***Тема 3.4. Символьные вычисления в среде MuPAD***

**3.4.1. Элементы основного окна MuPAD**

**3.4.2. Основные классы задач, решаемые в среде MuPAD**

**3.4.2.1. Описание функций**

**3.4.2.2. Вычисление производных**

**3.4.2.3. Вычисление интегралов**

**3.4.2.4. Решение алгебраических уравнений и неравенств**

**3.4.2.5. Решение дифференциальных уравнений**

##### 3.4.1. Элементы основного окна MuPAD

В конфигурацию MATLAB входит модуль MuPAD, позволяющий получить доступ к алгебраическим вычислениям и выполнить символьные преобразования.

Для его запуска достаточно в командном окне набрать ***mupad***, вследствие чего открывается окно MuPAD (рис. 3.4-1).

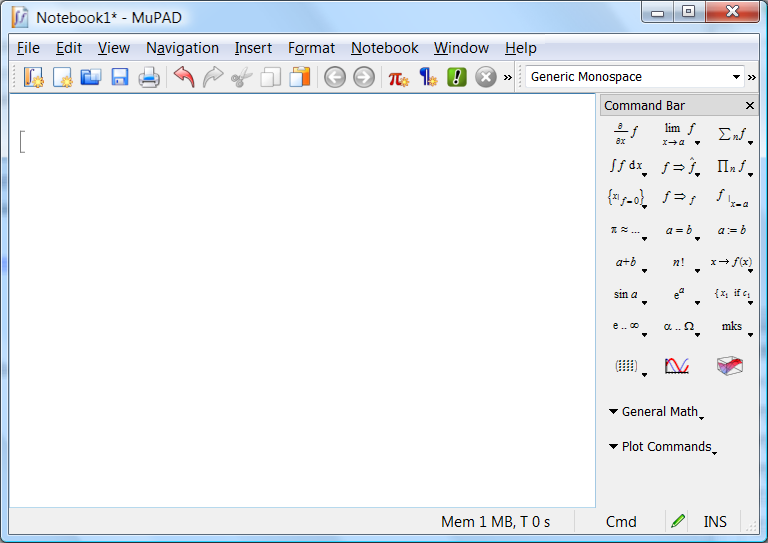


Рис. 3.4-1 Окно редактора среды MuPAD с новым пустым документом

В верхней части окна находится меню и панель инструментов, а в правой колонке палитры соответствующих математических функций и команд. Основная белая область окна - это, то рабочее пространство, в котором выполняются вычисления. Здесь вводятся команды, и отображается результат их выполнения. Подсказкой места ввода служит мигающий курсор и специальная метка в виде открытой квадратной скобки.

Ввод команд осуществляется в рабочей области, где мигает курсор (область ввода). Для ее выполнения следует нажать клавишу <Enter>. После чего, в области вывода, отображается результат.

***Чтобы результат не отображался в области вывода, в конце командной строки следует поставить двоеточие***.

***По умолчанию в MuPAD, команды ввода отображаются красным цветом, а результат – синим. Если вводится текст, то он отображается черным цветом***(рис. 2).

При записи математического выражения в MuPAD используются следующие основные операторы: сложение (+); вычитание (-); умножение (\*); деление (/); возведение в степень (^) и др.

Оператором присваивания является комбинация клавиш двоеточия и знака равенства (:=). Константа PI означает число π, мнимая единица обозначается как I.

Следует помнить, что по умолчанию все операции в MuPAD выполняются в символьном виде, то есть, например, при вычислении значения sin(PI/4) возвращается символьный результат вида , а приближенное значение числа. Однако если надо получить результат в форме числа с плавающей точкой, достаточно воспользоваться встроенной функцией ***float().***

Количество значащих цифр в результате, отображаемом в формате с плавающей точкой, можно указать, задав системной переменной DIGITS новое значение, например, ***DIGITS*** =30 (по умолчанию оно равно 10) . Чтобы вернуть настройку по умолчанию надо ввести команду - ***delete DIGITS***. Команда delete используется точно также и для «сброса» значений символьных переменных.

Ввод математических выражений в значительной степени облегчается за счет использованием элементов палитры, расположенной в правой части окна. Каждый из элементов палитры раскрывается щелчком по стрелке, и появляется многочисленные шаблоны, каждый из которых щелчком по его изображению переносится в место расположения курсора. При этом места заполнения отмечены символом решетка (#). Например, если щелкнуть по изображению действия a+b, то в окне отображается #a+#b, то есть a и b нужно заменить на необходимые переменные. На рис. 3.4-2 приведены элементарные вычисления, выполненные с использованием шаблонов палитры, расположенной в правой части окна.

