Математический пакет MAPLE

ВВЕДЕНИЕ

- В настоящее время научное программирование претерпевает серьезную трансформацию: развиваются интегрированные среды, основанные на алгоритмических языках, и растет применение универсальных математических систем (Maple, Mathematics, MATLAB. Mathcad и др.). Эти системы имеют дружественный интерфейс, реализуют множество стандартных и специальных математических операций, снабжены мощными графическими средствами и обладают собственными языками программирования. Все это предоставляет широкие возможности для эффективной работы специалистов разных профилей, о чем говорит активное применение математических пакетов в научных исследованиях и в преподавании. Система аналитических вычислений Maple – хороший выбор для проведения любого исследования, где требуется математика – от курсовой работы до научного открытия. С помощью этих пакетов проще готовить и выполнять задания, устраивать демонстрации и гораздо быстрее решать исследовательские и инженерные задачи.

- Математический пакет Maple – интеллектуальный лидер в своих классах и образец, определяющий развитие компьютерной математики. Компьютерная алгебра Maple вошла составной частью в ряд современных пакетов. Сам пакет постоянно совершенствуются, развивая аппарат и пополняя ресурсы. Пакет Maple – мощная и хорошо организованная система, надежная и простая в работе. Освоение даже части его возможностей даст несомненный эффект, а по мере накопления опыта придет настоящая эффективность от взаимодействия с ним. Еще одним достоинством пакета является неизменность набора основных команд и конструкций языка при появлении новых версий.
- Язык Maple – это функции и команды сравнительно небольшого по объему, но быстрого ядра, написанного на языке Си, основной библиотеки, содержащей около 500 команд и функций, написанных уже на собственном языке Maple, и большого количества специализированных библиотек, также написанных на собственном языке Maple и расширяющих “способности” Maple в различных областях математики. Пожалуй, наиболее важная особенность системы – открытость архитектуры, т.е. возможность редактировать и изменять подпрограммы библиотек, а также пополнять библиотеки собственными подпрограммами. Благодаря этому за короткое время было создано большое число Maple-подпрограмм, целиком написанных пользователями из самых разных областей науки и техники. Лучшие подпрограммы пополняют библиотеку пользователей, так называемую Share-библиотеку, которая распространяется вместе с пакетом Maple.

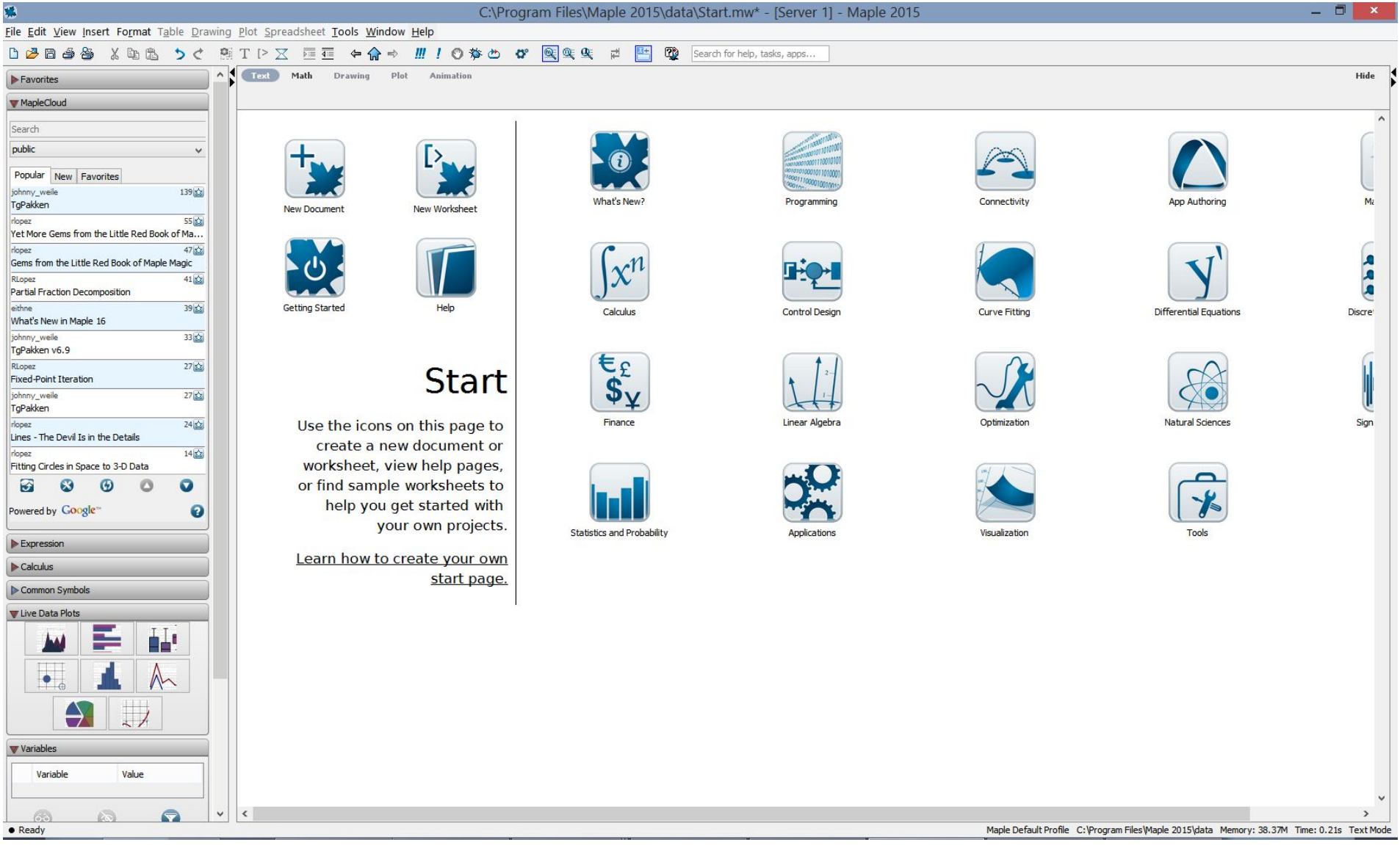
Назначение пакета Maple

- Maple (http://www.maplesoft.com/)
- Минимальные требования к системе:
- • процессор Pentium III 650 МГц;
- • 128 Мбайт оперативной памяти (рекомендуется 256 Мбайт);
- • 400 Мбайт дискового пространства;
- • операционные системы: Windows NT 4 (SP5)/98/ME/2000/2003 Server/XP Pro/XP Home.
- Программа Maple — своего рода патриарх в семействе систем символьной математики и до сих пор является одним из лидеров среди универсальных систем символьных вычислений. Она предоставляет пользователю удобную интеллектуальную среду для математических исследований любого уровня и пользуется особой популярностью в научной среде. Отметим, что символьный анализатор программы Maple является наиболее сильной частью этого ПО, поэтому именно он был позаимствован и включен в ряд других CAE-пакетов, таких как MathCad и MatLab, а также в состав пакетов для подготовки научных публикаций Scientific WorkPlace и Math Office for Word.

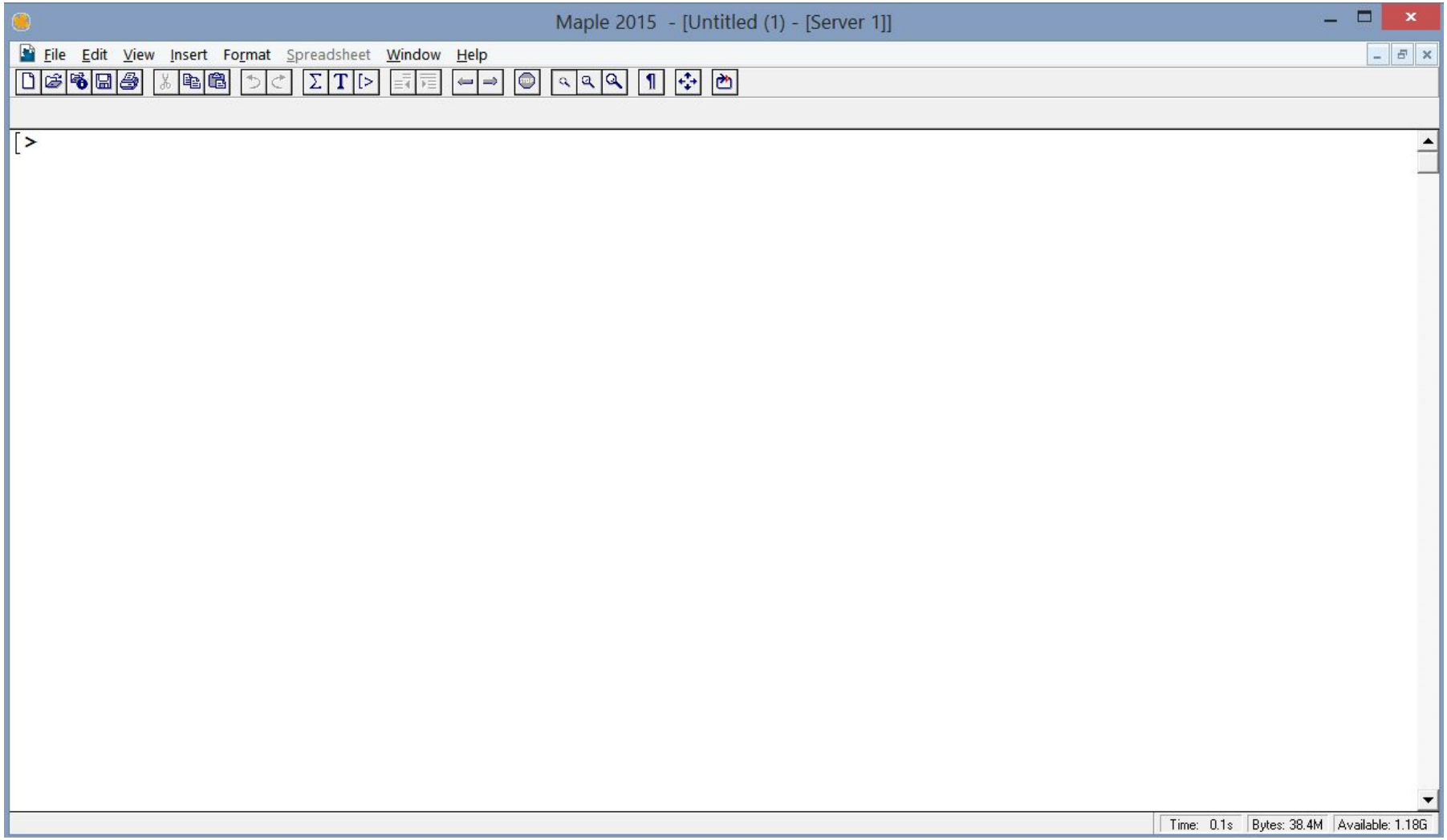
- Пакет Maple — совместная разработка Университета Ватерлоо (шт. Онтарио, Канада) и Высшей технической школы (ETHZ, Цюрих, Швейцария). Для его продажи была создана специальная компания — Waterloo Maple, Inc., которая, к сожалению, больше прославилась математической проработкой своего проекта, чем уровнем его коммерческой реализации. В результате система Maple ранее была доступна преимущественно узкому кругу профессионалов. Сейчас эта компания работает совместно с более преуспевающей в коммерции и в проработке пользовательского интерфейса математических систем фирмой MathSoft, Inc. — создательницей весьма популярных и массовых систем для численных расчетов MathCad, ставших международным стандартом для технических вычислений.

