Казанский государственный университет
им. В.И. Ульянова–Ленина

М.Ю. Першагин

РУКОВОДСТВО ПО ВЫПОЛНЕНИЮ ЛЕТНЕЙ ВЫЧИСЛИТЕЛЬНОЙ ПРАКТИКИ

Методическое пособие

Казань
2007

УДК 681.3.06

Печатается по решению
Учебно-методической комиссии
механико-математического факультета
Казанского государственного университета

Научный редактор
кандидат физико-математических наук, доцент Галимянов А.Ф.

Рецензенты:
кандидат физико-математических наук, доцент Липачев Е.К.
кандидат физико-математических наук, доцент Хабибуллин И.Ш.

Першагин М.Ю.

Руководство по выполнению летней вычислительной практики. Методическое пособие. – Казань: Казанский государственный университет им. В.И. Ульянова–Ленина, 2007. – 72 с.

В пособии приведена программа летней вычислительной практики у студентов 2 курса механико-математического факультета. Излагаются необходимые теоретические сведения по работе в пакетах TeX, Mathematica и созданию классов в языке программирования Java. Приведены варианты заданий и пример выполнения работы.

Для студентов 2 курса механико-математического факультета.

УДК 681.3.06

Предисловие научного редактора

В течение ряда лет на механико-математическом факультете КГУ проводится летняя вычислителтьная практика (ЛВП) после второго и третьего курсов. После второго курса студенты "методом погружения" в течение двух недель выполняют большое количество заданий, закрепляющих пройденный материал по курсу "Компьютерные науки", при этом делается упор на решение математических задач средствами ЭВМ. На третьем курсе решаются сложные задачи по вычислительной математике.

Хотя летняя вычислительная практика проводится во многих учебных заведениях, до сих пор отсутствует единый взгляд на ее содержание, а соответственно, методические пособия. Данное методическое пособие в какой-то мере восполняет этот пробел и обобщает опыт проведения ЛВП после второго курса на мехмате в течение ряда лет.

Методическое пособие условно можно разделить на шесть частей. Первый параграф включает в себя ту часть государственного стандарта по компьютерным наукам, которая относится к ЛВП, а также программу практики. Последующие три параграфа посвящены краткому изложению материала ЛВП.

В части, посвященной пакету LaTeX, изложены основные команды пакета, по существу – это краткий справочник по пакету, который будет полезен не только студентам, а всякому, кто набирает математический текст, используя данный пакет.

В части, посвященной пакету Mathematica, изложены основные операторы и функции пакета, приведено большое количество примеров.

Следующая часть посвящена решению задач с помощью объектно-ориентированного подхода. В качестве базового языка используется язык Java, изучаемый студентами-математиками на первых двух курсах. Также приводятся примеры.

В пятой части пособия приведены варианты заданий, предлагаемых студентам, выполняющим ЛВП.

В последней части приводится пример выполнения некоторых заданий.

1. Программа учебной практики «Летняя вычислительная практика»

Предназначена для студентов _2_ курса
по специальности: 010100 -математика

Аннотация практики:

Цели практики:

- студентам в режиме «погружения» («интенсива») отработать знания, умения и навыки, приобретенные в течение учебного года по предметам «Компьютерные науки», «Линейная алгебра и аналитическая геометрия», «Математический анализ», и др. как межпредметные через решение конкретных заданий с использованием компьютерных технологий;
- сформировать знания, умения, навыки у студентов при работе с информацией, представленной в виде целых, вещественных, строковых и др. данных при компьютерной обработке (точность, погрешности, ошибки …)
- сформировать навыки применения компьютерных технологий при проведении аналитических математических исследований.

Задачи практики:

- с помощью индивидуальных заданий проверить усвоение теоретических знаний и практических навыков решения математических задач с помощью компьютерных технологий;
- через самостоятельную работу с литературой и решение конкретных задач освоить новые численные методы решения математических задач с применением компьютера.

1. **Требования к уровню подготовки студента,** завершившего прохождение учебной практики ___летняя вычислительная практика___

Студенты, завершившие изучение данной дисциплины должны:

- понимать основные идеи, лежащие в основе компьютерных наук, их практическое применение;
- знать возможности издательского пакета LATEX, средства пакета Mathematica и среды визуальной разработки программ Java;
- обладать теоретическими знаниями основных методов алгоритмизации прикладных задач математики и других наук;
- уметь проводить анализ полученных результатов;

- ориентироваться в пакетах программ графического отражения расчетных данных, текстовых редакторах;
- приобрести навыки применения методов компьютерных наук для различного класса задач, умения довести их до числа.

2. Объем учебной практики.

Форма обучения Очная

Форма контроля; Зачет

Семестр 4

Всего часов 72 часа

Требования государственного образовательного стандарта к обязательному минимуму содержания программы

ЕН.Ф.01	КОМПЬЮТЕРНЫЕ НАУКИ Понятие информации, общая характеристика процессов сбора, передачи, обработки накопления информации; технические и программные средства реализации информационных процессов. Основные понятия: алгоритм для ЭВМ, базовые конструкции для записи алгоритмов, циклы “для”, “пока”, “если-то-иначе”, выбор, условный и безусловный переход; простейшие типы данных: целый, вещественный, символьный, логический и их представление в ЭВМ; массивы данных; организация ввода и вывода; понятие о файловой системе; файлы последовательного доступа и прямого доступа; форматный и бесформатный ввод/вывод; простейшие алгоритмы обработки данных: вычисление по формулам, последовательный и бинарный поиск, сортировка, итерационные алгоритмы поиска корней уравнений, индуктивная обработка последовательностей данных, рекуррентные вычисления. Структуры данных: вектор, матрица, запись (структура), стек, дек, очередь, последовательность, список, множество, бинарное дерево; реализация структур данных на базе линейной памяти ЭВМ; непрерывный и ссылочный способы реализации структур данных; реализации множества (битовая , непрерывная, хеш-реализация); алгоритмы обработки коллизий в хеш-реализации. Рекурсивные и итерационные алгоритмы обработки данных; условия, обеспечивающие завершение последовательности рекурсивных вызовов; идеи реализации рекурсивных вызовов в подпрограммах; инвариантная функция и инвариант цикла; взаимосвязь итерации и рекурсии, индуктивное вычисление функций на последовательности данных. Структуры данных в прикладных программах; примеры использования и реализации различных структур (редактор текстов, стековой

	калькулятор); принципы построения файловых систем; каталог, таблица размещения файлов, распределение блоков файла по диску. Компиляция и интерпретация: основные этапы компиляции, лексический, семантический анализ выражения, формальная грамматика, компилятор формулы, дерево синтаксического разбора. Понятие об операционной системе: процесс, состояние процесса, прерывание, планирование процессов, понятие о тупиках и способах их устранения. Надежность программного обеспечения: методы тестирования и отладки программ, переносимость программ, технология программирования, принципы создания пакетов стандартных программ, принципы обеспечения дружественного интерфейса прикладных программ. Понятие об архитектуре ЭВМ: процессор и система его команд, структура памяти ЭВМ и способы адресации, выполнение команды в процессоре, взаимодействие процессора, памяти и периферийных устройств. Локальные и глобальные сети ЭВМ; основы защиты информации и сведений, составляющих государственную тайну; методы защиты информации. Компьютерный и вычислительный практикум: реализация алгоритмов обработки данных, возникающих в задачах алгебры, математического анализа, математической статистики, задач обработки изображений, задачах линейного программирования; сети и работа в них.

ТРЕБОВАНИЯ ГОСУДАРСТВЕННОГО ОБРАЗОВАТЕЛЬНОГО СТАНДАРТА К ОБЯЗАТЕЛЬНОМУ МИНИМУМУ СОДЕРЖАНИЯ ПРОГРАММЫ ПРАКТИКИ

Индекс	Наименование дисциплин и их основные разделы	Всего недель
	Компьютерный и вычислительный практикум: реализация алгоритмов обработки данных, возникающих в задачах алгебры, математического анализа, математической статистики, задач обработки изображений, задачах линейного программирования; сети и работа в них.	2

Содержание разделов учебной практики

№ п/п	Название темы и ее содержание	Количество дней
1	Подготовка математического текста с помощью издательского пакета $\LaTeX$.	4
2	Решить средствами пакета Mathematica номера из «Сборника задач и упражнений по математическому анализу» Б.П. Демидовича (год издания 2003)	4
3	Описать один из математических объектов (точка, вектор, прямая, плоскость) средствами языка Java и написать программу, демонстрирующую работу описанного объекта.	4
	Итого дней:	12

2. Система TeX/LaTeX

Пример самого минимального LaTeX-файла, составленного по всем правилам:

```
\documentstyle{article}
\begin{document}
Hello, world!
\end{document}
```

Основные правила при наборе текста:

- исходный текст не должен содержать переносов (TeX сделает их сам);
- два пробела рядом считаются за один пробел;
- абзацы разделяются одной или несколькими пустыми строками;

2.1 Специальные символы

% комментарии
{ начало группы
} конец группы
$ ввод математики
_ нижние индексы математики
^ верхние индексы математики
~ неразрывный пробел
\ сигнальный символ (команд)
параметры в определениях команд
& табулятор
Их можно вывести на печать поставив перед ними \

2.2 Группы

Это одно из важнейших понятий TeX-а, как впрочем и любого другого языка программирования, где оно известно как "блочная структура". Внутри задаются локальные переменные и определения, которые не видны извне.

Пример:

- {\bf Это} слово полужирно.
- \centerline{Эта информация должна быть {\it в центре}}

В последнем примере два набора скобок имеют различные функции.

2.3 Окружения

\begin{Имя окружения} - начальные действия

|-----
| Тело
|-----
\end{Имя окружения} - конечные действия
Пример: \begin{center}
Все строки этого абзаца
будут центрированы
и переносов слов не будет
\end{center}

2.4 Простейший документ

\documentclass[a4paper,12pt]{article}
\usepackage{russian} – пакет поддержки русского языка
\begin{document}
\begin{…} % Кроме команд разбивающих документ на части,
\item … % очень полезными для улучшения зрительного восприятия
\item … % текста является создание перечней:
\begin{…} % В LaTeX-е описаны три вида перечней (окружения):
\item … % itemize - простые ("горошина" - *.)
\item … % enumerate - нумерованные (1 a I(римские) A)
\end{…} % description - описательные (конкретные названия)
\end{…}
\label{mylabel} % В LaTeX-е можно ссылаться на практически любой
% элемент текста: на страницу, на раздел документа,
% на номер рисунка, на номер элемента в нумерованном перечне.
Как указанно на стр~\pageref{mylabel} …
\begin{thebibliography}{99} % А в тексте можно ссылаться на этот
\bibitem{book1} % список литературы (занумерованный автом.)
\bibitem{book2} % с помощью команды \cite{...}
\bibitem{book3}
\end{thebibliography}
\end{document}

2.5 Набор текста

Многоточия

Набранные подряд три точки (…) для пропорциональных шрифтов окажутся расположены слишком близко. Поэтому используются специальные команды:

- \ldots нижнее многоточие
- \cdots центрированное (обычно в математике между +, -, и.т.д)
- \vdots вертикальное (обычно в математике)
- \ddots диагональное (в матрицах и т.д.)

Прочие значки (кроме математики) и т.д.

\S - знак номера параграфа
\copyright - знак авторского права

Промежутки между словами

\enskip - пробел в 0.5em
\quad - пробел в 1em
\qquad - пробел в 2em
\hspace{длина} - конкретный промежуток
\, - тонкий пробел (шпация)
\: - средний пробел
\; - толстый пробел

Формирование страниц

Закончить абзац (эквивалентные варианты):
- пустая строка
- команда \par

Оба способа, выполненные подряд, дополнительных промежутков не создадут (т.е. команда \par в вертикальной моде ничего не делает).

Задать "свой" промежуток между абзацами:
\smallskip маленький вертикальный пробел.
\medskip вертикальный пробел побольше.
\bigskip еще больше (точные размеры зависят от стиля и кегля).
\vspace{…} промежуток конкретного размера.
Можно задать промежуток не фиксированной, а переменной длины:
\vspace{x plus y minus z}, здесь: x,y,z – длины, plus, minus - ключевые слова (без backslash)

Принудительный разрыв страницы:
\newpage – дополняется снизу пустым пространством;
\clearpage – и еще допечатывает плавающие иллюстрации;

\pagebreak – с "растягиванием" страницы;
<u>Запрет разрыва страницы:</u>
\nopagebreak – локальный, т.е. между конкретными абзацами;
\samepage – глобальный, т.е. разрывы станут возможны только между абзацами, а не внутри и не между текстом и выключеной формулой;

2.6 Набор математических формул

Основные способы задания формул

- "внутренних" т.е. внутри текста:
  a) $...$ -- стандартный способ TeX;
- "выключеных" т.е. выделенных в отдельную строку:
  a) $$...$$ -- стандартный способ TeX;
  b) {equation} -- способ LaTeX с автоматической нумерацией
  (стилевые опции:
  [leqno] - нумерация слева,
  [flegn] - формулы слева).

Основные принципы набора формул

- пробелы игнорируются (TeX их сделает сам);
- пустые строки не разрешаются;
- математическая формула является группой;
- каждая буква рассматривается как имя переменной и набирается шрифтом "математический курсив";
- поэтому обычный текст включается командой \mbox.

Греческие буквы

Задаются командами по их английским названиям
(начинаются с большой буквы для прописных букв \psi \Psi).