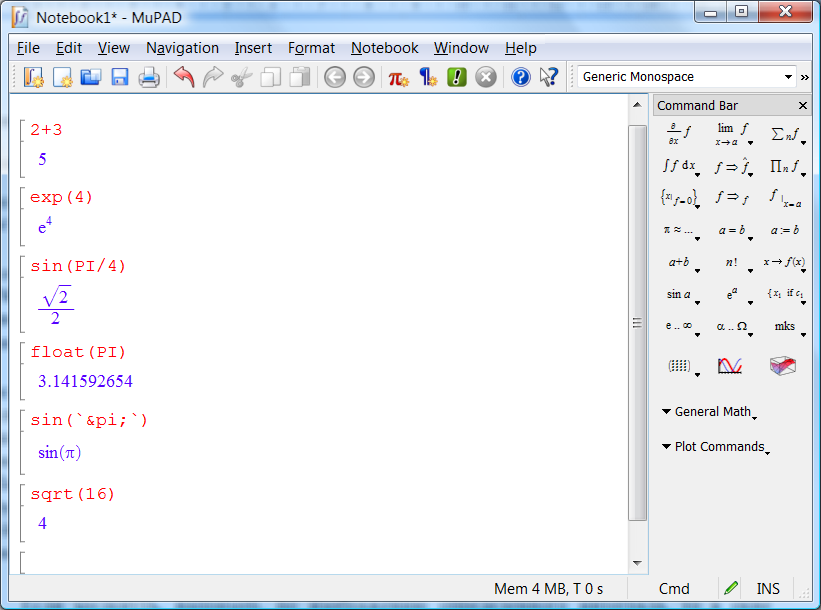


Рис. 3.4-2. Элементарные вычисления, выполненные с использованием шаблонов

Если щелкнуть, например, по изображению определенного интеграла, то в окне появляется шаблон команды вычисления определенного интеграла, где места ввода подынтегральной функции, аргумента и пределов интегрирования также отмечены символом # (рис. 3.4-3).

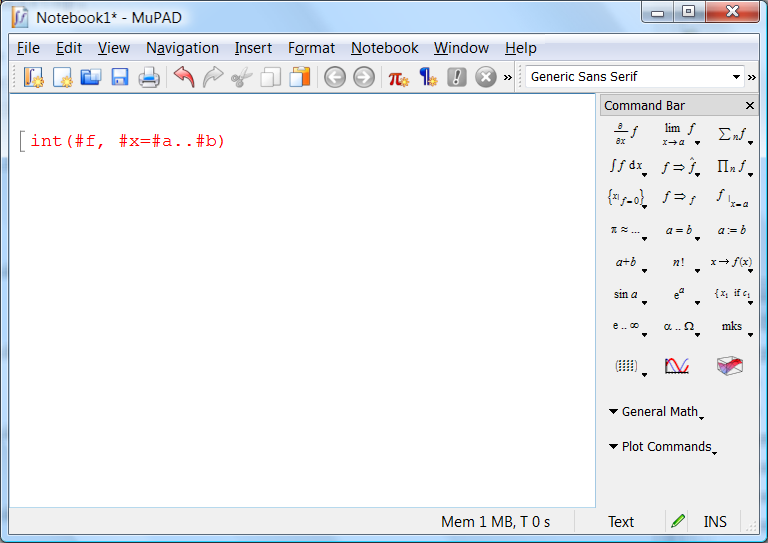


Рис.3.4-3 Ввод команды с использованием палитры шаблонов

***Использование палитры шаблонов в процессе вычислений не только ускоряет ввод командных строк, но и позволяет избежать многочисленных ошибок, связанных с неправильным вводом имени функции или команды.***

Еще одним достоинством MuPAD является возможность быстрой и качественной графики и анимации. На рис. 3.4-4 представлены графики функций от одной и двух переменных, созданные с использованием команд ***plot().***