Программная платформа пакета Maple

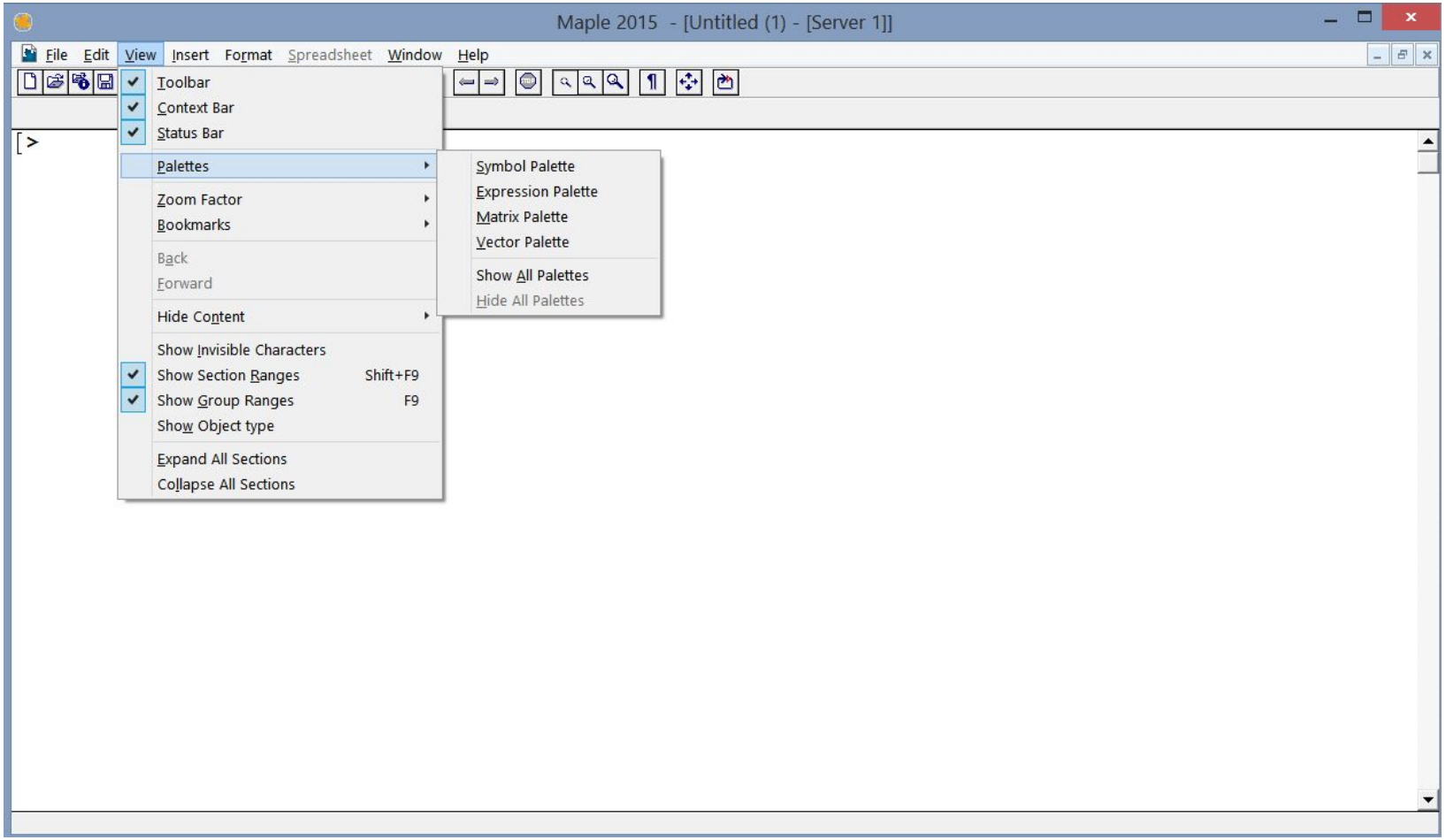
- Maple состоит из ядра – процедур, написанных на языке C, библиотеки оптимизированных процедур, написанных на языке Maple, и интерфейса. Ядро выполняет большинство базовых операций. Библиотека содержит команды – процедуры, выполняемые в режиме интерпретации. Библиотеку можно пополнять своими собственными процедурами и, следовательно, расширять возможности Maple. Интерфейс Maple в различных версиях может выглядеть различно. Так, например, в Maple имеется два вида рабочего документа (worksheet). Первый документ, предлагаемый в качестве основного, имеет следующий вид:


C:\Program Files\Maple 2015\data\Start.mw* - [Server 1] - Maple 2015
File Edit View Insert Format Table Drawing Plot Spreadsheet Tools Window Help
Search for help, tasks, apps...
Favorites
MapleCloud
Search
public
Popular New Favorites
johnny_weile 139
TgPakken
rlopez 55
Yet More Gems from the Little Red Book of Ma...
rlopez 47
Gems from the Little Red Book of Maple Magic
RLopez 41
Partial Fraction Decomposition
eithne 39
What's New in Maple 16
johnny_weile 33
TgPakken v6.9
RLopez 27
Fixed-Point Iteration
johnny_weile 27
TgPakken
rlopez 24
Lines - The Devil Is in the Details
rlopez 14
Fitting Circles in Space to 3-D Data
Powered by Google
Expression
Calculus
Common Symbols
Live Data Plots
Variables
Variable
Value
Text
Math
Drawing
Plot
Animation
Hide
New Document
New Worksheet
Getting Started
Help
Start
Use the icons on this page to create a new document or worksheet, view help pages, or find sample worksheets to help you get started with your own projects.
Learn how to create your own start page.
What's New?
Programming
Connectivity
App Authoring
Calculus
Control Design
Curve Fitting
Differential Equations
Finance
Linear Algebra
Optimization
Natural Sciences
Statistics and Probability
Applications
Visualization
Tools
Ready
Maple Default Profile C:\Program Files\Maple 2015\data Memory: 38.37M Time: 0.21s Text Mode

- Вид второго документа (классический вид) не изменяется в последних версиях Maple, начиная с четвертой. При установке системы Maple 10 на рабочий стол выводятся два ярлыка с названиями Maple и Classic Worksheet Maple . При вызове системы Maple с помощью второго ярлыка появляется классический вид документа. Он выглядит следующим образом.



- Как у всех приложений Windows интерфейс системы Maple имеет ряд характерных элементов. Эти элементы видны на рисунке. Перечислим их (сверху вниз).
  - строка состояния,
  - строка главного меню
  - главная панель инструментов,
  - контекстная панель инструментов (ее вид зависит от режима работы),
  - окно ввода и редактирования документа,



Меню системы Maple.

- Главное меню системы Maple расположено сразу под строкой заголовка и
- предоставляет доступ к основным действиям. В главное меню входят следующие пункты.
- File – работа с файлами, предварительный просмотр и печать документа,
- Edit – редактирование документа и действия с буфером обмена,
- View – управление видом окна,
- Insert – операции вставки текста, команд, графики и т.д.
- Format – форматирование документа,
- Spreadsheet – действия с таблицами,
- Windows – управление оконным режимом,
- Help – работа со справочным разделом.
- Доступность пунктов главного меню зависит от состояния документа.
- Назначение пунктов главного меню понятно большинству пользователей,
- работающих с Windows-приложениями.

Главная панель инструментов.

На панели инструментов находятся следующие кнопки

 - Create a new worksheet (создать новый документ),

 -Open an existing worksheet (открыть существующий документ),

 - Open a specified URL (открыть URL),

 Save the active worksheet (сохранить рабочий документ),

 Print the active worksheet (печатать рабочий документ),

 Cut the selection to the clipboard (вырезать выделенный фрагмент в буфер),

 Copy the selection to the clipboard (скопировать выделенный фрагмент в буфер),

 Paste the clipboard contents into the current worksheet (вставить из буфера),

 Undo the last operation (отменить последнюю операцию),

 Redo the previously undone operation (восстановить последнюю операцию),

 Insert nonexecutable Standard Math in a text region (вставить (неисполняемую) формулу в область текста),

 Insert text at the cursor (вставить текст в место перед курсором),

 Insert a new execution group after the cursor (вставить исполняемую группу операторов после курсора),

 Remove the section enclosing the selection (удалить выделенный раздел),

 Enclose the selection in a section or subsection (вставить фрагмент в новый раздел),

 Go backward in the hyperlink history (перейти назад по гиперссылке),

 Go forward in the hyperlink history (перейти вперед по гиперссылке),

 Cancel the computation in progress (остановить процесс вычислений),

 Set the zoom magnification to 100% (установить масштаб дисплея 100%),

 Set the zoom magnification to 150% (установить масштаб дисплея 150%),

 Toggle the display of nonprinting characters (изменение режима «невидимых» символов),

 Resize the active window to fill the available space (переход окна в полный экран),

 Restart Maple (рестарт Maple).

Панель форматирования текста (Context Bar for Text Regions)

Сюда входят следующие объекты.

Displays the text style (выбрать стиль текста),

Displays text font (выбрать шрифт),

 Bold the selected text (выбрать жирный шрифт),

 Italicize the selected text (выбрать курсив),

 Underline the selected text (выбрать подчеркнутый шрифт),

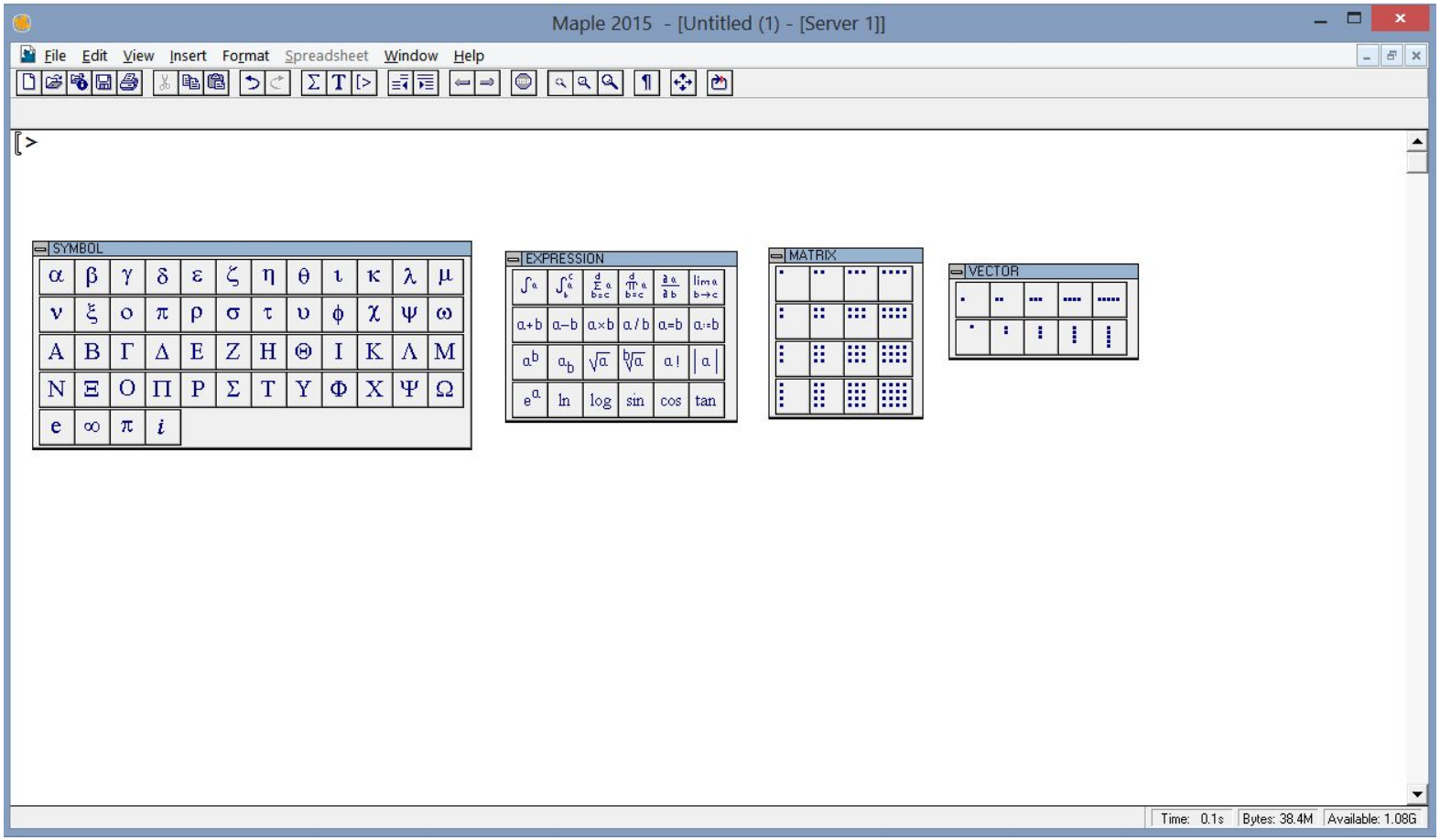
 Align the text on the left side (равнение по левому краю),

 Center the text (равнение по центру),

 Align the text on the right side (равнение по правому краю).

