МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Методические указания

по выполнению лабораторных работ по
дисциплине

Компьютерная алгебра на Python

для студентов направления подготовки

02.03.01 - Математика и компьютерные науки
Профиль подготовки
Анализ данных и искусственный интеллект
Квалификация выпускника **бакалавр**

Оглавление

Лабораторная работа №1

Система компьютерной алгебры Maple

Пакет Maple – интерактивная программа, позволяющая проводить аналитические выкладки и вычисления, снабженная средствами двумерной и трехмерной графики, имеющая мощный язык программирования и богатую библиотеку математических формул и сведений. Работа с Maple заключается в том, что пользователь вводит математические выражения и инструкции (команды), а система пытается их выполнить и представить ответ. Получив (или не получив) ответ, пользователь вводит новые инструкции и так далее – взаимодействие с пакетом происходит в диалоговом режиме. Благодаря собственному языку программирования высокого уровня, введенные выражения и инструкции, а также результаты выполнения команд – формулы, графики, таблицы и числа – запоминаются в едином документе, и итоговый документ становится (быть может, с минимальными дополнениями) научной статьей, разделом в учебнике, отчетом.

Графический интерфейс Maple аналогичен имеющемуся в системах редактирования и подготовки текста и использует обычные средства работы с файлами и редактирования (мышь и клавиатура). После запуска выполняемого модуля wmaple или xmaple в среде Unix появляется оболочка с новым документом (worksheet)

Работа в Maple проходит в режиме сессии (session) – пользователь вводит команды, математические выражения, процедуры, которые воспринимаются и интерпретируются Maple. Каждая команда должна завершаться точкой с запятой (;) или двоеточием (:). В первом случае в строке под предложением будет выведен результат исполнения команды или сообщение об ошибке, во втором случае результат не выводится. Для отмены всех сделанных назначений и начала нового сеанса без выхода из Maple используется команда restart.

Кроме того, в Maple можно вводить таблицы и текстовые параграфы, структурировать текст и документ, добавлять гиперссылки, объединяющие несколько документов в подобие электронной книги. В документ также можно вставлять объекты (рисунки и таблицы) из других программ, используя интерфейс OLE2.

Отметим, что различные документы, открытые в одном сеансе, используют общую область памяти, и значение, присвоенное переменной в одном документе, сохраняется при переходе к другому документу.

Результаты работы могут быть сохранены в файлах различных форматов. Текущий документ (области ввода и вывода, комментарии, текст, графика) записываются в файл с расширением .mws. При записи в файлы с другими расширениями сохраняются только области ввода и тексты комментариев. Кроме того, весь документ или его часть могут быть сохранены в форматах, допускающих их использование в других программах.

Пакет Maple постоянно развивается, приобретая новые команды и, соответственно, возможности для проведения математических выкладок, вычислений, графического отображения информации. При этом происходят изменения в синтаксисе и оболочке. Для сохранения преемственности с документами, подготовленными в старых версиях, фирма снабжает свой продукт специальными конверторами, при загрузке файла устаревшей версии Maple предлагает преобразовать нотацию файла согласно новым правилам.

Набор и исполнение команд производится по группам (Execution Group). Каждая группа состоит из нескольких частей: область ввода (Input Region) с командами, область вывода (Output Region) с результатами выполнения и тексты комментариев (Text Region). Область вывода может включать результаты выполнения математических и алгоритмических операций, а также графические образы (двумерная и трехмерная графика). Группы выделяются слева квадратными скобками и разделяются сепараторами (Separator), которые по умолчанию невидимы, но могут быть сделаны видимыми. Для структурирования документа используются параграфы.

Системы аналитических вычислений оперируют с множеством объектов, используя для работы различные типы данных. Это позволяет применять свои правила обработки для каждого типа. Простейшими объектами в Maple являются числа, константы, строки и имена. Под числами будем понимать собственно целые и вещественные числа, а также рациональные дроби и радикалы (корни из чисел). Приведём примеры чисел:

```
> 1.23, 4e5, I+Pi, sqrt(8);
```

Использование рациональных чисел, радикалов и констант (число π, мнимая единица) позволяет проводить абсолютно точные вычисления, так как при выполнении операций не возникает погрешности округления. Сразу отметим, что эти операции производятся гораздо медленнее.

Синтаксис и выражения

Каждая введенная команда должна завершаться разделителем: точкой с запятой (;) или двоеточием (:). Если ввод предложения завершается точкой с запятой, то в строке под предложением сразу будет отклик: результат исполнения команды или сообщение об ошибке. Разделитель (:) используется для отмены вывода, когда команда выполняется системой, но ее результат не выводится на экран. В Maple применяются круглые, квадратные и фигурные скобки. Назначение круглых скобок – задавать порядок при построении математических выражений и обрамлять аргументы функций и параметры в записи команд. Квадратные скобки нужны для работы с индексными величинами. Фигурные скобки используются для формирования множеств. В Maple две последовательные точки в параметрах команд применяются для определения интервала изменения переменных.

Теперь расскажем о других важных символах, используемых в Maple. Знаком процента (%) обозначается предшествующий вывод. Два знака процента отсылают к предпоследнему результату. Наконец, предшественник предпоследнего результата обозначается тремя знаками процента. Эта нотация может быть удобна при последовательной работе с документом, но чревата неприятностями при свободном перемещении по тексту, когда команды выполняются в произвольном порядке.

Используя константы, переменные, знаки арифметических и других операций, составляются выражения. Это основной объект для многих команд. Последовательность выполнения арифметических операций соответствует стандартным математическим правилам: сначала проводится возведение в степень (^), затем умножение (*) и деление (/), а в конце – сложение (+) и вычитание (-). Операции выполняются слева направо, для изменения порядка используются круглые скобки. Для операций отношения имеются знаки >, <, >=, <=, <>, =, а для конструирования булевых выражений используются также команды not, or, and. Обратный слеш (\) используется для переносов, а для комментирования в Maple предусмотрен символ #. Вся строка после этого символа не выполняется. Приведем примеры выражений:

```
> x+y^z-12<0;
  3.14159\26535\89793\23846; #число пи
```

Константы

В Maple представлены все математические константы. В табл. 1.2 представлены важнейшие из них. Напомним, что число π даётся при помощи Pi, а pi означает греческую букву π.

Таблица 1.2. Математические константы.

Имя	**Описание**
Pi	Число π
E	Основание натурального логарифма (число e)
I	Мнимая единица (квадратный корень из –1)
Infinity	Бесконечность
gamma	Константа Эйлера
true, false	Булевы константы (истина, ложь)

Имена этих констант являются зарезервированными, а их значения не могут быть переопределены в отличие от ряда переменных среды (управляющих констант).

Переменные

Переменная Maple идентифицируется именем – набором символов, начинающимся с буквы, причем большие и малые буквы различаются. Кроме букв могут употребляться цифры и знак подчеркивания. Приведем примеры различных имен: Old, old, o_1d

Длина имени зависит от платформы, и на 32-битных машинах допустимы имена из пятисот тысяч символов, а для 64-битных машин можно составлять имена длиной более миллиарда знаков (см. справку). Составные имена могут быть сформированы при помощи оператора конкатенации (||) или команды cat. В качестве имен переменных запрещено использовать термины языка Maple: and,

break, by, catch, description, do, done, elif, else, end, error, export, fi, finally, for, from, global, if, in, intersect, local, minus, mod, module, next, not, od, option, options, or, proc, quit, read, return, save, stop, then, to, try, union, use, while. Кроме того, не рекомендуется использовать имена всех команд Maple в качестве имен.

Для обозначения служебных констант используются имена, начинающиеся с подчеркивания. Неопределенные константы, возникающие при решении дифференциального уравнения, именуются _C1, _C2 и т. д. Произвольное целое число обозначается как _N1, _N2, и т.д., а комплексная величина соответственно как _Z1, _Z2 и т.д. Если последним символом имени идет тильда (~), то это имя переменной, относительно которой сделаны назначения (определена вещественность или положительность и т. д., см. подробнее команду assume).

Для присвоения значений переменной используется знак **:=**. Для просмотра содержимого переменной простого типа нужно лишь ввести имя переменной (для массивов и других составных типов используется команда eval). Напомним, что команда restart используется для отмены всех сделанных назначений в сеансе. Чтобы освободить конкретную переменную от предшествующих назначений, нужно этой переменной присвоить ее имя, заключенное в прямые одинарные кавычки (') (апострофы). Например:

```
> ex:=x^2+exp(y): ex;
> ex:='ex': ex;
```

Для защиты от изменений существует команда protect, а для снятия защиты – unprotect. В частности, защищенными являются многие константы Maple. Приведем пример использования последних двух команд. Сначала переменной присвоим значение, а затем ее защитим:

```
> a:=2: protect('a'): a;
```

Теперь попытаемся присвоить ей иное значение. Результатом будет сообщение об ошибке:

```
> a:=3;
Error, attempting to assign to `a` which is protected
```

Следующей командой снимем защиту с переменной, а затем присвоим ей новое значение:

```
> unprotect('a'): a:=3: a;
```

Переменные среды

Важными переменными среды являются Digits и Order, определяющие соответственно число знаков мантиссы для операций с плавающей точкой (по умолчанию десять цифр) и порядка разложений (по умолчанию разложения выписываются члены до шестого порядка). Для переопределения любой из этих величин достаточно просто присвоить ей новое значение. Приведем примеры, иллюстрирующие потерю точности при вычислениях с недостаточным числом значащих цифр и вычисление экспоненты:

```
> Digits:=1: sqrt(2.0)+0.01-71/50;
> Digits:=4: sqrt(2.0)+0.01-71/50;
> Digits:=20: exp(1.0);
```

Операции с вещественными числами проводятся по умолчанию с десятью значащими цифрами, но, переопределив Digits, можно работать с любой мантиссой. Этот способ может оказаться полезным и для определенных этапов символьных вычислений, поскольку операции с рациональными числами выполняются медленнее. Имеется также ряд других переменных среды. Так, переменная среды UseHardwareFloats определяет использование арифметического процессора компьютера для операций с вещественными числами, на котором вы работаете. Если эта переменная имеет значение true, то используется непосредственно процессор, а в случае false арифметические операции проделываются Maple программно. По умолчанию этой переменной присвоено значение true. В настоящее время этот режим действует для команд пакета LinearAlgebra и обслуживает операции с массивами, матрицами и векторами, которые основаны на новом классе rtable. Со временем, по утверждению разработчиков, переменная UseHardwarefloats будет определять режим вычислений для всех операций арифметики с плавающей точкой. Перечень переменных среды может быть выведен по команде

- **anames(environment);**

Строки и символы

Строкой (string) является любой набор символов, заключенный в двойные кавычки. Например:

```
> "Maple string:"";'o ?../'(}[]'-!G#tt*4*()_+";
```

Символ воспринимается Maple как единое целое, а строка состоит из символов, и с каждым из них можно работать отдельно. Например:

```
> v1:="String"; v2:=`Symbol`;
> v1[2]; v2[2];
```

Предупредим об опасности произвольного назначения имен переменных. Если имя переменной совпадает с именем какой-нибудь команды, то такая команда становится недоступной в текущем сеансе. Например:

```
> dchange:=2;
> with(PDEtiils.dchange):
Error, with expects its 1st argument, package, to be of type
{table, procedure, module, name}, but received 2*PDEtiils
```

Поэтому перед введением новой переменной name полезно удостовериться, что имя не занято, командой ?name. Появление окошка справки будет означать существование команды с именем name. При работе с Maple следует анализировать результат выполнения команды. Будучи гибкой системой, Maple старается выполнить любую введенную команду. Синтаксические ошибки выделяются программой и достаточно заметны. Описки не так бросаются в глаза. Если неправильно введено имя команды или не загружена библиотека, то Maple просто повторяет набранное в строке ввода без выполнения. Вообще, нужно постоянно проверять введенное и результат, иначе могут возникать «непонятные» ошибки. Например, если в строке ввода опустить знак умножения перед круглой скобкой в выражении x*(y-z), то Maple посчитает, что y-z является аргументом некоторой функции x(y-z). Сказанное иллюстрирует пример дифференцирования двух этих функций по переменной y:

```
> diff(x*(y-z),y);
> diff(x(y-z),y);
```

Другая часто встречающаяся ошибка связана с тем, что Maple различает большие и маленькие буквы. Например, если имя числа π написать с маленькой буквы, то пользователь не заметит видимых изменений при представлении результата команд, но многие команды будут работать неверно. Пример:

```
> sin(pi);
> sin(Pi);
```

Как уже отмечалось выше, прямые одинарные кавычки (апострофы) используются для того, чтобы освободить переменную от предшествующих значений. Кроме того, прямые кавычки полезны для предупреждения ошибки в том случае, когда для выполнения команды используется переменная, получившая значение ранее. Выполним последовательно команду присваивания и команду вычисления суммы, использовав в качестве индекса определённую ранее переменную:

```
> i:=3;
> sum(i^2,i=1..6);
Error, (in sum) summation variable previously assigned, second
argument evaluates to 3 = 1 .. 6
```

В результате появилось сообщение об ошибке, поскольку второй аргумент в команде суммирования sum воспринят как 3=1..6. Применение кавычек даст правильный результат:

```
> sum('i'^2,'i'=1..6);
```

Типы переменных

В Maple существует множество типов переменных: от известных вещественного (float), целого (integer) и строкового (string) до тех, которые необходимы для выполнения и программирования аналитических преобразований: дробь (fraction), функция (function), индексная переменная (indexed), процедура (procedure), множество (set), разложение (series), последовательность выражений (exprseq), массив (array), списки (переменные типа list, listlist, listlistlist) и некоторые другие. Списки используются для хранения коэффициентов полиномов,

множества удобны при формировании системы уравнений и решений уравнений, массивы позволяют хранить и использовать упорядоченную информацию и т. д.

По умолчанию переменная считается скалярной и имеет тип string. Это фактически математическая переменная, как x в формуле f(x). Для задания переменных других типов требуется явное их определение: при помощи оператора присваивания или команд, преобразующих тип. Информацию о типе той или иной переменной можно получить при помощи команды whattype, а проверить принадлежность переменной VAR типу TYPE помогает команда type (VAR,TYPE); Рассмотрим базовые типы: последовательности выражений, списки, множества, массивы и таблицы. Попутно представим ряд команд для работы с данными этих типов.

Переменная типа exprseq получается как последовательность выражений Maple, разделённых запятыми. Например:

> ex:=2,3,x^2,'abc',2;

Последовательность выражений в квадратных скобках образует переменную типа list (список):

> ex:=2,3,x^2,'abc',2: lex=[ex];

Тот же список получится по команде list(ex). Обращение к элементам типа list аналогично для последовательностей exprseq: в квадратных скобках указывается номер или диапазон номеров

Заключив последовательность выражений в фигурные скобки, получим переменную типа set (множество). Именно в виде множеств удобно задавать системы уравнений и получать найденные Maple решения уравнений:

> ex:=2,3,x^2,'abc',2: s:={ex};

Массивы или объекты типа array позволяют организовывать данные, используя для индексации отрицательные числа и нуль. Массив создается по команде array(FUN,DIA,LST). Параметры ее имеют следующее назначение: функция FUN задает свойства массива, переменная DIA определяет диапазоны изменения индексов, а LST есть список элементов массива. Каждый из параметров может быть опущен, но по крайней мере один диапазон или список элементов должны быть заданы. В качестве имени FUN могут быть использованы следующие стандартные методы: symmetric, antisymmetric, sparse, diagonal, identity, package. Это позволяет определись соответственно массивы симметричные и антисимметричные (кососимметричные), разреженные (нули для неупомянутых элементов), массивы с ненулевой диагональю и единичные. Имя package указывает на процедуру, служащую для ввода элементов. Например, для создания массива из четырех элементов с именем A используется следующая конструкция:

> A:=array(-1..2);

Сама переменная A при этом считается строковой (string), а любой элемент массива — индексной переменной (indexed). С элементами можно работать, как с обычными переменными: присваивать им значения, использовать при организации выражений. Например:

> A[-1]:=qwerty: A[1]:='йцукен': A[0]:=0: A[1]:=A[0]-A[2]:

Чтобы посмотреть содержимое массива, используем команду print:

> print(A);

При помощи команды table можно организовать данные в виде таблицы, отказавшись от целочисленной нумерации: table(FUN,LST). Здесь функция FUN определяет свойства таблицы, а список элементов LST формируется в виде пар равенств: Индекс-Значение. Для задания свойства можно использовать методы, перечисленные ранее. Например, создадим таблицу, устанавливающую соответствие между цифрами в английском и итальянском языках:

> T:=table(sparse,[one=uno,two=due,three=tre]);

Комбинируя данные разных типов, можно получить, например, списки множеств или последовательности списков и т. д. К составным типам относятся indexable, sequential, tabular, rtable. Встроенными типами (built-in) являются And, Or, Not, nonnegint, posint, nonposint, negint, negative, positive, nonnegative, nonpositive. Большое число типов связано с необходимостью адекватно обрабатывать многообразие математических конструкций. Кроме того, в дополнение к имеющимся типам можно создавать свои, расширяя возможности Maple для решения возникающих задач. Для работы с данными и извлечения элементов полезны команды nops (число операндов) и op (взятие операнда).

Опишем действие ряда команд, позволяющих эффективно обрабатывать данные (числа, выражения и другие), составляющие списки. Очень важной является команда map(FUN,LST). Эта

команда применяет функцию FUN к каждому элементу объекта LST, в качестве которого может выступать список, множество или выражение. Вычисление косинуса для нескольких значений из списка демонстрирует следующий пример:

```
> map(cos,[0,Pi/2,Pi]);
```

Таблица 1. Стандартные функции.

Maple	Математическая запись
exp(x)	e^x
ln(x) или log(x)	ln x
log10(x)	lg x
log[a](x)	$\log_a x$
sqrt(x)	$\sqrt{x}$
abs(x)	|x|
signum(x)	sgn x
n!	n!

Таблица 2. Функции округления.

Имя	Назначение
round	Округление к ближайшему целому
trunc	Округление отбрасыванием дробной части
floor	Округление к меньшему целому
ceil	Округление к большему целому

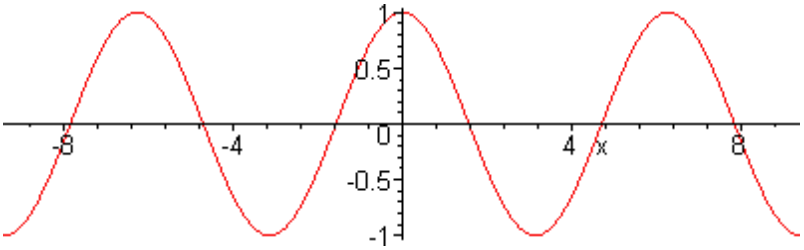
Таблица 3. Математические функции.

Имена	Назначение
sin, cos, tan, cot, sec, csc	Тригонометрические функции (аргументы в радианах)
arcsin, arccos, arctan, arccot, arcsec, arccsc	Обратные тригонометрические функции
sinh, cosh, tanh, coth, sech, csch	Гиперболические функции
arcsinh, arccosh, arctanh, arccoth, arcsech, arccsch	Обратные гиперболические функции
Dirac	Дельта-функция Дирака
Heaviside	Функция Хевисайда
BesselJ(p,x), BesselY(p,x), BesselI(p,x), BesselK(p,x),	Бесселевы функции
GAMMA(z), GAMMA(a,x), Beta(x,y)	Гамма- и бета-функции
Zeta(z)	Дзета-функция Римана
erf(x)	Интеграл ошибок
LegendreF, LegendreE, LegendrePi	Эллиптические интегралы в форме Лежандра первого, второго и третьего рода
LegendreFc, LegendreEc, LegendrePic	Полные эллиптические интегралы в форме Лежандра первого, второго и третьего рода
round, trunc	Округление к усечённому целому

Построение графиков функций в Maple

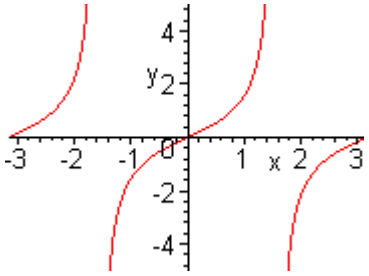
Для построения графиков в декартовых координатах служит функция plot

```
> plot(cos(x),x=-3*Pi..3*Pi);
```



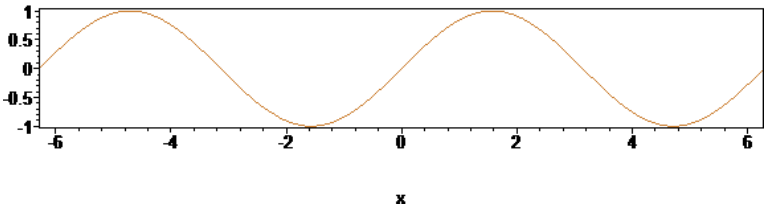
Если функция принимает бесконечное значение в какой-либо точке, то следует указать дополнительный параметр discont=true. Кроме того, может потребоваться ограничить диапазон отображения графика по оси y.

> **plot(tan(x),x=-Pi..Pi,y=-5..5,discont = true);**



Изменить параметры графика можно из контекстного меню, которое появляется, если нажать правую кнопку мыши в поле графика; из основного меню, пункты которого меняются, если график выделен; задавая необязательные параметры функции plot.