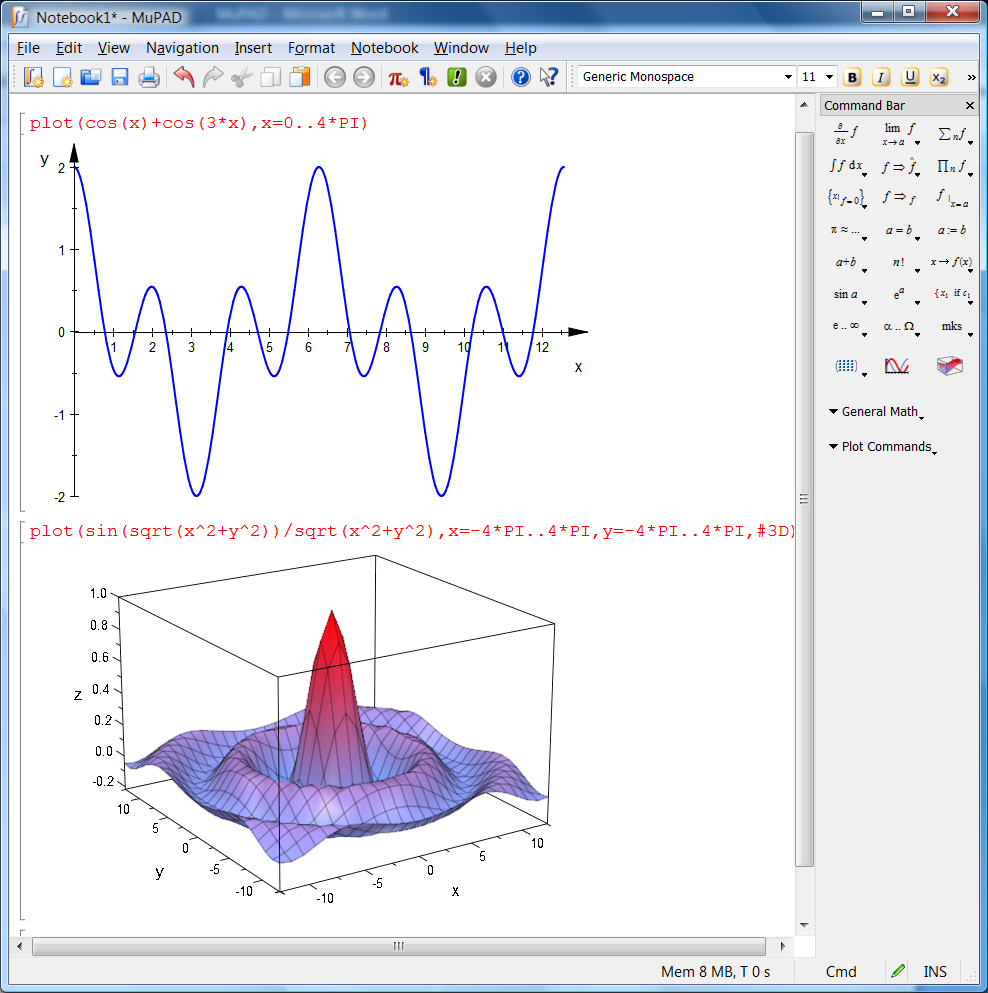


Рис. 3.4-4 Создание графиков с использованием команды plot().

Шаблон команды plot() появляется в месте ввода после щелчка по соответствующему изображению графика на палитре, расположенной в правой части окна. Основу команды при построении двумерного графика составляет функция plot(), имеющая два аргумента: первый - собственно выражение математической зависимости, а второй - независимая переменная и диапазон ее значений. Диапазон указывается по правилам: переменная, знак равенства и границы изменения переменной, разделенные двумя подряд идущими точками.

Для создания трехмерного графика шаблон команды также используется команда plot(), но после ввода выражения последовательно задаются диапазоны изменения каждой двух переменных, а в конце - опция #3D, что говорит о том, сто график трехмерный.

##### 3.4.2. Основные классы задач, решаемые в среде MuPAD

**3.4.2.1. Описание функций**

В среде MuPAD функции можно задавать несколькими способами. Рассмотрим самый простой - с помощью оператора «стрелка» (оператор ->).

При описании функции соблюдается следующая последовательность ввода:

* имя функции;
* оператор присваивания (:=);
* список аргументов (разделенных запятыми);
* оператор «стрелка» (->);
* выражение, в которое входят аргументы, представляющее собой функциональную зависимость.

На рис. 5 приведено описание и получение числового значения функции при х=2. Здесь y определяется как оператор, который действуя на переменную х, вычисляет значение x^2, а переменная х играет роль формального параметра. Как показано на рис. 3.4-5 функцию, объявленную таким способом, можно использовать при расчетах в разных выражениях, а также при построении графиков.