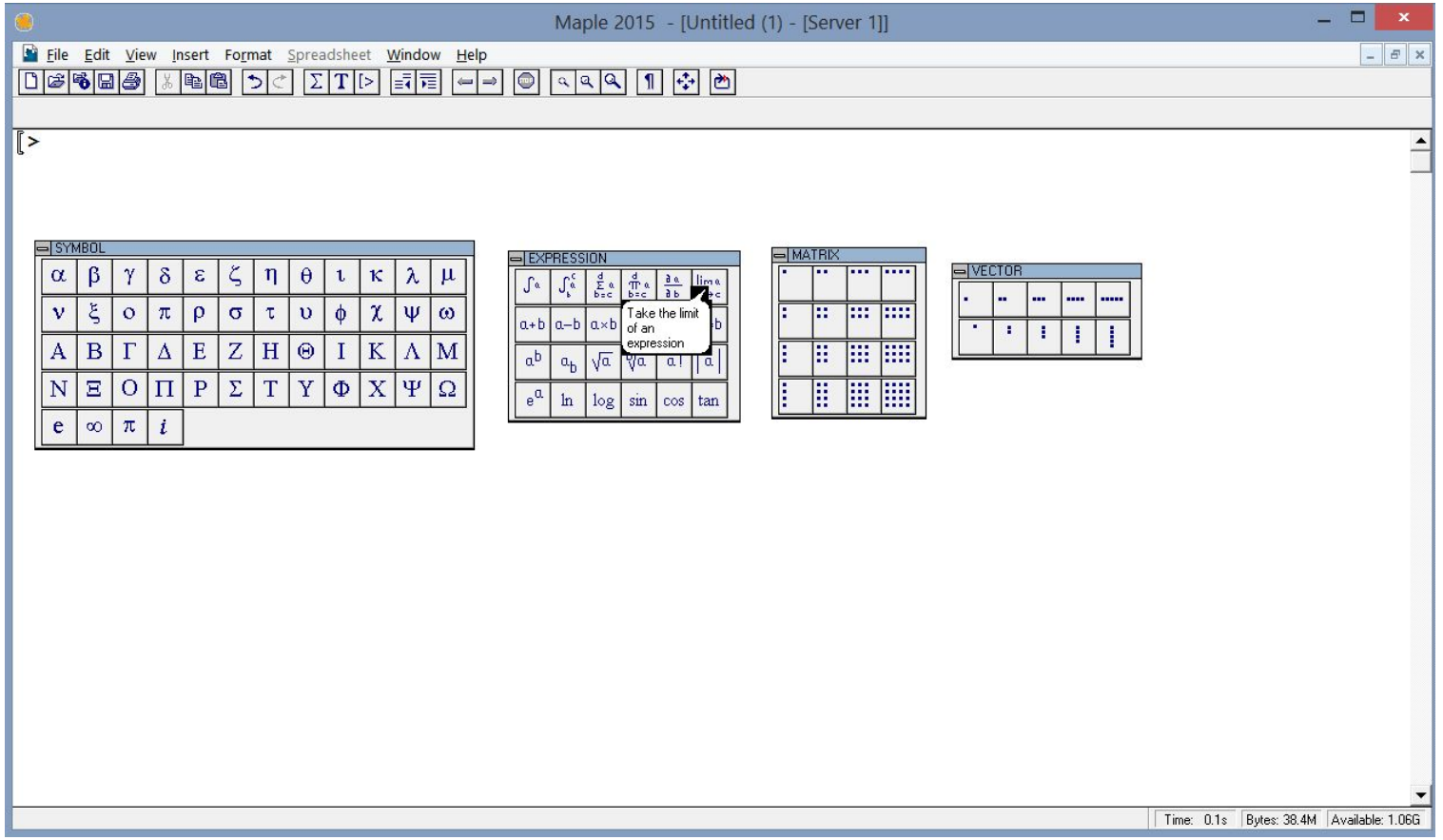
Палитры математических символов.

- Для ввода математических символов, выражений, букв греческого алфавита, матриц и векторов можно (наряду с вводом с клавиатуры) можно пользоваться палитрами



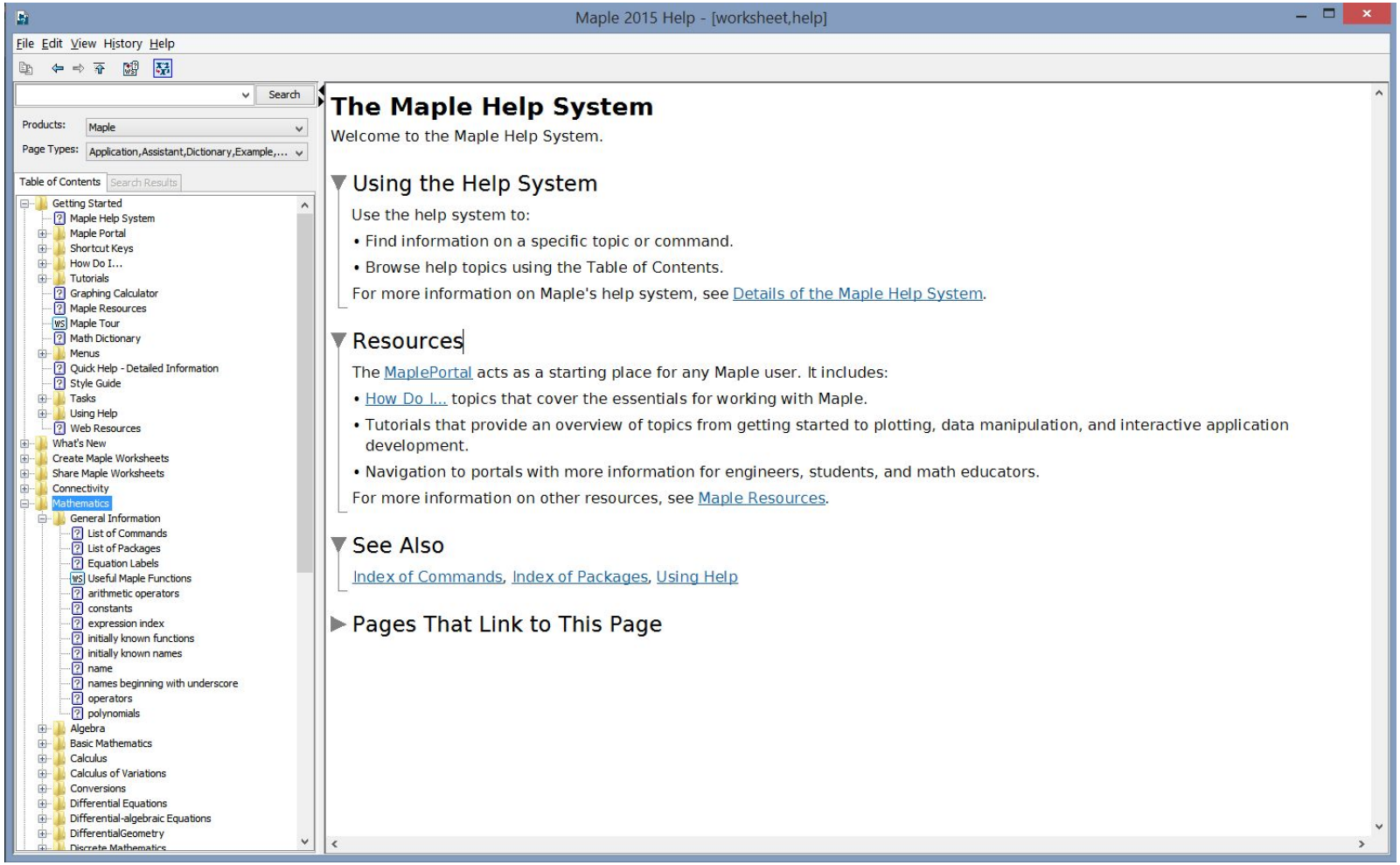
Всплывающие подсказки (balloon help).

- Удобным элементом интерфейса являются всплывающие подсказки, появляющиеся при наведении курсора мыши на некоторые элементы окна. Они имеют вид облачка, в котором имеется краткое пояснение. Например, при наведении курсора на элемент a+b в палитре Expression появляется всплывающая подсказка – addition (сложение).

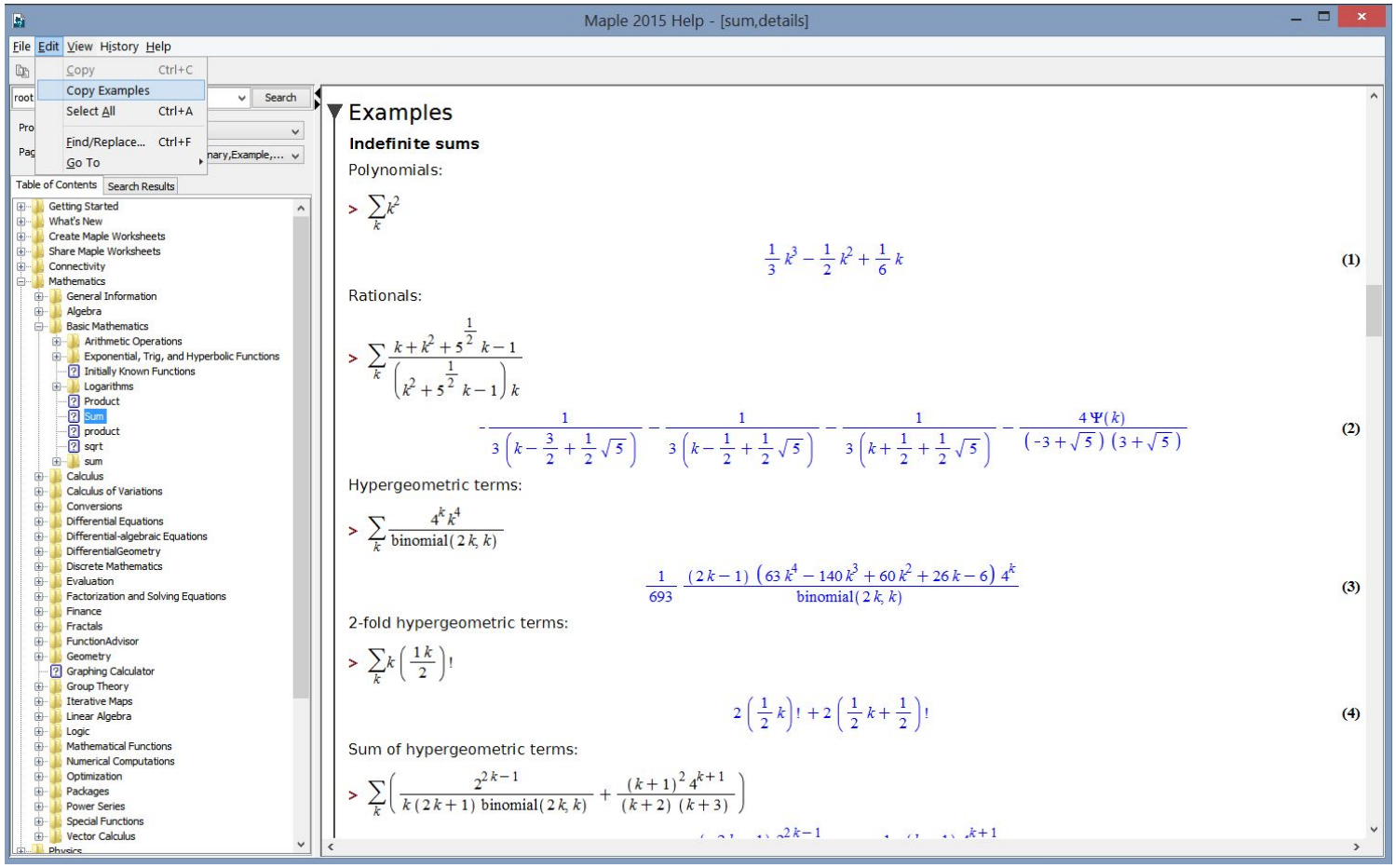


Help

- В любой момент пользователю доступна удобно организованная справочная система по среде и командам пакета Maple. Вызвать ее можно либо по нажатию клавиши <F1>, либо по выбору пункта меню Help и далее соответствующего пункта, например, Mathematics. Справочная система организована в виде гипертекстового документа



- Обычно внизу справки по каждой команде приводятся примеры по работе с данной командой. Эти примеры можно скопировать в буфер с помощью пункта меню Edit→Copy Examples (меню в справочной системе не русифицировано) и затем вставить в рабочий лист. Наличие большого количества примеров позволяет эффективно использовать Maple даже тем, которые недостаточно хорошо знают английский язык.



Элементы языка Maple

- Начать работать в среде Maple действительно просто даже начинающему пользователю. Для этого достаточно прочитать несколько первых глав. Искушенному программисту работа в этом пакете доставит особое удовольствие, так как не надо самому описывать алгоритмы интегрирования или решения дифференциального уравнения. Эта рутинная работа уже выполнена создателями программы. Достаточно ввести свои данные и многие задачи будут решены. Синтаксис Maple очень напоминает синтаксис таких языков программирования, как Паскаль и Фортран.

Символы и переменные

Как уже упоминалось раньше, команду необходимо заканчивать символом “:” или “;”. При определении выражения используются стандартные символы: “+”, “-”, “*”, “/”, “^”, “**”, “:=”, “=”, “!”. Например:

> **y:=5!/((x^2-3*x-5)**(x+2));**

$$y := \frac{120}{(x^2 - 3x - 5)^{(x+2)}}$$

Как видно из примера, возвести выражение в степень можно двумя способами: “^” и “**”.