\alpha \iota \sigma \beta \kappa \varsigma \gamma \lambda \tau \delta \mu \upsilon \epsilon \nu \phi \varepsilon \xi \varphi \zeta \pi \chi \eta \varpi \psi \theta \rho \omega \vartheta \varrho

Прописные греческие буквы не совпадающие по начертанию с латинскими (печатаются прямым шрифтом):

\Gamma \Delta \Theta \Lambda \Xi \Pi \Sigma \Upsilon \Phi \Psi \Omega

Если нужны наклонные прописные греческие буквы, то надо написать, например: {\mit\Sigma};

Бинарные операции

+ плюс

- минус

* умножение
\times умножение "крестиком"
\div деление (минус между точками)

Бинарные отношения

< меньше
> больше
= равно
\leq меньше либо равно
\geq больше либо равно
\neq не равно
\sim подобно (одна волна)
\approx приближенно (две волны)
\equiv эквивалентно ("тройное равенство")

Стрелки различных видов

\to тонкая стрелочка вправо
\Rightarrow двойная стрелочка вправо
\gets тонкая стрелочка влево
\Leftarrow двойная стрелочка влево

Стандартные функции

\sin \tan \exp \cos \arctan \dim \arcsin \log \lg \arccos \ln

"Элементарные" операции

\sum сумма \prod произведение \lim предел \inf инфимум \max максимум \int интеграл \min минимум \oint контурный интеграл

Скобки различных видов

() круглые скобки
[] квадратные скобки
\{ \} фигурные скобки
| знак модуля
\langle \rangle угловые скобки
Задание размера автоматическое:
\left(... \right) - по высоте фрагмента формулы.
Эти команды могут появляться только парами, однако скобку можно сделать невидимой, задав вместо нее точку: \left .
Ограничители не обязаны быть однотипными.
Задание размера явное:

\bigl ... \bigr - конкретные размеры зависят от кегля и подбираются экспериментально;
\Bigl ... \Bigr
\biggl ... \biggr
\Biggl ... \Biggr
Эти команды не обязаны появляться парами.

Переносы

а) в выключеных формулах:
никогда не переносятся автоматически, но можно это сделать используя некоторые трюки, например, окружение {array}:
$$ \begin{array}{l}
e^x=1+x+\frac{x^2}{2!}\\
\qquad +\frac{x^3}{3!}+\cdots
$$ \end{array}
b) во внутритекстовых формулах:
после знаков бинарных отношений и операций, но не может разорваться часть ограниченная фигурными скобками.

Буквы в формулах

а) набираются математическим курсивом \mit (по умолчанию) - эта команда непосредственно используется редко.
b) набираются полужирным прямым шрифтов (если в преамбуле задать \boldmath)
Если нужен другой шрифт нужно дать соответствующую команду.
Пример: Обозначение ${\bf P}^n$ используется для проективного пространства.

Текст в формулах

Набирается \mbox{текст}, а иначе TeX расставит свои маленькие "математические" пробелы.
Неправильно (вся правая часть сольется в одно слово):
$$ 2x=x+x {\rm для всех} x $$
Правильно:
$$ 2x=x+x \qquad \mbox{для всех} x. $$
Используется шрифт который был текущем перед началом формулы.

Надстрочные знаки

это дополнительные значки над буквой или фрагментом формулы:
а) \overline{...} - горизонтальная черта над любым фрагментом формулы;
b) \overrightarrow{...} - стрелка (вектор) над любым фрагментом формулы;

c) "узкие" значки:
\hat - шляпка
\tilde - волна
\bar - черточка
\vec - вектор
\dot - точка
\ddot - две точки
Пример: \vec a
d) "широкие" значки (но не безгранично):
\widehat{...}
\widetilde{...}
Пример: Тождество $ \widehat{f*g}=\hat f\cdot\hat g $ означает...

Элементарные мелочи

a) степени и индексы: набираются знаками ^ и _ соответственно:
R^i_{jkl} или $R_j{}^i{}_{kl}$
Во втором случае, оформив индексы к пустой формуле, мы добились чтобы они располагались не один под другим.
b) дроби, обозначаемые косой чертой, набираются непосредственно:
$x+1/x\geq 2$ верно для всех $x>0$
c) запятая в десятичной дроби записывается в фигурных скобках, иначе после нее будет поставлен дополнительный пробел:
$\pi\approx 3{,}14$
d) \sqrt[показатель] {подкоренное выражение} - корень
$\sqrt[3]{x^3}=x$
e) \log_{основание}{аргумент} - основание задается как нижний индекс.
e) штрихи обозначаются знаком ' и не оформляются как верхние индексы:
$$(fg)"=f"g+2f'g'+fg"$$
f) "пределы" у знака суммы (по умолчанию):
- в выключной формуле печатаются "над и под" знаком;
- во внутритекстовой "сбоку", как и индексы;
$$ \sum_{i=1}^n n^2 $$
"пределы" у знака интеграла (по умолчанию):
- печатаются как индексы "сбоку";
$$ \int_0^1 x^2 dx $$ - от 0 до 1 от x^2
\limits - обязать расположение "над и под";
\nolimits - обязать расположение "сбоку";
$$ \int\limits_0^1 x^2 dx $$

Одно над другим

a) \frac{числитель}{знаменатель} - запись для дроби обыкновенной
(одну букву или цифру можно не брать в скобки):
$$ \frac12+\frac x 2=\frac{1+x}2 $$
b) горизонтальная фигурная скобка:
\overbrace{фрагмент формулы}^надпись - над формулой
\underbrace{фрагмент формулы}_подпись - под формулой
c) расположение типа "над-под":
\atop - общий случай
{верхняя часть формулы \atop нижняя часть формулы}
$\left\{ij \atop k\right\}$ здесь {} выполняют две функции.
\choise - биноминальные коэффициенты
$n \choose k$
d) расположение "вровень-над" :
\stackrel{будет над строкой}{будет в строке}
$A\strackrel{f}{\longrightarrow}B$,почти: A-f->B

Матрицы

\begin{array}{преамбула}
|---------
| преамбула это ряд букв (по букве на столбец), описывающих столбцы:
| c - центрированы;
| l - выровнены по левому краю;
| r - выровнены по правому краю;
| сама матрица формируется с использованием:
| \\ - разделяет строки матрицы;
| & - разделяет элементы столбцов внутри строки;
|---------
\end{array}
Пример записи простой квадратной матрицы из n элементов
(скобки надо записать отдельно):
$$
\left(\begin{array}{cccc}
a_{11} & a_{12} & \ldots & a_{1n} \\
a_{21} & a_{22} & \ldots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{n1} & a_{n2} & \ldots & a_{nn}
\end{array}\right)
$$

Системы уравнений

Системы уравенений можно записывать используя {array}:

```
$$
\left\{
\begin{array}{rcl}
x^2+y^2 & = & 7 \\
x+y & = & 3 \\
\end{array}
\right.
$$
```

3. Система Mathematica

В приводящихся далее примерах в целях экономии места будем придерживаться следующего обозначения. Входные данные записываются жирным шрифтом, выходные данные (результаты вычисления) обычным шрифтом. Например,

Sum[i^2, {i,10}]

385

Кроме того, результаты вычислений приведены "в одну строчку", хотя вывод в пакете будет отличаться от этого. Например,

Integrate[1/(a^2+x^2), x]

ArcTan[x/a]/a

3.1 Вычисление сумм

В числе операций математического анализа прежде всего надо отметить суммы

Сумма от i = imin до imax по di

В этих операциях индекс i принимает целочисленные значения от минимального (начального) imin до максимального (конечного) imax с шагом, равным +1.

Суммы и произведения легко вычисляются численными математическими системами, такие вычисления просто описываются на всех языках программирования. Однако важным достоинством систем символьной математики, включая Mathematica, является вычисление сумм и произведений в аналитическом виде (если это возможно) и при большом числе членов — вплоть до стремящегося к бесконечности.

Для вычисления сумм в системе Mathematica предусмотрена функция **Sum**, используемая в ряде форм:

Sum[f, {i, imax}] — вычисляет сумму значений *f* при изменении индекса *i* от 1 до *imax* с шагом +1;

Sum[f, {i, imin, imax}] — вычисляет сумму значений *f* при изменении индекса i от минимального значения *i* = *imin* до максимального *i* = *imax* с шагом +1;

Sum[f, {i, imin, imax, di}] — вычисляет сумму значений *f* при изменении управляющей переменной вещественного типа от минимального значения *i* = *imin* до максимального *i* = *imax* с шагом *di*;

Sum[f, {i, imin, imax}, {j, jmin, jmax},...] — вычисляет многократную сумму значений *f* при изменении индексов *i* от *imin* до *imax* с шагом +1, *j* от *jmin* до jmax с шагом +1 и т. д. (число индексов не ограничено).

Таким образом, эта функция обеспечивает расширенные возможности вычисления сумм — как при целочисленных, так и при вещественных значениях управляющих переменных, задающих циклы вычислений. Примеры использования функций суммирования:

Sum[i^2, {i,10}]

385

Sum[i^2, {i,l,10}]

385

Sum[i^2, {i,1,2,0.25}]

11.875

Sum[i*j, {i,1,10}, {j,2,5}]

770

Обычно в математических системах недопустима перестановка *imin* и *imax*, хотя в математике известно школьное правило — от перестановки слагаемых сумма не изменяется. Рискнем проверить это:

Sum[i, {i,1,100}]

5050

Sum[i, {i,100,1}]

0

Sum[i, {i,100,1,-1}]

5050

Второй пример тут дал явно ошибочный результат, хотя третий с честью оправдал указанное правило.

Приведем еще ряд примеров выполнения операции суммирования:

Sum[x^n/n!, {n,0,8}]

$1 + x + x^2/2 + x^3/6 + x^4/24 + x^5/120 + x^6/720 + x^7/5040 + x^8/40320$

Sum[x^n/n!, {n,1,9,2}]

$x + x^3/3 + x^5/120 + x^7/5040 + x^9/362880$

Sum[x^i y^j, {i,1,4},{j,1,i}]

$xy + x^2y + x^3y + x^4y + x^2y^2 + x^3 y^2 + x^4y^2 + x^3 y^3 + x^4 y^3 + x^4y^4$

Sum[1/(n*n) , {n, 1, бесконечность}]

л^2/6

Sum[i^4 {i, 1, n}]

$1/30\ n\ (1+n)(1+2n)\ (-l + 3n+3n^2)$

Из этих примеров видно, что Mathematica обеспечивает возможность символьного вычисления сумм, в том числе с бесконечным пределом суммирования.

3.2 Вычисление произведений

Операции вычисления произведений

Произведение от i = imin до i = imax по di

представлены следующими функциями:

Product[f, {i, imax}] — возвращает произведения значений *f[i]* для значений *i*, изменяющихся от 1 до *imax*;

Product[f, {i, imin, imax}] — возвращает произведение значений *f[i]* при изменении *i* от *imin* до *imax* с шагом +1;

Product[f, {i, imin, imax, di}] — возвращает произведение *f[i]* при *i*, меняющемся от значения *imin* до значения *imax* с шагом *di*;

Product[f, {i, imin, imax}, {j, jmin, jmax},...] — вычисляет многократное произведение (произведение по нескольким переменным).

Примеры использования функций вычисления произведения.

Product[i, {i,10}]

3628800

Product[x+i^2, {i,1,5}]

(1 + x) (4 + x) (9 + x) (16 + x) (25 + x)

Об опасности перестановки сомножителей свидетельствуют следующие примеры:

Product[i, {i,10}]

3628800

Product[i, {i,10,1}]

1

Product[i, {i,10,l,-l}]

3628800

Как и в случае вычисления суммы, средний пример явно ошибочен. Он просто недопустим с точки зрения синтаксиса данной функции.

3.3 Вычисление производных

К числу наиболее часто используемых математических операций принадлежит вычисление производных функций как в аналитической, так и в символьной форме. Для этого используются следующие функции:

D [f, x] — возвращает частную производную функции *f* по переменной *x*;

D [f, {x, n}] — возвращает частную производную *n*-го порядка по *x*;

D[f, xl, x2,...] — возвращает смешанную производную;

Dt[f, x] — возвращает обобщенную производную функции *f* по переменной *x*;

Dt [f] — возвращает полный дифференциал *f*.

Название функции из одной буквы — это явно исключение из правил. Оно выбрано осознанно, в силу массовости этой операции.

Следующие примеры показывают применение функции D для вычисления производной в аналитическом виде:

Производная тригонометрической функции:

D[x*Sin[x], x]

xCos[x] + Sin[x]

Производная экспоненциальной функции:

D[Exp[x/b], x]

$e^{x/b}/b$

Производная логарифмической функции:

D[Log[3*x/4], x]

1/x

Производная степенного многочлена:

D[a*x^2+b*x+c, x]

b + 2ax

Пятая производная от x^n:

D[x^n, {x, 5}]

$(-4 + n)\ (-3 + n)\ (-2 + n)\ (-1 + n)\ n\ x^{-5+n}$

Производная функции двух переменных:

D[(x^m)*y^n, x, y]

$m\ n\ x^{-1+m}\ y^{-1+n}$

Следующие примеры иллюстрируют вычисление производных от первого до третьего порядка включительно для функции *f [x]*, заданной пользователем.

Ввод (In)	**Вывод (Out)**
f[x] :=x/(1+x^2)	
D[f[x],{x,l}]	$-2x^2/(1+x^2)^2+1/(1+x^2)$
D[%,x]	$8x^3/(1+x^2)^3-6x/(1+x^2)^2$
D[f[x],{x,2}]	$8x^3/(1+x^2)^3-6x/(1+x^2)^2$
D[D[D[f[x],x],x],x]	$-48x^4/(1+x^2)^4+48x^2/(1+x^2)^3-6/(1+x^2)^2$
D[f[x],{x,3}]	$-48x^4/(1+x^2)^4+48x^2/(1+x^2)^3-6/(1+x^2)^2$

Из предпоследнего примера видно, что для вычисления высших производных возможно последовательное применение функции D.

Использование функции Dt демонстрируют примеры, приведенные ниже.