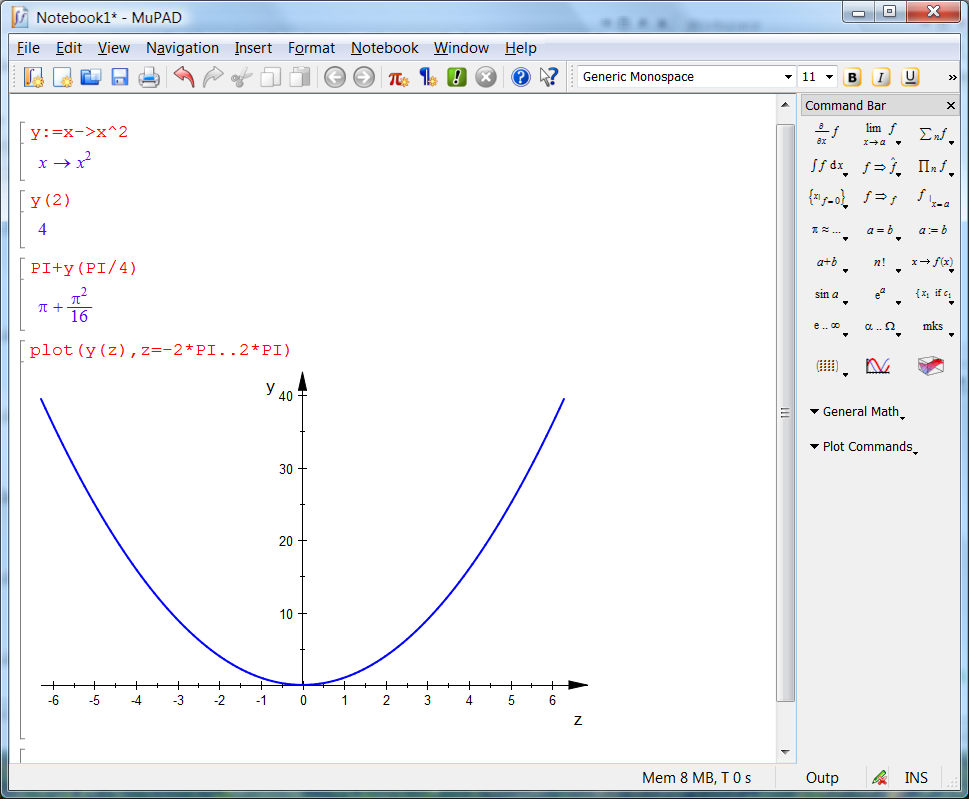
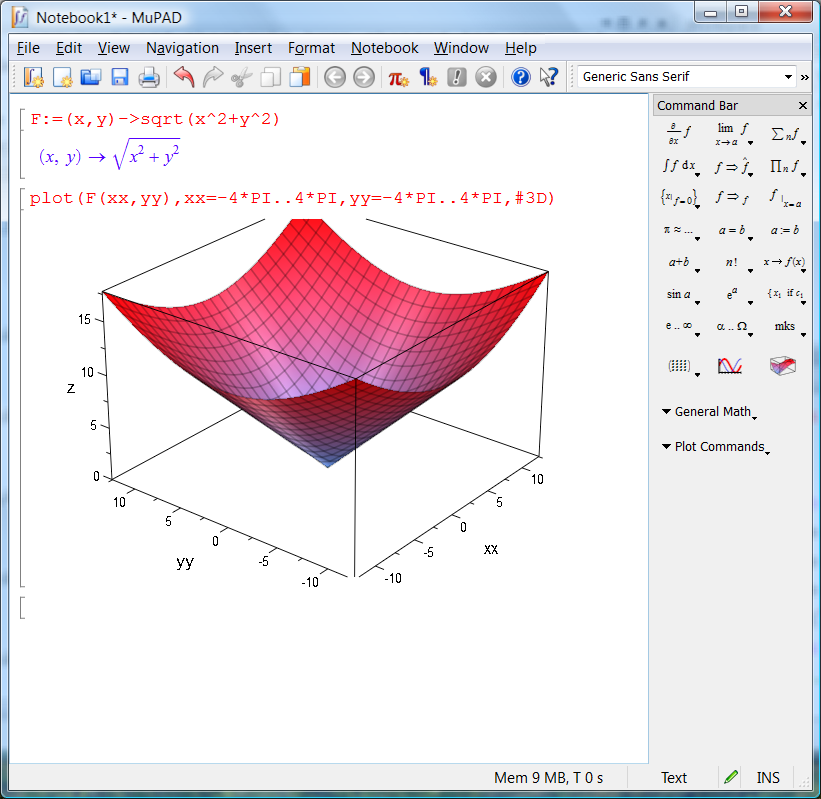


Рис. 3.4-5 Описание функции от одной переменной

При объявлении функции от двух переменных, как показано на рис. 3.4-6, аргументы вводятся через запятую, и заключены в скобки. Аналогично предыдущему примеру, после нажатия клавиши <Enter> в следующей строке появляется символьное выражение функциональной зависимости. Ниже показана возможность использования объявленной двумерной функции при построении графика. Использование в команде plot() вместо переменных х и y переменные хх и yy, показывает, что аргументы при описании функции также играют роль формальных параметров.