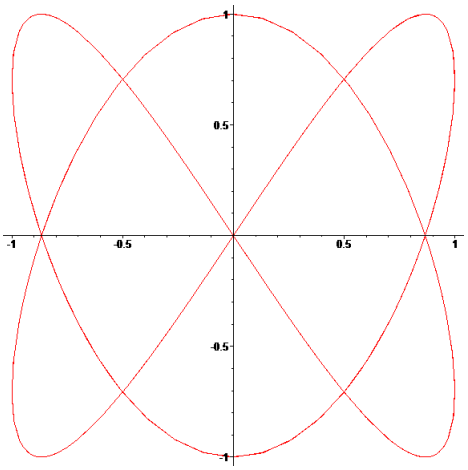
> **plot(sin(x),x=-2*Pi..2*Pi,color=gold,title=`График sin(x)`,axes=boxed,titlefont=[TIMES,ITALIC,10]);**



На рисунке вверху изменен цвет (color=gold), тип координатных осей (axes=boxed) и добавлен заголовок курсивом стиля Times размера 10 пунктов (titlefont=[TIMES,ITALIC,10]).

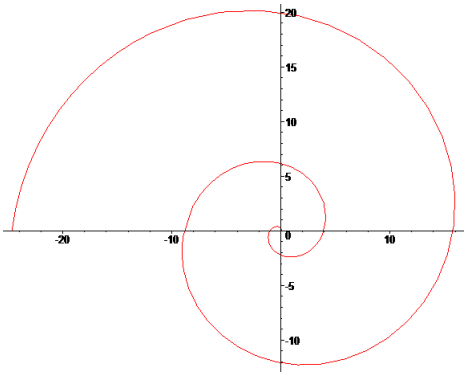
Параметрически заданную функцию можно изобразить также с помощью plot

> **plot([sin(2*t),cos(3*t),t=0..4*Pi]);**



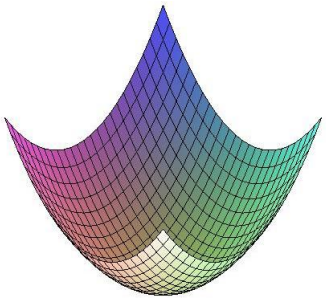
Для того, чтобы получить график в полярных координатах, следует указать coords = polar

> **r:=f^2/10:plot([r,f,f=0..5*Pi],coords=polar);**



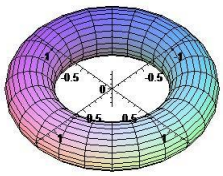
Для построения поверхностей служит функция plot3d. Она позволяет изображать поверхность, заданную как обычным способом, так и в параметрическом виде, а также несколько поверхностей одновременно.

> **plot3d(x^2+y^2,x=-1..1,y=-1..1);**

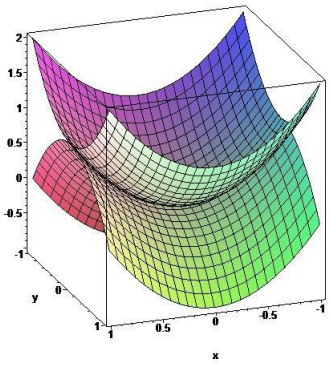


> **plot3d([(1+0.3*cos(phi))*cos(theta),**
(1+0.3*cos(phi))*sin(theta),
0.3*sin(phi)], phi=0..2*Pi,theta=0..2*Pi,
title=`Top`, titlefont=[TIMES,ITALIC,12],
scaling=CONSTRAINED,axes=normal);

Top



```
>plot3d({x^2+y^2,x^2-y^2},x= -1..1, y= 1..1, orientation = [70,70],
axes = boxed );
```



Задания:

1. **Построить графики функций**
   - $\begin{cases} x(t) = \sin t \\ y(t) = \ln \end{cases}$, $t \in [0,5]$;
   - $\rho = \varphi^{\frac{\varphi}{10}}$, $\varphi \in [0,2\pi]$;
   - $f(x) = 2\sin x$
2. **Построить графики**

Параметрические координаты							
№	Функция	интервал		№	Функция	Интервал	
		t_1	t_2			t_1	t_2
1	$\begin{cases} x = 2 + 5\cos t \\ y = -3 + 5\sin t \end{cases}$	0	2π	8	$\begin{cases} x = t(1 - \sin t \\ y = t\cos t \end{cases}$	0	π /2
2	$\begin{cases} x = 3\cos t \\ y = 4\sin t \end{cases}$	-π	π	9	$\begin{cases} x = 2e^t \\ y = e^{-t} \end{cases}$	-1.5	0.5
3	$\begin{cases} x = t^2 - 2t \\ y = t^2 - 2t \end{cases}$	-3	3	10	$\begin{cases} x = \sin t \\ y = \cos 2t \end{cases}$	-π /2	π /2

4	$\begin{cases} x = \cos t \\ y = t + 2\sin t \end{cases}$	-п	2п	11	$\begin{cases} x = 2\ln \operatorname{ctg} t + 1 \\ y = \operatorname{tg} t + \operatorname{ctg} t \end{cases}$	п /8	п /3
5	$\begin{cases} x = 2^{t-1} \\ y = \frac{1}{4}(t^2 + 1) \end{cases}$	-2	3	12	$\begin{cases} x = 3\cos t \\ y = 3\sin t \end{cases}$	-п	п
6	$\begin{cases} x = 3t \\ y = 6t - t^2 \end{cases}$	-10	20	13	$\begin{cases} x = 2(t - \sin t) \\ y = 2(1 - \cos t) \end{cases}$	0	2п
7	$\begin{cases} x = (t+1)/t \\ y = (t-1)/t \end{cases}$	-0.1	2	14	$\begin{cases} x = 4\cos^3 t \\ y = 4\sin^3 t \end{cases}$	-п	п

Полярные координаты

№	Функция	интервал		№	Функция	Интервал	
		φ_1	φ_2			φ_1	φ_2
1	$\rho = 2\sin\varphi$	0	п	8	$\rho = 1 + \operatorname{tg}\varphi$	-п /4	п /4
2	$\rho = 3\cos^2\frac{\varphi}{2}$	-п	п	9	$\rho = 3\varphi^3$	-2	2
3	$\rho = 2\sin^3\frac{\varphi}{3}$	0	п /2	10	$\rho = 2(2 + \cos\varphi)$	-п	п
4	$\rho = 4\sin^2\frac{\varphi}{2}$	-п	п	11	$\rho = 7\operatorname{tg}\varphi$	0	п /6
5	$\rho = 1 + \cos\varphi$	-п	п	12	$\rho = 3 + \cos 4\varphi$	-п	п
6	$\rho = 1 - \cos\varphi$	-п	п	13	$\rho = 2 - \cos 4\varphi$	-п	п
7	$\rho = \sin^3\varphi$	-п /2	п /2	14	$\rho = 2 + \cos 2\varphi$	-п	п

3. **Вычислить значение функции её график.** $f(x) = \frac{e^{x^2-2.6}}{п} + \cos 2x$ в точках $x = 1$ и $x = -1.46$. Построить
4. **Построить график функции** $z = \frac{x^2}{8} - \frac{y^2}{18}$, где $x \in [-8,8]$ и $y \in [-5,3]$ в трехмерной системе координат.

Лабораторная работа №2

Решение уравнений в Maple

Для решения нелинейных уравнений в системе Maple используется достаточно универсальная и гибкая функция **solve**, которая выдает решение в аналитическом виде. Чтобы получить корни уравнений в численном виде, приходится использовать функцию **evalf** или **convert**. Если результат решения представлен через функцию **RootOf**, то получить все корни можно с помощью функции **allvalues**. Эта функция применяется и самостоятельно. Функция solve имеет ряд производных от нее функций. Для получения численного решения нелинейного уравнения удобно использовать функцию **fsolve**, которая может использовать опции: **complex** — находит один или все корни полинома в комплексной форме, **maxsols** = n — задает нахождение только n корней и др. Рекуррентные уравнения решают с помощью функции **rsolve**. В системе Maple функции вызываются с двумя аргументами.

Основной функцией для решения уравнений является функция solve.

Решим алгебраическое уравнение

> **solve(x^4-x^3+x^2-x=0);**

$$0,\ 1,\ I,\ -I$$

Найдены все корни, включая комплексные.

> **eqn:=x^2=cos(x);**

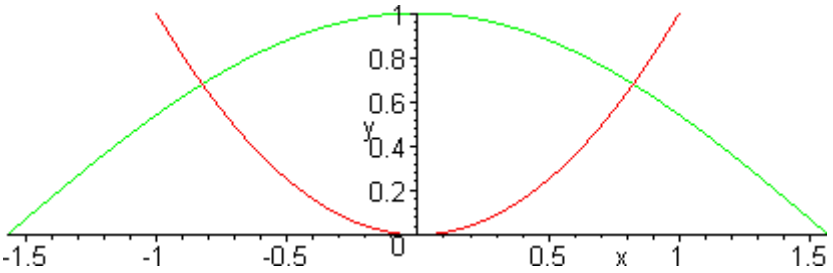
$$eqn := x^2 = \cos(x)$$

> **solve(eqn);**

$$\mathrm{RootOf}(-\cos(\_Z) + \_Z^2,\ label = \_L1)$$

Функция не выдала никакого решения, хотя, как видно по графику, уравнение имеет два действительных корня

> **plot({x^2,cos(x)},x=-Pi/2..Pi/2,y=0..1);**



Maple не смог найти аналитическое решение. В этом случае можно воспользоваться функцией fsolve, чтобы найти решение в виде действительного числа

> **fsolve(eqn,x);**

$$0.8241323123$$

Найден только один корень, чтобы найти второй, следует конкретизировать, где именно его следует искать (третий аргумент функции)

```
> fsolve(eqn,x,-2..0);
```

$$-0.8241323123$$

Решение систем уравнений

Функция solve может быть применена и для решения систем линейных уравнений

```
> first:=x+y+z=3:
second:=x-y-z=-1:
third:=x-y+x=1:
sys:={first,second,third};
```

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

```
> solve(sys);
```

$$\{x = 1, y = 1, z = 1\}$$

Функция может решать нелинейные системы

```
> solve({x^2+x*y+y^2=84,x+sqrt(x*y)+y=14});
```

$$\{x = 2, y = 8\}, \{x = 8, y = 2\}$$

Однако, в ряде случаем корни выражаются через функцию RootOf - корень выражения

```
> solve({x^2-y=23,x^2*y=50});
```

$$\{x = 5, y = 2\}, \{x = -5, y = 2\}, \{x = \mathrm{RootOf}(\_Z^2 + 2, label = \_L2), y = -25\}$$

С помощью функции allvalues можно найти значения RootOf

```
> allvalues(RootOf(_Z^2+2));
```

$$\sqrt{2}\,I, -I\sqrt{2}$$

Решение неравенств

Решение простых неравенств.

Команда **solve** применяется также для решения неравенств. Решение неравенства выдается в виде интервала изменения искомой переменной. В том случае, если решение неравенства полуось, то в поле вывода появляется конструкция вида RealRange(–∞ , Open(a)), которая означает, что

$x \in (-\infty, a)$, a – некоторое число. Слово Open означает, что интервал с открытой границей. Если этого слова нет, то соответствующая граница интервала включена во множество решений. Например:

```
> s:=solve(sqrt(x+3)<sqrt(x-1)+sqrt(x-2),x):
convert(s,radical);
```

$$\mathrm{RealRange}\left(\mathrm{Open}\left(\frac{2}{3}\sqrt{21}\right), \infty\right)$$

Если вы хотите получить решение неравенства не в виде интервального множества типа $x \in (a,b)$, а в виде ограничений для искомой переменной типа $a > x \quad x < b$, то переменную, относительно

,

которой следует разрешить неравенство, следует указывать в фигурных скобках. Например:

```
> solve(1-1/2*ln(x)>2,{x});
```

$$\{0 < x, x < \frac{1}{e^2}\}$$

Решение систем неравенств

С помощью команды **solve** можно также решить систему неравенств. Например:

```
> solve({x+y>=2,x-2*y<=1,x-y>=0,x-2*y>=1},{x,y});
```

$$\{y = \frac{x}{2} - \frac{1}{2}, \frac{5}{3} \le x\}$$

Задания:

1. **Решить уравнение** $y = 0$

№	Функция	№	Функция	№	Функция	№	Функция
1	$y = \frac{2^x}{xe^2}$	**5**	$y = \ln\sqrt{1+x^2}$	**9**	$y = x(1+\ln x)$	**13**	$y = \sqrt[3]{x^2-1}$
2	$y = \sin x \cos^3 x$	**6**	$y = x^2 2^x$	**10**	$y = \sqrt{x \cdot 2^x}$	**14**	$y = (x-1)^3$
3	$y = x^2\sqrt[3]{(1-x)^2}$	**7**	$y = \frac{x^3}{e^x - e^{-x}}$	**11**	$y = \sqrt{1+e^x}$		
4	$y = x^2(1+x)$	**8**	$y = x(1+x)^2$	**12**	$y = \cos x + \cos^2 x$		

2. **Решите неравенство $13x^3 - 25x^2 - x^4 - 129x + 270 > 0$.**
3. **Решите систему линейных алгебраических уравнений**

1. $\begin{cases} 3x_1 + 2x_2 + x_3 = 5, \\ 2x_1 + 3x_2 + x_3 = 1, \\ 2x_1 + x_2 + 3x_3 = 11. \end{cases}$	2. $\begin{cases} x_1 - 2x_2 + 3x_3 = 6, \\ 2x_1 + 3x_2 - 4x_3 = 20, \\ 3x_1 - 2x_2 - 5x_3 = 6. \end{cases}$	3. $\begin{cases} 4x_1 - 3x_2 + 2x_3 = 9, \\ 2x_1 + 5x_2 - 3x_3 = 4, \\ 5x_1 + 6x_2 - 2x_3 = 18. \end{cases}$
4. $\begin{cases} x_1 + x_2 + 2x_3 = -1, \\ 2x_1 - x_2 + 2x_3 = -4, \\ 4x_1 + x_2 + 4x_3 = -2. \end{cases}$	5. $\begin{cases} 2x_1 - x_2 - x_3 = 4, \\ 3x_1 + 4x_2 - 2x_3 = 11, \\ 3x_1 - 2x_2 + 4x_3 = 11. \end{cases}$	6. $\begin{cases} 3x_1 + 4x_2 + 2x_3 = 8, \\ 2x_1 - x_2 - 3x_3 = -4, \\ x_1 + 5x_2 + x_3 = 0. \end{cases}$

7. $\begin{cases} x_1 + x_2 - x_3 = 1, \\ 8x_1 + 3x_2 - 6x_3 = 3, \\ 4x_1 + x_2 - 3x_3 = 3. \end{cases}$	8. $\begin{cases} x_1 - 4x_2 - 2x_3 = -3, \\ 3x_1 + x_2 + x_3 = 5, \\ 3x_1 - 5x_2 - 6x_3 = -9. \end{cases}$	9. $\begin{cases} 7x_1 - 5x_2 = 31, \\ 4x_1 + 11x_3 = -43, \\ 2x_1 + 3x_2 + 4x_3 = -20. \end{cases}$
10. $\begin{cases} x_1 + 2x_2 + 4x_3 = 31, \\ 5x_1 + x_2 + 2x_3 = 20, \\ 3x_1 - x_2 + x_3 = 9. \end{cases}$	11. $\begin{cases} 4x_1 - 3x_2 + 2x_3 = 9, \\ 2x_1 + 5x_2 - 3x_3 = 4, \\ 5x_1 + 6x_2 - 2x_3 = 18. \end{cases}$	12. $\begin{cases} x_1 + x_2 + 2x_3 = -1, \\ 2x_1 - x_2 + 2x_3 = -4, \\ 4x_1 + x_2 + 4x_3 = -2. \end{cases}$
13. $\begin{cases} 2x_1 - x_2 - x_3 = 4, \\ 3x_1 + 4x_2 - 2x_3 = 11, \\ 3x_1 - 2x_2 + 4x_3 = 11. \end{cases}$	14. $\begin{cases} 3x_1 + 4x_2 + 2x_3 = 8, \\ 2x_1 - x_2 - 3x_3 = -4, \\ x_1 + 5x_2 + x_3 = 0. \end{cases}$	15. $\begin{cases} x_1 + x_2 - x_3 = 1, \\ 8x_1 + 3x_2 - 6x_3 = 3, \\ 4x_1 + x_2 - 3x_3 = 3. \end{cases}$

4. **Выполнить исследование функции на экстремум согласно индивидуальным заданиям.**
   1. **Задать функцию.**
   2. **Вычислить таблицу значений функции в заданном интервале.**
   3. **Построить график функции в заданном интервале, приблизительно определив экстремумы и точки пересечения.**
   4. **Найти точки пересечения функции с осью абсцисс (т.е. вычислить корни уравнения** $f(x) = 0$) разными способами. Сделать выводы о предпочтении того или иного способа.
   5. **Найти экстремумы функции несколькими способами. Сделать выводы о предпочтении того или иного способа.**

№	Функция	Интервал		№	Функция	Интервал	
		x_1	x_2			x_1	x_2
1	$y = \sin^3 x \cos x$	0	2п	**9**	$y = x \ln^2 x$	-0.3	3
2	$y = \sqrt[4]{1 - x^3}$	-1	5	**10**	$y = x \cdot 2^{x/2}$	-2	1.5
3	$y = x^3(1 - x)$	-2	2	**11**	$y = \frac{xe^x}{\sqrt{1 + e^x}}$	-3.5	1.5
4	$y = \ln\sqrt{1 + x^2}$	0.5	5	**12**	$y = (1 - \cos x)^2$	-п	п
5	$y = 2^{x/2}$	-2.5	3	**13**	$y = (1 - x)^2\sqrt{x}$	0	1.5

6	$y = \frac{1 + e^x}{1 + x^2}$	0	3.8	**14**	$y = x^4(1 - x)$	-1	2
7	$y = (\sin x + \cos x)^2$	-п	п	**15**	$y = x^2 \ln^3 x$	0.5	2.5

5. **Найдите значения a, при которых уравнение** $x^2 - 2(a-1) + 2a + 1 = 0$ имеет решение.

6. **Для каких значений a один из корней уравнения**

$$(a - 2)x^2 - 2(a + 3)x + 4a = 0$$

больше 3, а другой меньше ?

Лабораторная работа №3

Символьные вычисления в Maple

Бесконечные и конечные суммы и произведения

Вычислим конечную сумму $\sum_{i=1}^{10} \frac{1+i}{1+i^4}$

```
> sum( (1+i)/(1+i^4), i=1..10 );
```

$$\frac{5150805672759473291 3722}{4062664893881920008 8497}$$

затем определим значение бесконечной суммы $\sum_{k=1}^{\infty} \frac{1}{k^2}$

```
> sum( 1/k^2, k=1..infinity );
```

$$\frac{\pi^2}{6}$$

Аналогично действуем для нахождения конечных и бесконечных произведений, например

$$\prod_{i=0}^{10} \frac{i^2+3i-11}{i+3}$$

```
> product( ((i^2+3*i-11)/(i+3)), i=0..10 );
```

$$\frac{-7781706512657}{40435200}$$

Вычисление выражений

Зададим следующее выражение $(41x^2+x+1)^2(2x-1)$.

```
> expr1 := (41*x^2+x+1)^2*(2*x-1);
```

$$expr1 := (41x^2+x+1)^2(2x-1)$$

Бывает полезно уметь **раскрыть скобки** :

```
> expand(expr1);
```

$$3362x^5 - 1517x^4 + 84x^3 - 79x^2 - 1$$

затем снова **сгруппировать переменные**

```
> expr2:=factor(%);
```

$$expr2 := (41x^2+x+1)^2(2x-1)$$

Вы можете также **упрощать выражения** самого разного типа. Например, упростить

$$\cos(x)^5 + \sin(x)^4 + 2\cos(x)^2 - 2\sin(x)^2 - \cos(2x)$$

```
> simplify( cos(x)^5 + sin(x)^4 + 2*cos(x)^2 - 2*sin(x)^2 - cos(2*x) );
```

$$\cos(x)^4(\cos(x)+1)$$

Можно упрощать выражение при помощи команды **normal** , она позволяет **упрощать дроби**, сокращая их. Например, упростим следующую дробь $\frac{x^3-y^3}{x^2+x-y-y^2}$.

```
>  normal( (x^3-y^3)/(x^2+x-y-y^2) );
```

$$\frac{x^2+xy+y^2}{x+1+y}$$

Обратите внимание, что вы можете **вычислить значение** выражения, содержащее переменные в некоторой точке:

```
> eval(expr1 , x=1 );
```

$$1849$$

Дифференцирование и интегрирование

Зададим функцию $x \sin(a\,x) + b\,x^2$.

```
> f := x -> x*sin(a*x) + b*x^2;
```

$$f := x \to x \sin(a\,x) + b\,x^2$$

Вычислим производную данной функции $\frac{\partial}{\partial x}(x \sin(a\,x) + b\,x^2)$

и запишем в переменную f_prime .

```
> f_prime := diff( f(x), x);
```

$$f\_prime := \sin(a\,x) + x \cos(a\,x)\,a + 2\,b\,x$$

Вы можете также посчитать вторую производную: $\frac{d^2}{dx^2} \mathrm{f}(x)$

```
> f_second := diff( f(x), x$2 );
```

$$f\_second := 2 \cos(a\,x)\,a - x \sin(a\,x)\,a^2 + 2\,b$$

Для проверки полученного значения вы можете взять **неопределенный интеграл** от полученного выражения производной

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

$$-\frac{\cos(a\,x)}{a} + \frac{\cos(a\,x) + a\,x \sin(a\,x)}{a} + b\,x^2$$

```
> simplify(%);
```

$$x\,(\sin(a\,x) + b\,x)$$

Для вычисления определенного интеграла достаточно в оператор **int** добавить область изменения переменой.