Ввод (In)	**Вывод (Out)**
Dt[x^n,x]	x^n (n/x +Dt[n, x] Log[x])
Dt[x*Sin[x] ,x]	xCos[x] + Sin[x]

Dt[Exp[x/b],x]	$e^{x/b}$(1/b-xDt[b, x]/b^2)
Dt[a*x^2+b*x+c,x]	b+ 2 ax + x^2 Dt[a, x] + xDt[b, x] + Dt[c, x]
Dt[x^n,{x,2}]	x^n (n/x+Dt[n, x] Log[x]) + x^n(-n/x^2 + 2Dt[n, x]/x+ Dt[n, {x/2}] Log[x])
Dt[Log[3*x/4],x]	1/x

Обратите внимание на то, что порой результаты для одного и того же дифференцируемого выражения у функций D и Dt заметно различаются. Это вполне закономерно вытекает из различных определений данных функций.

3.4 Вычисление интегралов

Одна из важнейших операций — вычисление первообразных и определенных интегралов в символьном виде. Первообразная — это функция *F(x)*, удовлетворяющая уравнению $\int f(x)dx = F(x) + C$, где C — постоянная интегрирования. А вычисление определенного интеграла с пределами — верхним *b* и нижним *a* — производится по формуле $\int_a^b f(x)dx = F(b) - F(a)$

Для интегрирования в системе Mathematica используются следующие функции:

Integrate[f, x] — возвращает первообразную (неопределенный интеграл) подынтегральной функции *f* по переменной *x*;

Integrate[f, {x, xmin, xmax}] — возвращает значение определенного интеграла с пределами от *xmin* до *xmax*;

Integrate[f, {x, xmin, xmax}, {y, ymin, ymax},...] — возвращает значение кратного интеграла с пределами от *xmin* до *xmax* по переменной *x*, от *ymin* до *ymax* по переменной *y* и т. д. (кратность реально не ограничена).

Для обозначения бесконечных пределов используется константа Infinity. Эта константа означает положительную бесконечность, для задания отрицательной бесконечности она используется со знаком «минус». Пределы могут задаваться как константами, так и функциями.

Особый интерес, естественно, вызывает применение функции Integrate для вычисления заданных пользователем неопределенных интегралов в символьном виде. Это иллюстрируют примеры на вычисление неопределенных интегралов с алгебраическими подынтегральными функциями.

Integrate[a*x^b, x]

a x^{1+b} / (1 + b)

Integrate[{Log[x],Exp[x],Sin[x]}, x]

{-x + x Log[x], e^x, -Cos[x]}

Integrate[a x^2+b x+c, x]

$cx + bx^2/2 + ax^3/3$

Нетрудно заметить, что интегралы от ряда алгебраических функций дают выражения с тригонометрическими функциями.

Integrate[1/(a^2+x^2), x]

ArcTan[x/a]/a

Integrate[1/Sqrt[9 x^2+4], x]

1/3 ArcSin[3x/2]

Другая группа примеров показывает нахождение интегралов с тригонометрическими и гиперболическими подынтегральными функциями.

Integrate[1/(1-Cos[x]), x]

-(2 Cos[x/2] Sin[x/2])/(1-Cos[x])

Integrate[Cos[x] x^3, x]

3 (-2 + x^2) Cos[x] + x (-6 x^2) Sin[x]

Следующая серия примеров иллюстрирует вычисление определенных интегралов в символьном виде.

Integrate[x^2, {x, a, b}]

$-a^3/3+b^3/3$

Integrate[1/(x Sqrt[x^2+1]), {x, 1, Infinity}]

ArcSinh[1]

Вычисление определенных интегралов с пределами-функциями.

Integrate[x^2, {x, 0, Sqrt[x]}]

$x^{3/2}/3$

Integrate[Sqrt[x+1], {x, a x + b, x}]

2/3 (-1 – ax - b) Sqrt[1 + ax+ b] – 2/3 (-1 - x) Sqrt[1 + x]

Системы Mathematica имеют самые обширные возможности вычисления интегралов. Ядро системы вобрало в себя формулы интегрирования из всех известных справочников и даже древних рукописей.

Mathematica способна вычислять даже кратные интегралы с фиксированными и переменными верхним или нижним пределами. Кратный, например двойной, интеграл с фиксированными пределами имеет вид: $\int_a^b \int_c^d f(x,y)dxdy$

Ниже представлено вычисление двойных определенных интегралов.

Integrate[x^2 + y^2, {x, 0, a}, {y, 0, a}]

2 a^4/3

Integrate[Sin[x y^2], {x, 0, x^2}, {y, 0, y}]

1/2 x^4 Sin[x y^2]

Integrate[x y, {x, a, b}, {y, c, d}]

$(-a^2/2 + b^2/2) (-c^2/2 + d^2/2)$

3.5 Вычисление пределов функций

Многие функции при приближении аргумента к некоторому значению или к некоторой области значений стремятся к определенному *пределу*. Так, функция *sin(x)/x* при *x*, стремящемся к нулю (обозначим это как *x*—> 0), дает предел 1 в виде устранимой неопределенности 0/0.

Численные математические системы, равно как и большинство программ на обычных языках программирования, не воспринимают выражение 0/0 —> 1 как объективную реальность. Их защитный механизм настроен на примитивное правило — ничего нельзя делить на 0. Следовательно, вычисление *sin(x)/x* при *x* = 0 будет сопровождаться выдачей ошибки типа «Деление на 0». Конечно, в данном конкретном случае можно предусмотреть особый результат — выдать 1 при *x* = 0. Но это частный случай. В целом же подобные системы «не понимают» понятия предела.

Пределом некоторых функций может быть бесконечность, тогда как многие функции стремятся к конечному пределу при аргументе *x*, стремящемся к бесконечности. Система Mathematica не только численно находит пределы функций, заданных аналитически, но и позволяет найти предел в виде математического выражения.

Далее представлены примеры применения функции Limit. Они показывают, что возможно вычисление пределов функций, устремляющихся к бесконечности, и вычисление пределов при переменной *x*, стремящейся в бесконечность. Вычисление пределов функций в аналитическом виде — важное достоинство систем символьной математики.

Limit[1/Log[x]-1/x, x -> 1]

∞

Limit[(x^x – a^x)/(a^x – x^a), x -> a]

1/(-1 + Log[a])

Limit[Sin[x n]/x, x -> 0]

n

При работе с функцией Limit используются следующие опции:

Analytic — указывает, следует ли неопознанные функции интерпретировать как аналитические (значение по умолчанию — Automatic);

Direction — указывает направление, в котором происходит приближение к пределу. Опция используется в виде Direction -> -1 (или +1), по умолчанию выбор остается за системой (Automatic). Значение +1 означает предел слева, а -1 — справа (казалось бы, должно быть наоборот, но задано именно так).

3.6 Уравнения и системы уравнений

Решение уравнений

Многие математические задачи сводятся к решению в общем случае нелинейных уравнений вида *f(x)* = 0 или *f(x)* = expr.

В системе Mathematica они обозначаются как eqns (от слова equations — уравнения). Разумеется, могут решаться и системы, состоящие из ряда таких уравнений.

Для решения уравнений (как одиночных, так и систем) в численном и символьном виде Mathematica имеет функцию Solve:

Solve [eqns, vars] — предпринимает попытку решить уравнение или систему уравнений eqns относительно переменных vars;

Solve [eqns, vars, elims] — пытается решать уравнения eqns по переменным vars, исключая переменные elims.

Входные параметры этой функции могут быть представлены списками или записаны выражениями через объединительный знак «&&». В eqns в качестве знака равенства используется знак «= =».

Solve[x^7-1 = = 0, x]

{{x → 1}, {x → -$(-1)^{1/7}$}, {x → $(-1)^{2/7}$}, {x → $(-1)^{3/7}$}, {x → $(-1)^{4/7}$}, {x → $(-1)^{5/7}$}, {x → $(-1)^{6/7}$}}

Solve[Sqrt[x^3] = = 2, x]

{{x → $(-2)^{2/3}$}, {x → $2^{2/3}$}, {x → -$(-1)^{1/3}$ $2^{2/3}$}}

Solve[Sin[Cos[x]] = = 5, x]

{{x → -ArcCos[ArcSin[5]]}, {x → ArcCos[ArcSin[5]]}}

Решение систем нелинейных уравнений в символьном виде

Solve[{y = = x^2, x = = a + b}, {x, y}]

{{y → a^2 + 2 a b + b^2, x → a + b}}

N[Solve[x^3 – y^2 = = 6 && x^2 = = 3, {x, y}]]

{{y → 0. – 0.896575 ⅈ, x → 1.73205},
{y → 0. + 0.896575 ⅈ, x → 1.73205},
{y → 0. – 3.34607 ⅈ, x → 1.73205},
{y → 0. + 3.34607 ⅈ, x → 1.73205},}

eqns = {x = = 2+3 a x, y = = 5 + 2 x};

Solve[eqns, {x, y}]

{{y → 3 (-3 + 5 a) / (-1 + 3 a)}, x → - 2 / (-1 + 3 a)}

Приведенные примеры показывают решение систем нелинейных уравнений с помощью функции Solve.

Достаточно характерен пример с применением функции N. Если убрать в нем функцию N, то будет получен чрезвычайно громоздкий, хотя и точный результат (проверьте это сами, поскольку размеры результата делают нецелесообразным его приведение в книге). Функция N осуществляет выполнение всех промежуточных вычислений, благодаря чему результат получается вполне обозримым и представленным в комплексных числах.

Опции функции Solve

С функцией Solve можно использовать ряд опций. Их можно вывести командой Options [Solve]. Ниже описано их назначение:

InverseFunctions — указывает, следует ли использовать обратные функции;

MakeRules — указывает, должен ли результат быть представлен как объект AlgebraicRulesData;

Method — устанавливает алгоритм, используемый для вычисления результата (возможны методы 1, 2 и 3);

Mode — задает характер решения уравнения (возможны Generic, Modular и Rational);

Sort — устанавливает, нужна ли сортировка результатов;

Verif ySolutions — устанавливает, следует ли проводить проверку полученных решений и удаление посторонних решений;

WorkingPrecision — устанавливает число цифр промежуточных вычислений (по умолчанию Infinity).

Множество примеров решения систем нелинейных уравнений в символьном виде можно найти в справочной системе Mathematica.

Поиск корней уравнений

Для вычисления корней полиномиальных уравнений используется функция Roots:

Roots[lhs==rhs, var]

Ниже представлены примеры применения функции Roots.

Roots[x^2 + 2 * x + 15 = = 0, x]

x = = -1 – ⅈ Sqrt[14] | | x = = -1 + ⅈ Sqrt[14]

Roots[x^5 + 8 * x^4 + 31 * x^3 + 80 * x^2 + 94 * x + 20 = = 0, x]

x = = -2 – Sqrt[3] | | x = = -2 + Sqrt[3] | | x = = -1 – 3 ⅈ | | x = = -2

Формат выдачи результатов для функции Roots отличается от такового для функции Solve. Поэтому проверку решения подстановкой надо выполнять как в следующем примере:

e = x^2+3x==2

$3x + x^2 == 2$

N[Roots[e, x]]

x = = -3.56155 | | x = = 0.561553
r= {ToRules[%]}
{{x->-3.56155}, {x-> 0.561553}}
e /. r
{True, True}

Для преобразования результата вычислений в список решений (подобный решениям, получаемым с помощью функции Solve) здесь использована функция ToRules.

При затруднениях в решении уравнений с помощью функции Roots можно использовать следующие опции:

Опции функции Roots

Cubics — указывает, следует ли искать явные решения для неприводимых кубических уравнений;
EquatedTo — задает выражение для замещения переменной в решении;
Modulus — задает промежуточную факторизацию полинома;
Multiplicity— устанавливает кратность каждого из корней в конечном результате;
Quartics — задает точное решение квадратного уравнения и полинома четвертой степени;
Using — указывает какие-либо дополнительные уравнения, которые следует использовать для решения уравнений.

Применение опций нередко позволяет получать решения, которые не удаются с первого раза. Однако это требует определенного опыта и понимания сути решаемой задачи.

3.7 Дифференциальные уравнения

Дифференциальным*и* принято называть уравнения, в состав которых входят производные функции *y(x),* представляющей решение уравнения. Дифференциальные уравнения могут быть представлены в различной форме, например в общеизвестной форме Коши: y'(x) = = f(x,y).

Несколько дифференциальных уравнений образуют систему дифференциальных уравнений. Решение таких систем также возможно средствами Mathematica.

Для решения дифференциальных уравнений в символьном виде используются следующие средства:

DSolve[eqn, y[x], x] — решает дифференциальное уравнение относительно функций *y[x]* с независимой переменной *x*;

DSolve[{eqnl, eqn2,...}, {yl [xl,...],...}, {xl,...}] — решает систему дифференциальных уравнений.

У функции DSolve есть опция, на которую следует обратить внимание:

DSolveConstants — опция DSolve, определяющая постоянные интегрирования, которые будут использованы в результате;

В решении дифференциальных уравнений встречаются постоянные интегрирования. По умолчанию они обозначаются как *C[i]*.

Приведем примеры решения дифференциальных уравнений:

DSolve[{yl' [x] == 2 x^2, y2' [x] == 3 x}, {yl[x], y2[x]}, x]

{{yl[x] ->2x^3/3+C[1], y2[x] ->3x^2/2+C[2]}}

DSolve[y'[x] +y[x] ==x, y[x], x]

{{y[x] ->-1+x + e^{-x}C[1]}}

DSolve [y" [x] - y' [x] - 6 y [x] == 0, y [x] , x]

{{y[x] -> e^{-2x} C[1] + e^{3x} C[2]}}

DSolve [y" [x] + 4 y'[x] == 10 Sin [2 x] , y [x] , x]

{{y[x] -> e^{-4x} C[1] + C[2] - Cos[2x] - 1/2 Sin[2x]}}

3.8 Поиск максимального и минимального чисел в списке

В практике математических прикладных вычислений важная роль принадлежит задачам, поиска минимальных и максимальных значений функций одной или нескольких переменных. Mathematica дает разнообразные возможности — от поиска элементов списка с минимальным или максимальным значением до поиска локальных и даже глобальных минимумов функций, заданных аналитически.