Для обозначения последовательности чисел используется символ “$”

> **x!$x=1..4;**

$$1, 2, 6, 24$$

Символ “@” - композиционный оператор. Например, чтобы вычислить вторую производную необходимо написать следующее:

> **(D@@2)(ln);**

$$z \to -\frac{1}{z^2}$$

Численный параметр после символа “@” может принимать и отрицательные значения:

> **(sin@@(-1))(x);**

$$\arcsin(x)$$

Интересную роль играет символ ” (кавычки). Одинарные кавычки ссылаются на результат предыдущей команды. Двойные - на результат, полученный две команды назад, тройные - на три команды назад. В качестве примера решим численным методом систему линейных уравнений, очистив предварительно память Maple и определив точность:

> **restart: Digits:=2:**

> **2*x+5*y-z=2;**

$$2x + 5y - z = 2$$

> **x+2*y=5;**

$$x + 2y = 5$$

> **2*x-z=4!;**

$$2x - z = 24$$

- Решение выводится в форме множества. Множество - это группа выражений, заключенных в фигурные скобки. Более подробно о типах данных смотрите в параграфе «Типы данных».

Существует также ряд команд для выполнения операций над множествами:

union - объединение множеств;

intersect - пересечение множеств;

minus - вычитание множеств.

Стандартные логические операции прекрасно дополняют возможности Maple:

and - логическое “и”;

or - логическое “или”;

not - логическое отрицание.

Для более подробной информации нужно воспользоваться справочной системой.

Переменные в Maple характеризуются именем и типом. В качестве имени переменной может использоваться любой набор символов латинского алфавита, не зарезервированных программой. Следует отметить, что система различает заглавное и строчное написание букв.

Константы и внутренние функции

- Константы в Maple бывают целочисленными, числами с плавающей запятой и обыкновенными дробями. Кроме этих типов констант существуют символьные константы - зарезервированные имена. Например: false, true, infinity, Pi, I и т. д.
- Следует помнить, что не рекомендуется использовать эти имена для описания своих собственных переменных:

> **false:=1**

Error, attempting to assign to 'false' which is protected

Но вот так уже можно:

> **False:=1;**

$$False := 1$$

- В Maple используется общепринятые среди математиков названия для основных математических функций, хотя есть некоторые исключения.

ФУНКЦИЯ	ОПИСАНИЕ
abs	модуль
Re	действительная часть
Im	мнимая часть
factorial	факториал
log	обыкновенный логарифм
ln	натуральный логарифм
log10	десятичный логарифм
sqrt	квадратный корень
exp	экспонента
argument	аргумент комплексного числа
binomial	биномиальный коэффициент
round	округление
trunc	отсечение дробной части

Типы данных

- Большое разнообразие типов данных переводит Maple из разряда прикладной программы для решения математических задач в класс систем математического программирования. Около ста зарезервированных имен типов данных может встретить пользователь на необъятных просторах Maple . Существует большое разнообразие функций для работы с данными, в структуре последних можно просто запутаться. Чтобы хоть как-то пролить свет на типы данных, рассмотрим основные, с которыми мы встречаемся при выполнении различных вычислений.

- *Целые*
- В Maple выражение принадлежит к целому типу (тип integer), если оно состоит из последовательности цифр, не разделенных между собой никакими знаками. Длина последовательности ограничена лишь ресурсами системы, но, обычно, больше чем пользователь может себе представить - более 500000 цифр. Число типа integer может быть как положительным, так и отрицательным.
- С целыми числами возможны следующие операции:
- *abs* - модуль числа;
- factorial или n! - нахождение факториала.
- А также многие другие.
- > **abs(-10420);**

10420

- > **factorial(5); 5!;**

120

120

- Для проверки принадлежности выражения к определенному типу служит команда *type*. Формат команды:

type(x, t), где x - любое выражение, t - название типа.
Например: Синтаксис языка Maple

> **type(-102,integer);**

true

- *Дробные*

Тип fraction - дробный тип. Дроби представляются в виде: a/b, где a - целое число со знаком, b - целое число без знака. В выражении типа fraction обязательно присутствие двух полей: числитель и знаменатель.

Функция op от дроби возвращает два числа - числитель и знаменатель.

- > **op(2/7);**

2, 7

- *Числа с плавающей точкой*

Тип float - числа с плавающей точкой. Тип float в среде Maple определен как:

1. последовательность чисел, разделенных точкой:

а) <integer>.<integer>

б) <integer>.

в) .<integer>

2. число может быть представлено в виде:

Float(mantissa, exponent) т.е. <mantissa>*10A<exponent>

```
> type(.1234,float);
```

true

```
> Float(2,4);
```

20000.

- Обратное представление числа реализуется функцией op, которая возвращает два числа - мантиссу и экспоненту.

```
> op(0.02234);
```

2234, -5

- Для приближения чисел с плавающей точкой служит команда evalf:

```
> evalf(Pi,5);
```

3.1416

Строковые типы

В среде Maple определены тип string и тип name. Выражение типа string может содержать цифры и буквы, строчные и прописные. Строка с двух сторон должна быть окружена символом ' . Если же в строку требуется вставить символ ', то его надо удвоить.

```
> var:=`It``s a string`;
```

var := It`s a string

Из строки можно выделить подстроку:

```
> substring(abcdefgh,3..5);
```

cde

Определим длину строки:

```
> length(abcdef);
```

6

Булевы выражения

Для логических операций в среде Maple предусмотрен специальный тип данных - 'booleana зарезервированные слова true и false используются для работы с булевыми выражениями.

В булевских выражениях можно использовать следующие операторы: '<>', 'and', 'or', 'not'.

Для работы с булевыми выражениями предусмотрена команда evalb - вычисление сложного логического выражения:

> **evalb(f=f);**

true

> **ToBe or not ToBe;**

ToBe **or not** *ToBe*

Последовательности

Последовательность - это набор элементов, разделенных запятыми, без скобок.

> **S:=1,2,3,4,5,6;**

S := 1, 2, 3, 4, 5, 6

Множества

Множества принято обозначать фигурными скобками. Для них присущи все правила преобразования, принятые в классической математике.

> **set1:={sin,cos,tan,cos};**

$$set1 := \{\cos, \sin, \tan\}$$

Извлечение элементов с первого по второй из множества setl:

> **op(1..2,set1);**

cos, sin

Определение количества элементов в множестве setl:

> **nops(set1);**

3

Объединение двух множеств {a,b} и {b,c}:

> **{a,b} union {b,c};**

$$\{a, b, c\}$$

Пересечение:

> **{a,b} intersect {b,c};**

$$\{b\}$$

Вычитание:

> **{a,b} minus {b,c};**

$$\{a\}$$

Принадлежность элементов 'a' и 'cos' множеству setl:

> **member(cos,set1);**

$$true$$

> **member(a,set1);**

$$false$$

Зададим множество L в виде последовательности:

> **L := {seq(y[i],i=1. . 4) } ;**

$$L := \{y_0, y_4\}$$

Добавление элемента к множеству L:

> **L := {op (L), z[5]};**

$$L := \{y_0, y_4, z_5\}$$

Удаление второго элемента из множества L:

> **L := subsop(2=NULL,L);**

$$L := \{y_0, z_5\}$$

Массивы

Массив - конечномерный список с целочисленными индексами. Операции, применяющиеся к массивам: array - создание массива;

print - распечатка содержимого массива;

map - задание операции над всеми элементами массива;

op - извлечение элементов (уточнение задания массива).

Создадим массив v:

```
> v := array(1..4);
```

$$v := \mathrm{array}(1 .. 4, [\ ])$$

Заполним этот массив элементами и распечатаем его содержимое:

```
> for i to 4 do v[i] :=i od:
> print(v);
```

$$[1, 2, 3, 4]$$

Создадим одномерный массив ‘s’ с нулевыми значениями:

```
> s:=array(1..2,[0,0]) ;
```

$$s := [0, 0]$$

Создадим двумерный массив ‘m’:

```
> m:=
array(symmetric,1..2,1..2,[[cos(y),0],[0,sin(y)]]);
```

$$m := \begin{bmatrix} \cos(y) & 0 \\ 0 & \sin(y) \end{bmatrix}$$

Таблицы

В отличие от массивов, где индексы - целочисленные значения, расположенные по порядку номеров, индексы у таблиц - любые значения.

Таблица задается указанием слова "table":

```
> table();
```

table([])

Команда "table( )" создала таблицу с неопределенными значениями.

Определим значения таблицы, выполнив команду:

```
> table([22,42]);
```

table([1 = 22, 2 = 42])

Так как индексы таблицы не были определены, то программа сама присвоила им целочисленные значения, расположенные по порядку.

Индексы таблицы можно задать произвольным образом:

```
> S := table([(red)=45,(green)=61]);
```

$S := \text{table}([red = 45, green = 61])$

Обратимся к элементам таблицы:

```
> S[1],S[red];
```

$S_1, 45$

С таблицами возможны также следующие операции.

Зададим таблицу F, элементами которой являются операторы:

```
> F:=table([y=(x->x^2),cos=-sin]):
```

Распечатаем таблицу:

```
> print(F);
```

$\text{table}([y = (x \to x^2), \cos = -\sin])$

Операции с формулами

КОМАНДА	ОПИСАНИЕ
collect(w, x)	Приведение подобных членов в выражении w относительно переменной x
denom(d)	Выделение знаменателя дроби d
expand(w)	Раскрытие скобок выражения w
factor(w)	Факторизация (разложение на множители) выражения w
lhs(ur)	Выделение левой части уравнения ur
normal(w)	Нормализация (сокращение) дроби w
numer(w)	Выделение числителя дроби d
op(i..j, e)	Выделение подвыражения из выражения e
rhs(ur)	Выделение правой части уравнения ur
simplify(w)	Упрощение выражения w
subs(x=t, w)	Подстановка в выражение w вместо выражения x выражение t
subsop(eq1,..,eqN, expr)	Замена некоторого операнда в выражении expr
trigsub(w)	Определение всех тригонометрических эквивалентов выражения w

Начнем с самой распространенной операции - упрощения выражения. Переменной rex присвоим некоторую сумму из тригонометрических слагаемых.

```
> rex := cos(x)^3 + sin(x)^4 + 2*cos(x)^4 -2*sin(x)^4 - cos(2*x):
```

Далее упростим полученный полином:

```
> simplify(rex);
```

$$\cos(x)^3 (\cos(x) + 1)$$

Из приведенного примера видно, что Maple преобразовал выражение rex, подтверждая известные тригонометрические правила. Ниже приведен еще один пример с использованием экспоненты и натурального логарифма.

```
> w:=exp(a+ln(c*exp(c^a))); simplify(w);
```

$$w := \mathbf{e}^{\left(a + \ln\left(c\,\mathbf{e}^{(c^a)}\right)\right)}$$

$$c\,\mathbf{e}^{(c^a + a)}$$

Огромное количество операторов Maple используется в различных контекстах. Далее приведено описание некоторых самых употребимых и полезных операторов Maple.

Для выделения подвыражения из целого выражения служит команда op:

Формат команды:

op(i,e), op(i..j,e), op(e),

где i,j - положительные целые числа, определяющие позицию операнда в выражении, e - любое выражение.

```
> f:=[x,y,z];
```

$$f := [x, y, z]$$

```
> op(3,f);
```

$$z$$

Для того, чтобы заменить некоторый операнд в выражении, служит команда *subsop*.

Формат команды: subsop(eq1, eq2, ..., eqN, expr),

где eq1 - выражение(необязательное) вида: <numI> = <exprI>, numI - положительное целое, exprI - выражение, expr - выражение.

Заменим в определенном нами ранее списке f=[x,y,z] третий элемент этого списка на v:

```
> subsop(3=v,f);
```

$$[x, y, v]$$

При выполнении различных математических операций возникает необходимость подставить одно выражение в другое, а также проверить полученное решение путем подстановки его в исходное равенство. Для этого служит команда subs.

Формат вызова:

subs(s_1,s_2,...,s_n,expr),

где - уравнение, или множество, или список из уравнений,

expr - любое выражение.

При этом s_1,...,s_n подставляются в выражение expr.

Пример:

```
> subs( x=r^(1/3), 3*x*ln(x^3) );
```

$$3\,r^{(1/3)}\ln(r)$$

Другой пример использования функции subs - проверка полученного решения.

```
> eqs := {2*x*y = 1, x + z = 0, 2*x - 3*z = 2};
```

$$eqs := \{2\,x\,y = 1,\ x + z = 0,\ 2\,x - 3\,z = 2\}$$

```
> f:=solve(eqs,{x,y,z});
```

$$f := \{x = \frac{2}{5},\ y = \frac{5}{4},\ z = \frac{-2}{5}\}$$

Математические функции.

Maple имеет полный набор элементарных математических функций. Все они имеют один аргумент. Этот аргумент может быть целым, рациональным, вещественным или комплексным числом. В ответ на обращение к ним элементарные функции возвращают соответствующее значение. Если известно имя нужной функции, то обращение к ней происходит с помощью обращения к этому имени.

Например, при вычислении выражения **eπ** и присваивании результата переменной **A**, достаточно вызвать команду

> **A:=exp(Pi);** (в режиме Maple Input), или

> A:=**eπ** (в режиме Standard Math).