Вычислим $\int_1^2 [\sin(a\,x) + x \cos(a\,x)\,a + 2\,b\,x]\,dx$

```
> int( f_prime, x=1..2 );
```

$$-\sin(a) + 3\,b + 2 \sin(2\,a)$$

Вычисление пределов

Вы можете вычислять как конечные, так и бесконечные пределы для широкого класса функций.

Вы можете брать как левый, так и правый пределы.

Вычислим следующий предел: $\lim_{x \to \infty} \frac{2\,x + 3}{7\,x + 5}$.

```
> limit((2*x+3)/(7*x+5),x=infinity);
```

$$\frac{2}{7}$$

Теперь продемонстрируем вычисление левых и правых пределов: $\lim_{x \to 0} \left| \tan\left(x + \frac{\pi}{2} \right) \right|$.

```
> limit( tan(x+Pi/2), x=0, left );
```

$$\infty$$

```
> limit( tan(x+Pi/2), x=0, right );
```

$$-\infty$$

Maple может распознать, когда предел не определен $\lim_{x \to 0} \frac{\sqrt{1-\cos(x)}\sqrt{2}}{x}$.

```
> limit( sqrt(1-cos(x)) / x*sqrt(2), x=0 );
```

undefined

Степенные ряды

Maple с легкостью раскладывает аналитические функции в ряды Тейлора до заданного порядка

```
> expr := sin(4*x)*cos(x):
approx1 := series( expr, x=0, 6 );
```

$$approx1 := 4x - \frac{38}{3}x^3 + \frac{421}{30}x^5 + O(x^6)$$

Тот же результат можно достичь, используя функцию **taylor**

```
> taylor( expr, x=0, 8 );
```

$$4x - \frac{38}{3}x^3 + \frac{421}{30}x^5 - \frac{10039}{1260}x^7 + O(x^8)$$

Задания:

1. **Вычислить:** $\prod_{k=1}^{1000} \left(1 + \frac{1}{(2k-1)(2k+1)}\right)$ и $\left[\sum_{n=1}^{1000} \frac{(-1)^{n+1}}{2n-1} \right]$.
2. **Вычислить предел:** $\lim_{x \to 2} \frac{x^3 - 8}{2x - 4}$.
3. **Продифференцировать функцию:** $y = \frac{x^5 - 1}{3x^2 - 2} + \cos x$.
4. **Найти аналитически точки пересечения графиков функций: проверить графически.** $x - y = 3$ $xy = 4$. Результат и
5. **Исследовать функцию на наличие асимптот:** $y = \frac{x^3}{x^2 - 1}$. Построить график для проверки.
6. **Разложить на множители выражение** $y^3(z - x) - x^3(z - y) + z^3(x - y)$.
7. **Вычислить площадь области, ограниченной кривыми** $y = x^2$ $y = \sqrt{x}$, . Построить графики.
8. **Найти наибольшее и наименьшее значение для функции** $y = \frac{2}{\sqrt{x}} - x$ на отрезке [0,4]
9. **Построить касательную и нормаль к графику функции** $f(x) = x^4 - 1$ в точке (1, 0).

10. **Определить интервалы выпуклости, вогнутости и точки перегиба функций** $f(x) = \frac{2x}{x^2+1}$.

11. **Разложить функцию** $f(x) = \sin 4x$ в окрестности точки $x_0 = 0$ в ряд Тейлора.

Лабораторная работа №4

Линейная алгебра

Для того, чтобы использовать большинство операторов линейной алгебры необходимо подключить библиотеку **LinAlg**.

> > **with(linalg):**

Для знакомства с основными функциями пакета зададим несколько матриц, проиллюстрировав при этом различные способы их задания.

> **A:=matrix(3, 3, [-3, 9, -8, 18, -28, 27, 20, -36, 34]);**

$$A := \begin{bmatrix} -3 & 9 & -8 \\ 18 & -28 & 27 \\ 20 & -36 & 34 \end{bmatrix}$$

> **B:=matrix([[3, 0, 0], [8, -23, 24], [8, -26, 27]]);**

$$B := \begin{bmatrix} 3 & 0 & 0 \\ 8 & -23 & 24 \\ 8 & -26 & 27 \end{bmatrix}$$

> **C:=[[1, 2, 4], [-3, 1, -2], [1, -2, -1]];C:=convert(C, matrix);**

$$C := [[1, 2, 4], [-3, 1, -2], [1, -2, -1]]$$

$$C := \begin{bmatrix} 1 & 2 & 4 \\ -3 & 1 & -2 \\ 1 & -2 & -1 \end{bmatrix}$$

Вычислим определитель матрицы A

> **det(A);**

$$-4$$

Вычислим A^T:

> **transpose(A);**

$$\begin{bmatrix} -3 & 18 & 20 \\ 9 & -28 & -36 \\ -8 & 27 & 34 \end{bmatrix}$$

Если определитель на равен нулю, мы можем вычислить обратную матрицу

> **inverse(A);**

$$\begin{bmatrix} -5 & \frac{9}{2} & \frac{-19}{4} \\ 18 & \frac{-29}{2} & \frac{63}{4} \\ 22 & -18 & \frac{39}{2} \end{bmatrix}$$

Определим еще одну матрицу B, содержащие неизвестные θ и ϕ.

> **B:= Matrix(3, 3, [[phi, 0, 0], [0, 3, 2], [-1, 2/3, theta]]);**

$$B := \begin{bmatrix} \varphi & 0 & 0 \\ 0 & 3 & 2 \\ -1 & \frac{2}{3} & \theta \end{bmatrix}$$

Теперь определим матрицу C как произведение данных матриц

> **C := multiply(A , B);**

$$C := \begin{bmatrix} -3\varphi + 8 & \frac{65}{3} & 18 - 8\theta \\ 18\varphi - 27 & -66 & -56 + 27\theta \\ 20\varphi - 34 & \frac{-256}{3} & -72 + 34\theta \end{bmatrix}$$

Для умножения согласованных матриц в Maple используется операция **&***. Найдем разность произведений матриц A и B, взятых в противоположных порядках:

> **ABBA:=A&*B-B&*A;**

$$ABBA := \left(\begin{bmatrix} \varphi & 0 & 0 \\ 0 & 3 & 2 \end{bmatrix} \&* \begin{bmatrix} \varphi & 0 & 0 \\ 0 & 3 & 2 \end{bmatrix} \right) - \left(\begin{bmatrix} \varphi & 0 & 0 \\ 0 & 3 & 2 \end{bmatrix} \&* \begin{bmatrix} -1 & 3 & \theta \end{bmatrix} \right) \&* \left(\begin{bmatrix} -1 \\ 3 \\ \theta \end{bmatrix} \right)$$

Мы видим, что вместо привычного для нас окончательного результата Maple вернул в качестве ответа некий "полуфабрикат" (умножение и вычитание матриц обозначены, но не выполнены). Дело в том, что для сложных объектов (матриц, массивов, таблиц, процедур) в Maple используется так называемое "правило вычисления до последнего имени" (last name evaluation). Это правило (в отличие от "правила вычисления до конца", используемого для менее громоздких объектов) применяется во избежание лишнего вывода на экран. Для того, чтобы все же вывести на экран интересующую нас матрицу, воспользуемся функцией **evalm**:

> **AB:=evalm(ABBA);**

$$AB := \begin{bmatrix} 8 & \frac{65}{3}\varphi - 9 & 18 - 8\theta + 8\varphi \\ 18\varphi - 121 & 90 & -205 + 27\theta \\ 20\varphi - 49 - 20\theta & -\frac{173}{3} + 36\theta & -98 \end{bmatrix}$$

Найдем ранг полученной матрицы:

> **rank(AB);**

3

Для матрицы A найдем собственные значения и принадлежащие им собственные векторы:

> **eigenvals(A); eigenvectors(A);**

$$-1, 2, 2$$

$$\left[2, 2, \left\{ \begin{bmatrix} -1 \\ 3 \end{bmatrix}, \begin{bmatrix} 1, \frac{4}{3} \end{bmatrix} \right\} \right], \left[-1, 1, \left\{ \begin{bmatrix} -3 \\ 10 \end{bmatrix}, \begin{bmatrix} 1, \frac{6}{5} \end{bmatrix} \right\} \right]$$

Обратите внимание, что при обращении к функции **eigenvectors** (**eigenvects**) Maple возвращает собственные значения матрицы, их кратность как корней характеристического многочлена и базис подпространства из собственных векторов, принадлежащего каждому собственному значению.

Разумеется, Maple умеет работать не только с числовыми матрицами:

> **M:=matrix([[a, 2, c], [b, b, c], [0, a, 1]]);**

$$M := \begin{bmatrix} a & 2 & c \\ b & b & c \\ 0 & a & 1 \end{bmatrix}$$

> **M:=matrix([[a, 2, c], [b, b, c], [0, a, 1]]);**

$$M := \begin{bmatrix} a & 2 & c \\ b & b & c \\ 0 & a & 1 \end{bmatrix}$$

> **rank(M);**

3

> **evalm(M^2);**

$$\begin{bmatrix} a^2+2b & 2a+2b+ca & ca+3c \\ ba+b^2 & 2b+b^2+ca & 2cb+c \\ ba & ba+a & 1+ca \end{bmatrix}$$

> **f:=collect(charpoly(M, lambda), lambda);**

$$f := \lambda^3 + (-1-a-b)\lambda^2 + (-ca+a-b+ba)\lambda + ca^2 - acb - ba + 2b$$

В последнем примере мы нашли характеристический многочлен матрицы M и сгруппировали его относительно λ.

Положительная определенность матриц

negative_definite	отрицательно определена
negative_semidefinite	отрицательно полуопределена
positive_definite	положительно определена
positive_semidefinite	положительно полуопределена

Определим некоторую матрицу M

> **M := Matrix(3, 3, [[1, -3, 3], [3, -5, 3], [6, -6, 4]]);**

$$M := \begin{bmatrix} 1 & -3 & 3 \\ 3 & -5 & 3 \\ 6 & -6 & 4 \end{bmatrix}$$

> **with(LinearAlgebra):**

> **IsDefinite(M,'query' = 'negative_definite');**

false

> **IsDefinite(M,'query' = 'negative_semidefinite');**

false

> **IsDefinite(M,'query' = 'positive_semidefinite');**

false

> **IsDefinite(M,'query' = 'positive_definite');**

false

Задание

1. **Создать и заполнить матрицы A размерности 3×4, элемент которой, расположенный в i -й** строке, j -м столбце, равен значению $f(i, j)$ функции $f(x, y) = x + y$.
2. **Вычислить матрицу $2A - BA$, где** $A = \begin{pmatrix} 2 & 3 \\ 0 & 1 \\ -1 & 3 \end{pmatrix}$ и $B = \begin{pmatrix} 2 & -2 & 2 \\ 0 & 7 & 1 \\ 5 & & \end{pmatrix}$.
3. **Дана матрица** $M = \begin{pmatrix} 2 & 1 & 1 \\ 13 & 1 & 2 \\ 5 & -1 & 0 \end{pmatrix}$. Вычислить её определитель и найти значение выражения $\left(2M^3 - 3\left(M^{-1}\right)^2 + E\right)M^T$, где E – единичная матрица размером 3×3.
4. **Проверить, что матрица** $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & -1 & 6 \\ -1 & 8 & 9 \end{pmatrix}$ невырождена и найти матрицу A^{-1}.

5. **Проверить, что матрицы** $A = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{pmatrix}$ и $B = \begin{pmatrix} 2 & 0 & -1 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix}$ перестановочны, а матрицы A и $C = \begin{pmatrix} 2 & 8 & 3 \\ 3 & 0 & -2 \\ 7 & 2 & 1 \end{pmatrix}$ неперестановочны.

6. **Проверить, что матрица** $U = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix}$ ортогональна ($UU^T = E$).

7. **Вычислить определитель матрицы** $A = \begin{pmatrix} 1 & -2 & 3 & 0 \\ 2 & 3 & 0 & 1 \\ -7 & 5 & 6 & 7 \\ 3 & 10 & 12 & 13 \end{pmatrix}$ разложением по 1-ой строке

8. **Решить матричное уравнение:** $AX = b$, где $A = \begin{pmatrix} 1 & 2 & 3 \\ 1 & -3 & 2 \\ 1 & 1 & 1 \end{pmatrix}$, $b = \begin{pmatrix} 7 \\ 5 \\ 3 \end{pmatrix}$.

9. **Найти решение системы линейных уравнений по формулам Крамера и методом Гаусса**

$$\begin{cases} x_1 + 2x_2 + 3x_3 + 4x_4 = 30, \\ -x_1 + 2x_2 - 3x_3 + 4x_4 = 10, \\ x_2 - x_3 + x_4 = 3, \\ x_1 + x_2 + x_3 + x_4 = 10. \end{cases}$$

10. **Связь между тремя отраслями представлена матрицей прямых затрат A (табл.1). Спрос (конечный продукт) задан вектором $\overline{Y}$. Найти валовой выпуск продукции отраслей $\overline{X}$.**

Варианты заданий

1. $A = \begin{pmatrix} 0.2 & 0.3 & 0.2 \\ 0.4 & 0.5 & 0.3 \\ 0.3 & 0.1 & 0.1 \end{pmatrix}$ $\overline{Y} = \begin{pmatrix} 20 \\ 18 \\ 57 \end{pmatrix}$	2. $A = \begin{pmatrix} 0.2 & 0.3 & 0.2 \\ 0.1 & 0.2 & 0.3 \\ 0.3 & 0.1 & 0.1 \end{pmatrix}$ $\overline{Y} = \begin{pmatrix} 46 \\ 38 \\ 44 \end{pmatrix}$
3. $A = \begin{pmatrix} 0.3 & 0.1 & 0.2 \\ 0.1 & 0.25 & 0.3 \\ 0.3 & 0.1 & 0.1 \end{pmatrix}$ $\overline{Y} = \begin{pmatrix} 28 \\ 40 \\ 38 \end{pmatrix}$	4. $A = \begin{pmatrix} 0.25 & 0.1 & 0.2 \\ 0.1 & 0.25 & 0.3 \\ 0.3 & 0.15 & 0.2 \end{pmatrix}$ $\overline{Y} = \begin{pmatrix} 34 \\ 25 \\ 36 \end{pmatrix}$
5. $A = \begin{pmatrix} 0.05 & 0.1 & 0.4 \\ 0.1 & 0.1 & 0.3 \\ 0.3 & 0.15 & 0.2 \end{pmatrix}$ $\overline{Y} = \begin{pmatrix} 45 \\ 50 \\ 35 \end{pmatrix}$	6. $A = \begin{pmatrix} 0.2 & 0.1 & 0.4 \\ 0.1 & 0.1 & 0.3 \\ 0.3 & 0.15 & 0.2 \end{pmatrix}$ $\overline{Y} = \begin{pmatrix} 66 \\ 81 \\ 14 \end{pmatrix}$

7. $A=\begin{pmatrix}0.05 & 0.1 & 0.4\\ 0.1 & 0.1 & 0.3\\ 0.1 & 0.15 & 0.2\end{pmatrix}\quad \overline{Y}=\begin{pmatrix}47\\58\\81\end{pmatrix}$	8. $A=\begin{pmatrix}0.25 & 0.1 & 0.2\\ 0.1 & 0.1 & 0.3\\ 0.3 & 0.15 & 0.2\end{pmatrix}\quad \overline{Y}=\begin{pmatrix}55\\58\\81\end{pmatrix}$
9. $A=\begin{pmatrix}0.3 & 0.1 & 0.2\\ 0.1 & 0.1 & 0.3\\ 0.1 & 0.15 & 0.2\end{pmatrix}\quad \overline{Y}=\begin{pmatrix}62\\79\\53\end{pmatrix}$	10. $A=\begin{pmatrix}0.25 & 0.1 & 0.2\\ 0.1 & 0.25 & 0.3\\ 0.3 & 0.15 & 0.1\end{pmatrix}\quad \overline{Y}=\begin{pmatrix}32\\40\\42\end{pmatrix}$
11. $A=\begin{pmatrix}0.2 & 0.3 & 0.2\\ 0.4 & 0.1 & 0.3\\ 0.3 & 0.1 & 0.1\end{pmatrix}\quad \overline{Y}=\begin{pmatrix}20\\37\\43\end{pmatrix}$	12. $A=\begin{pmatrix}0.2 & 0.3 & 0.2\\ 0.1 & 0.2 & 0.3\\ 0.3 & 0.1 & 0.1\end{pmatrix}\quad \overline{Y}=\begin{pmatrix}96\\66\\46\end{pmatrix}$
13. $A=\begin{pmatrix}0.2 & 0.1 & 0.2\\ 0.1 & 0.25 & 0.3\\ 0.3 & 0.1 & 0.1\end{pmatrix}\quad \overline{Y}=\begin{pmatrix}70\\44\\48\end{pmatrix}$	14. $A=\begin{pmatrix}0.25 & 0.1 & 0.2\\ 0.1 & 0.25 & 0.3\\ 0.3 & 0.1 & 0.2\end{pmatrix}\quad \overline{Y}=\begin{pmatrix}58\\20\\42\end{pmatrix}$

Пример выполнения:

```
> restart:
> A:=array(1..3,1..3,[[0.2,0.3,0.2],[0.4,0.5,0.3],[0.3,0.1,0.1]]);
```

$$A := \begin{bmatrix}0.2 & 0.3 & 0.2\\ 0.4 & 0.5 & 0.3\\ 0.3 & 0.1 & 0.1\end{bmatrix}$$

```
> X:=array(1..3,[20,18,57]);
```

$$X := [20, 18, 57]$$

```
> E:=array(1..3,1..3,[[1,0,0],[0,1,0],[0,0,1]]);
```

$$E := \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}$$

```
> A2:=evalm(E-A);
```

$$A2 := \begin{bmatrix}0.8 & -0.3 & -0.2\\ -0.4 & 0.5 & -0.3\\ -0.3 & -0.1 & 0.9\end{bmatrix}$$

```
> with(linalg):det(A2);
```

$$0.163$$

```
> A3:=evalm(A2^(-1));
```

$$A3 := \begin{bmatrix}2.576687117 & 1.779141104 & 1.165644172\\ 2.760736196 & 4.049079755 & 1.963190184\\ 1.165644172 & 1.042944785 & 1.717791411\end{bmatrix}$$

```
> R:=evalm(A3&*X);
```

$$R := [150.0000000, 240.0000000, 140.0000000]$$

Лабораторная работа № 5

Матричные исчисления на Python и в системе MAPLE средствами библиотеки линейной алгебры

Цель работы: изучить команды библиотеки линейной алгебры и применить их к решению задач матричного исчисления.

Работа в математическом пакете MAPLE начинается с подключения библиотеки линейной алгебры.

> **restart:**
> **with(linalg); # подключается пакет линейной алгебры Далее выдается**

перечень команд (программ) этой библиотеки.

Ниже приводятся примеры выполнения этих команд. Выполните эти команды в системе MAPLE.