Для поиска максимального и минимального значений ряда чисел, входящих в список, система Mathematica предоставляет следующие средства:

Max [xl, x2,...] — возвращает наибольшее значение из x_i;

Max[{xl, x2,...}, {yl,...},...] — выбирает наибольший элемент из нескольких списков;

Min[xl, x2,...] — возвращает наименьшее значение из x_i;

Min[{xl, x2,...}, {yl,...},...] — выбирает наименьший элемент из нескольких списков.

Следующие примеры показывают действие этих простых функций.

Ввод (In)	Вывод(Out)
Max[1,5,2,6.5,3,4]	6.5
Max[{1,3,2},{4,5,6},{9,8,7}]	9
Min[1,5,2,6.5,-3,4]	-3
Min[{1,3,2},{4,5,6},{9,8,7}]	1

Поиск локального минимума аналитической функции

Если нужен поиск локального минимума некоторой аналитической функции, используется функция FindMinimum[f, {x, x0}], которая выполняет поиск локального минимума функции *f*, начиная со значения $x = x_0$, и возвращает его значение.

Для указания градиента минимизируемой функции используется опция Gradient.

Приведем примеры применения функции FindMinimum:

FindMinimum[-x Exp[-2 x] , {x, 1}]

{-0.18394, {x -> 0.5}}

FindMinimum[-x Exp[-2 x] , {x, 0.2, 6, 1}]

{-0.18394, {x -> 0.5}}

FindMinimum [-5 xExp[-x/2] (2 + Sin[3x]), {x, 1}]

{-7.17833, {x -> 0.783139}}

FindMinimum[-5xExp[- x/2] (2 + Sin[3 x]) , {x, 3}]

(-10.6299, {x -> 2.5805}}

FindMinimum[-5xExp[- x/2] (2+Sin[3x]), {x, 4}]

{-6.79134, {x -> 4.6179}}

Эти примеры показывают, что выбирая разные начальные значения *x*, можно найти ряд минимумов функции *f(x)*, разумеется, если таковые имеют место. Если необходимо разыскивать локальные максимумы, достаточно перед функцией поставить знак «минус» или умножить ее на -1.

Поиск глобального максимума и минимума аналитической функции

Следующие две функции служат для поиска глобального максимума и минимума аналитически заданной функции:

ConstrainedMax[f, {inequalities}, {x, y,...}] — ищет глобальный максимум функции *f* в области, определяемой неравенствами inequalities. Полагается, что все переменные *x*, *y*... неотрицательны;

ConstrainedMin[f, {inequalities}, {x, y,...}] — ищет глобальный минимум функции *f* в области, определяемой неравенствами inequalities. Все переменные *x*, *y*... полагаются неотрицательными.

3.9 Двумерная графика

Графическая функция Plot

Концептуально графики в системе Mathematica являются *графическими объектами*, которые создаются (возвращаются) соответствующими *графическими функциями*. Их немного, около десятка, и они охватывают построение практически всех типов математических графиков. Как уже отмечалось, достигается это за счет применения *опций* и *директив.*

Поскольку графики являются *объектами,* то они могут быть значениями переменных. Поэтому Mathematica допускает следующие конструкции:

Plot[Sin[x],{x,0,20}] — построение графика синусоиды;

g:=Plot [Sin [x], {x, 0, 20}] — задание объекта — графика синусоиды — с отложенным выводом;

g=Plot [Sin [x], {x, 0, 20}] — задание объекта — графика синусоиды — с немедленным выводом.

Начнем рассмотрение графических возможностей системы с построения простейших графиков функций одной переменной вида *y =f(x)* или просто *f(x).* График таких функций строится на плоскости, то есть в двумерном пространстве. При этом используется прямоугольная (декартова) система координат. График представляет собой геометрическое положение точек *(x, y)* при изменении независимой переменной (абсциссы) в заданных пределах, например от минимального значения *xmin* до максимального *xmax* с шагом *dx*. По умолчанию строятся и линии координатной системы.

Для построения двумерных графиков функций вида *f(x)* используется встроенная в ядро функция Plot:

Plot [f, {x, xmin, xmax}] — возвращает объект, представляющий собой график функции *f* аргумента *x* в интервале от *xmin* до *xmax*;

Plot[{f1, f2,...}, {x, xmin, xmax}] — возвращает объект в виде графиков ряда функций f_i.

Функция Plot используется для построения одной или нескольких линий, дающих графическое представление для указанных функций f, f_1, f_2 и т. д. На рис. 1 показано построение графика функции *sin(x)/x* без использования каких-либо опций (точнее, с набором опций по умолчанию).

Plot[Sin[x] / x, {x, -20, 20}]

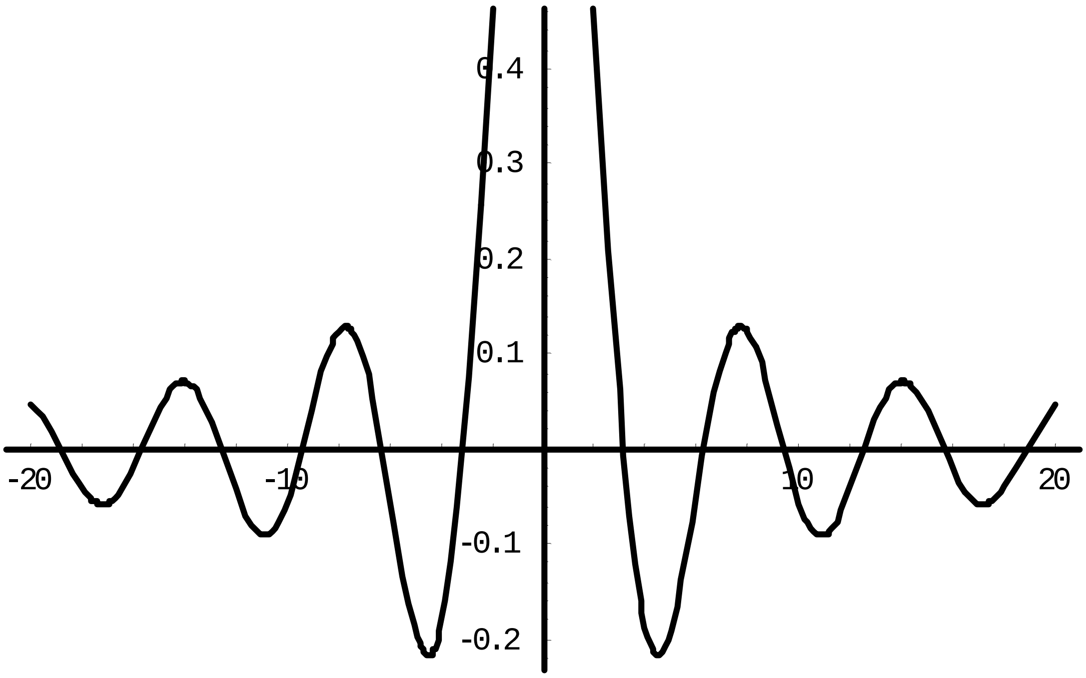


Рис. 1. Построение двумерного графика

Тут виден как раз тот случай, когда масштаб графика по вертикали выбран системой неудачно — часть графика сверху просто отсекается. В большинстве же случаев применение функции Plot позволяет получить вполне приемлимый график.

Опции функции Plot

По мере усложнения задач, решаемых пользователем, его рано или поздно перестанут устраивать графики, получаемые при автоматическом выборе их стиля и иных параметров. Для точной настройки графиков Mathematica использует специальные *опции* графических функций Для вывода их списка надо использовать команду Options [Plot].

Опции внутри.графических функций задаются своим именем name и значением value в виде name -> value

Наиболее распространённые символьные значения опций:

Automatic — используется автоматический выбор;

None — опция не используется;

All — используется в любом случае;

True — используется;

False — не используется.

Многие опции могут иметь числовые значения. В сомнительных случаях рекомендуется уточнять форму записи опций и их значений по оперативной справочной смстеме. Рассмотрим примеры применения опций двумерной графики.

Мы уже отметили неудачный выбор масштаба в случае, представленном на рис. 1. Очевидно, этот недостаток графика легко исправить, введя коррекцию масштаба по оси *y*. Это и сделано в примере, показанном на рис. 2. Для изменения масштаба использована опция PlotRange->{-.25, 1.2}. Нетрудно догадаться, что эта опция задает пределы отображения графика по вертикали от -0.25 до 1.2.

Plot[Sin[x] / x, {x, -20, 20}, PlotRange → {- .25, 1.2}]

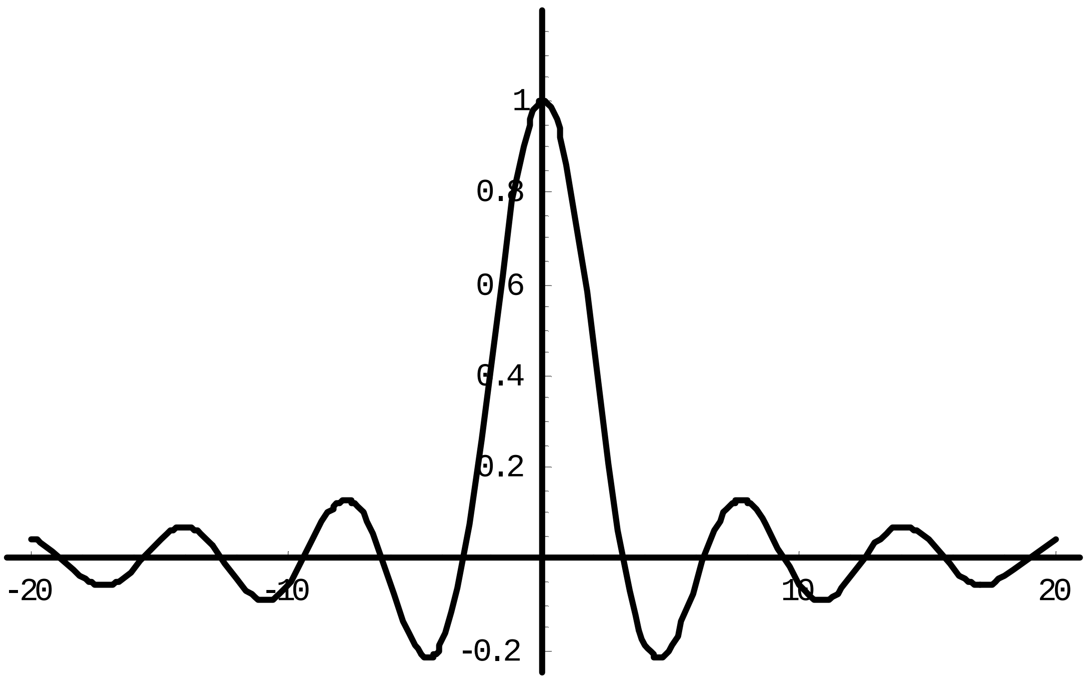


Рис. 2. График функции sin(*x*)/*x* *с масштабом, дающим его отображение в полном виде*

По умолчанию система строит графики, не указывая надписей ни по осям координат (кроме букв *x* и *y*), ни в верхней части графика. Такая надпись на графике по центру сверху называется титульной. Для создания таких надписей используется опция Axes Label. После нее указывается список, содержащий две надписи — одну для оси *x*, вторую — для оси *y*. Надписи указываются в кавычках. Таким образом, задание опции выглядит следующим образом: AxesLabel-> {"X value","f(x)}.

С помощью опции Axes со значением None можно убрать с графика отображение осей.

Перестроение и комбинирование графиков

При построении графиков часто требуется изменение их вида и тех или иных параметров и опций. Этого можно достичь повторением вычислений, но при этом скорость работы с системой заметно снижается. Для ее повышения удобно использовать специальные функции перестроения и вывода графиков, учитывающие, что узловые точки уже рассчитаны и большая часть опций уже задана. В этом случае удобно использовать следующую функцию-директиву:

Show[plot] — построение графика;

Show[plot, option -> value] — построение графика с заданной опцией;
Show[plotl, plot2,...] — построение нескольких графиков с наложением их друг на друга.

Директива Show полезна также и в том случае, когда желательно, не трогая исходные графики, просмотреть их при иных параметрах. Соответствующие опции, меняющие параметры графиков, можно включить в состав директивы Show. Другое полезное применение директивы — объединение на одном графике нескольких графиков различных функций или объединение экспериментальных точек и графика теоретической зависимости.

Разумеется, при использовании директивы Show надо побеспокоиться о выравнивании масштабов графиков, налагаемых друг на друга. Полезно особо обратить внимание на возможность присваивания графиков функций переменным (в нашем примере — g_1 и g_2) в качестве значений. Такие переменные становятся графическими объектами, используемыми директивой Show для вывода на экран дисплея.

Директива Show часто применяется, когда надо построить на одном графике кривую некоторой функции и представляющие ее узловые точки (например, при построении кривых регрессии в облаке точек исходных данных).

Графики функций, заданных в параметрической форме

Построение графиков в полярной системе координат возможно двумя способами. Первый способ основан на использовании обычной декартовой системы координат. Координаты каждой точки при этом задаются в *параметрическом* виде: $x = f_x(t)$ и $y = f_y(t)$, где независимая переменная t меняется от минимального значения t_{min} до максимального t_{max} с шагом dt. Особенно удобно применение таких функций для построения замкнутых линий, таких как окружности, эллипсы, циклоиды и т. д. Например, окружность радиусом R может быть задана в следующей параметрической форме: $x = R\cos(t)$ и $y = R\sin(t)$, если t меняется от 0 до 2π. В общем случае радиус также может быть функцией параметра t.

Для построения параметрически заданных функций используются следующие графические средства:

ParametricPlot[{fx, fy}, {t, tmin, tmax}]—строит параметрический график с координатами f_x и f_y (соответствующими x и y), получаемыми как функции от t;

ParametricPlot[{{fx, fy}, {gx, gy},...}, {t, tmin, tmax}] —строит графики нескольких параметрических кривых.

Функции f_x, f_y и т. д. могут быть как непосредственно вписаны в список параметров, так и определены как функции пользователя.