  Рис. 3.4-6. Описание функции от двух переменных

**3.4.2.2. Вычисление производных**

Производные в MuPAD, также как и в MatLAB, вычисляются с помощью функции ***diff()***. Дифференцируемое выражение указывается первым аргументом функции, а переменная, по которой происходит дифференцирование – вторым. Функцию ***diff()*** можно ввести с клавиатуры, а можно воспользоваться значком, изображающим производную.

Дифференцируемое выражение может быть непосредственно указано в функции diff(), как например, в команде ***diff(x^2\*sin(x), x)***, или объявлено перед использованием функции. После нажатия клавиши <Enter> в следующей строке появляется символьное выражение производной (рис.3.4-7).

Функция ***diff()*** вычисляются производные и высших порядков. В общем случае переменные, по которым вычисляются производные от выражения, являющимся первым аргументом, указываются через запятую в соответствии с порядком вычисления производной. В нашем случае переменная дифференцирования одна, следовательно, для получения второй производной, через запятую указывается та же переменная ***х***, а для третьей производной – еще один ***х***.

Следует отметить, что результат вычисления производной очень часто представлен в неупрощенном виде. Для упрощения выражения используется функция ***simplify()***, аргументом которой служит упрощаемое выражение. Если, как в нашем случае, упрощения требует результат предыдущей команды, то в качестве ссылки используется инструкция % (рис. 3.4-7).

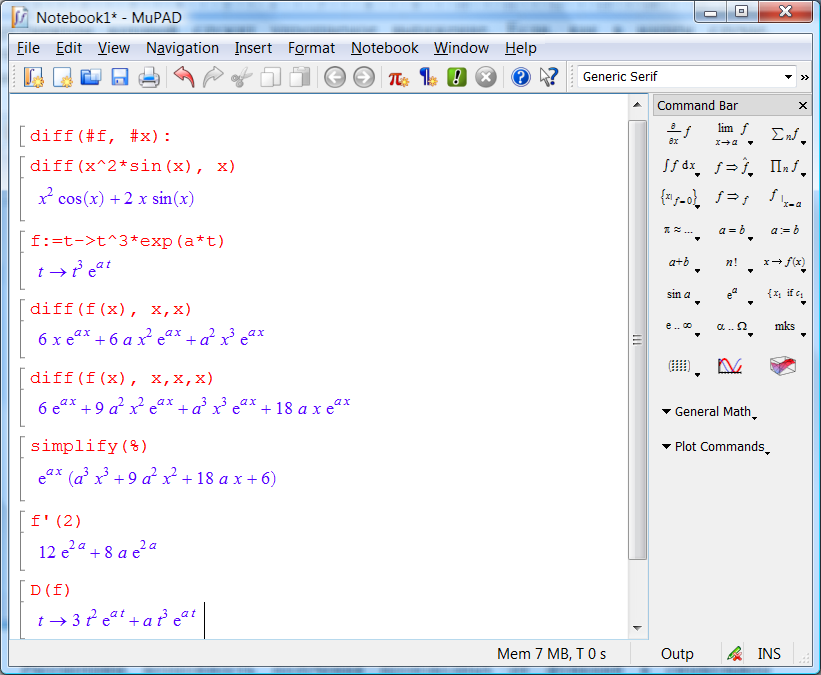


Рис.3.4-7. Вычисление производных

При работе с функцией от одной переменной производную можно вычислить также с помощью оператора производной – штриха (знак апостроф ‘) или с помощью функции ***D().*** Примеры использования этих операторов приведены в нижней части рис.7. Чтобы с использованием оператора штрих вычислить производную высшего порядка оператор штрих указывается несколько раз (например, f’’(t) – производная 2-го порядка). При вычислении производная 2-го порядка с использованием функции ***D()***, следует ввести ***D(@@2) (f) (t)***. Если вместо переменной t задать числовое значение (например, ***D(@@2)(f) (5),*** то вычисляется значение производная 2-го порядка при значении t=2.

Достаточно просто в MuPAD вычисляются и частные производные. В этом случае применяются функция ***diff()*** или ***D()***.