> **#Определение матриц**
> **a:=array(1..2,1..2,[[1,2],[2,4]]);**

$$a := \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$$

> **b:=array(1..2,1..3,[[1,2,3],[4,5,6]]);**

$$b := \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

> **c:=array(1..2,1..2,[[0,1],[2,4]]);**

$$c := \begin{bmatrix} 0 & 1 \\ 2 & 4 \end{bmatrix}$$

> **# Умножение матрицы на число p**
> **p:=2;**

$$p := 2$$

> **d:=scalarmul(a,p);**

$$d := \begin{bmatrix} 2 & 4 \\ 4 & 8 \end{bmatrix}$$

> **# перемножение матриц**
> **q:=multiply(a,c);**

$$q := \begin{bmatrix} 4 & 9 \\ 8 & 18 \end{bmatrix}$$

> **g:=multiply(a,b);**

$$g := \begin{bmatrix} 9 & 12 & 15 \\ 18 & 24 & 30 \end{bmatrix}$$

> **# сложение матриц**
> **q:=matadd(a,c);**

$$q := \begin{bmatrix} 1 & 3 \\ 4 & 8 \end{bmatrix}$$

> **# вычитание матриц**
> **w:=matadd(a,c,1,-1);**

$$w := \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}$$

> **# сложение матриц с умножением каждой на скаляр r=2*a+3*c**
> **r:=matadd(a,c,2,3);**

$$r := \begin{bmatrix} 2 & 7 \\ 10 & 20 \end{bmatrix}$$

> **# транспонирование матрицы**
> **at:=transpose(a);**

$$at := \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$$

> **bt:=transpose(b);**

$$bt := \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

> **# вычисление ранга матрицы u=g*bt**
> **u:=multiply(g,bt);**

$$u := \begin{bmatrix} 78 & 186 \\ 156 & 372 \end{bmatrix}$$

> **rank(u);**

$$1$$

>

```
s:=array(1..4,1..4,[[25,31,17,43],[75,94,53,132],[75,94,54,134],
[25,32,20,48]]);
```

$$s := \begin{bmatrix} 25 & 31 & 17 & 43 \\ 75 & 94 & 53 & 132 \\ 75 & 94 & 54 & 134 \\ 25 & 32 & 20 & 48 \end{bmatrix}$$

> **rank(s);**

$$3$$

> **# вычсление определителя s**
> **det(s);**

$$0$$

> **z:=array(1..3,1..3,[[3,-4,5],[2,-3,1],[3,-5,-1]]);**

$$z := \begin{bmatrix} 3 & -4 & 5 \\ 2 & -3 & 1 \\ 3 & -5 & -1 \end{bmatrix}$$

> **det(z);**

$$-1$$

> **# вычисление братной матрицы z**
> **zobrat:=evalf(inverse(z));**

$$zobrat := \begin{bmatrix} -8. & 29. & -11. \\ -5. & 18. & -7. \\ 1. & -3. & 1. \end{bmatrix}$$

> # проверка вычисления обратной матрицы
> # после перемножения обратной матрицы на исходную должна получиться единичная матрица
> Imatr:=multiply(zobrat,z);

$$Imatr := \begin{bmatrix} 1. & 0. & 0. \\ 0. & 1. & 0. \\ 0. & 0. & 1. \end{bmatrix}$$

> # вычисление собственных чисел матриц c и s
> evalf(eigenvals(c));

$$4.449489743, -0.449489743$$

> evalf(eigenvals(s));

$$0., 218.6259780 - 0.8\,10^{-9}\,I, -0.009594703 - 0.515615242510^{-7}\,I, 2.383616663 + 0.523615242510^{-7}\,I$$

> roc:=max(abs(4.449489743),abs(-.449489743));# вычисление спектрального радиуса матрицы c

$$roc := 4.449489743$$

> ros:=max(abs(0.0),abs(218.6259780-.8e-9*I),abs(-.9594703e-2-.5156152425e-7*I),abs(2.383616663+.5236152425e-7*I));

$$ros := 218.6259780$$

> # вычисление нормы 1, 2 и бесконечность для матрицы y
> y:=array(1..2,1..2,[[3,4],[5,6]]);

$$y := \begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix}$$

> normay1:=max(abs(y[1,1]+y[2,1]),abs(y[1,2]+y[2,2]));# вычисление нормы 1

$$normay1 := 10$$

> # вычисление нормы 2
> yt:=transpose(y);

$$yt := \begin{bmatrix} 3 & 5 \\ 4 & 6 \end{bmatrix}$$

> h:=multiply(yt,y);

$$h := \begin{bmatrix} 34 & 42 \\ 42 & 52 \end{bmatrix}$$

> evalf(eigenvals(h));

$$85.95346318, 0.04653682$$

>
> roy:=max(abs(85.95346318),abs(.4653682e-1));

$$roy := 85.95346318$$

> **normay2:=sqrt(roy);**

$$normay2 := 9.271109059$$

> **normaybesk:=max(abs(y[1,1]+y[1,2]),abs(y[2,1]+y[2,2]));#**
вычисление нормы бесконечность

$$normaybesk := 11$$

> **# возведение матрицы y в степень**
> **y2:=evalm(y^2);**

$$y2 := \begin{bmatrix} 29 & 36 \\ 45 & 56 \end{bmatrix}$$

> **y5:=evalm(y^5);**

$$y5 := \begin{bmatrix} 22683 & 28204 \\ 35255 & 43836 \end{bmatrix}$$

> **# формирование расширенной матрицы**
> **y6:=blockmatrix(1,4,[y,h,y2,y5]);**

$$y6 := \begin{bmatrix} 3 & 4 & 34 & 42 & 29 & 36 & 22683 & 28204 \\ 5 & 6 & 42 & 52 & 45 & 56 & 35255 & 43836 \end{bmatrix}$$

> **# выделение из матрицы y6 5-го столбца**
> **col(y6,5);**

$$[29, 45]$$

> **f:=array(1..4,1..4,[[2,5,7,9],[4,7,3,5],[8,6,1,0],[7,5,3,1]]);**

$$f := \begin{bmatrix} 2 & 5 & 7 & 9 \\ 4 & 7 & 3 & 5 \\ 8 & 6 & 1 & 0 \\ 7 & 5 & 3 & 1 \end{bmatrix}$$

> **# выделение из матрицы f 3-ей строки**
> **row(f,3);**

$$[8, 6, 1, 0]$$

> **# выделение из матрицы f 3-го и 4-го столбцов и 2-ой и 3-ей строк соответственно**
> **col(f,3..4);**

$$[7, 3, 1, 3], [9, 5, 0, 1]$$

> **row(f,2..3);**

$$[4, 7, 3, 5], [8, 6, 1, 0]$$

> **# сложение второй и третьей строки в матрице a**
> **addrow(f,2,3,1); # при этом вторая строка умножается на 1 после чего складывается с третьей**

$$\begin{bmatrix} 2 & 5 & 7 & 9 \\ 4 & 7 & 3 & 5 \\ 12 & 13 & 4 & 5 \\ 7 & 5 & 3 & 1 \end{bmatrix}$$

> **# сложение третьей строки и второй**
> **addrow(f,3,2,1);# при этом третья строка умножается на 1 после чего складывается со второй**

$$\begin{bmatrix} 2 & 5 & 7 & 9 \\ 12 & 13 & 4 & 5 \\ 8 & 6 & 1 & 0 \\ 7 & 5 & 3 & 1 \end{bmatrix}$$

> # сложение второй и третьей строки в матрице a
> addrow(f,2,3,3); # при этом вторая строка умножается на 3 после чего складывается с третьей

$$\begin{bmatrix} 2 & 5 & 7 & 9 \\ 4 & 7 & 3 & 5 \\ 20 & 27 & 10 & 15 \\ 7 & 5 & 3 & 1 \end{bmatrix}$$

> addrow(f,3,2,-2);# при этом третья строка умножается на -2
после чего складывается со второй

$$\begin{bmatrix} 2 & 5 & 7 & 9 \\ -12 & -5 & 1 & 5 \\ 8 & 6 & 1 & 0 \\ 7 & 5 & 3 & 1 \end{bmatrix}$$

> # вычитание строк
> addrow(f,3,2,-1);

$$\begin{bmatrix} 2 & 5 & 7 & 9 \\ -4 & 1 & 2 & 5 \\ 8 & 6 & 1 & 0 \\ 7 & 5 & 3 & 1 \end{bmatrix}$$

> # аналогичные операции со столбцами матрицы a
> addcol(f,2,3,1);

$$\begin{bmatrix} 2 & 5 & 12 & 9 \\ 4 & 7 & 10 & 5 \\ 8 & 6 & 7 & 0 \\ 7 & 5 & 8 & 1 \end{bmatrix}$$

> addcol(f,3,2,1);

$$\begin{bmatrix} 2 & 12 & 7 & 9 \\ 4 & 10 & 3 & 5 \\ 8 & 7 & 1 & 0 \\ 7 & 8 & 3 & 1 \end{bmatrix}$$

> addcol(f,2,3,3);

$$\begin{bmatrix} 2 & 5 & 22 & 9 \\ 4 & 7 & 24 & 5 \\ 8 & 6 & 19 & 0 \\ 7 & 5 & 18 & 1 \end{bmatrix}$$

> addcol(f,3,2,-2);

$$\begin{bmatrix} 2 & -9 & 7 & 9 \\ 4 & 1 & 3 & 5 \\ 8 & 4 & 1 & 0 \\ 7 & -1 & 3 & 1 \end{bmatrix}$$

> **addcol(f,3,2,-1);**

$$\begin{bmatrix} 2 & -2 & 7 & 9 \\ 4 & 4 & 3 & 5 \\ 8 & 5 & 1 & 0 \\ 7 & 2 & 3 & 1 \end{bmatrix}$$

> **#выделение подматрицы матрицы f**
> **submatrix(f,1..1,1..1);**

$$\begin{bmatrix} 2 \end{bmatrix}$$

> **submatrix(f,1..2,1..2);**

$$\begin{bmatrix} 2 & 5 \\ 4 & 7 \end{bmatrix}$$

> **submatrix(f,1..3,1..3);**

$$\begin{bmatrix} 2 & 5 & 7 \\ 4 & 7 & 3 \\ 8 & 6 & 1 \end{bmatrix}$$

> **submatrix(f,1..4,1..4);**

$$\begin{bmatrix} 2 & 5 & 7 & 9 \\ 4 & 7 & 3 & 5 \\ 8 & 6 & 1 & 0 \\ 7 & 5 & 3 & 1 \end{bmatrix}$$

> **submatrix(f,2..3,2..3);**

$$\begin{bmatrix} 7 & 3 \\ 6 & 1 \end{bmatrix}$$

> **submatrix(f,3..4,2..3);**

$$\begin{bmatrix} 6 & 1 \\ 5 & 3 \end{bmatrix}$$

> **# работа с векторами как с одномерными массивами**
> **n:=array(1..5,[4,5,2,7,9]);**

$$n := [4, 5, 2, 7, 9]$$

> **m:=array(1..5,[-4,8,-2,4,3]);**

$$m := [-4, 8, -2, 4, 3]$$

> **# сложение векторов**
> **matadd(n,m);**

$$[0, 13, 0, 11, 12]$$

> **# умножение векторов**
> **multiply(n,m);**

```
> # умножение вектора на число
> scalarmul(n,3);
```
[12, 15, 6, 21, 27]

```
> # вычитание векторов
> matadd(n,m,1,-1);
```
[8, -3, 4, 3, 6]

```
> # скалярное произведение векторов
> multiply(transpose(n),m);
```
75

```
> # норма вектора
> evalf(sqrt(multiply(transpose(n),n)));
```
13.22875656

```
> # угол между векторами
> p1:=evalf(multiply(transpose(n),m));
```
p1 := 75.

```
> p2:=evalf(sqrt(multiply(transpose(n),n)));
```
p2 := 13.22875656

```
> p3:=evalf(sqrt(multiply(transpose(m),m)));
```
p3 := 10.44030651

```
> p4:=evalf(p1/(p2*p3));
```
p4 := 0.5430364604

```
> teta:=arccos(p4);
```
teta := 0.9967473418

```
>
```

Задания на самостоятельную работу.

1) **Задайте матрицы: А размером n на n, В размером n на m и С размером n на n.**
   Для матрицы А выполните следующие действия: умножьте k-ю строку на a_1, l-ый столбец умножьте на a_2; сложите k-ю строку с l-той, предварительно умножив ее на a_3; сложите k-ый и p-ый столбец, предварительно умножив любой из них на a_4; транспонируйте матрицу; найдите ее ранг; найдите обратную к ней и выполните проверку; найдите ее собственные числа и спектральный радиус; найдите все три вида нормы для матрицы А; вычислите определитель матрицы; возведите матрицу в степень q; выделите из матрицы w-ю строку и r-ый столбец, а также подматрицу размером h на v. Вычислите сумму и разность матриц А и С, выполните то же, но с умножением каждой матрицы на скаляр a_5. Вычислите произведение матрицы А на В. Выполните формирование расширенной матрицы из матриц А, В и С. Определите ранг расширенной матрицы.
2) **Задайте два вектора размером j элементов. Выполните сложение, вычитание векторов, умножение векторов на скаляр, вычислите все нормы каждого вектора и угол между векторами.**

Номера вариантов

	Номер варианта						
	1	**2**	**3**	**4**	**5**	**6**	**7**
n	4	5	3	5	4	5	5
m	5	6	5	7	8	5	6
k	1	4	2	4	5	2	4
l	2	3	2	3	3	3	3
k	3	4	1	2	3	4	2
p	2	2	3	4	2	1	1
w	1	1	2	3	1	2	2
r	4	3	1	2	3	3	3
h	2	3	2	3	3	3	4
v	2	3	2	3	2	3	4
q	2	3	4	2	4	3	2
a_1	-6	4	-4	-5	-2	-4	4
a_2	8	-6	-2	-3	-1	-3	5
a_3	-5	-7	4	7	2	-4	6
a_4	9	-4	3	2	4	5	-3
a_5	2	3	6	4	5	6	-5
j	8	7	5	6	7	8	8
	Номер варианта						
	8	**9**	**10**	**11**	**12**	**13**	**14**
n	4	6	4	5	4	5	5
m	5	8	5	8	6	7	7
k	1	4	2	4	5	3	4
l	2	5	3	4	2	3	3
k	3	4	1	2	3	4	2
p	2	3	3	5	2	5	4
w	1	1	4	3	2	2	2
r	4	3	1	2	3	3	3
h	2	3	2	4	3	3	4
v	2	3	2	4	2	3	4
q	3	4	2	6	8	2	2
a_1	-6	-4	-4	5	-12	4	-4
a_2	-8	-6	2	-3	-1	-3	5
a_3	-5	7	4	-7	32	4	16
a_4	5	-4	-3	12	4	5	-3
a_5	2	3	6	4	5	6	-25
j	8	7	5	6	7	8	8
	Номер варианта						
	15	**16**	**17**	**18**	**19**	**20**	**21**
n	5	6	4	5	4	5	5
m	5	7	7	6	7	7	8
k	5	2	2	4	5	3	4
l	2	5	2	3	3	1	2
k	2	4	1	2	3	4	2
p	2	2	3	5	2	5	4
w	1	1	2	3	2	1	2
r	4	3	1	1	3	3	1

h	1	3	2	4	3	3	4
v	2	3	2	4	2	3	4
q	4	5	3	4	2	3	2
a_1	-6	-4	-4	5	-12	4	-4
a_2	-18	-6	12	-31	-11	-3	15
a_3	-5	17	4	-7	32	14	16
a_4	5	-4	-3	12	4	5	-3
a_5	2	3	16	4	15	6	25
j	6	6	8	7	8	9	5

Отчет по работе должен содержать название работы, ее цель, задание на самостоятельное исполнение в соответствии с вариантом и распечатку программы с результатами выполнения.

Лабораторная работа №6

Программирование управляющих конструкций и подпрограмм в Maple и Python

Maple является не только прекрасным инструментом для проведения разнообразных вычислений, но и содержит полноценный внутренний язык программирования. Рассмотрим основные конструкции: условия, циклы и процедуры. Для написания кода полезно уметь писать многострочные выражения в **Maple** . Это делается несложно: **Shift+Enter** .

Для записи строк используйте символ " ` ".

Условный оператор

Использование условий и циклов требует умения манипулировать логическими операторами: **OR, AND, NOT** . Составной оператор условного перехода имеет структуру:

if < *cond* > **then** < *block* > **elif** < *cond* > **then** < *block* >... **else** < *block* > **fi** ;

Здесь *cond* - выражение булевского типа, а *block* - любая группа команд Maple.

Конструкция **elif** < *cond* > **then** < *block* > может повторяться несколько раз или вовсе отсутствовать. Часть **else** < *block* > тоже не является обязательной.

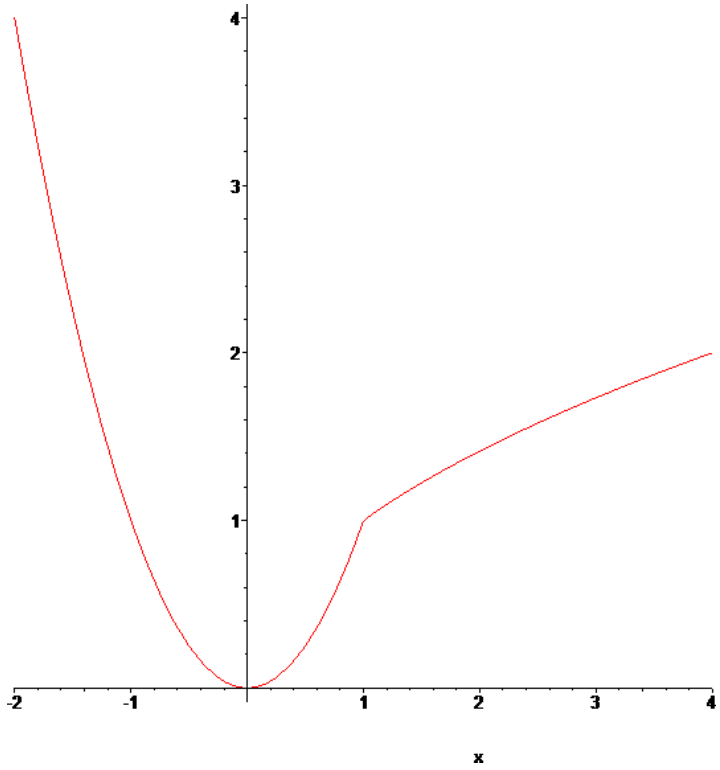
```
> i:=4:
if i > 0 then
print(`Positive number.`);
elif i < 0 then
print(`Negative number.`);
else
print(`Zero number.`);
fi:
```

Positive number.

Часто вместо составного оператора ветвления бывает удобнее использовать функцию ветвления - **`if` `**. Она имеет три аргумента, первый из которых является условием, а второй и третий - возвращаемыми значениями. Если первый аргумент принимает значение true, то возвращаемым значением функции является второй аргумент, а если false - третий. Приведем пример использования функции **if** :

```
> f:=`if`(x>1,sqrt(x),x^2);plot(f,x=-2..4);
```

$$f := \mathrm{if}(1 < x, \sqrt{x}, x^2)$$



Циклы

В Maple существует несколько разновидностей циклов. Стандартный арифметический цикл имеет следующую структуру:

for < *var* > **from** < *exp1* > **by** < *exp2* > **to** < *exp3* > **do** < *block* > od;

Здесь группа команд *block* выполняется для каждого значения переменной цикла *var*, которая меняется от значения *exp1* до значения *exp3* с шагом *exp2*. Если конструкция **from** < *exp1* > и/или **by** < *exp2* > отсутствует, то соответствующие им выражения принимаются равными 1. Отметим, что переменная цикла может изменять свое значение не только при переходе к следующему шагу цикла, но и внутри тела цикла. Проиллюстрируем это на примере:

```
> for i from 1 by 1 to 10 do i:=i+3 od;
```

i := 4

i := 8

i := 12

Итерационный цикл имеет структуру:

while < *cond* > **do** < *block* > **od** ;

Возможен и "гибрид" арифметического и итерационного циклов. Соответствующая конструкция имеет вид:

for < *var* > **from** < *exp1* > **by** < *exp2* > **while** < *cond* > **do** < *block* > **od** ;

```
> tot := 0;
for i from 11 by 2 while i < 20 do
tot := tot + i;
print(tot);
end do:
```

tot := 0

11

24

39

56

75

Рассмотренные варианты организации циклов характерны (с теми или иными изменениями) для большинства языков программирования. Кроме них, в Maple имеется еще один специфический вид цикла. Как уже упоминалось, выражения в Maple имеют структуру дерева. Перебрать все ветви (операнды) первого уровня можно с помощью специального цикла, имеющего такой синтаксис:

for < *var* > **in** < *exp* > **do** < *block* > **od** ;

Проиллюстрируем работу этого цикла на примере:

```
> s:=0:for i in numtheory[divisors](60) do if numtheory[mobius](i)=1 then s:=s+i fi od;s;
```

32

В последнем примере мы нашли сумму тех делителей числа 60, значение функции Мёбиуса от которых, равно 1.

В качестве иллюстрации использования циклов и условных операторов решим такую задачу: найти все пары простых чисел близнецов (так называют простые числа, отличающиеся друг от друга на 2) на отрезке натурального ряда чисел от 8000 до 9000.

```
> printlevel:=0:
p:=nextprime(8000):
while p<8998 do
p1:=p+2:
if isprime(p1) then print(p,p1) fi;
p:=nextprime(p1)
od;
```

8009, 8011

8087, 8089

8219, 8221

8231, 8233

8291, 8293

8387, 8389

8429, 8431

8537, 8539

8597, 8599

8627, 8629

8819, 8821

8837, 8839

8861, 8863

8969, 8971

Обратите внимание на то, что перед началом вычислений мы присвоили значение 0 зарезервированной переменной **printlevel**. Дело в том, что на команды, вызываемые из вложенных структур, не распространяется правило регулирования вывода результатов выполнения команды на экран с помощью помещения точки с запятой или двоеточия после соответствующей команды. В этом случае отображаются результаты выполнения всех команд, глубина вложенности которых не превышает текущего значения зарезервированной переменной **printlevel**. Это делается для того, чтобы программа не печатала слишком много промежуточных результатов. Если же мы все же хотим вывести на экран результаты, получаемые на глубине вложенности, превышающей значение **printlevel**, достаточно использовать оператор **print**. По умолчанию значение **printlevel** равно 1. (Именно этим объясняется, что в примере, где мы вычисляли сумму части делителей числа 60, не были отображены на экране результаты присваиваний s **:=** s+i)

Процедуры и функции

В **Maple** процедуры и функции не различаются в записи. Для их задания используется оператор **proc** .

```
> > f := proc(x, y) local z; if x<0 then
z:=x+y;
elif x>0 then
z:=x-y;
else
z:=y;
fi;
RETURN(z);
end proc;
```

f := **proc**(*x*, *y*)
local *z*;
if *x* < 0 **then** *z* := *x* + *y* 0 < *x* **then** *z* := *x* − *y* **else** *z* := *y* **end if** ; RETURN(*z*)
elif end proc

```
> f(2,1);
f(2,-1);
f(2,0);
```

1

3

2

Задания:

1. **Составить программу вычисления факториала *n*!**
2. **Создать массив A размерностью 10. Задать функцию**

$$F(x) = \begin{cases} F_1(x), & x \le 3 \\ F_2(x), & 3 < x \le 6 \\ F_3(x), & 6 < x \le 10 \end{cases}$$

согласно индивидуальному заданию (табл.1). Заполнить массив значениями функции $F(i)$, где $i = \overline{1,10}$. Провести селективную обработку полученного массива согласно индивидуальному заданию (табл.2). Вывести в текстовый файл: а) исходный массив; б) результат расчета.