4. Язык программирования Java

4.1 Классы как новые типы данных. Поля данных и методы

Системная библиотека классов языка Java содержит классы и пакеты, реализующие различные базовые возможности языка. Методы классов, включенных в эту библиотеку, вызываются из JVM во время интерпретации Java-программы. В системе Java вся программа интерпретируется на JVM, так что возможности расширять язык, вводя все новые и новые классы и пакеты в системную библиотеку, практически неограничены. Один класс (подкласс) может наследовать переменные и методы другого класса (суперкласса), используя ключевое слово extends. Подкласс имеет доступ ко всем открытым переменным и методам родительского класса, как будто они находятся в подклассе. В то же время подкласс может иметь методы с тем же именем и сигнатурой, что и методы суперкласса. В этом случае подкласс переопределяет методы родительского класса. В следующем примере перегружаемый метод show() находится в двух классах Bird и Eagle. По правилам Java вызывается метод, наиболее близкий к текущему объекту.

```
// пример #1: наследование и перегрузка методов : BirdSample.java
class Bird {
float price;
String wage;
Bird(float p, String v) { price = p; wage = v; }//конструктор
void show(){
System.out.println("вес: " + wage + ", цена: "+ price);}
}
class Eagle extends Bird {
boolean fly;
Eagle(float p, String v, boolean f) {
super(p, v); //вызов конструктора суперкласса
fly = f;
}
void show(){
System.out.println("вес:"+ wage+", цена: "+ price+ ", полет:"+fly);
}
}
public class BirdSample {
```

```
public static void main(String[] args) {
Eagle whiteEagle = new Eagle(10.55F, "Белый Орел",true);
Bird goose = new Bird(0.85F, "Гусь");
goose.show(); // вызов show() класса Bird
whiteEagle.show(); // вызов show() класса Eagle
}
}
```

Способность Java делать выбор метода, исходя из типа времени выполнения, называется полиморфизмом. Поиск метода происходит сначала в данном классе, затем в суперклассе, пока метод не будет найден или не будет достигнут Object - суперкласс для всех классов. Отметим, что статические методы могут быть переопределены в подклассе, но не могут быть полиморфными, так как их вызов не затрагивает объекты. Нельзя создать подкласс для класса, объявленного со спецификатором final:

```
// ОШИБКА! class First не может быть суперклассом
final class First {/* код*/} // следующий class неверен
class Second extends First{/* код*/}
```

Методы классов

Все функции Java объявляются внутри классов и называются методами. Невозможно объявить функцию вне класса. Как и в С++ определение функции имеет вид:

```
return_type function_name(arglist)
{//instructions
return value;}
```

Если метод не возвращает значение, ключевое слово return отсутствует, тип возвращаемого значения будет void или отсутствует. Вместо пустого списка аргументов void не указывается. Вызов методов осуществляется из объекта или класса (для методов класса):

```
Object_name.method_name(arglist);
```

Отметим, что методы-конструкторы вызываются автоматически при создании объекта класса с помощью оператора new. Автоматически вызывается метод main() при загрузке приложения, содержащего класс и метод main(). При загрузке апплетов автоматически запускаются методы init(), start(), stop(), paint(). Для того чтобы создать метод, нужно внутри объявления класса написать объявление метода и затем реализовать его тело. Объявление метода, как минимум, должно содержать тип возвращаемого значения (возможен void) и имя метода. В объявлении метода элементы, заключенные в квадратные скобки, являются необязательными.

[доступ][static][abstract][final][native][synchronized] тип_возвращаемого_значения имя_метода(список параметров) [throw список_исключений]

Как и для полей данных спецификатор доступа к методам может быть public, private, protected и friendly (по умолчанию).

Статические методы

Переменные, объявленные как static, являются общими для всех объектов класса и называются переменными класса. Если один объект изменит значение такой переменной, то это изменение увидят все объекты. Для работы со статическими переменными используются статические методы, объявленные со спецификатором static. Такие методы являются методами класса и не содержат указателя this на конкретный объект. Вызов такого метода возможен с помощью указания: имя_класса.имя_метода

```
public class MyStaticShow {
public static int callMethod(){/* код*/}
}
x = MyStaticShow.callMethod(); // вызов статического метода.
```

Можно вызывать такие методы, например, из класса Math без объявления объекта:

```
z = Math.max(x,y);
k = Math.random(); // случайное значение
```

Final-методы

Спецификатор final используется для определения констант. Методы, объявленные как final, нельзя замещать в подклассах. Например:

```
class A {
final void method() { System.out.println("Это final метод."); }
}
class B extends A {
void method() { /* ОШИБКА! Нельзя переопределять.*/ }
}
```

Абстрактные методы

Абстрактные методы помещаются в абстрактных классах или интерфейсах, тела таких методов отсутствуют и реализуются в подклассах.

```
public abstract class AbstractShow{
public abstract void abstractMethod();
}
```

Модификатор native

Программа может вызывать методы, написанные на C++. Такие методы объявляются с ключевым словом native, которое сообщает компилятору, что метод реализован в другом месте. Например:

static native void outFunction(int x);// outFunction() функция C++

Модификатор synchronized

При использовании нескольких потоков необходимо синхронизировать методы, обращающиеся к общим данным. Когда компилятор обнаруживает synchronized, он включает код, блокирующий доступ к данным при запуске потока и снимающий блок при его завершении.

4.2 Полиморфизм. Конструкторы

Перегрузка и переопределение методов

Полное имя метода включает его метода, класс и параметры. Если два метода с одинаковыми именами находятся в одном классе, списки параметров должны отличаться. Такие методы являются перегружаемыми. Если метод подкласса совпадает с методом суперкласса (порождающего класса), то метод подкласса переопределяет метод супер-класса. Все методы Java являются виртуальными (ключевое слово virtual как в C++ не используется). Переопределение методов является основой концепции динамического связывания, реализующей полиморфизм. Когда переопределенный метод вызывается через ссылку суперкласса, Java определяет, какую версию метода вызвать, основываясь на типе объекта, на который имеется ссылка. Таким образом, тип объекта определяет версию метода на этапе выполнения. В следующем примере рассматривается реализация полиморфизма на основе динамического связывания. Так как суперкласс содержит методы, переопределенные подклассами, то объект суперкласса будет вызывать методы различных подклассов, в зависимости от того, на объект какого подкласса у него имеется ссылка.

```
//пример #1: динамическое связывание методов : DynDispatch.java
class A {
int i, j;
A(int a, int b) {
i = a;
j = b;
}
void show() { // вывод i и j
System.out.println("i и j: " + i + " " + j); }
```

```
}
class B extends A {
int k;
B(int a, int b, int c) { super(a, b); k = c; }
void show() { // вывод k: переопределенный метод show() из A
super.show(); // вывод значений из A
System.out.println("k: " + k);
}
}
class C extends B {
int m;
C(int a, int b, int c, int d) { super(a, b, c); m = d; }
void show() { // вывод m: переопределенный метод show() из B
super.show(); //вывод значений из B
System.out.println("m: " + m);
}
}
public class DynDispatch {
public static void main(String[] args) {
A Aob;
B Bob = new B(1, 2, 3);
C Cob = new C(5, 6, 7, 8);
Aob = Bob; // установка ссылки на Bob
Aob.show(); // вызов show() из B
System.out.println();
Aob = Cob; // установка ссылки на Cob
Aob.show(); // вызов show() из C
}
}
```

Результат: i и j: 1 2
k: 3
i и j : 5 6
k:7
m: 8

Отметим, что при вызове show() обращение super всегда происходит к ближайшему суперклассу.

В следующем примере экземпляр подкласса создается с помощью new, ссылка на него передается объекту суперкласса. При вызове из суперкласса соответственно вызывается метод подкласса.

```
// пример #2 : динамического вызов метода : Dispatch.java
class A {
void method() { System.out.println("метод класса A");}
}
class B extends A {
void method () { System.out.println("метод класса B");}
}
public class Dispatch {
public static void main(String[] args) {
A ob_a = new B();
ob_a. method ();
}
}
```

Вывод: метод класса B

В следующим примере приведение к базовому типу происходит в выражении: People s = new Men(); People s = new Women();

Базовый класс People предоставляет общий интерфейс для всех наследников. Порожденные классы перекрывают эти определения для обеспечения уникального поведения.

```
//пример # 3: полиморфизм: Peoples.java
class People {
void add() {/*пустая реализация*/}
void delete() {}
}
class Men extends People {
void add() { System.out.println("добавлен сотрудник-мужчина");}
void delete() {System.out.println("удален сотрудник-мужчина");}
}
class Women extends People {
void add() {System.out.println("добавлен сотрудник-женщина ");}
void delete() {System.out.println("удален сотрудник-женщина");}
}
public class Peoples {
public static People randPeople() {
switch((int)(Math.random() * 2)) {
case 0: return new Men();
case 1: return new Women();
}
```

```
}
public static void main(String[] args) {
People[] s = new People[10];
for(int i = 0; i < s.length; i++) // заполнение массива сотрудниками
s[i] = randPeople();
for(int i = 0; i < s.length; i++)
s[i].add();// вызов полиморфного метода
}
}
```

Главный класс Peoples содержит static метод randPeople(), который возвращает ссылку на случайно выбранный объект подкласса класса People каждый раз, когда он вызывается. Приведение к базовому типу производится оператором return, который возвращает ссылку на Men или Women. Функция main() содержит массив из ссылок People, заполненный вызовами randPeople(). На этом этапе известно, что имеется некоторое множество ссылок на объекты базового типа и ничего больше (не больше, чем знает компилятор). Когда происходит перемещение по этому массиву, метод add() вызывается для каждого случайным образом выбранного объекта.

Конструкторы

Конструктор - это метод, который автоматически вызывается при создании объекта класса и выполняет действия по инициализации объекта. Конструктор имеет то же имя, что и класс; вызывается не по имени, а только вместе с ключевым словом new . Конструктор не возвращает значение, но может иметь параметры и быть перегружаемым.

```
//пример # 4 : перегрузка конструктора: Book.java
class Book{
String title, publisher;
float price;
Book(String title, String publisher, float price) {/* конструктор*/}
Book(String t, float p){ //второй конструктор
title = new String(t);
price = p;
}
void printBookInfo(){/*реализация*/}
}
```

Объект класса Book может быть создан двумя способами, вызывающими один из конструкторов:

Book tips = new Book("Java 2","П.Ноутон, Г.Шилдт",49.95);

Book tips = new Book("Java 2",49.95);

Объявление можно отделить от инициализации:

```
Book tips; // объявление
tips = new Book("Java 2",49.95);// инициализация
```

Если конструктор в классе не определен, Java предоставляет конструктор по умолчанию, который инициализирует объект значениями по умолчанию.

Использование super и this

Ключевое слово super используется для вызова конструктора суперкласса и для доступа к члену суперкласса. Например:

```
super(список_параметров); // вызов конструктора суперкласса
super.i = a; //присваивание значения члену суперкласса
super(x,y); //передача параметров конструктору базового класса
```

Вторая форма super подобна ссылке this на экземпляр класса. Каждый экземпляр класса имеет неявную ссылку this на себя, которая передается также и методам. После этого можно писать this.good; this.price, хотя и необязательно. Ссылку this можно использовать в методе для доступа к переменной класса, если в методе есть локальная переменная с тем же именем. Следующий код показывает, как, используя this и перегрузку методов, можно строить одни конструкторы на основе других.

```
//пример # 5 : this в конструкторе : Locate.java
class Locate {
int x, y;
Locate(int x, int y) {
this.x = x;
this.y = y;
}
Locate() { this(-1, -1); }
}
```

В этом фрагменте второй конструктор для завершения инициализации объекта обращается к первому конструктору.

Абстрактные классы

Абстрактные классы объявляются с ключевым словом abstract и содержат объявления абстрактных методов, которые не реализованы в этих классах. Объекты таких классов создать нельзя, можно создать объекты подклассов, которые реализуют эти методы.

```
//пример #6: абстрактные методы и классы: AbstractDemo.java
abstract class Square {
abstract int squareIt(int i); //абстрактный метод
void show(){/*реализация*/}
```

```
}
//squareIt() должен быть реализован подклассом Square
class SquareTwo extends Square {
int squareIt(int i) { return i*i; }
}
public class AbstractDemo {
public static void main(String[] args) {
SquareTwo ob = new SquareTwo();
System.out.println("10 в квадрате равно " + ob.squareIt(10));
}
}
```

Результат: 10 в квадрате равно 100

Методы finalize аналогичны деструкторам в С++. Исполняющая среда Java будет вызывать его каждый раз, когда сборщик мусора соберется уничтожить объект этого класса.

4.3 Интерфейсы. Пакеты

Интерфейсы представляют полностью абстрактные классы: ни один из объявленных методов не может быть им реализован. Все методы автоматически трактуются как public и abstract, а все переменные - как public, static и final. Каждый интерфейс может быть реализован одним или несколькими классами, не связанными по иерархии наследования. Класс может реализовывать любое число интерфейсов, указываемых после ключевого слова implements, дополняющего определение класса. На множестве интерфейсов тоже определена иерархия по наследованию, но она не имеет отношения к иерархии классов. В языке Java интерфейсы обеспечивают большую часть той функциональности, которая в С++ представляется с помощью механизма множественного наследования. Определение интерфейса имеет вид:

[public] interface имя [extends I1,I2,…,IN]{/*реализация*/}

Класс также может наследовать любое число интерфейсов:

[доступ] class имя_ класса implements I1,I2, …,IN {/*…реализация…*/}

Класс, который реализует интерфейс, должен предоставить полную реализацию всех методов, объявленных в интерфейсе. Кроме этого, данный класс может объявлять свои собственные методы. Если класс включает интерфейс, но полностью не реализует его методы, то этот класс должен быть объявлен как abstract.

```
// пример #1 : интерфейс и его реализации : InterfacesDemo.java
interface Auto{
```

```
public String getMark();
public String getPassangers();
public String getWheels();
}
class Bus implements Auto{
public String getMark(){return "Mercedes";}
public String getPassangers(){return "90 человек";}
public String getWheels(){return "6 колес";}
}
class Car implements Auto{
public String getMark(){return "BMV";}
public String getPassangers(){return "5 человек";}
public String getWheels(){return "4 колеса";}
}
public class InterfacesDemo {
public static void main(String[] args){
Car c = new Car();
Bus b = new Bus();
printFeatures(c);
printFeatures(b);
}
public static void printFeatures(Auto f){
System.out.println("марка:"+f.getMark()+" вместимость:"+
f.getPassangers()+" количество колес: "+f.getWheels());
}
}
```

Класс InterfacesDemo содержит метод printFeatures(), который вызывает методы того объекта, который передается ему в качестве параметра. Вначале ему передается объект, соответствующий легковому автомобилю, затем автобусу (объекты c и b). Каким образом метод printFeatures() может обрабатывать объекты двух различных классов? Все дело в типе передаваемого этому методу аргумента - класса, реализующего интерфейс Auto. Вызывать, однако, можно только те методы, которые были объявлены в интерфейсе.