На рис.3.4-8 показан процесс вычисления частных производных от функции . Функция описана с использованием оператора «стрелка» и получено ее символьное выражение, а затем приведены примеры вычисления частных производных по двум аргументам с использованием функции ***diff()*** и частных производных по первой переменной и по второй переменной с использованием функции ***D().***

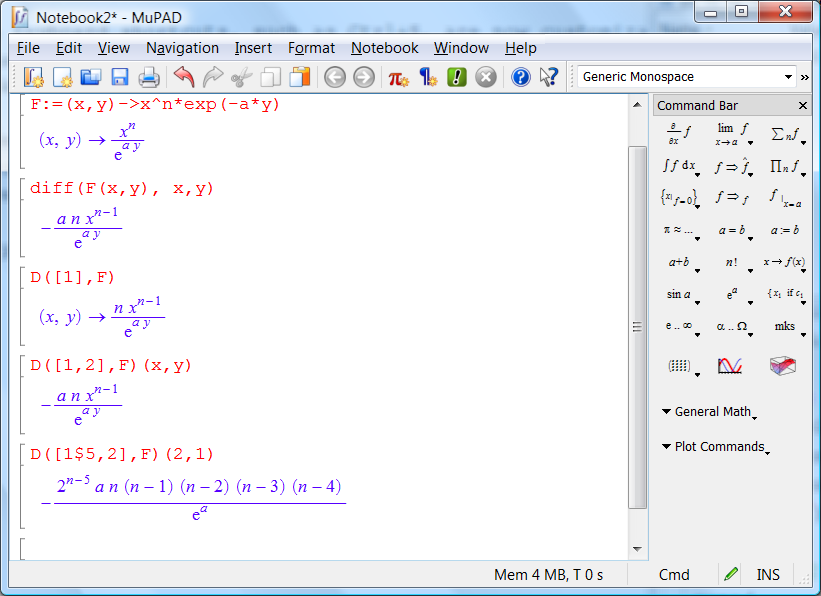


Рис.3.4-8. Вычисление производных

* + - 1. **Вычисление интегралов**

Интегралы в MuPAD вычисляются с использованием функции ***int()***. Первым аргументом указывается подынтегральное выражение, а вторым – переменная интегрирования. На рис.9 приведены примеры вычисления неопределенных и определенных интегралов (ключевым словом ***infinity*** обозначается бесконечность), получение интеграла в аналитическом аналитического выражения и его числового значения, для чего функцию ***int()*** пришлось дополнить внешней функцией ***float().***

К сожалению не любой интеграл может быть представлен в аналитическом виде, как показано на рис.3.4-9. В таком случае можно прибегнуть к вычислению интеграла в числовом виде. При использовании команды **float(int(exp(-x)\*cos(5\*x),x=0..infinity))** вычислить интеграл в символьном виде не удается, поэтому на экран выдается его числовое значение. Команда **numeric::int(sin(2\*PI\*x)/sqrt(1+x),x=0..1)** сразу использует числовой алгоритм.

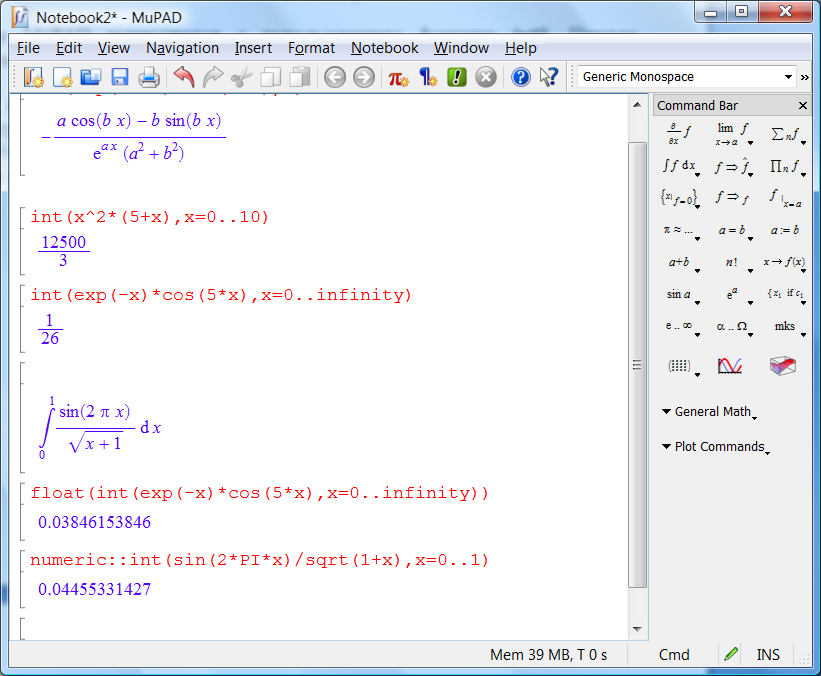


Рис. 3.4-9. Вычисление интегралов

* + - 1. **Решение алгебраических уравнений и неравенств**

Для решения **алгебраических уравнений**, систем алгебраических уравнений или неравенств используется функция ***solve()***.