Таблица 1 – Функции

№	$F_1(x)$	$F_2(x)$	$F_3(x)$
1	tg(2x)	sin(3x)	cos(x-2)
2	4x+2	5/(X+0,4)	0,5/(2sin(4x))
3	$\sqrt[3]{x-1}$	$x^4/7$	$\sin^3 2x$
4	$\sqrt[3]{\sin^2 x + \cos^4 x}$	ctg(X+0,4)	$\ln(2x + 0.5)$
5	$x^3 - \ln\|x\|$	$\ln^3(x+4)$	x^4-x^{2-x}
6	$\sin(x^2)$	$e^{-x} + \sqrt[4]{x}$	$\ln(x^3+x^2)$
7	$(3x-1)/x^5$	$\ln^2\sqrt{x+5}$	$\sqrt{1+x^2}$
8	$x\cos x$	1/(tg2x+1)	x^2e^{-x}
9	$x^{1.2}\sin 3x$	$x^2\cos x$	$\sin x^2 + x^{0.25}$
10	x^{2x+1}	$\sin x^2$	$\ln^2 x + \sqrt{x}$
11	$\sin^2 x^3$	$\sin 2x$	$2\sin(x - e^{-x})$
12	$2xe^{-x}$	$\cos 2x$	$x^x - \cos x$
13	$\sin(2x + 1)$	$(x+1)^2\cos x^2$	$\sqrt{x^3-1} + \sin x^2$
14	$\cos x$	$\sqrt{x^3}\sin x$	$x^2 + \sin 5x$
15	$x(\sin x + 2)$	$2\ln(4x+1)$	$\ln\sqrt[5]{x+x^2}$
16	$x^4 + 2x^3 - x$	$\sin 2x$	$x^{x+1}\sin x$

17	$x^5 ctg\left(2x^3\right)$	$\ln(x+1)$	$e^{-2x} - \sqrt[3]{x}$
18	$\sin 4x$	$\sqrt{6x - x^2 + 1}$	$\sin x^{2x} - \cos x$
19	$ctg\left(3x - 1\right)^2$	$2 + xe^{-x}$	$\sin^3 x^2$
20	$x\sin\left(x - 1\right)$	$5 + 18tgx$	$2\sqrt{x^3 \sin x^3}$
21	$\frac{2x+1}{x^5}$	$e^{1+x} + \cos x$	$\ln\left(3 + \sqrt[5]{\sin x + x^2}\right)$
22	$3x^5 - ctgx^3$	$\ln^2\left(\sin 4x + 1\right)$	$\ln\left(1 + \sqrt[3]{2x+3}\right)$
23	$1.3\sqrt{4 + x^2}$	3^{x+3}	$5^{x+1} \sin 2x$
24	e^{-3x}	$\sin^3 x^4$	$e^{-x} - \sqrt[3]{3x}$

Таблица 2 – Содержание задачи

№	Содержание задачи
1	Найти сумму наибольшего и наименьшего элементов массива.
2	Вычислить среднее арифметическое элементов, удовлетворяющих условию $5 \le A_i \le 15$
3	Найти сумму квадратов максимального и минимального чисел.
4	Найти количество элементов, не больших заданного числа K.
5	Вычислить среднее арифметическое элементов, не меньших 10.
6	Найти максимальный и минимальный элементы и поменять их местами.
7	Найти количество элементов, принадлежащих интервалу $10 \le A_i \le 50$.
8	Найти среднее геометрическое элементов, имеющих четный индекс
9	Найти разность суммы положительных и произведения отрицательных чисел

10	Найти номер наибольшего положительного элемента
11	Найти сумму и количество элементов, у которых индекс кратен 3.
12	Найти частное от деления минимального элемента на максимальный элемент.
13	Вычислить сумму и количество элементов, имеющих нечетный индекс.
14	Найти среднее арифметическое отрицательных элементов, стоящих на четных местах
15	Поменять местами первый и максимальный элемент.
16	Найти сумму минимального положительного элемента и его номера.
17	Поменять местами второй и наибольший положительный элементы.
18	Вычислить среднее арифметическое максимального и минимального элементов.
19	Найти сумму и количество положительных элементов, стоящих на четных местах.
20	Вычислить частное от деления максимального элемента на его порядковый номер.
21	Найти среднее геометрическое положительных элементов, стоящих на нечетных местах.
22	Найти номер и значение наибольшего по модулю элемента.
23	Вычислить сумму квадратов положительных элементов, имеющих нечетные индексы.
24	Поменять местами максимальный положительный и минимальный отрицательный элементы.

Пример вывода в файл

```
> filename:=`c:/lab5.txt`: fd:=fopen(filename,WRITE):
writeline(fd,`Массив A:`): st:=``:
for i from 1 by 1 to 10 do st:=cat(st,convert(A[i],string),` `): od:
writeline(fd,st):
st:=cat(`Сумма положительных элементов = `,convert(s,string)):
writeline(fd,st):
```

```
fclose(fd);
```

3. **Дана действительная квадратная матрица порядка *n*. Найти сумму элементов, расположенных в заштрихованной части матрицы**

4. **Даны целые числа** $a_1,\ldots,a_{30}$. Пусть M - наибольшее, а m - наименьшее из них. Получить в порядке возрастания все целые числа из интервала (m,M), которые не входят в последовательность $a_1,\ldots,a_{30}$.

Лабораторная работа № 7

Матричные исчисления в Python и выполняемые в системе MAPLE средствами встроенного языка программирования

Цель работы: изучить команды языка программирования и применить их к решению задач матричного исчисления.

Возможности встроенного языка программирования в основном такие же, как и в любом другом языке высокого уровня. Алфавит языка, синтаксис и семантика основных операндов и операторов являются стандартными. Отдельные особенности языка будут рассматриваться постепенно по мере необходимости в данной и последующих работах. В настоящей работе будем рассматривать три типа операторов.

Оператор присваивания := . Его основное предназначение - присвоить значение некоторой переменной, чтобы в дальнейшем использовать ее в вычислениях. Например, *x* := 3; .

Условный оператор:

if булево_выражение **then**
последовательность_операторов 1:
[**else**
последовательность_операторов 2:]
end if;

Семантика оператора: если истинно булево_выражение, то выполняется последовательность_операторов 1: в противном случае выполняется последовательность_операторов 2. Квадратные скобки означают, что отмеченный блок может отсутствовать.

Оператор цикла:

for имя **from** выражение [**by** выражение] **to** выражение **do**
последовательность_операторов;
end do;

В блоке **for** задается имя переменной цикла**,** блоки **from** и **to** определяют соответственно, начальное и конечное значение диапазона изменения переменной цикла, а в блоке **by** задается шаг ее изменения (он может быть и отрицательным). Выполнение цикла начинается с присваивания переменной цикла начального значения, после чего проверяется, не превосходит ли оно конечного значения, и в случае положительного ответа выполняются операторы тела цикла, заданные в блоке **do….. end do.** Затем переменная цикла увеличивается на значение шага, и алгоритм проверки начинается снова. Если значение переменной цикла превосходит конечное значение, то цикл прекращает свое выполнение. Блок [**by** выражение] может отсутствовать, тогда шаг полагается по умолчанию равным 1.

Применение этих операторов рассмотрим на примере матричных исчислений:

> **restart:**
>
> **# работа с векторами как с одномерными массивами**
> **x:=array(1..5,[-4,3,-6,-7,1]);**

$$x := [-4, 3, -6, -7, 1]$$

> **y:=array(1..5,[-5.8,-5.3,4.3,8.2,-1.3]);**

$$y := [-5.8, -5.3, 4.3, 8.2, -1.3]$$

```
> # сложение векторов
> c:=array(1..5);
```

$$c := \text{array}(1 .. 5, [\ ])$$

```
> for i from 1 to 5 do #переход на следующую строку
<shift>+<enter>
c[i]:=x[i]+y[i];
end do:
print(c); # распкчатка массива
```

$$[-9.8, -2.3, -1.7, 1.2, -0.3]$$

```
> # умножение вектора на число
> d:=array(1..5):
alfa:=2;
for i from 1 to 5 do
d[i]:=alfa*c[i];
end do:
print(d);
```

$$alfa := 2$$

$$[-19.6, -4.6, -3.4, 2.4, -0.6]$$

```
> #скалярное произведение векторов
> s:=0:
for i from 1 to 5 do
s:=s+x[i]*y[i];
end do:
scalpxy:=s;
```

$$scalpxy := -77.2$$

```
> # норма вектора x (2-норма)
> s:=0:
for i from 1 to 5 do
s:=s+x[i]^2;
end do:
normax:=evalf(sqrt(s));
```

$$normax := 10.53565375$$

```
> # норма вектора y
> s:=0:
for i from 1 to 5 do
s:=s+y[i]^2;
end do:
normay:=evalf(sqrt(s));
```

$$normay := 12.21269831$$

```
> # угол между векторами x и y
> teta:=arccos(scalpxy/(normax*normay));
```

$$teta := 2.214285229$$

```
> #определение максимального элемента в массиве x
> xmax:=x[1]:
for i from 2 to 5 do
```

```
if x[i]>xmax then
xmax:=x[i];
end if;
end do;
print(x);
print("xmax=",xmax);
```

[-4, 3, -6, -7, 1]

"xmax=", 3

```
> # работа с двумерными массивами
> a:=array(1..3,1..3,[[-5,2,6],[-5,7,-1],[-8,13,2]]);
```

$$a := \begin{bmatrix} -5 & 2 & 6 \\ -5 & 7 & -1 \\ -8 & 13 & 2 \end{bmatrix}$$

```
> b:=array(1..3,1..3,[[-15,-2,6],[-5,2,-12],[-2,3,2]]);
```

$$b := \begin{bmatrix} -15 & -2 & 6 \\ -5 & 2 & -12 \\ -2 & 3 & 2 \end{bmatrix}$$

```
> # умножение матрицы на число 2
# результат переписывается в эту же матрицу
> for i from 1 to 3 do for j from
    1 to 3 do
      a[i,j]:=2*a[i,j];
    end do;
end do;
print("a=",a);
```

$$\text{"a="}, \begin{bmatrix} -10 & 4 & 12 \\ -10 & 14 & -2 \\ -16 & 26 & 4 \end{bmatrix}$$

```
> # транспонирование матрицы a
at:=array(1..3,1..3):
for i from 1 to 3 do
    for j from 1 to 3 do
      at[i,j]:=a[j,i];
    end do;
end do;
print("at=",at);
```

$$\text{"at="}, \begin{bmatrix} -10 & -10 & -16 \\ 4 & 14 & 26 \\ 12 & -2 & 4 \end{bmatrix}$$

```
> # умножение матриц a*b
g:=array(1..3,1..3): for i from 1
to 3 do
    for j from 1 to 3 do
     s:=0;
     for k from 1 to 3 do
      s:=s+a[i,k]*b[k,j];
     end do;
```

```
   g[i,j]:=s;
   end do;
end do;
print("g=",g);
```

$$\text{"g="}, \begin{bmatrix} 106 & 64 & -84 \\ 84 & 42 & -232 \\ 102 & 96 & -400 \end{bmatrix}$$

> **# суммирование элементов матрицы a**

```
suma:=0:
for i from 1 to 3 do
   for j from 1 to 3 do
     suma:=suma+a[i,j];
   end do;
end do;
print("suma=",suma);
```

"suma=", 22

> **# произведение элементов матрицы a**

```
proizva:=1:
for i from 1 to 3 do
   for j from 1 to 3 do
     proizva:=proizva*a[i,j];
   end do;
end do;
print("proizva=",proizva);
```

"proizva=", 223641600

> **# вычисление суммы отрицательных элементов матрицы a suma:=0:**

```
for i from 1 to 3 do
   for j from 1 to 3 do
     if a[i,j]<0 then
     suma:=suma+a[i,j];
     end if;
   end do;
end do;
print("suma=",suma);
```

"suma=", -38

> **# вычисление суммы элементов матрицы a, попадающих в интервал от -5 до 15**

```
suma:=0:
for i from 1 to 3 do
   for j from 1 to 3 do
     if a[i,j]> -5 and a[i,j]<15 then
     suma:=suma+a[i,j];
     end if;
   end do;
end do;
print("suma=",suma);
```

"suma=", 32

> **# вычисление произведения элементов матрицы a, не попадающих в интервал от -5 до 15**

```
prizva:=1:
for i from 1 to 3 do
   for j from 1 to 3 do
     if a[i,j]<-5 or a[i,j]>15 then
     prizva:=prizva*a[i,j];
     end if;
   end do;
end do;
print("prizva=",prizva);
```

"prizva=", -41600

> **#определение максимального элемента в массиве a**

```
amax:=a[1,1]:
for i from 1 to 3 do
 for j from 1 to 3 do
  if a[i,j]>amax then
   amax:=a[i,j];
  end if;
 end do;
end do;
print(a);
print("amax=",amax);
```

>

$$\begin{bmatrix} -10 & 4 & 12 \\ -10 & 14 & -2 \\ -16 & 26 & 4 \end{bmatrix}$$

"amax=", 26

> **# вычисление суммы элементов главной диагонали элементов матрицы a**

```
sdiag:=0:
for i from 1 to 3 do
sdiag:=sdiag+a[i,i];
end do:
print("sdiag=",sdiag);
```

"sdiag=", 8

> **# формирование одномерного массива, состоящего из максимальных элементов каждой строки матрицы a**

> **v:=array(1..3):**

```
for i from 1 to 3 do
 maxstr:=a[i,1]:
  for j from 1 to 3 do
   if a[i,j]>maxstr then
    maxstr:=a[i,j]:
   end if;
  end do;
 v[i]:=maxstr:
end do:
print(a);
print("v=",v);
```

$$\begin{bmatrix} -10 & 4 & 12 \\ -10 & 14 & -2 \\ -16 & 26 & 4 \end{bmatrix}$$

"v=", [12, 14, 26]

> **# формирование одномерного массива, состоящего из минимальных элементов каждого столбца матрицы a**

```
w:=array(1..3):
for j from 1 to 3 do
 minstolb:=a[1,j]:
  for i from 1 to 3 do
   if a[i,j]<minstolb then
    minstolb:=a[i,j]:
   end if;
  end do;
 w[j]:=minstolb:
end do:
print(a);
print("w=",w);
```

$$\begin{bmatrix} -10 & 4 & 12 \\ -10 & 14 & -2 \\ -16 & 26 & 4 \end{bmatrix}$$

"w=", [-16, 4, -2]

Задание на самостоятельное исполнение:

1) **Наберите указанные фрагменты программ и выполните их в системе Maple. Дополните эту программу выполнением следующих действий: выполните**

   вычитание векторов x и y, вычислите $\|\vec{x}\|_\infty$ и $\|\vec{x}\|_1$; найдите минимальный элемент в массиве y; осуществите вычитание матриц a и b, произведение положительных элементов матрицы a; определите минимальный элемент в массиве b; найдите минимальный элемент в массиве v и максимальный – в массиве w.

2) **Для матриц A, B и C из задания на самостоятельную работу практического**

   занятия 1 (в соответствии со своим вариантом) напишите программу на языке программирования для выполнения следующих вычислений:

   - **Произведение $A \times B = R$, умножение R на число, транспонирование R и**

     получение R^T, определение минимального элемента для каждой строки R^T и максимального элемента для каждого столбца R^T;
   - **Сумма $A + C = H$, произведение элементов матрицы H, попадающих в**

     интервал $[z_1, z_2]$ (значения z_1 и z_2 подберите самостоятельно) и сумму элементов матрицы H не попадающих в этот интервал;
   - **Произведение суммы диагональных элементов матрицы C на максимальный элемент матрицы B.**

3) **Для векторов из 2-го задания на самостоятельную работу практического занятия 1 (в соответствии со своим вариантом) напишите программу на языке программирования для выполнения следующих вычислений: сложение, вычитание векторов, умножение векторов на скаляр, вычислите все нормы каждого вектора и угол между векторами.**

Отчет по работе должен содержать название работы, ее цель, задание на самостоятельное исполнение в соответствии с вариантом и распечатку программы с результатами выполнения.

Лабораторная работа № 8

Решение систем линейных алгебраических уравнений (СЛАУ) средствами библиотеки линейной алгебры Maple и в Python. Численные исследования обусловленности СЛАУ, алгоритма решения плохо обусловленной системы и ее решений на устойчивость.

Цель работы: освоить программные средства решения СЛАУ в системе Maple, алгоритмы вычисления числа обусловленности матрицы системы, решения плохо обусловленных систем, провести численные исследования свойств матрицы системы и ее решений на устойчивость по правой части.

Краткие теоретические сведения, реализация соответствующих алгоритмов в Maple.

Постановка задачи: система линейных алгебраических уравнений (СЛАУ) имеет вид:

$$A\vec{x} = \vec{b}, \quad (1)$$

где A - вещественная квадратная матрица, порядка n, $\vec{b} = (b_1, b_2, ..., b_n)^T$ - **заданный** вектор, $\vec{x} = (x_1, x_2, ..., x_n)^T$ - **искомый вектор. Если определитель матрицы *A* отличен** нуля, то система (1) имеет единственное решение. Наиболее часто применяется численный метод решения СЛАУ – метод исключения Гаусса. В системе Maple в соответствующих командах он применяется автоматически по умолчанию, т.е. в том случае если явно не указан какой-либо другой метод.

В качестве примера рассмотрим систему уравнений:

$$\begin{cases} x + y + z = 4 \\ 2x + 3y + z = 9 \\ x - y - z = -2 \end{cases}. \quad (2)$$

Соответствующая матрица $A = \begin{pmatrix} 1 & 1 & 1 \\ 2 & 3 & 1 \\ 1 & -1 & -1 \end{pmatrix}$ и заданный $b = \begin{pmatrix} 4 \\ 9 \\ -2 \end{pmatrix}$. Искомый вектор $\vec{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, где обозначено $x_1 = x$, $x_2 = y$, $x_3 = z$. Размерность СЛАУ $n = 3$. Зададим

$$\left| \left(x_3 \right) \right|$$

исходные данные в системе Maple.

```
> restart:
with(linalg):# подключается библиотека линейной алгебры
> #Исследование и решение СЛАУ
n:=3;
# Решение системы Ax=b
```

$$n := 3$$

```
> A:=array(1..n,1..n,[[1,1,1],[2,3,1],[1,-1,-1]]): b:=array(1..n,[4,9,-2]):
> print("A=",A);
print ("b=",b);
```

$$\text{"A="}, \begin{bmatrix} 1 & 1 & 1 \\ 2 & 3 & 1 \\ 1 & -1 & -1 \end{bmatrix}$$

$$\text{"b="}, [4, 9, -2]$$

Прежде чем решать любую систему уравнений, необходимо провести анализ матрицы системы. Определитель системы может быть $\det A = \begin{cases} \neq 0 \\ = 0 \\ \approx 0 \end{cases}$. В первом случае решение существует и единственно, во втором – не существует, в третьем – решение может получиться не верным, а может и не существовать. Другой важной характеристикой системы является ее обусловленность. Для пояснения этого понятия введем некоторые определения. Если получено точное решение системы (1) - $\vec{x}^*$, то при подстановке его в исходное уравнение (1) получим тождество: $A\vec{x}^* \equiv b^*$, что означает $\vec{x}^* = A^{-1}b^*$, где b^* - точное значение правой части (1). Если решение получено приближенно из-за возможных погрешностей применяемого вычислительного метода или накопления вычислительных ошибок, то будем иметь ситуацию при которой $A\vec{x} = b$, т.е. $\vec{x} = A^{-1}b$, где x - приближенно вычисленное решение и b - приближенное значение правой части (1). Тогда можно оценить значения величин: погрешности решения $\delta = \left\| \vec{x}^* - \vec{x} \right\|$ $\left\| A^{-1}\vec{b}^* - A^{-1}\vec{b} \right\| \le \left\| A^{-1} \right\| \cdot \left\| \vec{b}^* - b \right\|$. Обозначим $\sigma = \left\| \vec{b}^* - b \right\|$ и назовем эту величину невязкой. Тогда $\delta \le \left\| A^{-1} \right\| \cdot \sigma$ или $\delta \le M_A \cdot \sigma$, где $M_A = \left\| A^{-1} \right\| = const$. Из последнего выражения следует, что чем ближе к нулю определитель матрицы A, тем больше значение M_A и, следовательно, тем сильнее погрешность правой части (1) может исказить искомое решение.

Заметим, что $\delta = \left\| \vec{x}^* - \vec{x} \right\|$ и $\sigma = \left\| \vec{b}^* - \vec{b} \right\|$ представляют собой абсолютные погрешности решения и правой части соответственно. При решении систем на ЭВМ с числами с плавающей точкой более естественными характеристиками являются относительные погрешности, а именно:

$$\tilde{\delta} = \frac{\delta}{\left\| \vec{x}^* \right\|} = \frac{\left\| \vec{x}^* - \vec{x} \right\|}{\left\| \vec{x}^* \right\|} \quad \text{и} \quad \tilde{\sigma} = \frac{\sigma}{\left\| \vec{b}^* \right\|} = \frac{\left\| \vec{b}^* - \vec{b} \right\|}{\left\| \vec{b}^* \right\|}.$$

Получим оценку, выражающую погрешность решения через относительную погрешность

правой части:

$$\begin{cases} \|x\| = \|A^{-1} \cdot b\| \le \|A^{-1}\| \cdot \|b\| \\ \|b\| = \|Ax\| \le \|A\| \cdot \|x\| \end{cases} \Rightarrow \|x^* - x\| \cdot \|b\| \le \|A^{-1}\| \cdot \|b^* - b\| \cdot \|b\| =$$

$$= \|A^{-1}\| \cdot \|b^* - b\| \cdot \|Ax\| \le \|A^{-1}\| \cdot \|b^* - b\| \cdot \|A\| \cdot \|x\| \Rightarrow$$

$$\Rightarrow \frac{\|x^* - x\|}{\|x\|} \le \|A^{-1}\| \cdot \|A\| \frac{\|b^* - b\|}{\|b\|} = \tilde{\delta} \le \tilde{M}_A \tilde{\sigma},$$

где $\tilde{M}_A = \|A^{-1}\| \cdot \|A\|$. (3)

Число $\tilde{M}_A$ называется числом обусловленности матрицы A и характеризует степень зависимости относительной погрешности решения от относительной погрешности правой части. Матрицы с большим числом обусловленности называются плохо обусловленными матрицами. При численном решении систем с плохо обусловленными матрицами возможно сильное накопление погрешностей.