В следующем примере объявляется объектная ссылка, которая использует интерфейсный тип. В такой переменной можно сохранять экземпляр любого класса, который реализует объявленный интерфейс. Когда вызывается метод через такую ссылку, то будет вызываться его правильная версия, основанная на текущем экземпляре. Выполняемый метод разыскивается динамически во время выполнения, что позволяет создавать классы позже кода, который вызывает их методы.

```
// пример #2: динамический вызов функций: TestCall.java
interface Callback {
void callB(int param);
}
class Client implements Callback {
public void callB(int p) {
System.out.println("метод callB() вызван со значением = " + p);
}
void myMethod() {
System.out.println("класс, реализующий интерфейс " +
"может иметь свои собственные методы");
}
}
class XClient implements Callback {
public void callB(int p) {
System.out.print("другая версия метода callB():");
System.out.println("p в квадрате = " + (p*p));
}
}
public class TestCall{
public static void main(String[] args) {
Callback c = new Client();
XClient ob = new XClient();
c.callB(11);
c = ob; // c присваивается ссылка на другой объект
c.callB(11);
}
}
```

Результат: метод callB() вызван со значением = 11
другая версия метода callB(): p в квадрате = 121

Пакеты

Оператор package, помещаемый в начале исходного программного файла, определяет пакет, т. е. область в пространстве имен классов, в которой определяются имена классов, содержащихся в этом файле. Внутри указанной области можно выделить подобласти, используя тот же оператор package; действие оператора package аналогично действию объявления директории на имена файлов. Для обеспечения возможности использования коротких имен классов, помещенных в другие пакеты, используется оператор import.

Приведем общую форму исходного файла Java:
одиночный оператор package (необязателен);
любое количество операторов import (необязательны);
одиночное объявление открытого (public) класса {/*код*/}
любое количество закрытых (private) классов пакета (необязательны)
{/*код*/}

Каждый класс принадлежит некоторому пакету и добавляется в пакет при компиляции. Для добавления класса в какой-то пакет указывается этот пакет после слова package. Например:

```
// пример #3 : простейший пакет : AccountBalance.java
package com.MyPack;
class Balance {
String name; double bal;
Balance(String n, double b) {name = n; bal = b; }
void show() {
if(bal < 0) System.out.print("-->> ");
else System.out.println(name + ": $" + bal);
}
}
public class AccountBalance {
public static void main(String[] args) {
Balance current[] = new Balance[3];
current[0] = new Balance("Билл Гейтс", 150000000.0);
current[1] = new Balance("Борис Березовский", -170000.02);
current[2] = new Balance("Ясир Арафат", 666000.11);
for(int i = 0; i < 3; i++) current[i].show();
}
}
```

Файл начинается с объявления того, что данный класс принадлежит пакету MyPack. Другими словами это значит, что файл AccountBalance.java находится в каталоге /com/MyPack. Нельзя переименовывать пакет, не переименовав каталог, в котором хранятся его классы.

Если пакет не существует, он создается при первой компиляции, если пакет не указан, класс добавляется в неименованный (unnamed) пакет. Чтобы получить доступ к классу из другого пакета, перед именем класса указывается имя пакета: Pakagename.Classname. Чтобы избежать таких длинных имен, используется ключевое слово import.

```
import javax.swing.JApplet;
import java.awt.*;
```

Каталог, который транслятор Java будет рассматривать как корневой для иерархии пакетов, можно задавать с помощью переменной среды окружения CLASSPATH.

5. Варианты заданий

Вариант 1

1.С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
101-103, 253, 266, 278, 307, 435-440, 1188, 1638-1643, 1674-1679, 2986-2989, 3462, 3463, 3474.

2. Ввести класс комплексных чисел, содержащий как поля действительную и мнимую часть числа и как методы – действия над комплексными числами. Используя введенный класс, вычислить sin z с заданной точностью, используя разложение в ряд Тейлора. Аргументы вводятся из компонентов.

Вариант 2

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
103-106, 254, 267, 279 а, 308, 441-446, 1189, 1644-1649, 1679-1684, 2989-2993, 3462, 3463, 3474.

2. Ввести класс векторов, содержащий как поля координаты и как методы – действия над векторами. Вычислить |b|/(b, c)x[b, c]

Вариант 3

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
106-109, 255, 268, 279 б, 309, 447-452, 1190, 1649-1654, 1684-1689, 2993-2997, 3462, 3463, 3474.

2. Ввести класс матриц третьего порядка, содержащий как поля – элементы, как методы – действия над матрицами. Вычислить|AB|/|BC|

Вариант 4

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
109-112, 256, 269, 279 в, 310, 453-458, 1191, 1654-1659, 1690-1694, 2998-3002, 3462, 3463, 3474.

2. Ввести класс прямых в пространстве, содержащий как поля – параметры, определяющие прямую и как методы, определяющие угол пересечения прямой и произвольной плоскости и определяющие соотношение этой прямой с другой прямой. Привести пример применения введенного класса.

Вариант 5

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
112-115, 257, 270, 279 г, 308, 459-464, 1192, 1659-1664, 1695-1699, 3003-3005, 3462, 3463, 3474.

2. Ввести класс плоскостей в пространстве, содержащий как поля – параметры, определяющие плоскость и как методы, определяющие угол пересечения произвольной прямой и плоскости и определяющие соотношение этой плоскости с другой плоскостью. Привести пример применения введенного класса.

Вариант 6

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
116-119, 258, 271, 279 д, 309, 465-470, 1193, 1664-1669, 1700-1704, 3005-3007, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля – координаты, как методы – функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 7

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
119-121, 259, 272, 279 е, 310, 471-476, 1194, 1668-1673, 1705-1709, 3008-3010, 3462, 3463, 3474.

2. Ввести класс точек на плоскости, содержащий как поля – координаты, как методы – функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами.

Вариант 8

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
142, 144, 260, 273, 280, 311, 477-482, 1195, 1638-1643, 1710-1714, 3011-3013, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля – цилиндрические

координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 9

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
145, 261, 274, 281, 308, 483-488, 1196, 1643-1648, 1715-1719, 3014-3016, 3462, 3463, 3474.

2. Ввести класс точек на плоскости, содержащий как поля - полярные координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами.

Вариант 10

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
147, 148 262, 275, 282, 309, 489-494, 1197, 1648-1653, 1720-1724, 3018-3020, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля – цилиндрические координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 11

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
74, 75, 263, 276, 283, 310, 499-504, 1198, 1654-1659, 1725-1729, 3021 -3023, 3462, 3463, 3474.

2. Ввести класс комплексных чисел, содержащий как поля действительную и мнимую часть числа и как методы - действия над комплексными числами. Используя введенный класс, вычислить tg z с заданной точностью, используя разложение в ряд Тейлора. Аргументы вводятся из компонентов.

Вариант 12

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
77-79, 264, 277, 284, 311, 505-510, 1199, 1660-1665, 1730-1734, 3024-3026, 3462, 3463, 3474.

2. Ввести класс векторов, содержащий как поля координаты и как методы - действия над векторами. Вычислить (d, c)/(b, c)x[b, d].

Вариант 13

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
80-83, 265, 278, 285, 312, 511-516, 1200, 1665-1670, 1735-1739, 3027-3029, 3462, 3463, 3474.

2. Ввести класс матриц третьего порядка, содержащий как поля - элементы, как методы - действия над матрицами. Вычислить |AB|/|BC|xA

Вариант 14

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
96-98, 253, 279, 286, 313, 517-522, 1201, 1668-1673, 1740-1744, 3031-3033, 3462, 3463, 3474.

2. Ввести класс прямых в пространстве, содержащий как поля - параметры, определяющие прямую и как методы, определяющие точку пересечения прямой и произвольной плоскости и определяющие соотношение этой прямой с другой прямой. Привести пример применения введенного класса.

Вариант 15

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
99-101, 254, 280, 287, 314, 523-528, 1202, 1638-1643, 1745-1749, 2986-2988, 3462, 3463, 3474.

2. Ввести класс плоскостей в пространстве, содержащий как поля - параметры, определяющие плоскость и как методы, определяющие точку пересечения произвольной прямой и плоскости и определяющие соотношение этой плоскости с другой плоскостью. Привести пример применения введенного класса.

Вариант 16

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
102-104, 255, 281, 288, 315, 529-534, 1203, 1644-1649, 1750-1754, 2989-2991, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля - координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 17

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
105-108, 256, 282, 289, 316, 535-540, 1204, 1650-1654, 1754-1759, 2992-2995, 3462, 3463, 3474.

2. Ввести класс точек на плоскости, содержащий как поля - координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами.

Вариант 18

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
109-112, 257, 283, 290 , 317, 541-546, 1205, 1654-1659, 1760-1764, 2997-2999, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля - цилиндрические

координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 19

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
113-116, 258, 284, 291, 318, 547-552, 1206, 1659-1664, 1765-1769, 3000-3002, 3462, 3463, 3474.

2. Ввести класс точек на плоскости, содержащий как поля - полярные координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами.

Вариант 20

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
117-120, 259, 285, 292, 319, 553-558, 1207, 1665-1670, 1770-1774, 3003-3005, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля – цилиндрические координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 21

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
142, 144, 260, 286, 293, 320, 563-569, 1208, 1668-1673, 1775-1779, 3006-3008, 3462, 3463, 3474.

2. Ввести класс комплексных чисел, содержащий как поля действительную и мнимую часть числа и как методы - действия над комплексными числами. Используя введенный класс, вычислить tg z с заданной точностью, используя разложение в ряд Тейлора. Аргументы вводятся из компонентов.

Вариант 22

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
145, 261, 287, 294, 321, 570-575, 1209, 1638-1643, 1780-1784, 3009-3011, 3462, 3463, 3474.

2. Ввести класс векторов, содержащий как поля координаты и как методы - действия над векторами. Вычислить |b|/(b, c)x[b, c]

Вариант 23

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
146, 147, 262, 288, 295, 322, 576-580, 1210, 1644-1649, 1785-1789, 3012-3015, 3462, 3463, 3474.

2. Ввести класс матриц третьего порядка, содержащий как поля - элементы, как методы - действия над матрицами. Вычислить |AB|/|BC|

Вариант 24

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
101-103, 263, 289, 296, 323, 581-585, 1211, 1650-1654, 1790-1794, 3016-3018, 3462, 3463, 3474.

2. Ввести класс прямых в пространстве, содержащий как поля - параметры, определяющие прямую и как методы, определяющие угол пересечения прямой и произвольной плоскости и определяющие соотношение этой прямой с другой прямой. Привести пример применения введенного класса.

Вариант 25

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)

104-106, 264, 290, 297, 324.1а, 586-570, 1212, 1654-1659, 1795-1799, 3019-3021, 3462, 3463, 3474.

2. Ввести класс плоскостей в пространстве, содержащий как поля - параметры, определяющие плоскость и как методы, определяющие угол пересечения произвольной прямой и плоскости и определяющие соотношение этой плоскости с другой плоскостью. Привести пример применения введенного класса.

Вариант 26

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
107-110, 265, 291, 298, 324.1б, 571-575, 1188, 1659-1664, 1800-1804, 3023-3025, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля - координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 27

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
111-114, 253, 266, 279 а, 307, 576-580, 1189, 1665-1670, 1805-1809, 3026-3028, 3462, 3463, 3474.

2. Ввести класс точек на плоскости, содержащий как поля - координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами.

Вариант 28

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
115-118, 254, 267, 279 б, 308, 581-585, 1190, 1638-1643, 1810-1814, 3029-3031, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля – цилиндрические координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 29

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)

101-103, 255, 268, 279 в, 309, 586-590, 1191, 1644-1649, 1815-1819, 3031-3033, 3462, 3463, 3474.

2. Ввести класс точек на плоскости, содержащий как поля - полярные координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами.

Вариант 30

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)

104-106, 256, 269, 279 г, 310, 591-595, 1192, 1650-1654, 1824-1829, 2986-2988, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля – цилиндрические координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 31

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)

107-110, 257, 270, 279 д, 311, 596-600, 1193, 1655-1659, 1830-1834, 2989-2991, 3462, 3463, 3474.

2. Ввести класс комплексных чисел, содержащий как поля действительную и мнимую часть числа и как методы - действия над комплексными числами. Используя введенный класс, вычислить tg z с заданной точностью, используя разложение в ряд Тейлора. Аргументы вводятся из компонентов.

Вариант 32

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)

111-114, 258, 271, 279 е, 312, 435-440, 1194, 1650-1654, 1835-1839, 2991-2993, 3462, 3463, 3474.

2. Ввести класс векторов, содержащий как поля координаты и как методы - действия над векторами. Вычислить (d, c)/(b, c)x[b, d].