Опишем соответствие математической записи некоторых функций с записью в Maple.

ex – exp(x),

$ln\ x$ - ln(x),

$log10x$ – log10(x),

x - sqrt(x),

x - abs(x),

$sgn\ x$ – signum(x).

К целочисленным функциям относятся, например, следующие функции:

factorial (n) – вычисление факториала (можно использовать оператор !),

iquo(a,b) – целочисленное деление a на b,

irem(a,b) – остаток от деления a на b,

igcd(a,b) – наибольший общий делитель,

ilcm(a,b) – наименьшее общее кратное.

Примеры.

```
> irem(23,4,'q');
                                3
> q;
                                5
> iquo(23,4,'r');
                                5
> r;
                                3
> irem(-23,4);
                               -3
> irem(23,-4);
                                3
```

В Maple определены следующие тригонометрические функции.

sin – синус,

cos – косинус,

tan – тангенс,

cot – котангенс,

sec – секанс,

csc – косеканс.

Примеры.

```
> sin(0);
```

$$0$$

```
> cos(Pi);
```

$$-1$$

```
> expand(sin(x+y));
```

$$\sin(x)\cos(y)+\cos(x)\sin(y)$$

```
> combine(%,trig);
```

$$\sin(x+y)$$

К степенным и логарифмическим функциям в Maple относятся следующие функции.

exp – показательная функция (экспонента),

ilog10 – целая часть логарифма по основанию 10,

ilog – целая часть натурального логарифма,

ln – натуральный логарифм,

log[b] – логарифм по заданному основанию b (библиотечная функция),

sqrt – квадратный корень

Примеры.

```
> ln(1);
```

$$0$$

```
> diff(ln(x),x);
```

$$1x$$

```
> exp(-1);
```

$$\mathbf{e}(-1)$$

```
> exp(1.379);
```

$$3.970928713$$

```
> diff(exp(-x),x);
```

$$-\mathbf{e}^{(-x)}$$

```
> sqrt(3.0);
```

$$1.732050808$$

Функции с элементами сравнения.

abs – абсолютное значение (модуль) числа,

ceil – наименьшее целое, не меньшее аргумента,

floor – наибольшее целое, не превосходящее аргумента,

frac – дробная часть,

trunk – целое округленное, ближайшее к нулю,

round – целое округленное (без дробной части),

signum – сигнум (известная функция «знак»).

Примеры.

```
> abs(-11);
```

$$11$$

```
> abs(3-4*I);
```

$$5$$

```
> abs(cos(3));
```

$$-\cos(3)$$

```
> a:=abs(2*x-3);
```

$$a := |2x - 3|$$

Функции комплексного аргумента.

Для комплексных чисел и данных, кроме уже перечисленных функций определены следующие функции.

argument – аргумент комплексного числа,

conjugate – комплексно-сопряженное число,

Im – мнимая часть числа,

Re – вещественная часть числа,

Polar – полярное представление комплексного числа (библиотечная функция),

Примеры.

```
> conjugate(3+5*I);
```

$$3 - 5\,I$$

```
> conjugate((3+5*I)*z);
```

$$(3 - 5\,I)\,\overline{z}$$

```
> conjugate(sin(exp(1)));
```

$$\sin(e)$$

```
> conjugate(exp(3*I));
```

$$\frac{1}{e^{(3\,I)}}$$

Специальные математические функции.

Ниже приводятся наиболее важные специальные функции, имеющиеся в Maple.

AiryAi (Bi), AiryAi (Bi) Zeros – функции Эйри,

AngerJ – функция Ангера,

Bernoulli – числа и полиномы Бернулли,

BesselI (J,K,Y), BesselI (J,K,Y) Zeros – функции Бесселя,

Beta – бета функция

Binomial – биномиальные коэффициенты,

ChebyshevT(U) – функции Чебышева,

Chi – интегральный гиперболический косинус,

Ci – интегральный косинус

CylinderD(U,V) – параболические функции,

csgn – комплексная сигнум-функция,

dilog – дилогарифм,

Dirac – дельта-функция Дирака,

Ei – экспоненциальный интеграл,

EllipticCE(CK, CPi, E, F, K, Modulus, Nome, Pi) – эллиптические интегралы,

Примеры.

```
> WeierstrassP(1.0,2.0,3.0);
```

$$1.214433709$$

```
> JacobiSN(z,k) = 'sin'(JacobiAM(z,k));
```

$$\mathrm{JacobiSN}(z, k) = \sin(\mathrm{JacobiAM}(z, k))$$

```
> JacobiCN(z,k) = 'cos'(JacobiAM(z,k));
```

$$\mathrm{JacobiCN}(z, k) = \cos(\mathrm{JacobiAM}(z, k))$$

```
> JacobiDN(z,k) = Diff(JacobiAM(z,k),z);
```

$$\mathrm{JacobiDN}(z, k) = \frac{\partial}{\partial z}\mathrm{JacobiAM}(z, k)$$

```
JacobiSN( InverseJacobiAM(z,k), k);
```

$$\sin(z)$$

```
> JacobiCN( InverseJacobiAM(z,k), k);
```

$$\cos(z)$$

Элементы программирования в Maple.

- Простейшим способом задания функции пользователя является применение функционального оператора. При этом используется следующая конструкция.

> name:=(x,y,,,)->expr. После этого вызов функции осуществляется в виде **name:=(x,y…)->expr**, где (x,y…) список формальных параметров функции пользователя с именем **name**. Переменные, указанные в списке параметров, являются локальными. При подстановке на их место фактических параметров они сохраняют их значения только в теле функции **expr**. За пределами этой функции переменные с этими именами оказываются либо неопределенными, либо сохраняют ранее присвоенные им значения. Иногда удобно перед группой действий вставлять команду **restart**. Эта команда приводит все ячейки и имена в исходное состояние.

Пример.

```
> restart;
> x:=0; y:=0;
```

$$x := 0$$

$$y := 0$$

```
> m:=(x,y)->sqrt(x^2+y^2);
```

$$m := (x, y) \rightarrow \sqrt{y^2 + x^2}$$

```
> m(3,4);
```

- Для написания разветвляющихся программ в Maple имеется условный оператор **if**, позволяющий создавать следующие конструкции**if**<условие **>then<** действия**>elif<** условие **>then<** действия **>else <** действия **>|end if**; В прямых скобках находятся необязательный элемент конструкции. Чаще всего используются две следующих конструкции. **if**<условие А **>then<** действия 1> **end if** – если условие А выполняется, то выполняются действия 1, в противном случае ничего не выполняется; **if**<условие А **>then<** действия 1> **else<** действия 2> **end if** - если условие А выполняется, то выполняются действия 1, в противном случае выполняются действия 2. В задании условий используются любые логические конструкции со знаками сравнения (<, <=, >, >=, <>) и логические операторы **and**, **or** и **not**, конструкции с которыми возвращают значения **true** и **false**.

Пример.

```
> a := 3; b := 5;
```

$$a := 3$$

$$b := 5$$

```
> if (a > b) then a else b end if;
```

$$5$$

```
> 5*(Pi + `if`(a > b,a,b));
```

$$5\,p + 25$$

```
> x := `if`(a < b,NULL,b);
```

$$x :=$$

```
> if FAIL then 3 else 5 end if;
```

$$5$$

Циклы for и while.

Часто требуется циклические повторения выражения заданное число раз или до тех пор, пока выполняется заданное условие. В Maple имеется общая конструкция цикла, которая задается следующим образом.

| for **<name>** | | from **<expr1>** | | by <**expr3** > | | to <**expr2**> | | while **<expr4>** |
do <statement sequence> end do;

Здесь **name** - имя управляющей переменной цикла (счетчик), **expr1, expr2,**и **expr3** - выражения задающие начальное значение, конечное значение и шаг изменения переменной **name**, **expr4** - выражение, задающее условие, пока цикл (**statement sequence** – действия между словами **do** и **od**) будет выполняться.

В ходе выполнения цикла управляющая переменная меняется от начального значения **expr1** до значения **expr2** с шагом, заданным **expr3**. Если блок by <**expr3** > отсутствует, то управляющая переменная будет меняться с шагом, равным +1 при **expr1<expr2**.

Примеры.

1) > for i from 1 to 5 do print (i) od;

1

2

3

4

5

Для того, чтобы пропустить заданное значение управляющей переменной или остановить выполнение цикла после заданного значения управляющей переменной используются оператор пропуска (**next**) и оператор прерывания (**break**). Заметим,что вместе
с признаком конца цикла **od** используется его эквивалент - слово **end do**.

Примеры.

```
> for n from 1 to 3 do
if n=2 then next; end if;
print(n);
end do;
```

1

3

```
> L := [1, 2, "abc", "a", 7.0, infinity]:
for x in L do
if type(x, 'string') then
print(x);
break;
end if;
end do;
```

"abc"

Вычисления и операции с формулами

Вычисление производных.

Для вычисления производных (обычных и частных) используются команды D, d, diff и Diff.

Команда diff имеет следующий вид:

diff(f, x1, ..., xj), diff(f, [x1$n]), diff(f, x1$n, [x2$n, x3], ... xi, [xj$m]).

В инертной форме используется команда Diff.

Примеры.

```
> diff(sin(x),x);
```

$$\cos(x)$$

```
> diff(sin(x),y);
```

$$0$$

```
> diff(sin(x),x$3);
```

$$-\cos(x)$$

```
> diff(x*sin(cos(x)),x);
```

$$\sin(\cos(x)) - x \sin(x) \cos(\cos(x))$$

Вычисление определенных и неопределенных интегралов.

Для вычисления неопределенных интегралов используются функции int(f,x) и Int(f,x) (инертная форма). Для вычисления определенных интегралов используются функции int(f,x=a..b) и Int(f,x) (инертная форма).

Примеры.

```
> int( sin(x), x );
```

$$-\cos(x)$$

```
> int( sin(x), x=0..Pi );
```

$$2$$

```
> int( sin, a..b );
```

$$\cos(a) - \cos(b)$$

```
> int( x/(x^3-1), x );
```

$$-\frac{1}{6}\ln(x^2+x+1)+\frac{1}{3}\sqrt{3}\arctan\left(\frac{(2x+1)\sqrt{3}}{3}\right)+\frac{1}{3}\ln(x-1)$$

```
> int( exp(-x^2), x );
```

$$\frac{1}{2}\sqrt{\pi}\,\mathrm{erf}(x)$$

Разложение функций в ряды.

Для разложения функции в степенной ряд используются следующие функции:

series(expr, eqn) и **series(expr, eqn, n)**, где **expr** – разлагаемое выражение, **eqn** - условие (например, в виде x=a) или имя переменной (например, x) и **n** – необязательный параметр, задающий число членов ряда (при его отсутствии оно равно 6 и может переустанавливаться

системной переменной **order**)/

Примеры.

```
> series(x/(1-x-x^2), x=0);
```

$$x + x^2 + 2x^3 + 3x^4 + 5x^5 + O(x^6)$$

```
> convert(%,polynom);
```

$$5x^5 + 3x^4 + 2x^3 + x^2 + x$$

```
> series(x+1/x, x=1, 3 );
```

$$2 + (x-1)^2 + O((x-1)^3)$$

```
> series(exp(x), x=0, 8 );
```

$$1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3 + \frac{1}{24}x^4 + \frac{1}{120}x^5 + \frac{1}{720}x^6 + \frac{1}{5040}x^7 + O(x^8)$$

Решение систем линейных и нелинейных уравнений.

Для решения систем линейных уравнений существуют мощные матричные методы, которые будут рассмотрены позже. Однако функция **solve** также с успехом справляется с этой задачей. Формат команды **solve** для решения систем уравнений имеет вид:

solve({eq1,…,eqn}, [x1,,...,xn)

Примеры.

```
> solve( {3.2*x + 1.3*y + 4.2*z = 5, 8.7*x + 19*y + 11.2*z = 94, x
+ y/4 + z = 1}, [x, y, z]);
```

$$[[x = 0.4969502408, y = 5.187800963, z = -0.7939004815]]$$

```
> solve( {x + y = 10, x^2 = 9}, [x, y] );
```

$$[[x = 3, y = 7], [x = -3, y = 13]]$$

```
> RootOf(x^2+1=0,x);
```

$$\mathrm{RootOf}(\_Z^2 + 1)$$

```
> allvalues(%);
```

Некоторые действия с полиномами.

Для выделения коэффициентов полиномов используются следующие функции:

coeff(p, x), coeff(p, x, n), coeff(p, x^n), collect(p, x)

Здесь функция **coeff(p, x)** возвращает коэффициент при x в полиноме **p**, **coeff(p, x, n)** – возвращает коэффициент для члена со степенью **n**, **coeff(p, x^n)** –возвращает коэффициенты

при x^n, **collect(p, x)** – возвращает полином, объединяя коэффициенты при степенях x.