Вычислим определитель матрицы системы (2) и ее число обусловленности в системе Maple.

> **# исследование определителя матрицы**
det(A);

$$-4$$

Определитель не равен 0, следовательно, матрица системы не вырождена и существует единственное решение (2). Вычислим число обусловленности (3).

> **# вычисление числа обусловленности для матрицы A**

 # вычисление нормы 2 для матрицы A (см. работу 1)
At:=transpose(A);
AtA:=multiply(At,A);

$$At := \begin{bmatrix} 1 & 2 & 1 \\ 1 & 3 & -1 \\ 1 & 1 & -1 \end{bmatrix}$$

$$AtA := \begin{bmatrix} 6 & 6 & 2 \\ 6 & 11 & 5 \\ 2 & 5 & 3 \end{bmatrix}$$

> **lamdaAtA:=evalf(eigenvals(AtA));**

$$lamdaAtA := 16.99581363 - 0.4\,10^{-9}\,I,\ 0.355412918 + 0.2\,10^{-9}\,I,\ 2.648773460 + 0.2\,10^{-9}\,I$$

> **lamdaAtA[2];**

$$0.355412918 + 0.2\,10^{-9}\,I$$

> **roAtA:=max(abs(lamdaAtA[1]),abs(lamdaAtA[2]),abs(lamdaAtA[3]));**

$$roAtA := 16.99581363$$

> **norma2A:=sqrt(roAtA);**

$$norma2A := 4.122597922$$

> **# вычисление нормы 2 для матрицы Aobrat**
Aobrat:=evalf(inverse(A));
det(Aobrat);
multiply(Aobrat,A); # проверка правильности вычисления обратной матрицы

$$Aobrat := \begin{bmatrix} 0.5000000000 & 0. & 0.5000000000 \\ -0.7500000000 & 0.5000000000 & -0.2500000000 \\ 1.250000000 & -0.5000000000 & -0.2500000000 \end{bmatrix}$$

$$-0.2500000000$$

$$\begin{bmatrix} 1.000000000 & 0. & 0. \\ 0. & 1.000000000 & 0. \\ 0. & 0. & 1.000000000 \end{bmatrix}$$

> **Atobrat:=transpose(Aobrat);**

$$Atobrat := \begin{bmatrix} 0.5000000000 & -0.7500000000 & 1.250000000 \\ 0. & 0.5000000000 & -0.5000000000 \\ 0.5000000000 & -0.2500000000 & -0.2500000000 \end{bmatrix}$$

> **AtAobr:=multiply(Atobrat,Aobrat);**

$$AtAobr := \begin{bmatrix} 2.375000000 & -1.000000000 & 0.1250000000 \\ -1.000000000 & 0.5000000000 & 0. \\ 0.1250000000 & 0. & 0.3750000000 \end{bmatrix}$$

> **lAtAobr:=evalf(eigenvals(AtAobr));**

$$lAtAobr := 0.05883801870, 0.3775332305, 2.813628751$$

> **lAtAobr[2];**

$$0.3775332305$$

> **roAtAobr:=max(abs(lAtAobr[1]),abs(lAtAobr[2]),abs(lAtAobr[3]));**

$$roAtAobr := 2.813628751$$

> **nor2AtAobr:=sqrt(roAtAobr);**

$$nor2AtAobr := 1.677387478$$

> **#вычисление числа обусловленности матрицы A**
> **MA:=nor2AtAobr*norma2A;**

$$MA := 6.915194131$$

Число обусловленности $M_{\sim A} \approx 6.92$ не является большим, что должно гарантировать получение качественного решения (2). Получим решение (2), выполним проверку полученного решения, вычислим погрешность решения $\sigma = \left\| \vec{b}^{*} - \vec{b} \right\|$.

> **# решение СЛАУ Ax=b**

```
x:=linsolve(A,b);
```

$$x := [1, 2, 1]$$

> **x[2]:**
> **# проверка решения**

```
bp:=multiply(A,x); # приближенное значение правой части
# погрешность решения
```

> **sigma:=evalf(sqrt(add((b[i]-bp[i])^2,i=1..n)));**

$$bp := [4, 9, -2]$$

$$\sigma := 0.$$

Полученное значение $\sigma = 0$ говорит о том, что мы получили точное решение $\vec{x}^{*}$.

Далее рассмотрим пример системы с плохо обусловленной матрицей.

> **# Пример плохообусловленной матрицы в системе Ax=b**
> **n:=3:**

```
A:=array(1..n,1..n):
b:=array(1..n):
for i from 1 to n do
for j from 1 to n do
A[i,j]:=evalf((1+2*sin(5*i+1.5))*(1+3*cos(-4*j+2.5)))+2.5;
end do:
b[i]:=evalf(1+1.85*cos(5*i+2));
end do:
print("A=",A);
print("b=",b);
```

$$\text{"A="}, \begin{bmatrix} 4.233753497 & 6.970943499 & -0.348346467 \\ 1.589745034 & 0.152666004 & 3.995438378 \\ 1.986542700 & 1.175914090 & 3.343547995 \end{bmatrix}$$

$$\text{"b="}, [2.394719170, 2.561129824, 0.4909478245]$$

> **# анализ матрицы системы**

```
det(A);
```

$$-0.1\,10^{-7}$$

> **Aobrat:=evalf(inverse(A));**

$$Aobrat := \begin{bmatrix} 0.418784617210^{9} & 0.237173096810^{10} & -0.279051558510^{10} \\ -0.262172012210^{9} & -0.148477631510^{10} & 0.174694832710^{10} \\ -0.156612604910^{9} & -0.886954652910^{9} & 0.104356725810^{10} \end{bmatrix}$$

> **det(Aobrat);**

```
multiply(Aobrat,A); # проверка правильности вычисления обратной матрицы
```

$$-0.156612605010^{18}$$

$$\begin{bmatrix} 1. & -1. & 1. \\ 0. & 1. & -2. \\ -1. & 0. & 1. \end{bmatrix}$$

Анализ полученных результатов позволяет сделать вывод о том, что определитель $\det A \approx 0$ - матрица практически вырождена, т.е. решение, скорее всего, не существует. Обратная матрица имеет очень большое значение определителя, при этом обратная матрица формируется не верно, т.е. фактически не формируется. Вычислим число обусловленности.

> **# вычисление числа обусловленности для матрицы A**

```
# вычисление нормы 2 для матрицы A
At:=transpose(A):
```

> **AtA:=multiply(At,A):**

> **lamdaAtA:=evalf(eigenvals(AtA)):**

> **lamdaAtA[2];**

$$28.41393706$$

> **roAtA:=max(abs(lamdaAtA[1]),abs(lamdaAtA[2]),abs(lamdaAtA[3]));**

$$roAtA := 73.24869320$$

> **norma2A:=sqrt(roAtA);**

$$norma2A := 8.558545040$$

> **# вычисление нормы 2 для матрицы Aobrat**

> **Atobrat:=transpose(Aobrat);**

$$Atobrat := \begin{bmatrix} 0.418784617210^{9} & -0.262172012210^{9} & -0.156612604910^{9} \\ 0.237173096810^{10} & -0.148477631510^{10} & -0.886954652910^{9} \\ -0.279051558510^{10} & 0.174694832710^{10} & 0.104356725810^{10} \end{bmatrix}$$

> **AtA:=multiply(Atobrat,Aobrat);**

$$AtA := \begin{bmatrix} 0.268642227610^{18} & 0.152141951910^{19} & -0.179006174610^{19} \\ 0.152141951910^{19} & 0.861635704710^{19} & -0.101377765610^{20} \\ -0.179006174610^{19} & -0.101377765610^{20} & 0.119278383110^{20} \end{bmatrix}$$

> **lAtAobr:=evalf(eigenvals(AtA));**

$$lAtAobr := -0.219536812910^{9}, 0.478532510610^{10}, 0.208128375810^{20}$$

> **lAtAobr[2];**

$$0.478532510610^{10}$$

> **roAtAobr:=max(abs(lAtAobr[1]),abs(lAtAobr[2]),abs(lAtAobr[3]));**

$$roAtAobr := 0.208128375810^{20}$$

> **norma2Aobr:=sqrt(roAtAobr);**

$$norma2Aobr := 0.456210889610^{10}$$

> **#вычисление числа обусловленности**
> **MA:=norma2Aobr*norma2A;**

$$MA := 0.390450144610^{11}$$

Число обусловленности очень большое – система плохо обусловлена. Попытаемся получить решение данной системы.

> **# решение СЛАУ Ax=b**
> **x:=linsolve(A,b);**

$$x :=$$

Решение не найдено.

Попытаемся все же решить данную систему, используя так называемый «параметр регуляризации» $0 \le \alpha \le 1$. Построим матрицу системы следующим образом $A_\alpha = A + \alpha I$ При достаточно малом значении $\alpha = 0.001$ исходная матрица изменится не значительно. В итоге получим систему

$$A_\alpha \vec{x} = b \qquad (4)$$

Исследуем полученную матрицу системы (4). Обозначим $A_\alpha = Ap$.

> **alfa:=0.001;**

```
Ap:=array(1..n,1..n): #приближенное значение матрицы A - A(alfa)
```

$$alfa := 0.001$$

> **for i from 1 to n do for j**
from 1 to n do

```
if i=j then
Ap[i,j]:=A[i,j]+alfa;
else
Ap[i,j]:=A[i,j];
end if;
end do:
end do:
print(Ap);
```

$$\begin{bmatrix} 4.234753497 & 6.970943499 & -0.348346467 \\ 1.589745034 & 0.153666004 & 3.995438378 \\ 1.986542700 & 1.175914090 & 3.344547995 \end{bmatrix}$$

> **det(Ap);**

$$0.00023198$$

> **Apobrat:=evalf(inverse(Ap));**

$$Apobrat := \begin{bmatrix} -18037.54185 & -102268.6465 & 120292.7158 \\ 11294.63909 & 64037.16463 & -75323.21196 \\ 6742.561885 & 38229.03102 & -44966.31244 \end{bmatrix}$$

> **det(Apobrat);**
multiply(Apobrat,Ap);

$$12716.38$$

$$\begin{bmatrix} 1.0001 & 0.0001 & 0.0003 \\ 0. & 0.99989 & 0.0001 \\ 0.00002 & 0. & 1.0000 \end{bmatrix}$$

Сравнивая между собой исходную матрицу A и $A_a = Ap$ видим, что эти матрицы почти одинаковые. Но значение определителя A_a намного больше, а значение определителя обратной матрицы A_a^{-1} - намного меньше: $\det A = -0.1 \cdot 10^{-7}$ $\det A_a = 0.23 \cdot 10^{-3}$; $\det A^{-1} = -0.16 \cdot 10^{18}$ $\det A_a{}^{-1} = 0.1271638 \cdot 10^{5}$. Кроме того, видно, что формируется почти правильно единичная матрица $I = A^{-1} \cdot A_a$, т.е. – почти правильная обратная матрица.

Определим число обусловленности полученной матрицы.

> **# вычисление числа обусловленности для матрицы Ap**

```
# вычисление нормы 2 для матрицы Ap
Apt:=transpose(Ap);
```

$$Apt := \begin{bmatrix} 4.234753497 & 1.589745034 & 1.986542700 \\ 6.970943499 & 0.153666004 & 1.175914090 \\ -0.348346467 & 3.995438378 & 3.344547995 \end{bmatrix}$$

> **AptAp:=multiply(Apt,Ap);**

$$AptAp := \begin{bmatrix} 24.40677835 & 32.10052068 & 11.52065430 \\ 32.10052068 & 50.00044046 & 2.118560622 \\ 11.52065430 & 2.118560622 & 27.27087438 \end{bmatrix}$$

> **lamdaAptAp:=evalf(eigenvals(AptAp));**

$$lamdaAptAp := 0.204746442010^{-9}, 28.41659632, 73.26149687$$

> **lamdaAptAp[2];**

$$28.41659632$$

>

roAptAp:=max(abs(lamdaAptAp[1]),abs(lamdaAptAp[2]),abs(lamdaAptAp[3]));

$$roAptAp := 73.26149687$$

> **norma2Ap:=sqrt(roAptAp);**

$$norma2Ap := 8.559293012$$

> **# вычисление нормы 2 для матрицы Apobrat**
> **Aptobrat:=transpose(Apobrat):**
> **AptobrApobr:=multiply(Aptobrat,Apobrat);**

$$AptobrApobr := \begin{bmatrix} 0.498383929010^{9} & 0.282571326210^{10} & -0.332372153310^{10} \\ 0.282571326210^{10} & 0.160210933210^{11} & -0.188446767110^{11} \\ -0.332372153310^{10} & -0.188446767110^{11} & 0.221658929810^{11} \end{bmatrix}$$

> **lApobrat:=evalf(eigenvals(AptobrApobr));**

$$lApobrat := -9.234398228, 0.1919369661, 0.386853702410^{11}$$

> **lApobrat[2];**

$$0.1919369661$$

>

roApobrat:=max(abs(lApobrat[1]),abs(lApobrat[2]),abs(lApobrat[3]));

> **nor2Apobrat:=sqrt(roApobrat);**

$$nor2Apobrat := 196685.9686$$

> **#вычисление числа обусловленности**
> **MA:=nor2Apobrat*norma2Ap;**

$$MA := 0.168349283710^{7}$$

Результаты вычислений показывают, что $\|A\| = 8.558545040$ отличается от $\|A_a\| = 8.559293012$ на 0,000747972, т.е. не значительно. В то же время $\|A^{-1}\| \approx 0.46 \cdot 10^{10}$ отличается от $\|A_a^{-1}\| \approx 0.20 \cdot 10^{6}$ в 2300 раз в сторону уменьшения нормы. Число обусловленности $M_{A_a} \approx 0.17 \cdot 10^{7}$ в 2300 раз меньше чем $M_A \approx 0.39 \cdot 10^{11}$. Эти результаты дают надежду получить решение системы.

> **# решение СЛАУ Apx=b**
print("b=",b);
x:=linsolve(Ap,b);

$$"b=", [2.394719170, 2.561129824, 0.4909478245]$$

$$x := [-246077.1190, 154085.5069, 91986.08518]$$

Как видно решение найдено, выполним проверку решения путем его подстановки в исходное уравнение.

> **# проверка решения**
bp:=multiply(Ap,x); #
погрешность решения

- **sigma:=evalf(sqrt(add((b[i]-bp[i])^2,i=1..n)));**
-

$$bp := [2.39521, 2.5612, 0.4910]$$

$$\sigma := 0.0004985589661$$

Очевидно, что получено решение с погрешностью $\sigma \approx$ - очень хороший результат. 0.0005

Исследуем устойчивость полученного выше решения по правой части. Сформируем вектор $\vec{b} = \{b_i\}_n$, $b_i = b_i + \beta$ $0 \le \beta \le 1$, например, положим $\beta = 0.0001$, $\vec{b} \approx \vec{b}$, решим , т.е.
систему $A_a \vec{x} = \vec{b}$ и получим приближенное решение $\vec{x}$, вычислим $\|\vec{x} - \vec{x}\|$ и $\|\vec{b} - \vec{b}\|$ и проверим выполнение свойства (4). Обозначим $\vec{x}$ как xp, $\vec{b}$ как bp.

> **#Исследование решений СЛАУ на устойчивость к погрешностям в правой части**
> **# решение СЛАУ Ap*x=bp**

```
beta:=0.0001;
for i from 1 to n do
bp[i]:=b[i]+beta;
end do:
xp:=linsolve(Ap,bp):
x:=linsolve(Ap,b):
# погрешность решения
print("b=",b); print("bp=",bp);
print("x=",x);
print("xp=",xp);
sigmaxp:=evalf(sqrt(add((x[i]-xp[i])^2, i=1..n)));
sigmab:=evalf(sqrt(add((b[i]-bp[i])^2, i=1..n)));
```

$$\beta := 0.0001$$

"b=", [2.394719170, 2.561129824, 0.4909478245]

"bp=", [2.394819170, 2.561229824, 0.4910478245]

"x=", [-246062.3181, 154076.2391, 91980.55250]

"xp=", [-246062.3194, 154076.2398, 91980.55300]

$$sigmaxp := 0.001558845727$$

$$sigmab := 0.0001732050808$$

Устойчивость решения при $\beta = 0.0001$ очевидна, поскольку $\|\vec{x} - \vec{x}\|_b \approx 0.0016 \le M \cdot \|\vec{b} - \vec{b}\| \approx M \cdot 0.00017$, $M \approx 0.1$. Постепенно увеличивая β, откуда
будем получать: при $\beta = 0.001$, $\|\vec{x} - \vec{x}\|_b \approx 0.016 \le M \cdot \|\vec{b} - \vec{b}\| \approx M \cdot 0.0017$; при получим

$\beta = 0.01$, получим $x - x \approx 0.16 \leq M \cdot b - \approx M \cdot 0.017$. Последний результат
b

$\left\|\vec{x} - \tilde{x}\right\| \approx 0.16$ говорит о том, что погрешность в решении составляет уже около 16%. Это достаточно большая погрешность, поэтому на этом этапе вычислительный эксперимент необходимо остановить. Вывод: при $0 \le \beta \le 0.01$ получаемые решения устойчивы по правой части СЛАУ (4).

Задание на самостоятельную работу.

1. **Сформировать матрицу системы A и ее правую часть с помощью функций согласно варианту.**
2. **Провести анализ матрицы системы: ее определителя, определителя обратной матрицы, числа обусловленности. Попытаться получить решение системы.**
3. **Сформировать матрицу** A_α, подобрать значение α, позволяющее уменьшить число обусловленности, с тем, чтобы было возможно получить решение системы. Провести сравнительный анализ характеристик A и A_α так, как это было выполнено в работе. Получить решение новой системы, сделать выводы.
4. **Исследовать последнюю систему – ее решений на устойчивость по правой части. Определить возможный диапазон изменения значений β, при которых решения будут устойчивыми и будут иметь при этом приемлемую точность.**

Варианты:

Номер варианта	
1	$a_{ij} = (1+ 2.5\sin(5i +1.25))(1+ 3.6\cos(-2j + 2.5)) + 3.5;$ $b_i = 1-1.85\cos(5i + 2);\ n = 5$
2	$a_{ij} = (1+ 2.5\sin(-5i +1.25))(1- 3.6\cos(-2j + 2.5)) + 3.5;$ $b_i = 1-1.85\cos(5i - 2);\ n = 4$
3	$a_{ij} = (1.5 + 2.5\sin(-5i +1.25)) + (1- 3.6\cos(-2j + 2.5)) + 3.5;$ $b_i = 2-1.85\cos(5i - 2);\ n = 6$
4	$a_{ij} = (1.75 + 2.25\sin(-5i +1.25)) + (1+ 3.6\cos(-2j + 2.5)) +1.5;$ $b_i = 2 + 2.85\cos(5i - 2);\ n = 5$
5	$a_{ij} = (1.75 + 2.65\sin(5i +1.25)) - (1+ 3.6\cos(-2j + 2.5)) + 5.5;$ $b_i = 2.76 + 2.85\cos(5i + 2.5);\ n = 5$
6	$a_{ij} = (1.75 + 2.65\sin(5i +1.25)) - (1+ 3.6\sin(-2j + 2.5)) + 5.5;$ $b_i = 2.76 + 2.85\sin(5i + 2.5);\ n = 5$
7	$a_{ij} = (1.75 + 2.65\sin(5i +1.25))(1+ 3.6\sin(-2j + 2.5)) + 3.5;$ $b_i = 2.76 + 2.85\sin(5i + 2.5);\ n = 6$
8	$a_{ij} = (1.75 + 2.65\sin(5i +1.25))(2 + 3.6\sin(-2j + 2.5)) + 3.5;$ $b_i = 2.76 + 2.45\sin(5i + 2.5);\ n = 4$
9	$a_{ij} = (1.75 + 2.65\sin(5i +1.25))(2.8 + 3.6\sin(-3j + 2.5)) + 3.5;$ $b_i = 2.76 + 2.45\sin(5i + 2.5);\ n = 6$
10	$a_{ij} = (1.75 + 2.65\sin(5i +1.25))(2.8 + 3.6\sin(-9j + 2.5)) + 5.5;$ $b_i = 2.76 + 2.45\sin(5i + 2.5);\ n = 5$

Отчет по работе должен содержать название работы, ее цель, задание на самостоятельное исполнение в соответствии с вариантом и распечатку программы с результатами выполнения.

Лабораторная работа № 9

Итерационные методы решения СЛАУ Якоби и Гаусса-Зейделя

Цель работы: освоить технологию программирования алгоритмов методов Якоби и Гаусса-Зейделя в системе Maple и методологию исследования итерационных методов решения СЛАУ в вычислительном эксперименте.