Вариант 33

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)

115-118, 259, 272, 280, 308, 441-445, 1195, 1655-1660, 1840-1844, 2995-2997, 3462, 3463, 3474.

2. Ввести класс матриц третьего порядка, содержащий как поля - элементы, как методы - действия над матрицами. Вычислить |AB|/|BC|xA

Вариант 34

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
119-121, 260, 273, 281, 309, 446-450, 1196, 1661-1665, 1845-1849, 2998-3000, 3462, 3463, 3474.

2. Ввести класс прямых в пространстве, содержащий как поля - параметры, определяющие прямую и как методы, определяющие точку пересечения прямой и произвольной плоскости и определяющие соотношение этой прямой с другой прямой. Привести пример применения введенного класса.

Вариант 35

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
101-103, 261, 274, 282, 310, 451-455, 1197, 1666-1670, 1850-1854, 3001-3003, 3462, 3463, 3474.

2. Ввести класс плоскостей в пространстве, содержащий как поля - параметры, определяющие плоскость и как методы, определяющие точку пересечения произвольной прямой и плоскости и определяющие соотношение этой плоскости с другой плоскостью. Привести пример применения введенного класса.

Вариант 36

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
104-106, 262, 275, 283, 311, 456-460, 1198, 1668-1673, 1859-1864, 3004-3006, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля - координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 37

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
107-109, 263, 276, 284, 312, 461-465, 1199, 1638-1643, 1865-1869, 3007-3009, 3462, 3463, 3474.

2. Ввести класс точек на плоскости, содержащий как поля - координаты, как методы - функции, определяющие расстояние этой точки до другой точки

и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами.

Вариант 38

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
110-113, 264, 267, 285, 313, 466-470, 1200, 1644-1649, 1870-1874, 3010-3012, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля – цилиндрические координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 39

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
114-117, 265, 268, 286, 314, 471-475, 1201, 1650-1654, 1875-1879, 3013-3015, 3462, 3463, 3474.

2. Ввести класс точек на плоскости, содержащий как поля - полярные координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами.

Вариант 40

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
117-120, 265, 268, 286, 314, 476-480, 1202, 1659-1664, 1880-1884, 3016-3018, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля – цилиндрические координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 41

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
142, 144, 260, 273, 280, 311, 477-482, 1195, 1638-1643, 1710-1714, 3011-3013, 3462, 3463, 3474.

2. Ввести класс комплексных чисел, содержащий как поля действительную и мнимую часть числа и как методы - действия над

комплексными числами. Используя введенный класс, вычислить sin z с заданной точностью, используя разложение в ряд Тейлора. Аргументы вводятся с компонентов.

Вариант 42

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
145, 261, 274, 281, 308, 483-488, 1196, 1643-1648, 1715-1719, 3014-3016, 3462, 3463, 3474.

2. Ввести класс векторов, содержащий как поля координаты и как методы - действия над векторами. Вычислить |b|/(b, c)x[b, c]

Вариант 43

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
147, 148 262, 275, 282, 309, 489-494, 1197, 1648-1653, 1720-1724, 3018-3020, 3462, 3463, 3474.

2. Ввести класс матриц третьего порядка, содержащий как поля - элементы, как методы - действия над матрицами. Вычислить |AB|/|BC|

Вариант 44

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
74, 75, 263, 276, 283, 310, 499-504, 1198, 1654-1659, 1725-1729, 3021 -3023, 3462, 3463, 3474.

2. Ввести класс прямых в пространстве, содержащий как поля - параметры, определяющие прямую и как методы, определяющие угол пересечения прямой и произвольной плоскости и определяющие соотношение этой прямой с другой прямой. Привести пример применения введенного класса.

Вариант 45

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
105-108, 253, 266, 278 а, 307, 435-440, 1188, 1638-1648, 1674-1674, 2578-2581, 2986-2989, 3462, 3463, 3474.

2. Ввести класс плоскостей в пространстве, содержащий как поля - параметры, определяющие плоскость и как методы, определяющие угол пересечения произвольной прямой и плоскости и определяющие соотношение этой плоскости с другой плоскостью. Привести пример применения введенного класса.

Вариант 46

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
77-79, 264, 277, 284, 311, 505-510, 1199, 1660-1665, 1730-1734, 3024-3026, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля - координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 47

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
80-83, 265, 278, 285, 312, 511-516, 1200, 1665-1670, 1735-1739, 3027-3029, 3462, 3463, 3474.

2. Ввести класс точек на плоскости, содержащий как поля - координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами.

Вариант 48

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
96-98, 253, 279, 286, 313, 517-522, 1201, 1668-1673, 1740-1744, 3031-3033, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля – цилиндрические координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

Вариант 49

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
99-101, 254, 280, 287, 314, 523-528, 1202, 1638-1643, 1745-1749, 2986-2988, 3462, 3463, 3474.

2. Ввести класс точек на плоскости, содержащий как поля - полярные координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами.

Вариант 50

1. С помощью пакета Mathematica решить следующие номера из «Сборника задач и упражнений» Б.П. Демидовича (год издания 2003)
102-104, 255, 281, 288, 315, 529-534, 1203, 1644-1649, 1750-1754, 2989-2991, 3462, 3463, 3474.

2. Ввести класс точек в пространстве, содержащий как поля – цилиндрические координаты, как методы - функции, определяющие расстояние этой точки до другой точки и расстояние точки до произвольной прямой. Найти площадь треугольника, заданного своими координатами в пространстве.

6. Пример выполнения работы

6.1 Работа в системе TeX

Исходный файл в TeX

```
\documentclass[12pt]{article}

\usepackage[T2A]{fontenc}
\usepackage[cp866]{inputenc}
\usepackage{amsthm,amssymb}
\usepackage[russian]{babel}

\selectlanguage{russian}

\sloppy
\oddsidemargin=0pt
\evensidemargin=0pt
\textwidth=16cm
\textheight=23cm
\topmargin=-2cm
\renewcommand{\thefootnote}{*)}

\begin{document}

{\textbf {\emph{Лемма}. }} {\it Если $u$ --- полином степени не
выше $n$, то справедливы неравенства}

$$
\|u\|_{{\bf C}_D}\leqslant A_1n^2\|u\|_{{\bf L}^2_D}, \quad
\|u\|_{{\bf C}_{D_1}}\leqslant B_1n\|u\|_{{\bf L}^2_D};
\eqno{(13)}
$$

$$
\|u\|_{{\bf C}_D^{(1)}}\leqslant A_2n^2\|u\|_{{\bf W}_2^{(1)}(D)},
\quad \|u\|_{{\bf C}_{D_1}^{(1)}}\leqslant B_1n\|u\|_{{\bf
W}_2^{(1)}D}, \eqno{(14)}
$$
```

{\noindent \it где D_1 означает внутреннюю подобласть области
D, а через ${\bf C}^{(1)}$ обозначено пространство
непрерывно-дифференцируемых функций.}

{Д\,о\,к\,а\,з\,а\,т\,е\,л\,ь\,с\,т\,в\,о}. Ограничимся случаем,
когда подобласть D
--- круг с центром в начале координат (общий случай требует
небольших дополнительных рассуждений). Полагая

$$
\varphi(s,t)=\int\limits_0^s \int\limits_0^t p^2(s,t) \, ds \, dt,
$$

{\noindent имеем}

$$
|\varphi(s,t)|\leqslant \|p\|^2_{\bf L} \quad ((s,t) \in D),
$$

{\noindent а на основании двукратного применения неравенства (11)}

$$
|p(s,t)|^2=\left|\frac{\partial^2\varphi(s,t)}{\partial s \partial
t}\right|\leqslant A^2(2n+2)^2(2n+1)^2 \max|\varphi(s,t)|\leqslant
A^2_1n^4\|p\|_{{\bf L}^2}^2,
$$

{\noindent откуда получается первое из неравентсв (13). Второе
неравентство устанавливается аналогично, надо только вместо (11)
использовать (12).}

Применяя установленное неравенство к производной, будем иметь

$$
\left|\frac{\partial p(s,t)}{\partial s}\right| \leqslant A_1n^2
\left\|\frac{\partial p}{\partial s}\right\|_{{\bf L}^2}\leqslant
A_1n^2\|p\|_{{\bf W}_2^{(1)}};
$$

```
{\noindent аналогичное неравенство верно и для производной
$\frac{\partial p}{\partial t}$. Наконец, в силу (11) и на
основании теоремы вложения}

$$
|p(s,t)|\leqslant An^2\|p\|_{{\bf L}^2}\leqslant CAn^2\|p\|_{{\bf
W}_2^{(1)}}.
$$

{\noindent Сопоставляя все это, приходим к первому из неравенств
(14):}

$$
\|p\|_{{\bf C}_D^{(1)}}=\max_D|p(s,t)|+\max_D\left|\frac{\partial
p(s,t)}{\partial s}\right|+\max_D\left|\frac{\partial p
(s,t)}{\partial t}\right|\leqslant A_2n^2\|p\|_{{\bf W}_2^{(1)}}.
$$

Второе неравенство получается аналогично.

\end{document}
```

Результат на экране и печати

(см. на следующей странице)

Лемма. *Если u — полином степени не выше n, то справедливы неравенства*

$$\|u\|_{\mathbf{C}_D} \leqslant A_1 n^2 \|u\|_{\mathbf{L}^2_D}, \quad \|u\|_{\mathbf{C}_{D_1}} \leqslant B_1 n \|u\|_{\mathbf{L}^2_D}; \tag{13}$$

$$\|u\|_{\mathbf{C}^{(1)}_D} \leqslant A_2 n^2 \|u\|_{\mathbf{W}^{(1)}_2(D)}, \quad \|u\|_{\mathbf{C}^{(1)}_{D_1}} \leqslant B_1 n \|u\|_{\mathbf{W}^{(1)}_2 D}, \tag{14}$$

где D_1 означает внутреннюю подобласть области D, а через $\mathbf{C}^{(1)}$ обозначено пространство непрерывно-дифференцируемых функций.

Доказательство. Ограничимся случаем, когда подобласть D — круг с центром в начале координат (общий случай требует небольших дополнительных рассуждений). Полагая

$$\varphi(s,t) = \int\limits_0^s \int\limits_0^t p^2(s,t)\, ds\, dt,$$

имеем

$$|\varphi(s,t)| \leqslant \|p\|^2_{\mathbf{L}} \quad ((s,t) \in D),$$

а на основании двукратного применения неравенства (11)

$$|p(s,t)|^2 = \left|\frac{\partial^2 \varphi(s,t)}{\partial s \partial t}\right| \leqslant A^2 (2n+2)^2 (2n+1)^2 \max |\varphi(s,t)| \leqslant A_1^2 n^4 \|p\|^2_{\mathbf{L}^2},$$

откуда получается первое из неравентсв (13). Второе неравентство устанавливается аналогично, надо только вместо (11) использовать (12).

Применяя установленное неравенство к производной, будем иметь

$$\left|\frac{\partial p(s,t)}{\partial s}\right| \leqslant A_1 n^2 \left\|\frac{\partial p}{\partial s}\right\|_{\mathbf{L}^2} \leqslant A_1 n^2 \|p\|_{\mathbf{W}^{(1)}_2};$$

аналогичное неравенство верно и для производной $\frac{\partial p}{\partial t}$. Наконец, в силу (11) и на основании теоремы вложения

$$|p(s,t)| \leqslant A n^2 \|p\|_{\mathbf{L}^2} \leqslant C A n^2 \|p\|_{\mathbf{W}^{(1)}_2}.$$

Сопоставляя все это, приходим к первому из неравенств (14):

$$\|p\|_{\mathbf{C}^{(1)}_D} = \max_D |p(s,t)| + \max_D \left|\frac{\partial p(s,t)}{\partial s}\right| + \max_D \left|\frac{\partial p(s,t)}{\partial t}\right| \leqslant A_2 n^2 \|p\|_{\mathbf{W}^{(1)}_2}.$$

Второе неравенство получается аналогично.

6.2 Работа в системе Mathematica

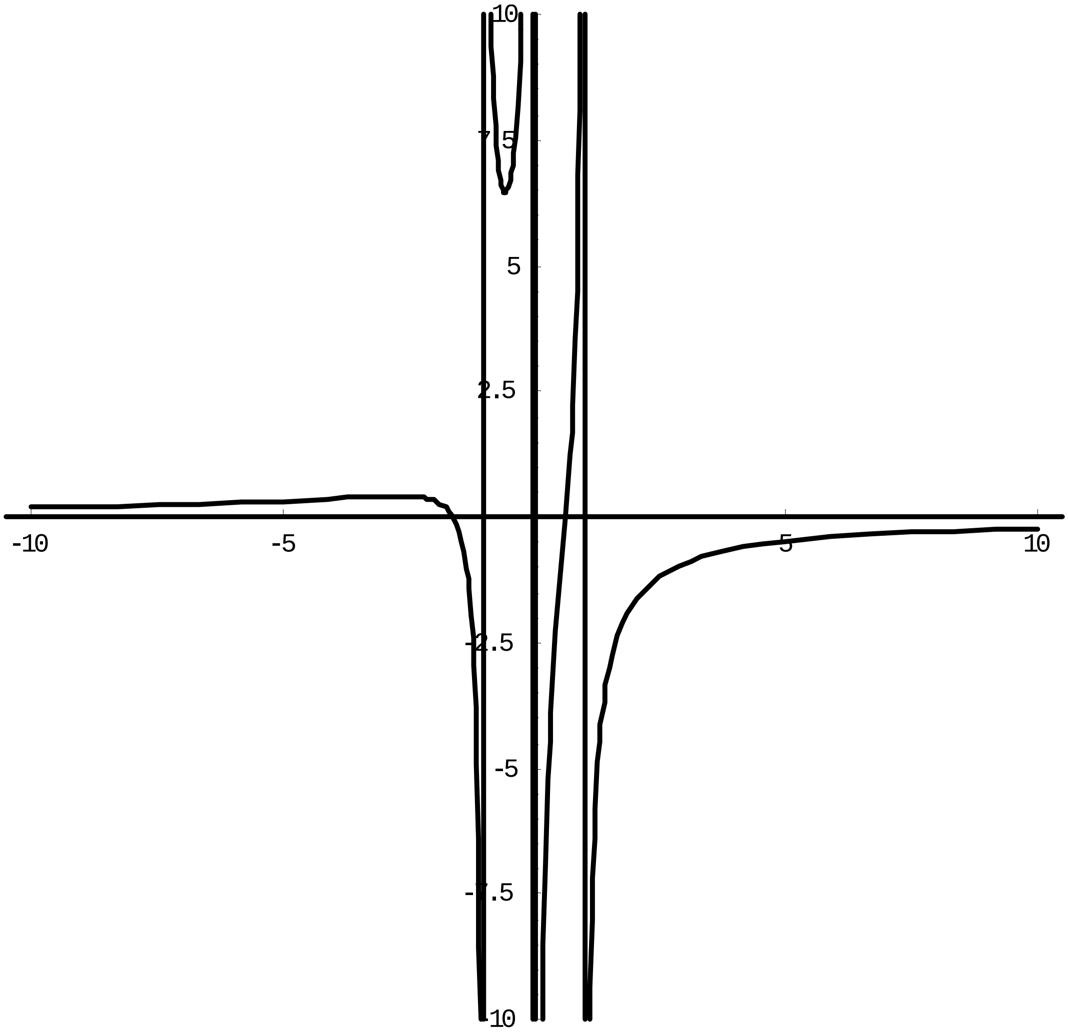
№ 119. Найти частичные пределы последовательности $x_n = 3\left(1 - \frac{1}{n}\right) + 2(-1)^n$

Limit[Re[3 (1 - 1/n) + 2 (-1)^n], n -> Infinity]
Interval[{1, 5}]

Ответ: Верхний предел = 5; нижний предел = 1.

