); odesolve(eqy,y,x);
```

Спочатку директивою `depend` повідомляємо REDUCE, що змінна  $y$  залежить від  $x$ . Далі записуємо диференціальне рівняння (у даному разі вільного осцилятора із затуханням) і присвоюємо його змінній `eqy`. Після цього викликаємо `odesolve`.

Аргументи оператора `odesolve`:

1. рівняння,
2. залежна змінна,
3. незалежна змінна.

Отримуємо результат:

$$eqy := \frac{\partial^2 y}{\partial x^2} = -\frac{\partial y}{\partial x} g - y;$$

$$\left\{ y = \frac{\text{arbconst}(2) + e^{\sqrt{g^2-4}x} \text{arbconst}(1)}{\exp\left(\frac{\sqrt{g^2-4}x + gx}{2}\right)} \right\}.$$

Тут  $\text{arbconst}(1)$  та  $\text{arbconst}(2)$  – довільні сталі інтегрування, значення яких визначаються з початкових умов.

## 8. Робота з поліномами

REDUCE, як і інші системи комп'ютерної алгебри, надає широкі можливості для роботи з поліномами. Розглянемо деякі з них на прикладах.

Візьмемо поліном з раціональними коефіцієнтами

$$y := 2 + 3/2 * x + a * x - 5/3 * x^2 + b * x^2;$$

За замовчуванням перемикачі мають стандартні значення, тому результат подається зі спільним знаменником:

$$y := \frac{6ax + 6bx^2 - 10x^2 + 9x + 12}{6}.$$

Якщо потрібно отримати запис із раціональними коефіцієнтами, вмикаємо перемикач `rational`:

`on rational; y;`

Отримаємо:

$$ax + bx^2 - \frac{5}{3}x^2 + \frac{3}{2}x + 2.$$

Збираємо коефіцієнти при  $x$ :

`factor x; y;`

Результат:

$$x^2 \left( b - \frac{5}{3} \right) + x \left( a + \frac{3}{2} \right) + 2.$$

Вимкнемо `rational` і, скасовуючи збирання за  $x$ , знайдемо чисельник і знаменник початкового запису:

`off rational; remfac x; yn:=num(y); yd:=den(y);`

Отримаємо:

$$y_n := 6ax + 6bx^2 - 10x^2 + 9x + 12;$$

$$y_d := 6.$$

Тепер знайдемо найбільший степінь  $x$  у чисельнику  $y_n$ , коефіцієнт при  $x^1$  та доданок з найвищим степенем  $x$ :

$$\text{deg}(y_n, x); \text{coeffn}(y_n, x, 1); \text{lterm}(y_n, x);$$

Результати:

$$2;$$

$$3(2a + 3);$$

$$2x^2(3b - 5).$$

Таким чином, REDUCE надає зручні інструменти для аналізу структури поліномів.

## 9. Трохи тригонометрії

За тригонометричні перетворення в REDUCE відповідає пакет **trigsimp**. За замовчуванням він завантажений на початку роботи системи. Продемонструємо деякі його можливості на прикладах перетворення тригонометричних формул:

$$w := \sin(2x) + \cos(y)^2 + \sin(x+y);$$

$$\text{trigsimp}(w, \text{expand});$$

У результаті отримаємо:

$$w := \cos^2(y) + \sin(x+y) + \sin(2x);$$

$$2 \cos(x) \sin(x) + \cos(x) \sin(y) + \cos(y) \sin(x) - \sin^2(y) + 1.$$

Тут використано параметр `expand`; в результаті розкрито формули подвійного кута та суми аргументів. Тут ми звели вираз до тригонометричних функцій одного аргументу.

Тепер виконаємо обернене перетворення — понизимо степені тригонометричних функцій і зведемо вираз до функцій лінійних комбінацій аргументів:

$$u := \sin(x)^3 + \cos(y) * \sin(x);$$

$$\text{trigsimp}(u, \text{combine});$$

Отримаємо:

$$u := \sin(x) (\cos(y) + \sin^2(x));$$

$$\frac{2 \sin(x - y) + 2 \sin(x + y) - \sin(3x) + 3 \sin(x)}{4}.$$

У цьому випадку використання параметра `combine` зумовлює застосування формул перетворення добутоків і степенів тригонометричних функцій до сум і різниць аргументів.

## 10. Некомутуючі оператори

Ще одна цікава можливість REDUCE – робота з некомутуючими операторами, що може бути корисною, зокрема, під час розв'язування задач з квантової механіки.

Розглянемо приклад:

```
operator a, b; noncom a, b;
commutator := a(1)*b(1) - b(1)*a(1);
```

Отримаємо:

$$\text{commutator} := a(1)b(1) - b(1)a(1).$$

Тут формально введено аргумент оператора. Тепер задамо правило перестановки, аналогічне співвідношенню для операторів народження та знищення в квантовій механіці:

```
let a(1)*b(1)=>b(1)*a(1)+1; commutator;
```

Результат:

1.

Перевіримо, як система враховує це правило в інших виразах:

```
(a(1)+b(1))^2;
```

Отримаємо:

$$a(1)^2 + b(1)^2 + 2b(1)a(1) + 1.$$

Таким чином можна задати правила перестановки для некомутуючих операторів, а REDUCE автоматично враховує їх під час перетворення виразів.

## 11. Висновки

У статті наведено частину команд системи комп'ютерної алгебри REDUCE, яка може слугувати основою для самостійного опанування цієї системи. Для первинного ознайомлення з можливостями REDUCE достатньо скористатися вебверсією на сайті розробників (саме так виконувалися обчислення під час підготовки цієї статті). Однак для систематичної та тривалої роботи доцільно встановити REDUCE на власний комп'ютер.

Зрозуміло, що наведений огляд далеко не вичерпний. Обсяг повного посібника REDUCE User's Manual (Hearn та Schöpf, 2025) становить 1389 сторінок. Поза межами статті залишилися такі можливості системи, як читання команд із файлу та запис результатів у файл, побудова графіків, робота з масивами, матрицями, векторами й тензорами, спеціальні функції, оператори заміни змінних у диференціальних рівняннях, неперервні дробі, обчислення границь, лінійні оператори, задання правил перетворення (зокрема, саме цей механізм використовується в пакеті trigsimp), програмування (функції, процедури, умовні оператори, цикли), пакети для задач диференціальної геометрії та небесної механіки тощо.

REDUCE є прикладом того, як класичні ідеї комп'ютерної алгебри органічно поєднуються із сучасними потребами дослідників і студентів.

### **Джерела фінансування**

Ця робота не отримала жодного гранту від фінансових установ у державному, комерційному чи некомерційному секторах.

#### **Список використаних джерел**

- Axiom Development Team. (n.d.). Axiom: The scientific computation system. Retrieved February 8, 2026, from <https://www.nongnu.org/axiom/>
- MacCallum, M. A. (2018). Computer algebra in gravity research. *Living reviews in relativity*, 21 (1), 6. <https://doi.org/10.1007/s41114-018-0015-6>
- Maxima Manual Version 5.48.1. <https://maxima.sourceforge.io/ext/maxima.pdf>
- Hearn, A. C. (1968). REDUCE: A user oriented interactive system for algebraic simplification. In M. Klerer & J. Reinfelds (Eds.), *Interactive systems for experimental applied mathematics* (pp. 79–90). Academic Press. <https://dl.acm.org/doi/10.1145/2402536.2402544>
- Hearn, A. C. (2005). REDUCE: The first forty years [Conference presentation]. A3L Conference in Honor of the 60th Birthday of Volker Weispfenning, Passau, Germany. <http://www.reduce-algebra.com/reduce40.pdf>
- Hearn, A. C., & Schöpf, R. (2025). REDUCE user's manual: Free version (Rev. 7146). Retrieved February 8, 2026, from <https://reduce-algebra.sourceforge.io/>
- Rayna, G. (1987). *Reduce: Software for algebraic computation*. Springer. <https://doi.org/10.1007/978-1-4612-4806-4>

#### **References**

- Axiom Development Team. (n.d.). Axiom: The scientific computation system. Retrieved February 8, 2026, from <https://www.nongnu.org/axiom/>
- MacCallum, M. A. (2018). Computer algebra in gravity research. *Living reviews in relativity*, 21 (1), 6. <https://doi.org/10.1007/s41114-018-0015-6>
- Maxima Manual Version 5.48.1. <https://maxima.sourceforge.io/ext/maxima.pdf>
- Hearn, A. C. (1968). REDUCE: A user oriented interactive system for algebraic simplification. In M. Klerer & J. Reinfelds (Eds.), *Interactive systems for experimental applied mathematics* (pp. 79–90). Academic Press. <https://dl.acm.org/doi/10.1145/2402536.2402544>
- Hearn, A. C. (2005). REDUCE: The first forty years [Conference presentation]. A3L Conference in Honor of the 60th Birthday of Volker Weispfenning, Passau, Germany. <http://www.reduce-algebra.com/reduce40.pdf>
- Hearn, A. C., & Schöpf, R. (2025). REDUCE user's manual: Free version (Rev. 7146). Retrieved February 8, 2026, from <https://reduce-algebra.sourceforge.io/>

Rayna, G. (1987). Reduce: Software for algebraic computation. Springer. <https://doi.org/10.1007/978-1-4612-4806-4>

Отримано редакцією журналу: 20.01.2026

Прийнято до публікації: 19.06.2026

Опубліковано: 30.07.2026

**Victor ROMANENKO, Dr. of Sci. (Phys&Math), Leading Researcher**

**ORCID: 0000-0001-8554-0214**

**e-mail: victor.romanenko@gmail.com**

**Institute of Physics of the National Academy of Sciences of Ukraine, Kyiv, Ukraine**

**Nataliya KORNILOVSKA, Ph.D (Tech. Sci.), Assoc. Prof.**

**ORCID ID: 0000-0002-8331-8027**

**e-mail: kornilovska.natalya@kntu.net.ua**

**Kherson National Technical University, Khmelnytskyi, Ukraine**

## **REDUCE – A CLASSIC OF COMPUTER ALGEBRA**

**Abstract.** *The article provides a brief overview of the computer algebra system REDUCE. Despite its considerable age, REDUCE is still actively used by researchers. The purpose of the work is to provide basic knowledge of REDUCE that will help readers begin working with the system. Most of the examples presented relate to mathematics; their selection is intended to demonstrate the widest possible range of the system's basic capabilities within the scope of a short publication.*

**Keywords:** *computer algebra; mathematical analysis; elementary functions.*

Received: 20.01.2026

Accepted for publication: 19.06.2026

Published: 30.07.2026