Краткие теоретические сведения и практическая реализация итерационных методов в системе Maple.

Алгоритм исключения Гаусса решения системы

$$A\vec{x} = b \tag{1}$$

представляет собой конечный, или прямой метод. В конце вычислений получается точный ответ, если оставить без внимания ошибки округления. Итерационные методы решения $A\vec{x} = b$ являются бесконечными методами и вычисляют только приближенные ответы.

Итерационные методы привлекательны для разреженных матриц с диагональным преобладанием элементов матрицы A:

$$|a_{ii}| > \sum_{\substack{j=1 \\ i \neq j}}^{n} |a_{ij}| \quad \text{или} \quad |a_{jj}| > \sum_{\substack{i=1 \\ i \neq j}}^{n} |a_{ij}|. \tag{2}$$

В этом случае методы Якоби и Гаусса-Зейделя обычно сходятся и при этом требуют гораздо меньше оперативной памяти, чем прямые методы. Обычно метод Якоби сходится медленнее, чем метод Гаусса-Зейделя, хотя и не всегда. Вопросы сходимости итерационных методов, а также скорости сходимости требуют исследования в вычислительном эксперименте в каждом конкретном случае.

Алгоритм метода Якоби имеет вид:

$$x_1^{(v)} = -\sum_{j=2}^{n} \frac{a_{1j}}{a_{11}} x_j^{(v-1)} + \frac{b_1}{a_{11}} \quad \text{при } i = 1; \tag{3}$$

$$x_i^{(v)} = -\sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} x_j^{(v-1)} - \sum_{j=i+1}^{n} \frac{a_{ij}}{a_{ii}} x_j^{(v-1)} + \frac{b_i}{a_{ii}}, \text{ при } 1 < i < n; \tag{4}$$

$$x_n^{(v)} = -\sum_{j=1}^{n-1} \frac{a_{nj}}{a_{nn}} x_j^{(v-1)} + \frac{b_n}{a_{nn}}, \text{ при } i = n. \tag{5}$$

В выражениях (3)-(5) обозначено: n - размерность задачи (1), v - номер итерации, $v = 1,2,3\ldots.$, $\vec{x}^{(0)} = \left(x_1^{(0)}, x_2^{(0)} \ldots\ldots x_n^{(0)}\right)$ - нулевое приближение к решению системы, задается произвольно, обычно равным нулю. Критерий остановки алгоритма:

$$\sigma_v = \frac{\left\| \vec{x}^{(v)} - \vec{x}^{(v-1)} \right\|}{\left\| \vec{x}^{(v)} \right\|} = \sqrt{\frac{\sum_{i=1}^{n} \left(x_i^{(v)} - x_i^{(v-1)} \right)^2}{\sum_{i=1}^{n} \left(x_i^{(v)} \right)^2}} \leq \varepsilon. \tag{6}$$

Окончательное решение $\vec{x}^{*} = \vec{x}^{(v)}$. Проверка правильности решения осуществляется путем подстановки $\vec{x}^{*} = \vec{x}^{(v)}$ в систему (1). При этом оценивается относительная погрешность правых частей (1):

$$\delta = \frac{\left\| \vec{b} - \tilde{\vec{b}} \right\|}{\left\| \vec{b} \right\|} = \frac{\sqrt{\sum_{i=1}^{n} \left(b_i - \tilde{b} \right)^2}}{\sqrt{n \left(b \right)^2}}, \quad \sigma_v \leq M \cdot \delta, \quad M = const, \quad \tilde{\vec{b}} = A \cdot \vec{x}^{*}. \tag{7}$$

$$\sum_{i=1} i$$

В методе Гаусса-Зейделя вычисления почти такие же, как и в методе Якоби. Различие состоит в том, что новые значения x_i используются здесь сразу же по мере получения, в то время как в методе Якоби они не используются до следующей итерации:

$$x_1^{(v)} = -\sum_{j=2}^{n} \frac{a_{1j}}{a_{11}} x_j^{(v-1)} + \frac{b_1}{a_{11}} \quad \text{при } i = 1; \tag{8}$$

$$x_i^{(v)} = -\sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} x_j^{(v)} - \sum_{j=i+1}^{n} \frac{a_{ij}}{a_{ii}} x_j^{(v-1)} + \frac{b_i}{a_{ii}}, \text{ при } 1 < i < n; \tag{9}$$

$$x_n^{(v)} = -\sum_{j=1}^{n-1} \frac{a_{nj}}{a_{nn}} x_j^{(v)} + \frac{b_n}{a_{nn}}, \text{ при } i = n. \tag{10}$$

Реализуем данные методы в системе Maple. Для этого сформируем матрицу системы с почти диагональным преобладанием по следующему алгоритму. Пусть $A = \{a_{ij}\}$, $i, j = \overline{1,n}$, где $a_{ij} = \sin(i) \cdot \cos(j)$. В результате все элементы данной матрицы будут по модулю меньше 1. Затем увеличим значения элементов главной и двух побочных диагоналей следующим образом:

$a_{i,i-1} = a_{i,i-1} + (m/2)$, для $i > 1$;

$a_{i,i} = a_{i,i} + m$; $\quad i = \overline{1,n}$.

$a_{i,i+1} = a_{i,i+1} + (m/2)$, для $i < n$;

Положим $m = n$. Для анализа полученной матрицы вычислим ее определитель. Правую часть системы (1) сформируем так: $b = (b_1,...,b_n)^T$, $b_i = w \cdot \sin(i) \cdot \cos(i)$, $i = \overline{1,n}$. Положим $w = n$.

```
> restart:
with(linalg):# подключается библиотека линейной алгебры
> #Исследование и решение СЛАУ итерационными методами
n:=5; #размерность СЛАУ
m:=n: w:=n:
# Решение системы Ax=b
```

$$n := 5$$

```
> # формирование матрицы системы с преобладанием #диагональных
элементов на главной диагонали и двух побочных #диагоналях - ленточная
матрица
A:=array(1..n,1..n):
for i from 1 to n do
for j from 1 to n do
A[i,j]:=evalf(sin(i)*cos(j));
end do:
end do:
#print(A);
#det(A):
for i from 1 to n do
if i>1 then
A[i,i-1]:=A[i,i-1]+evalf((m/2));
```

```
end if;
A[i,i]:=A[i,i]+m;
if i<n then
A[i,i+1]:=A[i,i+1]+evalf((m/2));
```

```
end if;
end do:
print(A);
det(A);
```

$$\begin{bmatrix} 5.454648713 & 2.149824512 & -0.8330499611 & -0.5500221414 & 0.2386934986 \\ 2.991295496 & 4.621598752 & 1.599802370 & -0.5943564625 & 0.2579332954 \\ 0.07624746579 & 2.441273355 & 4.860292251 & 2.407757807 & 0.04003040992 \\ -0.4089021333 & 0.3149409643 & 3.249228792 & 5.494679123 & 2.285323750 \\ -0.5181089968 & 0.3990533034 & 0.9493278368 & 3.126794735 & 4.727989444 \end{bmatrix}$$

589.4211417

> **# формирование вектора правой части системы**

```
b:=array(1..n):
for i from 1 to n do
b[i]:=evalf(w*sin(i)*cos(i));
end do:
print(b);
```

[2.273243567, -1.892006238, -0.6985387455, 2.473395616, -1.360052778]

Определитель матрицы системы отличен от нуля, поэтому система (1) имеет единственное решение.

Реализуем алгоритм метода Якоби (3)-(5) с условием останова (6) и проверкой полученного решения $\tilde{x}$ (7). При этом обозначим $v = niu$ $\vec{x}^{(v)} = xtek$ $\vec{x}^{(v-1)} = xpred$, $\sigma_v = sigmaotn$, $b = bp$, $\delta = delta$ $\varepsilon = eps$. Зададим начальные значения $\vec{\sigma}^{(0)}_x = \left(x^{(0)}_1 = 0, x^{(0)}_2 = 0, .. x^{(0)}_n = 0\right)$, $= 10$, $\varepsilon = 0.1$.

> **niu:=0: # номер итерации**

```
xtek:=array(1..n): xpred:=array(1..n):
```

> **sigmaotn:=10.0: eps:=0.1;**

```
# задается вектор начальных значений
for i from 1 to n do
xpred[i]:=0;
end do:
```

$$eps := 0.01$$

> **# метод Якоби;**

```
while sigmaotn > eps do # проверка условия сходимости
niu:=niu+1; # вычисление количества итераций
#  вычисление текущего вектора xtek на основе предыдущего xpred
for i from 1 to n do
if i=1 then
p:=0;
else
p:=evalf(add((A[i,j]/A[i,i])*xpred[j],j=1..i-1));
end if;
if i=n then
g:=0;
else
g:=evalf(add((A[i,j]/A[i,i])*xpred[j],j=i+1..n));
end if;
h:=evalf(b[i]/A[i,i]);
```

```
xtek[i]:=evalf(-p-g+h);
end do;
# вычисление значения относительной погрешности решения
sigmaotn:=(sqrt(add((xtek[i]-xpred[i])^2,
i=1..n)))/(sqrt(add((xtek[i])^2, i=1..n)));
# текущий вектор переписывается в предыдущий
for i from 1 to n do
xpred[i]:=xtek[i];
end do;
print("x=",xtek);
end do;
```

niu := 1

sigmaotn := 1.000000000

"x=", [0.4167534312, -0.4093834925, -0.1437236095, 0.4501437774, -0.2876598592]

niu := 2

sigmaotn := 0.3365908597

"x=", [0.6141310001, -0.5554282723, -0.1652617039, 0.7092544125, -0.4762763617]

niu := 3

sigmaotn := 0.1546901757

"x=", [0.7027831726, -0.6318744115, -0.2218096549, 0.8234986746, -0.6093554048]

niu := 4

sigmaotn := 0.07880807476

"x=", [0.7416198608, -0.6475598854, -0.2403021409, 0.9232664680, -0.6573881348]

```
> # проверка решения
bp:=multiply(A,xtek); # приближенное значение правой части
# погрешность решения
print("b=",b);
> delta:=evalf(sqrt(add((b[i]-bp[i])^2,
i=1..n))/sqrt(add((b[i])^2, i=1..n)));
>
```

bp := [2.188588119, -1.877105424, -0.4955761625, 2.282718557, -1.092035776]

"b=", [2.273243567, -1.892006238, -0.6985387455, 2.473395616, -1.360052778]

δ := 0.09546354132

Таким образом, при $n = 5$ $\varepsilon = 0.1$ потребовалось в методе Якоби 4 итерации. При этом

, $\sigma_4 \approx 0.079$ и $\delta \approx 0.095$. Поскольку $\leq M \cdot \delta$, $M = 0.83$.

σ_V то

Реализуем алгоритм метода Гаусса-Зейделя, проанализируем полученный результат.

```
> # Метод Гаусса-Зейделя
> sigmaotn:=10.0: niu:=0:
for i from 1 to n do
xpred[i]:=0;
end do:
> while sigmaotn > eps do # проверка условия сходимости
```

```
niu:=niu+1; # вычисление количества итераций
#  вычисление текущего вектора на основе предыдущего
for i from 1 to n do
if i=1 then
p:=0;
else
p:=evalf(add((A[i,j]/A[i,i])*xtek[j],j=1..i-1));
end if;
if i=n then
g:=0;
else
g:=evalf(add((A[i,j]/A[i,i])*xpred[j],j=i+1..n));
end if;
h:=evalf(b[i]/A[i,i]);
xtek[i]:=evalf(-p-g+h);
end do;
# вычисление значения относительной погрешности
sigmaotn:=(sqrt(add((xtek[i]-xpred[i])^2,
i=1..n)))/(sqrt(add((xtek[i])^2, i=1..n)));
# текущий вектор переписывается в предыдущий
for i from 1 to n do
xpred[i]:=xtek[i];
end do;
print("x=",xtek);
end do;
```

niu := 1

sigmaotn := 1.000000000

"x=", [0.4167534312, -0.6791240582, 0.1908552576, 0.4072227915, -0.4923041165]

niu := 2

sigmaotn := 0.3556029017

"x=", [0.7761681174, -0.8979723742, 0.0974611585, 0.7064982870, -0.6136164220]

niu := 3

sigmaotn := 0.1478424100

"x=", [0.8836449069, -0.8899482955, -0.0555153486, 0.8549535331, -0.6699789778]

niu := 4

sigmaotn := 0.1003404667

"x=", [0.8745553415, -0.8088735659, -0.1691753003, 0.9402839715, -0.7114284641]

niu := 5

sigmaotn := 0.08042361103

"x=", [0.8356612683, -0.7310682336, -0.2495766998, 0.9977140916, -0.7440945172]

> # проверка решения

```
bp:=multiply(A,xtek); # приближенное значение правой части
# погрешность решения
print(b);
```

> **delta:=evalf(sqrt(add((b[i]-bp[i])^2,**
i=1..n))/sqrt(add((b[i])^2, i=1..n)));

bp := [2.468104748, -2.063192213, -0.5615685605, 2.398743109, -1.360052777]

[2.273243567, -1.892006238, -0.6985387455, 2.473395616, -1.360052778]

δ := 0.07297455601

Таким образом, при $n = 5$ $\varepsilon = 0.1$ потребовалось в методе Гаусса-Зейделя 5 итераций. При этом $\sigma_5 \approx 0.080$ и $\delta \approx 0.073$. Поскольку $\sigma_V \le M \cdot \delta$, $M = 1.09$. то

По результатам вычислений построим таблицу 1

$n = 5,\ \varepsilon = 0.1$			
Метод Якоби		Метод Гаусса-Зейделя	
Номер итерации V	Относительная погрешность решения σ_V	Номер итерации V	Относительная погрешность решения σ_V
1	1.000	1	1.000
2	0.336	2	0.355
3	0.154	3	0.147
4	0.078	4	0.100
		5	0.080
Относительная погрешность правой части системы (1) $\delta = 0.0954$		Относительная погрешность правой части системы (1) $\delta = 0.0729$	

По данной таблице построим графики зависимостей относительной погрешности решения σ_V от количества итераций, определив тем самым скорость сходимости итерационного процесса.

> **with(plots):**
> **p1:=plot([[1,1.00],[2,0.336],[3,0.154],[4,0.078]],**
col-
or=[black],labels=["niu","sigmaotn"],linestyle=[3],thickness=2,
legend=["Method Yarobe(n=5,eps=0.1)"]):
p2:=plot([[1,1.00],[2,0.355],[3,0.147],[4,0.100],[5,0.080]],
col-
or=[black],labels=["niu","sigmaotn"],linestyle=[4],thickness=2,
legend=["Method Gaussa-Zeidelya(n=5,eps=0.1)"]):
display(p1,p2);

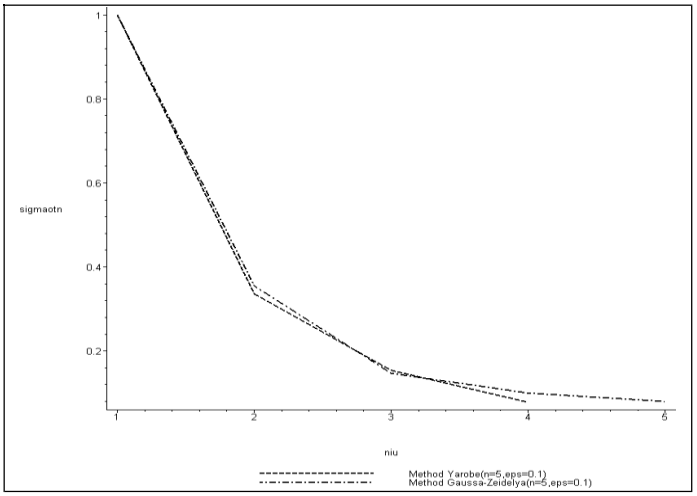


Рисунок 1.

Анализируя рисунок, можно сделать вывод о том, что при $n = 5$ и $\varepsilon = 0.1$ методы работают приблизительно одинаково.

Исследуем поведение вычислительного процесса при увеличении точности расчетов. По результатам вычислений построим следующую таблицу 2.

$n = 5$						
Точность расчетов ε	Метод Якоби			Метод Гаусса-Зейделя		
	Количество итераций V	Относительная погрешность решения σ_V	Относительная погрешность правой части (1) δ	Количество итераций V	Относительная погрешность решения σ_V	Относительная погрешность правой части (1) δ
0,100	4	0,078	0,095	5	0,080	0,073
0,050	6	0,034	0,055	7	0,045	0,040
0,010	14	0,009	0,016	12	0,010	0,009
0,005	18	0,005	0,009	15	0,0040	0,0035
0,001	29	0,0010	0,0018	20	0,0009	0,0008

Анализируя данные таблицы, делаем вывод о том, что при повышении точности расчетов (уменьшении ε) метод Гаусса-Зейделя работает более эффективно: затрачивается меньшее количество итераций, чем в методе Якоби и при этом точность получаемых решений выше, если сравнивать значения относительных погрешностей правых частей уравнения системы (1).

Исследуем поведение вычислительного процесса при увеличении размерности системы. По результатам вычислений построим следующую таблицу 3.

$\varepsilon = 0.01$						
Размерность системы n	Метод Якоби			Метод Гаусса-Зейделя		
	Количество итераций V	Относительная погрешность решения σ_V	Относительная погрешность правой части (1) δ	Количество итераций V	Относительная погрешность решения σ_V	Относительная погрешность правой части (1) δ
5	14	0,0090	0,0160	12	0,0100	0,0090
10	25	0,0097	0,0160	8	0,0090	0,0068
15	43	0,0099	0,0169	14	0,0096	0,0081
20	29	0,0098	0,0178	21	0,0097	0,0090
25	38	0,0099	0,0190	20	0,0097	0,0092

Анализ таблицы показывает, что при больших размерностях системы второй метод Гаусса-Зейделя так же сходится значительно быстрее метода Якоби, обеспечивая при этом большую точность расчетов.

Задание на самостоятельную работу.

1. **Сформировать матрицу системы A и вектор b по алгоритму**

$$a_{i,i-1} = a_{i,i-1} + (m/2), \text{ для } i > 1;$$
$$a_{i,i} = a_{i,i} + m; \qquad i = \overline{1,n}.$$
$$a_{i,i+1} = a_{i,i+1} + (m/2), \text{ для } i < n;$$

где

$A = \{a_{ij}\}$ и $b = \{b_i\}$ определяются согласно варианту.

2. **Вычислить определитель матрицы, сделать соответствующий вывод.**
3. **Реализовать метод Якоби и метод Гаусса-Зейделя, по результатам построить таблицу 1 и рисунок 1, сделать выводы.**
4. **Исследовать вычислительный процесс и построить таблицы 2 и 3, сделать выводы.**

5. Построить графики зависимостей относительной погрешности решения σ_v

от количества итераций при $\varepsilon = 0.01$ и $n = 20$. Сделать вывод о скорости сходимости.

Варианты:

Номер варианта	
1	$a_{ij} = (1+2.5\sin(5i+1.25))(1+3.6\cos(-2j+2.5))$; $b_i = w\left(1-1.85\cos(5i+2)\right)$; $n = 5$, $m = 2\cdot n$, $w = m-n$.
2	$a_{ij} = (1+2.5\sin(-5i+1.25))(1-3.6\cos(-2j+2.5))$; $b_i = w\left(1-1.85\cos(5i-2)\right)$; $n = 4$, $m = 2+n$, $w = n$.
3	$a_{ij} = (1.5+2.5\sin(-5i+1.25))+(1-3.6\cos(-2j+2.5))$; $b_i = w\left(2-1.85\cos(5i-2)\right)$; $n = 6$, $m = n$, $w = 2n$
4	$a_{ij} = (1.75+2.25\sin(-5i+1.25))+(1+3.6\cos(-2j+2.5))$; $b_i = w\left(2+2.85\cos(5i-2)\right)$; $n = 5$, $m = 3n$, $w = 2n$
5	$a_{ij} = (1.75+2.65\sin(5i+1.25))-(1+3.6\cos(-2j+2.5))$; $b_i = w\left(2.76+2.85\cos(5i+2.5)\right)$; $n = 5$, $m = 2n$, $w = n$
6	$a_{ij} = (1.75+2.65\sin(5i+1.25))-(1+3.6\sin(-2j+2.5))$; $b_i = w\left(2.76+2.85\sin(5i+2.5)\right)$; $n = 5$, $m = 1.5n$, $w = 2n$
7	$a_{ij} = (1.75+2.65\sin(5i+1.25))(1+3.6\sin(-2j+2.5))$; $b_i = w\left(2.76+2.85\sin(5i+2.5)\right)$; $n = 6$, $m = n$, $w = 2n$
8	$a_{ij} = (1.75+2.65\sin(5i+1.25))(2+3.6\sin(-2j+2.5))$; $b_i = w\left(2.76+2.45\sin(5i+2.5)\right)$; $n = 4$, $m = 3+n$, $w = 2+n$
9	$a_{ij} = (1.75+2.65\sin(5i+1.25))(2.8+3.6\sin(-3j+2.5))$; $b_i = w\left(2.76+2.45\sin(5i+2.5)\right)$; $n = 6$, $m = 3n$, $w = 3n$
10	$a_{ij} = (1.75+2.65\sin(5i+1.25))(2.8+3.6\sin(-9j+2.5))$; $b_i = w\left(2.76+2.45\sin(5i+2.5)\right)$; $n = 5$, $m = 3+n$, $w = 2n$

Отчет по работе должен содержать название работы, ее цель, задание на самостоятельное исполнение в соответствии с вариантом и распечатку программы с результатами выполнения.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Методические указания
по организации и проведению самостоятельной работы
по дисциплине
«Компьютерная алгебра на Python»
для студентов направления подготовки
02.03.01 Математика и компьютерные науки

Общие положения

Самостоятельная работа - планируемая учебная, учебно-исследовательская, научно-исследовательская работа студентов, выполняемая во внеаудиторное (аудиторное) время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия (при частичном непосредственном участии преподавателя, оставляющем ведущую роль за работой студентов).