Примеры.

```
> p := 2*x^2 + 3*y^3 - 5:
coeff(p,x,2);
```

$$2$$

```
> coeff(p,x^2);
```

$$2$$

```
> coeff(p,x,0);
```

$$3\,y^3 - 5$$

```
> q := 3*a*(x+1)^2 + sin(a)*x^2*y - y^2*x + x - a:
coeff(q,x);
```

$$-y^2 + 6\,a + 1$$

Задача из раздела уравнений математической физики

Метод Даламбера (метод характеристик)

Колебания бесконечной струны. Формула Даламбера

Рассмотрим свободные колебания бесконечной струны. Для этого решается однородное уравнение

$$\frac{\partial^2}{\partial t^2}\mathrm{u}(t,x) = a^2\left(\frac{\partial^2}{\partial x^2}\mathrm{u}(t,x)\right)$$

с начальными условиями

$$\mathrm{u}(0,x) = \mathrm{F}(x),$$
$$\frac{\partial}{\partial t}\mathrm{u}(0,x) = \mathrm{f}(x).$$

```
> restart;
```

Однородное уравнение и его решение:

```
> PDE:=diff(u(t,x),t,t)=a^2*diff(u(t,x),x,x);
pdsolve(PDE);
```

$$PDE := \frac{\partial^2}{\partial t^2}\mathrm{u}(t,x) = a^2\left(\frac{\partial^2}{\partial x^2}\mathrm{u}(t,x)\right)$$

$$\mathrm{u}(t,x) = \_\mathrm{F1}(-x - a\,t) + \_\mathrm{F2}(x - a\,t)$$

Переобозначение решений:

```
> u(t,x):=U1(x-a*t)+U2(x+a*t);
```

$$u(t,x) := U1(x - a\,t) + U2(x + a\,t)$$

Учет начальных условий:

```
> u_0(x):=subs(t=0,u(t,x))-F(x)=0;
ut_0(x):=subs(t=0,diff(u(t,x),t))-f(x)=0;
```

$$u\_0(x) := U1(x) + U2(x) - F(x) = 0$$

$$ut\_0(x) := -D(U1)(x)\,a + D(U2)(x)\,a - f(x) = 0$$

```
> int(diff(-a*U1(xi),xi)+diff(a*U2(xi),xi)-f(xi),xi=0..x);
```

$$\int_0^x -a\left(\frac{d}{d\xi}U1(\xi)\right) + a\left(\frac{d}{d\xi}U2(\xi)\right) - f(\xi)\,d\xi$$

```
> -a*(U1(x)-U2(x))+a*(U1(0)-U2(0))=int(f(xi),xi=0..x);
```

$$-a\,(U1(x) - U2(x)) + a\,(U1(0) - U2(0)) = \int_0^x f(\xi)\,d\xi$$

```
> -a*(U1(x)-U2(x))=int(f(xi),xi=0..x)+a*C;
```

$$-a\,(U1(x) - U2(x)) = \int_0^x f(\xi)\,d\xi + a\,C$$

Выражение начальных функций:

```
> solve({U1(x)+U2(x)-F(x)=0,-a*(U1(x)-U2(x)) = int(f(xi),xi = 0
.. x)+a*C}, {U1(x),U2(x)});
```

$$\left\{ U2(x) = \frac{1}{2}\,\frac{a\,C + a\,F(x) + \int_0^x f(\xi)\,d\xi}{a},\ U1(x) = \frac{1}{2}\,\frac{-a\,C + a\,F(x) - \int_0^x f(\xi)\,d\xi}{a} \right\}$$

```
> U1(x):=1/2*(a*F(x)-int(f(xi),xi = 0 .. x)-a*C)/a;
U2(x):=1/2*(a*F(x)+int(f(xi),xi = 0 .. x)+a*C)/a;
```

$$U1(x) := \frac{1}{2}\,\frac{-a\,C + a\,F(x) - \int_0^x f(\xi)\,d\xi}{a}$$

$$U2(x) := \frac{1}{2}\,\frac{a\,C + a\,F(x) + \int_0^x f(\xi)\,d\xi}{a}$$

Представление общего решения:

```
> u(t,x):=simplify(subs(x=x-a*t,U1(x))+(subs(x=x+a*t,U2(x))));
```

$$u(t,x) := \frac{1}{2}\,\frac{a\,F(x - a\,t) - \int_0^{x - a\,t} f(\xi)\,d\xi + a\,F(x + a\,t) + \int_0^{x + a\,t} f(\xi)\,d\xi}{a}$$

```
> u(t,x):=collect(1/2*(a*F(x-a*t)-int(f(xi),xi = 0 .. x-
a*t)+a*F(x+a*t)+int(f(xi),xi = 0 .. x+a*t))/a, a);
```

$$u(t,x) := \frac{1}{2}F(x-at) + \frac{1}{2}F(x+at) + \frac{-\frac{1}{2}\int_0^{x-at} f(\xi)\,d\xi + \frac{1}{2}\int_0^{x+at} f(\xi)\,d\xi}{a}$$

Формула Даламбера:

```
> u(t,x):=1/2*F(x-a*t)+1/2*F(x+a*t)+1/2*int(f(xi),xi = x-a*t ..
x+a*t)/a;
```

$$u(t,x) := \frac{1}{2}F(x-at) + \frac{1}{2}F(x+at) + \frac{1}{2}\left(\frac{1}{a}\int_{x-at}^{x+at} f(\xi)\,d\xi\right)$$

Убедимся, что полученная функция $u(t,x)$ удовлетворяет уравнению и начальным условиям.

Для этого подставим $u(t,x)$ в уравнение и начальные условия:

```
> simplify(diff(u(t,x),`$`(t,2)) - a^2*diff(u(t,x),`$`(x,2)));
eval(subs(t=0,u(t,x)));
subs(t=0,diff(u(t,x),t));
```

$$0$$

$$F(x)$$

$$f(x)$$

Построение графиков

Двумерная графика.

Для построения двумерного графика используется функция plot. Она задается в виде:
Plot(f, h, v), Plot(f, h, v, o),

где f - функция, h - переменная с указанием области изменения, v – необязательная переменная с указанием области изменения,o - параметр или набор параметров, задающих стиль построения графика (толщина и цвет кривой, тип кривых и т.д.).

Таблица параметров функции plot

ПАРАМЕТРЫ	ЗНАЧЕНИЯ	ОПИСАНИЕ
adaptive	true, false	использование адаптивного алгоритма построения
axes	FRAME, BOXED, NORMAL, NONE	тип координатных осей
axesfont	[family, style, sizel	шрифт для осей
color	зарезервированное слово или процедура	цвет графика
coords	имя системы координат	тип системы координат
discont	true, false	для построения выражений с разрывами
font	[family, style, size]	шрифт для текста
labelfont	[family, style, size]	шрифт для меток осей
labels	[strl, str2l	названия осей
linestyle	целое число	тип линии

numpoints	целое число	точки по оси абсцисс
resolution	целое число	горизонтальное разрешение устройства вывода
sample	[x1,x2,...xn]	список точек, в которых будет построена функция (adaptive=false)
scaling	CONSTRAINED, UNCONSTRAINED	масштабирование
style	POINT, LINE,	тип интерполяции
symbol	BOX, CROSS, CIRCLE, POINT, DIAMOND	символ точек чертежа
thickness	0, 1, 2, 3	толщина линий
title	строка	заголовок чертежа
titlefont	[family, style, size]	шрифт для заголовка
view	[x1..x2, y1..y2]	окно координатной плоскости
xtickmarks, ytickmarks	целое число	количество отметок на осях X и Y

С помощью команды plot можно строить помимо графиков функций y=f(x), заданной явно, также графики функций, заданных параметрически y=y(t), x=x(t), если записать команду plot([y=y(t), x=x(t), t=a..b], parameters).

Примеры.

> **plot(cos(x/2) + sin(2*x), x = 0..4*Pi);**

```
> plot([sin(x), x-x^3/6], x=0..2, color=[red,blue],
style=[point,line]);
```

```
> s := t->100/(100+(t-Pi/2)^8): r := t -> s(t)*(2-sin(7*t)-
cos(30*t)/2):
plot([r(t),t,t=-
Pi/2..3/2*Pi],numpoints=2000,coords=polar,axes=none);
```

Трехмерные графики. Анимация

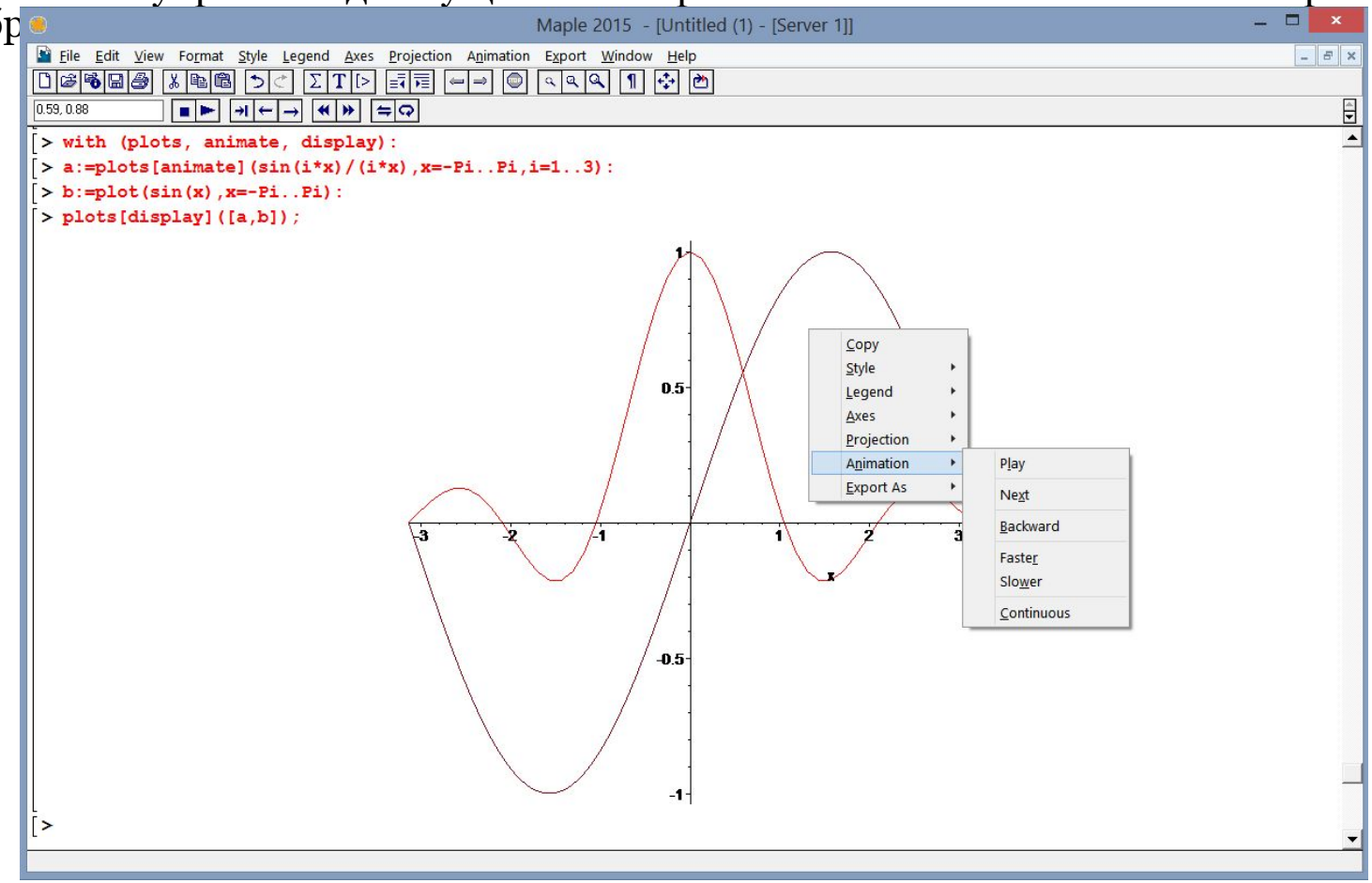
Для построения графиков 2-мерных поверхностей (3-мерный график) используется функция plot 3d.