При решении алгебраических уравнений в качестве первого аргумента выступает само уравнение и вторым аргументом является переменная, относительно которой оно решается. На рис. 3.4-10 приведен пример решения классического квадратного уравнения, имеющего общий вид . Предварительно это уравнение командой  присваивается в качестве значения переменной ***ur1***. Для решения таких уравнений результат решения представлен для всех случаев: общий - при , решение уравнения  при a=0, множество комплексных чисел С при a=b=c=0 и пустое множество при a=b=0 и , т.е. уравнение решения не имеет.

Второе алгебраическое уравнение, описанное переменной ***ur2***, имеет числовые значения коэффициентов, поэтому результатом выполнения команды ***solve()*** служит набор действительных и мнимых решений уравнения в символьной форме записи.

Ниже на рис.10 приведен пример решения системы алгебраических уравнений. В этом случае команды ***solve()***  имеет также два параметра. В качестве первого аргумента (в квадратных скобках, через запятую) записаны сами уравнения системы, а вторым параметром (также в квадратных скобках, через запятую) служит список переменных, относительно которых решается система.

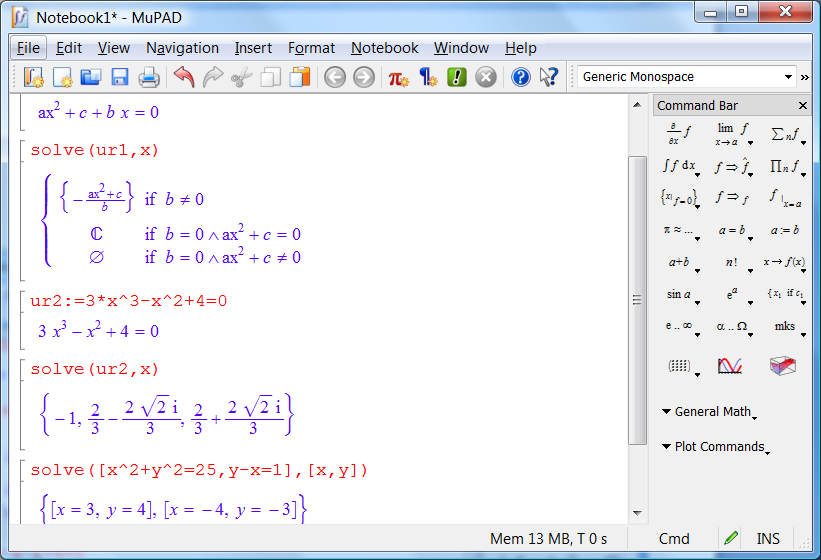


Рис. 3.4-10. Решение алгебраических уравнений и систем

Следующий пример (рис. 3.4-11) показывает использование команды ***solve()***  для решения **неравенств**. В данном случае решается неравенство . Решение найдено на множестве действительных чисел от 2 до 3. Кроме того решение найдено и на множестве мнимых чисел , где через I обозначена мнимая единица, а через y обозначена действительная переменная.

С одной стороны, получение таких полных решений удобно, но, с другой стороны, простые задачи зачастую этого не требуют. Поэтому нередко указывают, что решение ищем только на множестве действительных чисел: solve(x^2-5\*x+6<0,x)assuming x in R\_.