№ 260 Построить график дробно-рациональной функции $y = \frac{1}{1+x} - \frac{2}{x} + \frac{1}{1-x}$

Plot[1/(1+x) – 2/x + 1/(1-x), {x, -3, 3}]



№ 448 Найти предел $\lim\limits_{x \to 0} \frac{\sqrt{1+x} - \sqrt{1-x}}{\sqrt[3]{1+x} - \sqrt[3]{1-x}}$

Limit[((1+x)^1/2 – (1-x)^1/2)/((1+x)^1/3 – (1-x)^1/3), x -> 0]
3/2

№ 1196 Найти $y^{(n)}$, если $y=cos^3x$

TrigReduce[Cos[x]^3]

1/4 (3 Cos[x] + Cos[3x])

Найдем $Cos^{(n)}$[x]

D[Cos[x], {x, 1}]

D[Cos[x], {x, 2}]

D[Cos[x], {x, 3}]

D[Cos[x], {x, 4}]

-Sin[x]

-Cos[x]

Sin[x]

Cos[x]

Индукционное предположение $Cos^{(n)}$[x]=Cos[π n/2 + x]

D[Cos[π n/2 + x], x]

-Sin[n π/2 + x]

Simplify[-Sin[n π/2 + x]- Cos[π n/2 + x]]

0

Следовательно, $Cos^{(n)}$[x]=Cos[π n/2 + x]

и $Cos^{(n)}$[3x]=3^nCos[π n/2 + 3x]

Ответ: $(Cos^3[x])^{(n)}$ = 3/4 Cos[π n/2 + x] + 3^{n+1}Cos[π n/2 + 3 x]

№ 1664 Найти интеграл $\int \frac{dx}{\sqrt{-2+3x^2}}$

Integrate[1/Sqrt[-2 + 3 x^2], x]

$$\frac{Log[3x+\sqrt{-6+9x^2}]}{\sqrt{3}}$$

№ 1845 Найти интеграл $\int \frac{dx}{\sin x + 2\cos x + 3}$

Integrate[1 / (Sin[x] + 2 Cos[x] + 3), x]

-ArcTan[2 Cos[x/2] / (Cos[x/2] + Sin[x/2])]

FullSimplify[%]

-ArcTan[2 / (1+Tan[x/2])

№ 2999 Найти сумму ряда $\sum_{n=0}^{\infty} \frac{(-1)^n n}{(2n+1)!}$

Sum[(-1)^n n/(2 n + 1)!,{n, 0, Infinity}]

$$-\frac{1}{2}\sqrt{\frac{\pi}{2}}BesselJ\left[\frac{3}{2},1\right]$$

N[%]
-0.150584
№ 3462 Принимая *u* и *v* за независимые переменные, преобразовать уравнение $x\frac{\partial z}{\partial x}+\sqrt{1+y^2}\frac{\partial z}{\partial y}=xy$, если $u=\ln x$ и $v=\ln\left(y+\sqrt{1+y^2}\right)$.

Solve[u = = Log[x], x]
{{x → e^u}}
Solve[v = = Log[y + Sqrt[1 + y^2]], y]
{{y → 1/2 e^{-v}(-1+e^{2v})}}
u[x_, y_] = Log[x]
v[x_, y_] = Log[y + Sqrt[1 + y^2]]

x D[z[x,y],x] + Sqrt[1 + y^2] D[z[x,y],y] = = x y /.
{D[z[x,y],x] -> D[z[u,v],u] D[u[x,y],x] + D[z[u,v],v] D[v[x,y],x],
D[z[x,y],y] -> D[z[u,v],u] D[u[x,y],y] + D[z[u,v],v] D[v[x,y],y]}

$$\frac{\sqrt{1+y^2}\left(1+y\big/\sqrt{1+y^2}\right)z^{(0,1)}[u,v]}{y+\sqrt{1+y^2}}+z^{(1,0)}[u,v]==xy$$

% /. {x -> Exp[u], y -> 1/2 Exp[-v] (-1+Exp[2 v])}

$$\left(\sqrt{1+\frac{1}{4}e^{-2v}(-1+e^{2v})^2}\left(1+\frac{e^{-v}(-1+e^{2v})}{2\sqrt{1+e^{-2v}(-1+e^{2v})^2/4}}\right)z^{(0,1)}[u,v]\right)\Big/$$
$$\left(\frac{1}{2}e^{-v}(-1+e^{2v})+\sqrt{1+\frac{1}{4}e^{-2v}(-1+e^{2v})^2}\right)+z^{(1,0)}[u,v]==\frac{1}{2}e^{u-v}(-1+e^{2v})$$

FullSimplify[%]
$z^{(0,1)}$[u,v] + $z^{(1,0)}$[u,v] = = e^uSinh[v]

Ответ: $\frac{\partial z}{\partial u}+\frac{\partial z}{\partial v}=e^u\,sh\,v$

6.3 Работа в Java

Ввести класс прямых в пространстве, содержащий как поля – параметры, определяющие прямую, и методы, определяющие точку пересечения прямой и произвольной плоскости и определяющие соотношение этой прямой с другой прямой. Привести пример применения введенного класса.

Класс Line задает прямые.

class Line {

```
  double a,b,c;
  double x,y,z;
  //(a,b,c) - направляющий вектор, (x,y,z) - точка на прямой.

Line(double a1, double b1, double c1, double x1, double y1, double z1) {
    this.a=a1;
    this.b=b1;
    this.c=c1;
    this.x=x1;
    this.y=y1;
    this.z=z1;
  }

double det2(double u, double v, double w, double t){
    return (u*t-v*w);
  }

 double det3(double u1, double u2, double u3, double v1, double v2, double v3,
double w1, double w2, double w3){
    return (u1*v2*w3+u2*v3*w1+v1*w2*u3-u3*v2*w1-u1*v3*w2-u2*v1*w3);
  }

  String WithLine(Line line2){
    String s="";
    double eps=0.000001;
    if (Math.abs(det2(this.a,line2.a,this.b,
line2.b))<eps&&Math.abs(det2(this.b,line2.b,this.c,line2.c))<eps&&Math.abs(det
2(this.a,line2.a,this.c,line2.c))<eps){
       s="прямые параллельны";
    }else
      {if(det3(this.a,this.b,this.c,line2.a,line2.b,line2.c,this.x-line2.x,this.y-
line2.y,this.z-line2.z)==0){
        s="прямые пересекаются";
      }else s="прямые скрещиваются";}

    return s;
  }

String WithPlane(double A, double B, double C, double D){
```

```
String s="";
if(A*this.a+B*this.b+C*this.c==0){
  if(A*this.x+B*this.y+C*this.z+D==0){
    s="прямая лежит в данной плоскости";
  }else {s="прямая и плоскость параллельны";}
}else {
 double t=(-D-A*this.x-B*this.y-C*this.z)/(A*this.a+B*this.b+C*this.c);
 s="Прямая и плоскость пересекаются в точке "+"("+(this.x+this.a*t)+", "
+(this.y+this.b*t)+", "+(this.z+this.c*t)+")";
}
return s;
 }
}
```

Описание:

- Методы *det2* и *det3* – вспомогательные, считают определители 2-го и 3-го порядка соответственно.
- Метод *String WithLine(Line line2)* определяет соотношения данной прямой с прямой *line2* класса *Line*. В качестве результата выдает строку “прямые пересекаются”, либо “прямые параллельны”, либо “прямые скрещиваются”.
- Метод *String WithPlane(double A, double B, double C, double D)* определяет соотношение данной прямой с плоскостью, заданной уравнением Ax+By+Cz+D=0. В качестве результата выдает строку “прямая и плоскость параллельны”, либо “прямая лежит в плоскости”, либо “прямая и плоскость пересекаются в точке (p,q,s)”, где (p,q,s) – координаты точки пересечения.

Пример использования:

Класс *LineTest2* определяет соотношения данной прямой *myLine* с прямыми *line1*, *line2* и *line3* и плоскостями, заданными уравнениями *3x-z+1=0*, *x+y-z=0* и *5x+y-3z+6=0.*

```
public class LineTest2{
  public static void main(String args[]){
    System.out.println("Прямая задается направляющим вектором и своей
точкой");
```

```
    Line myLine=new Line(1,2,3,1,2,3); //Данная прямая
    System.out.println("Данная прямая: направляющий вектор ("+myLine.a+", "+myLine.b+", "+myLine.c+"); "+"точка на прямой ("+myLine.x+", "+myLine.y+", "+myLine.z+")");
    Line line1=new Line(1,2,4,2,3,5);
    Line line2=new Line(2,4,6,2,2,2);
    Line line3=new Line(0,0,1,2,4,5);
    Line line4=new Line(2,4,6,3,6,9);
    System.out.println("1-я прямая: направляющий вектор ("+line1.a+", "+line1.b+", "+line1.c+"); "+"точка на прямой ("+line1.x+", "+line1.y+", "+line1.z+"):   "+myLine.WithLine(line1));
    System.out.println("2-я прямая: направляющий вектор ("+line2.a+", "+line2.b+", "+line2.c+"); "+"точка на прямой ("+line2.x+", "+line2.y+", "+line2.z+"):   "+myLine.WithLine(line2));
    System.out.println("3-я прямая: направляющий вектор ("+line3.a+", "+line3.b+", "+line3.c+"); "+"точка на прямой ("+line3.x+", "+line3.y+", "+line3.z+"):   "+myLine.WithLine(line3));
    System.out.println("4-я прямая: направляющий вектор ("+line4.a+", "+line4.b+", "+line4.c+"); "+"точка на прямой ("+line4.x+", "+line4.y+", "+line4.z+"):   "+myLine.WithLine(line4));
    System.out.println("Соотношения с плоскостью, заданной уравнением 3x-z+1=0: "+myLine.WithPlane(3,0,-1,1));
    System.out.println("Соотношения с плоскостью, заданной уравнением x+y-z=0: "+myLine.WithPlane(1,1,-1,0));
    System.out.println("Соотношения с плоскостью, заданной уравнением 5x+y-3z+6=0: "+myLine.WithPlane(5,1,-3,6));
  }
}
```

Результат выполнения:

Прямая задается направляющим вектором и своей точкой
Данная прямая: направляющий вектор (1.0, 2.0, 3.0);
точка на прямой (1.0, 2.0, 3.0)
1-я прямая: направляющий вектор (1.0, 2.0, 4.0);
точка на прямой (2.0, 3.0, 5.0):   прямые пересекаются
2-я прямая: направляющий вектор (2.0, 4.0, 6.0);
точка на прямой (2.0, 2.0, 2.0):   прямые параллельны
3-я прямая: направляющий вектор (0.0, 0.0, 1.0);

точка на прямой (2.0, 4.0, 5.0):   прямые пересекаются
4-я прямая: направляющий вектор (2.0, 4.0, 6.0);
точка на прямой (3.0, 6.0, 9.0):   прямые совпадают
Соотношения с плоскостью, заданной уравнением 3x-z+1=0:
прямая и плоскость параллельны
Соотношения с плоскостью, заданной уравнением x+y-z=0:
прямая лежит в данной плоскости
Соотношения с плоскостью, заданной уравнением 5x+y-3z+6=0:
прямая и плоскость пересекаются в точке (3.0, 6.0, 9.0)

Литература

1. Демидович Б.П. Сборник задач и упражнений по математическому анализу: Учеб. Пособие для ун-тов и пед.вузов / Б.П. Демидович. – СПб.: Мифрил, 2003. – 489 с.
2. Дьяконов В. Mathematica 4: Учеб. курс / В.Дьяконов. и др. — СПб.: Питер, 2001.— 654 с.
3. Львовский С.М. Набор и верстка пакета LaTeX. –М.: Космоинформ, 1994. – 328 с.
4. Основы программирования на языке Java: Метод. Указ. По курсу "Методы программирования": Для студентов механико-мат. фак. / Авт. сост. В.С. Романчик, И.Н. Блинов. – Мн.: БГУ, 2002. – 80 с.
5. Хабибуллин И. Ш.. Самоучитель Java. — СПб.: BHV-Петербург, 2001. — 464 с.

Содержание