Таблица параметров функции plot3d

ПАРАМЕТРЫ	ЗНАЧЕНИЯ	ОПИСАНИЕ
ambientlight	[red, green, blue]	освещение рассеянным светом
axes	NORMAL, BOXED, FRAME, NONE	тип осей координат
axesfont	[family, style, size]	шрифт для осей
contours	целое число или список целых чисел	число контуров
coords	имя системы координат	тип системы координат
font	[family, style, size]	шрифт для текста
grid	[n, m], n, m-целые	количество ячеек сетки
gridstyle	rectangular или triangular	тип сетки

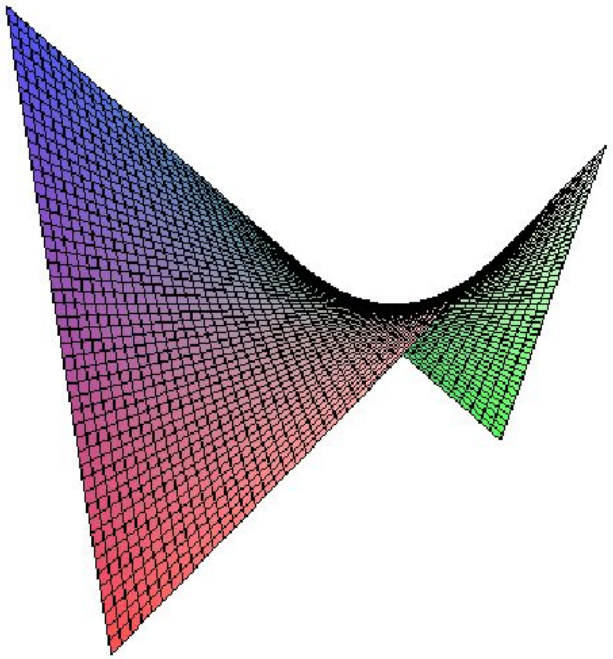
labelfont	[family, style, size]	шрифт для меток осей
labels	[str1, str2]	названия осей
light	[phi, theta, red, green, blue]	источник света
lightmodel	none, light 1, light2, light3, light4	модели освещения
linestyle	целое число	тип линий
orientation	[theta, phi]	точка взгляда в сферических координатах
projection	0<x<1 или FISHEYE, NORMAL, ORTHOGONAL	точка взгляда
shading	XYZ, XY, Z, Z GREYSCALE, Z HUE, NONE	наложение теней
style	POINT, HIDDEN, PATCH, WIREFRAME, CONTOUR, PATCHNOGRID, PATCHCONTOUR, LINE	стиль рисования поверхности
symbol	BOX, CROSS, CIRCLE, POINT, DIAMOND	символ для точек чертежа
thickness	0, 1, 2, 3	толщина линий
title	строка	заголовок
titlefont	[family, style, size]	шрифт для заголовка
view	z1..z2 или [x1..x2, y1..y2, z1..z2]	окно пространства

- Анимация. Maple позволяет выводить на экран движущиеся изображения с помощью команд animate (двумерные) и animate3d (трехмерные) из пакета plot. Среди параметров команды animate3d есть frames – число кадров анимации (по умолчанию frames=8). Трехмерные изображения удобнее настраивать не при помощи опций команды plot3d, а используя контекстное меню программы. Для этого следует щелкнуть правой кнопкой мыши по изображению. Тогда появится контекстное меню настройки изображения. Команды этого меню позволяют изменять цвет изображения, режимы подсветки, устанавливать нужный тип осей, тип линий и управлять движущимся изображением. Контекстное меню настройки изобр

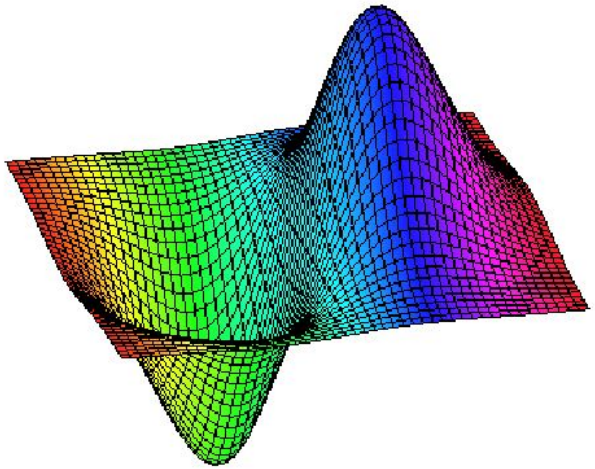


Примеры.

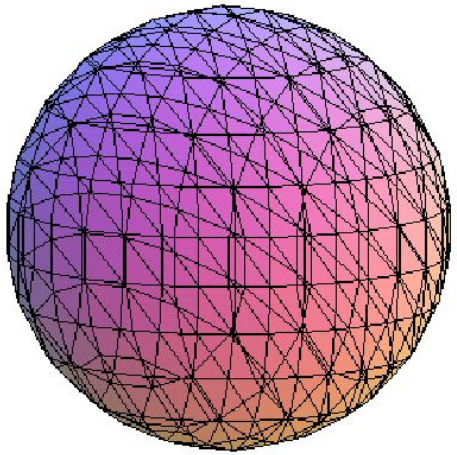
> **plot3d(x*y, x=-Pi..Pi, y=-Pi..Pi);**



```
> plot3d(x*exp(-x^2-y^2), x=-2..2, y=-2..2, color=x);
```



```
> with(plots): implicitplot3d(x^2+y^2+z^2=4,
x=-2..2, y=-2..2, z=-2..2, scaling=CONSTRAINED);
```

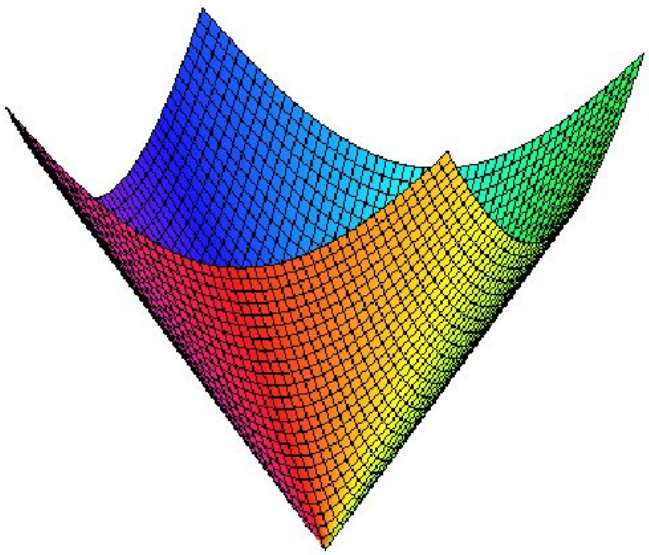


Библиотека plots

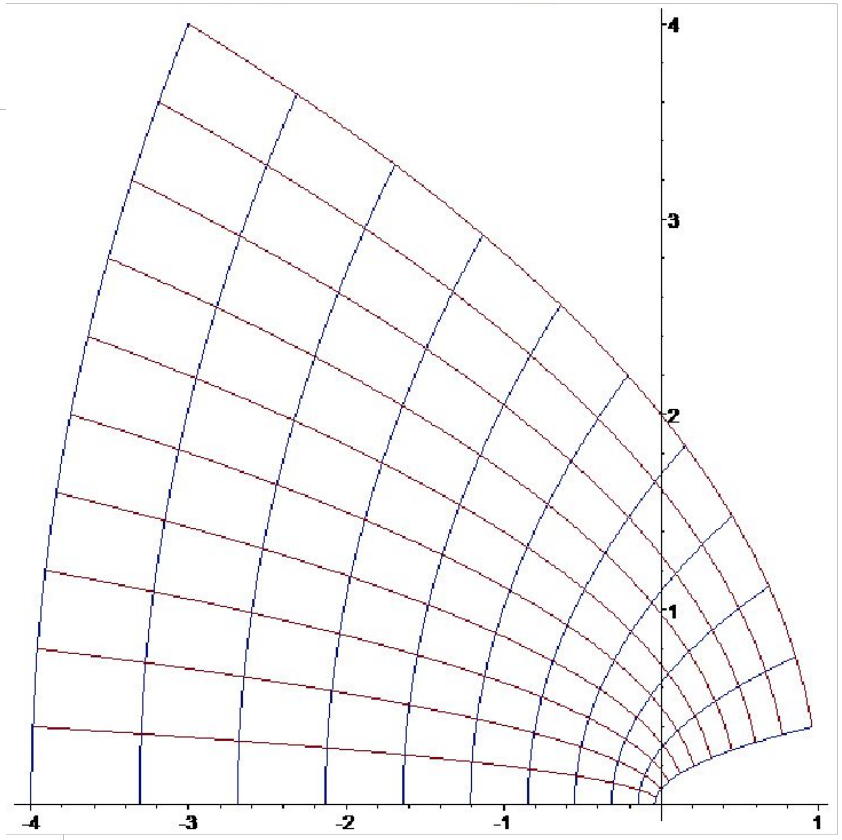
- Кроме основных функций построения графиков plot и plot3d Maple содержит две библиотеки, значительно расширяющие графические возможности пакета. Библиотека plots - позволяет строить разнообразные двухмерные и трехмерные графики для различных математических объектов. Библиотека plottools - содержит функции для построения различных графических объектов (примитивов). Библиотека plots содержит 48 функций для построения различных типов графиков . При таком количестве функций подробное рассмотрение формата каждой функции займет достаточно много места, поэтому ограничимся кратким описанием каждой команды и приведем небольшие примеры, иллюстрирующие работу команд. Заметим, что для использования функций, содержащихся в библиотеки plots, применимы обычные приемы подключения библиотеки целиком или отдельных функций при помощи оператора with.

Примеры:

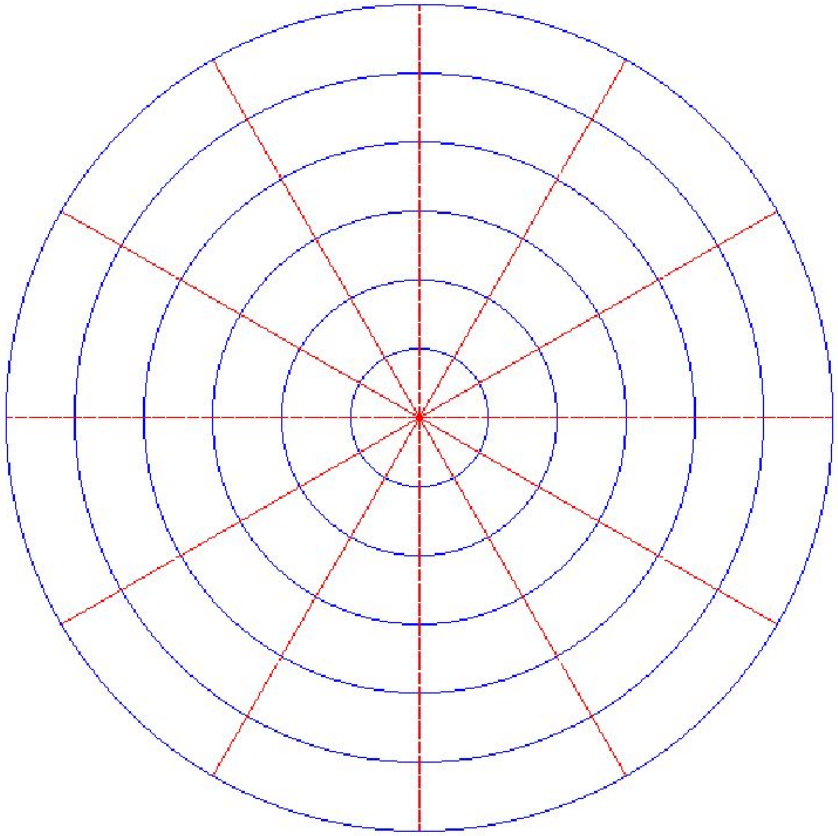
```
> complexplot3d( (z) , z = -2 - 2*I .. 2 + 2*I
);
```



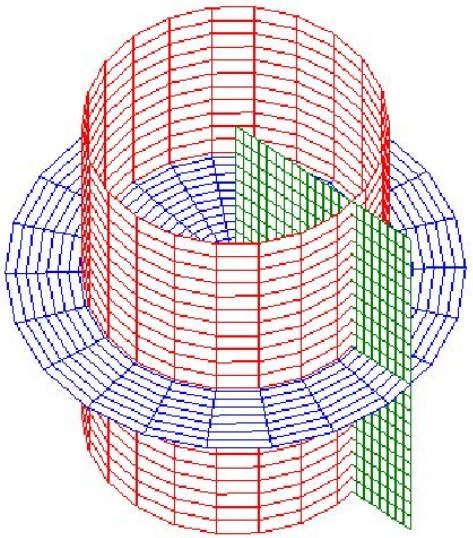
```
> conformal(z^2,z=.1.2*I..1+2*I);
```



> coordplot(polar);



> coordplot3d(cylindrical);



Сравнение с Mathematica

Примеры:

Maple:

```
> x:= 3;
```

$$3$$

Mathematica:

```
In[9]:=
        x = 3
Out[9]= 3
```

Maple:

```
> 1+sum(binomial(n,k),k=1..n);
```

$$2^n$$

Mathematica:

```
In[1]:=
        1 + Sum[Binomial[n, k], {k, 1, n}]
Out[1]= 2^n
```