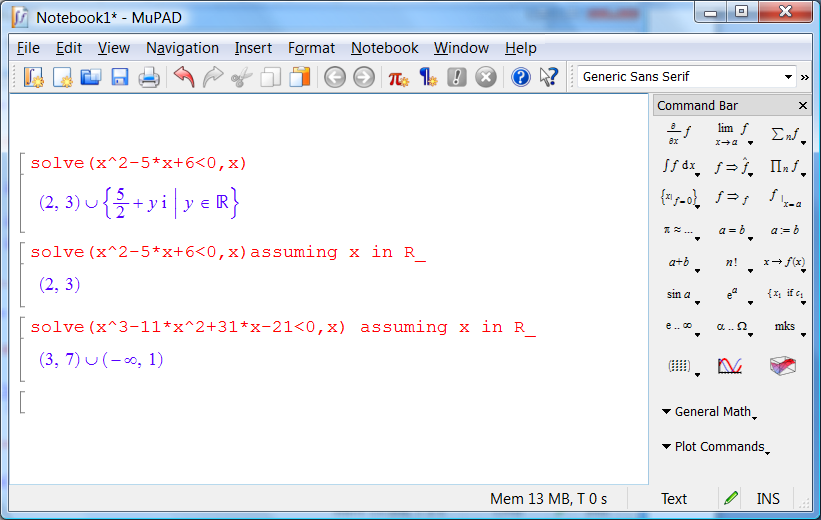


Рис. 3.4-11. Решение неравенств\

* + - 1. **Решение дифференциальных уравнений**

Дифференциальные уравнения также как и алгебраическое уравнение можно решать с использованием функции ***solve()***, однако в этом случае указывается, что уравнение не алгебраическое, а именно дифференциальное. Чтобы в явном виде выполнить это указание до применения команды ***solve()***  указывается дополнение ***ode::***. Тогда синтаксис вызова функции ***solve()***  имеет такой вид: ***ode:: solve()***, где первым аргументом указывается решаемое дифференциальное уравнение (или, при решении системы дифференциальных уравнений, множество уравнений, заключенные в фигурных скобках), а вторым – неизвестная функция (или - функции, заключенные в фигурные скобки).

На рис. 3.4-12 приведены примеры решения **дифференциальных уравнений**. При решении первого уравнение , получено общее решение , где С1 и С2 – произвольные постоянные.

Далее на рис. 3.4-12 рассмотрен пример решения **обыкновенного дифференциального уравнения** с начальными условиями y(0)=3 и (0)=2. При записи команды в этом примере аргументы заключены в фигурные скобки. Результат получается точным и достаточно компактным.

Третий пример ни рис.3.4-12 приведен пример решения **системы** **дифференциальных уравнений**

**** с начальными условиями x(0)=1; y(0)=2.

Здесь и уравнения, перечисленные через запятые, и аргументы заключены в фигурные скобки. Результатом решения данной системы дифференциальных уравнений являются символьные выражения искомых функция y(t) и x(t).

Наконец четвертый пример на рис.3.4-12 показывает решение **дифференциального уравнения второго** порядка  с начальными значениями y(1)=1,  (1)=0. В результате найдено решение дифференциального уравнения, удовлетворяющее начальным условиям.

Зачастую символьные решения дифференциальных уравнений крайне громоздки, поэтому после получения их решений целесообразно их упростить, применив функцию simplify(%).

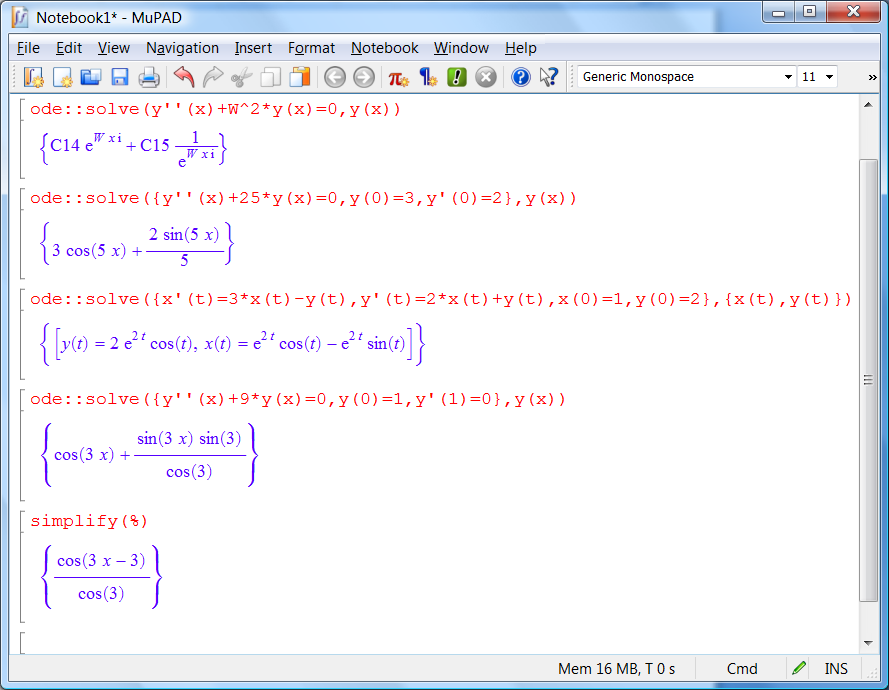


Рис.3.4-12. Решение дифференциальных уравнений