Maple:

```
> sin(x^2+2*x);
```

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

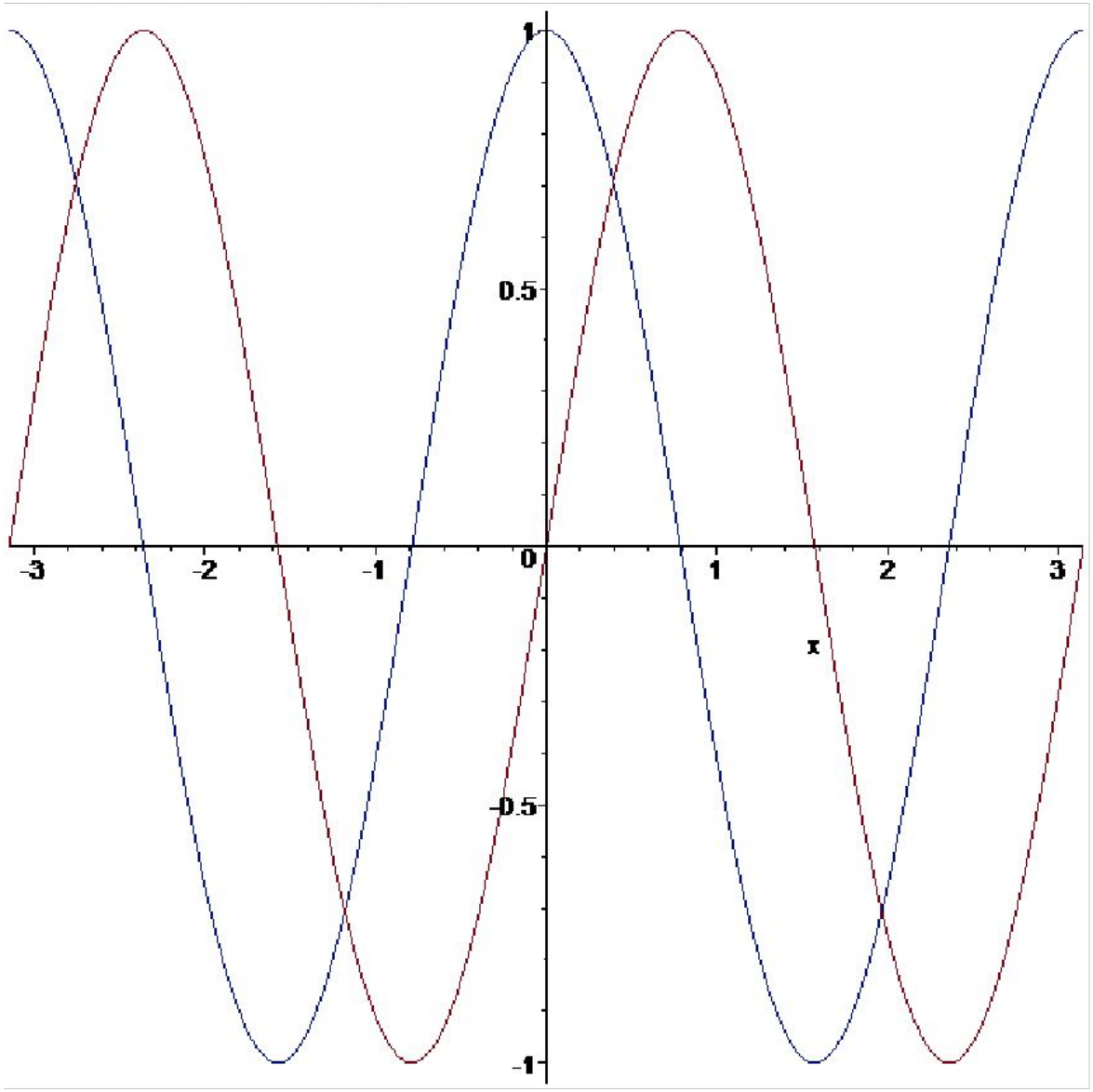
Mathematica:

```
In[20]:=
sin[X^2 + 2 * X]
```

Out[20]= $\sin\left[2X + X^2\right]$

Maple:

```
> plot([ sin(2*x), cos(2*x)], x=-Pi..Pi );
```

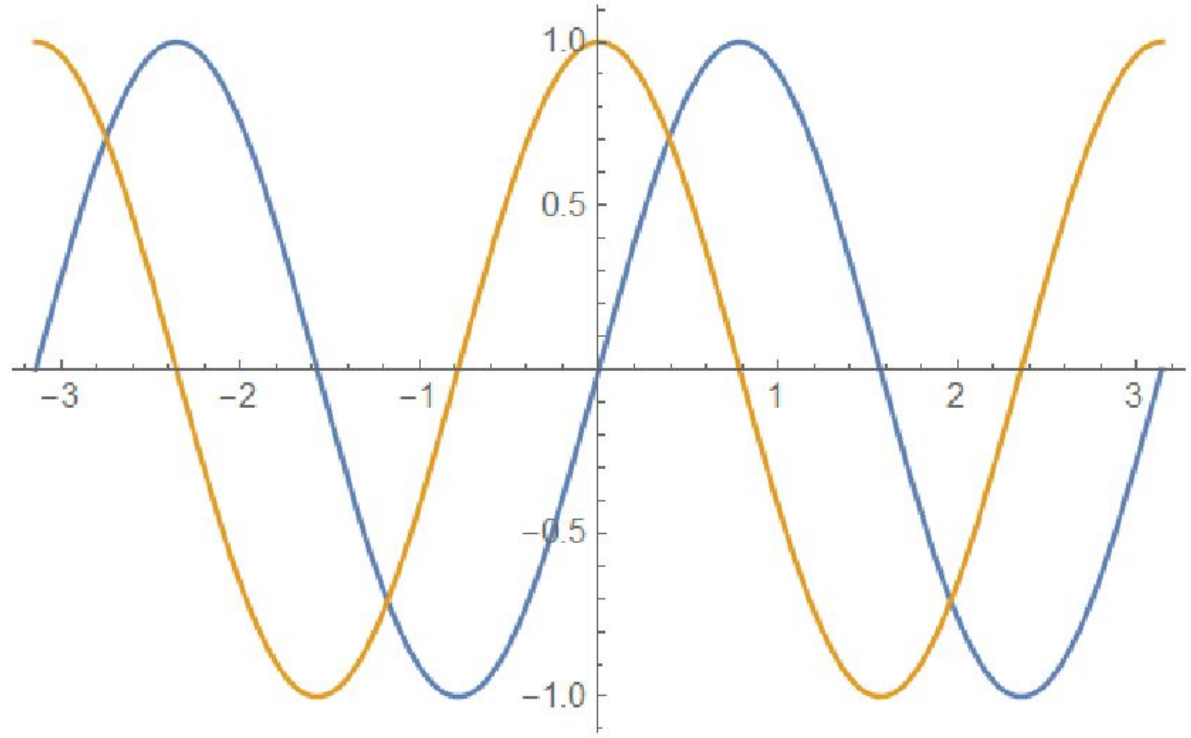


Mathematica:

In[23]:=

```
Plot[{Sin[2 * x], Cos[2 * x]}, {x, -Pi, Pi}]
```

Out[23]=



Maple	VS	Mathematica
$\sin(2x)$	See notes 1 and 2.	`Sin[2 x]`
$5x - 7 = 3x + 2$	See notes 3.	`5 x - 7 == 3 x + 2`
$2x^2 + \cos\left(\frac{x}{2}\right)$	See notes 4.	`2 x^2 + Cos[x / 2]`
$\lim_{x \to 0} \frac{\sin(x)}{x}$	See notes 5.	`Limit[Sin[x] / x, x → 0]`

Maple	VS	Mathematica
$\sum_{i=1}^{3} a_i = a_1 + a_2 + a_3$		In[2]:= $\sum_{i=1}^{3} a_i$ Out[2]= $a_1 + a_2 + a_3$
$\prod_{i=m}^{n} x_i$		$\prod_{i=m}^{n} x_i$
$\frac{d}{dx}\sqrt{x} = \frac{1}{2\sqrt{x}}$		In[3]:= $\partial_x \sqrt{x}$ Out[3]= $\frac{1}{2\sqrt{x}}$
$\int_a^b f(x)\,dx$		$\int_a^b f(x)\,dx$

Maple

```
hailstone := proc( N )
    local n := N, HS := Array();
    HS(1) := n;
    while n > 1 do
        if type(n,even) then
            n := n/2;
        else
            n := 3*n+1;
        end if;
        HS(numelems(HS)+1) := n;
    end do;
    HS;
end proc;
```

VS

Mathematica

```
HailstoneFP[n_] :=
  Drop[
    FixedPointList[
      If[# != 1,
        Which[Mod[#, 2] == 0,
          #/2,
          True,
          ( 3*# + 1)
        ],
        1
      ] &, n
    ], -1
  ]
```

Сравниваемый пакет/ Объект сравнения	**Пакет Mathematica**			**Пакет Maple**	
Циклические операции общий вид	Формат команды *While[test, body]*	Формат использования цикла с заголовком For: *For[start, test, incr, body].*	Формат команды: *Do[expr, {i, imin, imax, di}]*	Maple имеет обобщённую конструкцию цикла, которая задаётся следующим образом: *|for<name>| |from<expr1>| |to<expr3>| |by<expr2>| |while<expr4>| do<Statement sequence>od;*	Есть ещё одна, более специфическая конструкция цикла: *|for<name>| |in<expr1>| |while<expr2>| do<Ststement sequence>od:*
Объяснение циклической операции	Подсчитываются повторно условие test и тело цикла body до тех пор, пока истинность условия не нарушится.	Данный цикл вычисляет start, потом, повторяясь, вычисляет incr и body, пока test не примет значение False.	Подсчитывается выражение expr для значений i, принимающих значения от imin до imax с шагом di.	Здесь name – имя управляющей переменной цикла, expr1, expr2, expr3 – выражения, задающие начальное значение, конечное значение и шаг изменения переменной name, expr4 – выражение, задающее условие, пока цикл (набор объектов между словами do и od), будет выполняться. В ходе выполнения цикла управляющая переменная меняется от значения expr1 до значения expr2 с шагом, заданным expr3. Если блок by<expr2> отсутствует, то управляющая переменная будет меняться с шагом +1 при expr1>expr2.	Здесь expr1 задает список значений, которые будет принимать управляющая переменная name. Цикл будет выполняться, пока не будет исчерпан список и пока выполняется условие, заданное выражением expr2. В цикле этого вида управляющая переменная может меняться произвольно.

Директивы прерывания и продолжения циклов	*Abort[]* — вызывает прекращение вычислений с сообщением $Aborted. Команда может быть использована в любом месте программы. *Break[]* — выполняет экстренный выход из тела цикла или уровня вложенности программы, содержащего данный оператор (возвращает Null — значение без генерации секции выхода). *Continue[]* — переход на следующий шаг ближайшего содержащего эту функцию оператора Do, For или While. *Interrupt[]* — приостанавливает вычисления, делая запрос об их возобновлении. *Return[]* — прерывает выполнение с возвращением значения Null. *Return[expr]* — возвращает значение выражения expr, выходя из всех процедур и циклов.	*Next[]* – позволяет пропустить определённый цикл *Break[]* – прерывает выполнение фрагмента программы (или цикла), как только он встречается в ходе её выполнения. *Return[]* – позволяет возвратить значение последнего выражения в тело процедуры или выражения. Любой из операторов *Quit[], Done[], Stop[]* обеспечивает также прерывание выполнения текущей программы (в частности, цикла), но при этом окно текущего документа закрывается.
Подведение итогов сравнения.	*При сравнении языков программирования данных пакетов можно выделить явные отличие:* 1. Можно увидеть явное отличие используемых директив. А именно довольно различный набор функций для продолжения или прерывания цикла. Но имеются и директивы, встречающиеся в обоих пакетах: Break[], Return[]. 2. Также в отличие от Maple Mathematica не имеет циклических конструкций, позволяющих производить циклические вычисления по подвыражениям заданного выражения. Maple в свою циклическую конструкцию вобрал основные конструкции циклов for и while.Тем самым обеспечивая для себя возможность на основе подвыражений создавать интересные конструкции. *В остальном использование циклических операций и директив прерывания и продолжения циклов этих двух математических пакетов весьма схоже.*	

ЗАКЛЮЧЕНИЕ

- С помощью программных пакетов можно сэкономить массу времени и избежать многих ошибок при математических вычислениях. Естественно, системы не ограничиваются только этими возможностями. Отметим, что спектр задач, решаемых подобными системами, очень широк:
- проведение математических исследований, требующих вычислений и аналитических выкладок;
- разработка и анализ алгоритмов;
- математическое моделирование и компьютерный эксперимент;
- анализ и обработка данных;
- визуализация, научная и инженерная графика;
- разработка графических и расчетных приложений.

Спасибо за внимание!