Самостоятельная работа студентов (СРС) в ВУЗе является важным видом учебной и научной деятельности студента. Самостоятельная работа студентов играет значительную роль в рейтинговой технологии обучения.

К основным видам самостоятельной работы студентов относятся:

– формирование и усвоение содержания конспекта лекций на базе рекомендованной лектором учебной литературы, включая информационные образовательные ресурсы (электронные учебники, электронные библиотеки и др.);

– написание докладов;

– подготовка к семинарам, практическим и лабораторным работам, их оформление;

– составление аннотированного списка статей из соответствующих журналов по отраслям знаний (педагогических, психологических, методических и др.);

– выполнение учебно-исследовательских работ, проектная деятельность;

– подготовка практических разработок и рекомендаций по решению проблемной ситуации;

– выполнение домашних заданий в виде решения отдельных задач, проведения типовых расчетов, расчетно-компьютерных и индивидуальных работ по отдельным разделам содержания дисциплин и т.д.;

– компьютерный текущий самоконтроль и контроль успеваемости на базе электронных обучающих и аттестующих тестов;

– выполнение курсовых работ (проектов) в рамках дисциплин;

– выполнение выпускной квалификационной работы и др.

Методика организации самостоятельной работы студентов зависит от структуры, характера и особенностей изучаемой дисциплины, объема часов на ее изучение, вида заданий для самостоятельной работы студентов, индивидуальных качеств студентов и условий учебной деятельности.

Процесс организации самостоятельной работы студентов включает в себя следующие этапы:

- подготовительный (определение целей, составление программы, подготовка методического обеспечения, подготовка оборудования);
- основной (реализация программы, использование приемов поиска информации, усвоения, переработки, применения, передачи знаний, фиксирование результатов, самоорганизация процесса работы);
- заключительный (оценка значимости и анализ результатов, их систематизация, оценка эффективности программы и приемов работы, выводы о направлениях оптимизации труда).

Самостоятельная работа по дисциплине **«Компьютерная алгебра на Python»** направлена на формирование следующих **компетенций**:

ПК-1

Способен демонстрировать базовые знания математических и естественных наук, основ программирования и информационных технологий

1. Цель и задачи самостоятельной работы

Ведущая цель организации и осуществления СРС совпадает с целью обучения студента – формирование соответствующего набора компетенций будущего бакалавра по направлению подготовки «Математика и компьютерные науки».

При организации СРС важным и необходимым условием становятся формирование умения самостоятельной работы для приобретения знаний, навыков и возможности организации учебной и научной деятельности. Целью самостоятельной работы студентов является овладение фундаментальными знаниями, профессиональными умениями и навыками деятельности по профилю, опытом творческой, исследовательской деятельности. Самостоятельная работа студентов способствует развитию самостоятельности, ответственности и организованности, творческого подхода к решению проблем учебного и профессионального уровня.

Задачами СРС являются:

- систематизация и закрепление полученных теоретических знаний и практических умений студентов;
- углубление и расширение теоретических знаний;
- формирование умений использовать нормативную, правовую, справочную документацию и специальную литературу;
- развитие познавательных способностей и активности студентов: творческой инициативы, самостоятельности, ответственности и организованности;
- формирование самостоятельности мышления, способностей к саморазвитию, самосовершенствованию и самореализации;
- развитие исследовательских умений;
- использование материала, собранного и полученного в ходе самостоятельных занятий на семинарах, на практических и лабораторных занятиях, при написании курсовых и выпускной квалификационной работ, для эффективной подготовки к итоговым зачетам и экзаменам.

2. Технологическая карта самостоятельной работы студента

Коды реализуемых компетенций	Вид деятельности студентов	Средства и технологии оценки	Объем часов, в том числе (акад.)		
			СРС	Контактная работа с преподавателем	Всего
4семестр					
ПК-1	Самостоятельное изучение литературы	Собеседование	28	2	30
ПК-1	Решение практических задач	Конспект, отчет	22	2	24
ПК-1	Подготовка к экзамену	Вопросы к экзамену	52	2	54
		Итого за 4 семестр	**102**	**6**	**108**

Итого	**102**	**6**	**108**

3. Порядок выполнения самостоятельной работы студентом

3.1. Методические рекомендации по работе с учебной литературой

При работе с книгой необходимо подобрать литературу, научиться правильно ее читать, вести записи. Для подбора литературы в библиотеке используются алфавитный и систематический каталоги.

Важно помнить, что рациональные навыки работы с книгой - это всегда большая экономия времени и сил.

Правильный подбор учебников рекомендуется преподавателем, читающим лекционный курс. Необходимая литература может быть также указана в методических разработках по данному курсу.

Изучая материал по учебнику, следует переходить к следующему вопросу только после правильного уяснения предыдущего, описывая на бумаге все выкладки и вычисления (в том числе те, которые в учебнике опущены или на лекции даны для самостоятельного вывода).

При изучении любой дисциплины большую и важную роль играет самостоятельная индивидуальная работа.

Особое внимание следует обратить на определение основных понятий курса. Студент должен подробно разбирать примеры, которые поясняют такие определения, и уметь строить аналогичные примеры самостоятельно. Нужно добиваться точного представления о том, что изучаешь. Полезно составлять опорные конспекты. При изучении материала по учебнику полезно в тетради (на специально отведенных полях) дополнять конспект лекций. Там же следует отмечать вопросы, выделенные студентом для консультации с преподавателем.

Выводы, полученные в результате изучения, рекомендуется в конспекте выделять, чтобы они при перечитывании записей лучше запоминались.

Опыт показывает, что многим студентам помогает составление листа опорных сигналов, содержащего важнейшие и наиболее часто употребляемые формулы и понятия. Такой лист помогает запомнить формулы, основные положения лекции, а также может служить постоянным справочником для студента.

Чтение научного текста является частью познавательной деятельности. Ее цель – извлечение из текста необходимой информации. От того на сколько осознанна читающим собственная внутренняя установка при обращении к печатному слову (найти нужные сведения, усвоить информацию полностью или частично, критически проанализировать материал и т.п.) во многом зависит эффективность осуществляемого действия.

Выделяют ***четыре основные установки в чтении научного текста***:

информационно-поисковый (задача – найти, выделить искомую информацию)

усваивающая (усилия читателя направлены на то, чтобы как можно полнее осознать и запомнить как сами сведения излагаемые автором, так и всю логику его рассуждений)

аналитико-критическая (читатель стремится критически осмыслить материал, проанализировав его, определив свое отношение к нему)

творческая (создает у читателя готовность в том или ином виде – как отправной пункт для своих рассуждений, как образ для действия по аналогии и т.п. – использовать суждения автора, ход его мыслей, результат наблюдения, разработанную методику, дополнить их, подвергнуть новой проверке).

Основные виды систематизированной записи прочитанного:

Аннотирование – предельно краткое связное описание просмотренной или прочитанной книги (статьи), ее содержания, источников, характера и назначения;

Планирование – краткая логическая организация текста, раскрывающая содержание и структуру изучаемого материала;

Тезирование – лаконичное воспроизведение основных утверждений автора без привлечения фактического материала;

Цитирование – дословное выписывание из текста выдержек, извлечений, наиболее существенно отражающих ту или иную мысль автора;

Конспектирование – краткое и последовательное изложение содержания прочитанного.

Конспект – сложный способ изложения содержания книги или статьи в логической последовательности. Конспект аккумулирует в себе предыдущие виды записи, позволяет всесторонне охватить содержание книги, статьи. Поэтому умение составлять план, тезисы, делать выписки и другие записи определяет и технологию составления конспекта.

Методические рекомендации по составлению конспекта:

1. Внимательно прочитайте текст. Уточните в справочной литературе непонятные слова. При записи не забудьте вынести справочные данные на поля конспекта;
2. Выделите главное, составьте план;
3. Кратко сформулируйте основные положения текста, отметьте аргументацию автора;
4. Законспектируйте материал, четко следуя пунктам плана. При конспектировании старайтесь выразить мысль своими словами. Записи следует вести четко, ясно.
5. Грамотно записывайте цитаты. Цитируя, учитывайте лаконичность, значимость мысли.

В тексте конспекта желательно приводить не только тезисные положения, но и их доказательства. При оформлении конспекта необходимо стремиться к емкости каждого предложения. Мысли автора книги следует излагать кратко, заботясь о стиле и выразительности написанного. Число дополнительных элементов конспекта должно быть логически обоснованным, записи должны распределяться в определенной последовательности, отвечающей логической структуре произведения. Для уточнения и дополнения необходимо оставлять поля.

Овладение навыками конспектирования требует от студента целеустремленности, повседневной самостоятельной работы.

3.2. Методические рекомендации по подготовке к практическим и лабораторным занятиям

Для того чтобы практические и лабораторные занятия приносили максимальную пользу, необходимо помнить, что упражнение и решение задач проводятся по вычитанному на лекциях материалу и связаны, как правило, с детальным разбором отдельных вопросов лекционного курса. Следует подчеркнуть, что только после усвоения лекционного материала с определенной точки зрения (а именно с той, с которой он излагается на лекциях) он будет закрепляться на практических занятиях как в результате обсуждения и анализа лекционного материала, так и с помощью решения проблемных ситуаций, задач. При этих условиях студент не только хорошо усвоит материал, но и научится применять его на практике, а также получит дополнительный стимул (и это очень важно) для активной проработки лекции.

При самостоятельном решении задач нужно обосновывать каждый этап решения, исходя из теоретических положений курса. Если студент видит несколько путей решения проблемы (задачи), то нужно сравнить их и выбрать самый рациональный. Полезно до начала вычислений составить краткий план решения проблемы (задачи). Решение проблемных задач или примеров следует излагать подробно, вычисления располагать в

строгом порядке, отделяя вспомогательные вычисления от основных. Решения при необходимости нужно сопровождать комментариями, схемами, чертежами и рисунками.

Следует помнить, что решение каждой учебной задачи должно доводиться до окончательного логического ответа, которого требует условие, и по возможности с выводом. Полученный ответ следует проверить способами, вытекающими из существа данной задачи. Полезно также (если возможно) решать несколькими способами и сравнить полученные результаты. Решение задач данного типа нужно продолжать до приобретения твердых навыков в их решении.

3.3. Методические рекомендации по самопроверке знаний

После изучения определенной темы по записям в конспекте и учебнику, а также решения достаточного количества соответствующих задач на практических занятиях и самостоятельно студенту рекомендуется, провести самопроверку усвоенных знаний, ответив на контрольные вопросы по изученной теме.

В случае необходимости нужно еще раз внимательно разобраться в материале.

Иногда недостаточность усвоения того или иного вопроса выясняется только при изучении дальнейшего материала. В этом случае надо вернуться назад и повторить плохо усвоенный материал. Важный критерий усвоения теоретического материала - умение решать задачи или пройти тестирование по пройденному материалу. Однако следует помнить, что правильное решение задачи может получиться в результате применения механически заученных формул без понимания сущности теоретических положений.

3.4. Методические указания по подготовке к экзамену

Изучение многих общепрофессиональных и специальных дисциплин завершается экзаменом. Подготовка к экзамену способствует закреплению, углублению и обобщению знаний, получаемых, в процессе обучения, а также применению их к решению практических задач. Готовясь к экзамену, студент ликвидирует имеющиеся пробелы в знаниях, углубляет, систематизирует и упорядочивает свои знания. На экзамене студент демонстрирует то, что он приобрел в процессе обучения по конкретной учебной дисциплине.

Экзаменационная сессия - это серия экзаменов, установленных учебным планом. Между экзаменами интервал 3-4 дня. Не следует думать, что 3-4 дня достаточно для успешной подготовки к экзаменам.

В эти 3-4 дня нужно систематизировать уже имеющиеся знания. На консультации перед экзаменом студентов познакомят с основными требованиями, ответят на возникшие у них вопросы. Поэтому посещение консультаций обязательно.

Требования к организации подготовки к экзаменам те же, что и при занятиях в течение семестра, но соблюдаться они должны более строго. Во-первых, очень важно соблюдение режима дня; сон не менее 8 часов в сутки, занятия заканчиваются не позднее, чем за 2-3 часа до сна. Оптимальное время занятий, особенно по математике - утренние и дневные часы. В перерывах между занятиями рекомендуются прогулки на свежем воздухе, неутомительные занятия спортом. Во-вторых, наличие хороших собственных конспектов лекций. Даже в том случае, если была пропущена какая-либо лекция, необходимо во время ее восстановить (переписать ее на кафедре), обдумать, снять возникшие вопросы для того, чтобы запоминание материала было осознанным. В-третьих, при подготовке к экзаменам у студента должен быть хороший учебник или конспект литературы, прочитанной по указанию преподавателя в течение семестра. Здесь можно эффективно использовать листы опорных сигналов.

Вначале следует просмотреть весь материал по сдаваемой дисциплине, отметить для себя трудные вопросы. Обязательно в них разобраться. В заключение еще раз целесообразно повторить основные положения, используя при этом листы опорных сигналов.

Систематическая подготовка к занятиям в течение семестра позволит использовать время экзаменационной сессии для систематизации знаний.

Правила подготовки к экзамену:

• Лучше сразу сориентироваться во всем материале и обязательно расположить весь материал согласно экзаменационным вопросам (или вопросам, обсуждаемым на семинарах), эта работа может занять много времени, но все остальное – это уже технические детали (главное – это ориентировка в материале!).

• Сама подготовка связана не только с «запоминанием». Подготовка также предполагает и переосмысление материала, и даже рассмотрение альтернативных идей.

• Готовить «шпаргалки» полезно, но пользоваться ими рискованно. Главный смысл подготовки «шпаргалок» – это систематизация и оптимизация знаний по данному предмету, что само по себе прекрасно – это очень сложная и важная для студента работа, более сложная и важная, чем простое поглощение массы учебной информации. Если студент самостоятельно подготовил такие «шпаргалки», то, скорее всего, он и экзамены сдавать будет более уверенно, так как у него уже сформирована общая ориентировка в сложном материале.

• Как это ни парадоксально, но использование «шпаргалок» часто позволяет отвечающему студенту лучше демонстрировать свои познания (точнее – ориентировку в знаниях, что намного важнее знания «запомненного» и «тут же забытого» после сдачи экзамена).

• Сначала студент должен продемонстрировать, что он «усвоил» все, что требуется по программе обучения (или по программе данного преподавателя), и лишь после этого он вправе высказать иные, желательно аргументированные точки зрения.

Критерии оценивания компетенций

Содержание и объем материала, подлежащего проверке, определяется программой дисциплины. При проверке усвоения этого материала производится выявление полноты, прочности усвоения студентами теории и умения применять ее на практике в знакомых и незнакомых ситуациях.

При оценке в первую очередь учитывается показанные студентами знания и умения (их полноту, глубину, прочность, использование). Оценка также зависит от наличия и характера погрешностей. Среди погрешностей выделяют ошибки и недочеты. Погрешность считается ошибкой, если она свидетельствует о том, что студент не овладел основными знаниями, умениями, указанными в программе.

К недочетам относятся погрешности, свидетельствующие о недостаточно полном или недостаточно прочном усвоении основных знаний и умений или о отсутствии знаний не считающихся основными в соответствии с программой. Недочетами также являются: погрешности, которые не привели к искажению смысла полученного студентом задания или способа его выполнения, неаккуратная запись, небрежное выполнение чертежа, графика.

Ответ на теоретический вопрос считается безупречным, если по своему содержанию полностью соответствует вопросу, содержит все необходимые теоретические факты и обоснованные выводы, а устное изложение и письменная запись ответа математически грамотна и отличается последовательностью и аккуратностью.

Решение задачи считается безупречным, если правильно выбран способ решения, само решение сопровождается необходимыми объяснениями, верно выполнены нужные вычисления и преобразования, получен верный ответ, последовательно и аккуратно записано решение.

Оценка на экзамене выставляется в соответствии с соответствующим критерием.
Оценка **«отлично»** выставляется студенту, если студент

- имеет глубокие и всесторонние знания фундаментальных и специальных разделов аналитической геометрии, позволяющие эффективно решать разнообразные профессиональные задачи с использованием соответствующего математического аппарата;
- умеет применять фундаментальные методы аналитической геометрии для решения задач в профессиональной сфере, в том числе в области компьютерных наук; умеет использовать на высоком уровне методы анализа и синтеза для решения творческих логических задач прикладного значения;
- уверенно владеет проблемно-задачной формой представления логических знаний; навыками решения творческих и нестандартных задач аналитической геометрии; навыками использования на высоком уровне современных образовательных и информационных технологий в области изучения аналитической геометрии.

Оценка **«хорошо»** выставляется студенту, если студент

- показывает наличие твердых знаний фундаментальных разделов аналитической геометрии при недостаточно уверенном владении некоторыми понятиями и категориями;
- умеет решать базовые задачи аналитической геометрии, но при этом допускает нарушение последовательности действий и недостаточно полно использует понятия, свойства и определения;
- владеет проблемно-задачной формой представления базовых знаний аналитической геометрии и навыками решения основных типов задач; на достаточном уровне владеет навыками использования современных образовательных и информационных технологий в области изучения аналитической геометрии.

Оценка **«удовлетворительно»** выставляется студенту, если студент

- в основном предмет знает, однако знания имеют разрозненный, поверхностный и не систематизированный характер;
- имеет основные умения, однако допускает отдельные ошибки и погрешности при решении практических задач и применении соответствующих теоретических положений на практике;
- в основном владеет проблемно-задачной формой представления базовых знаний аналитической геометрии и навыками решения соответствующих базовых задач, но при этом допускаются фрагментарность и непоследовательность, ошибки и недочеты, недостаточно используются современные образовательные и информационные технологии в процессе изучения дисциплины.

Оценка **«неудовлетворительно»** выставляется студенту, если студент

- не знает базовые понятия, определения и теоремы аналитической геометрии;
- не умеет решать базовые задачи аналитической геометрии;
- не владеет проблемно-задачной формой представления базовых знаний аналитической геометрии и навыками решения соответствующих базовых задач.

Описание шкалы оценивания

Максимально возможный балл за весь текущий контроль устанавливается равным **55.** Текущее контрольное мероприятие считается сданным, если студент получил за него не менее 60% от установленного для этого контроля максимального балла. Рейтинговый балл, выставляемый студенту за текущее контрольное мероприятие, сданное студентом в установленные графиком контрольных мероприятий сроки, определяется следующим образом:

Уровень выполнения контрольного задания	Рейтинговый балл (в % от максимального балла за контрольное

	задание)
Отличный	**88-100**
Хороший	**72-87**
Удовлетворительный	**53-71**
Неудовлетворительный	**<53**

Примерные вопросы для подготовки к экзамену по дисциплине «Компьютерная алгебра на Python»

Запись арифметических выражений. Программирование алгоритмов линейной структуры

1. Что такое алгоритм?
2. Какова блок-схема полной (неполной) формы команды ветвления?
3. Какова блок-схема цикла с предусловием?
4. Какова блок-схема цикла с постусловием?
5. Какова блок-схема цикла с параметром?
6. Что такое программа?
7. Что такое транслятор?
8. Что такое компилятор?
9. Что такое интерпретатор?
10. Что такое оператор и операнд?
11. В каких случаях при записи математических выражений на языке Python используются круглые скобки?
12. Как используются стандартные математические функции в Python?
13. Что такое программа на Python?
14. Какая программа является линейной?
15. Какова структура программы на Python?
16. Что такое идентификатор?
17. По каким правилам создаются идентификаторы в Python?
18. Что такое тип величины?
19. Какие типы величин используются в языке Python?
20. Какие типы величин относятся к простым типам?
21. В каком диапазоне могут изменяться переменные типа **int** и **float**?
22. Как описываются переменные в Python? (Привести пример.)
23. Как выполняется команда присваивания?
24. Какая инструкция предназначена для вывода на экран сообщений?
25. Какая инструкция предназначена для ввода данных?

Программирование алгоритмов разветвляющейся структуры

1. Что такое составной оператор?
2. Какова полная (неполная) форма команды ветвления (блок-схема)?
3. Каков алгоритм выполнения команды ветвления?
4. Каков алгоритм выполнения команды множественного ветвления (выбора)?
5. Какие операторы сравнения используются в Python?
6. Что называется простым условием? Приведите примеры.
7. Что такое составное условие? Приведите примеры.
8. Какие логические операторы допускаются при составлении сложных условий?
9. Каков общий вид (формат) инструкции «Ветвление»?
10. Каков алгоритм выполнения условной (тернарной) операции (?:)? Приведите пример.
11. Каков общий вид (формат) инструкции «Выбор»?

12. Может ли оператор ветвления содержать внутри себя другие ветвления?

Программирование алгоритмов циклической структуры

1. Какие базовые алгоритмические конструкции можно реализовать средствами языка Python?
2. Что называется циклом?
3. Что такое тело цикла?
4. Что такое параметр цикла?
5. Что такое итерация?
6. Что такое зацикливание?
7. Какие типы циклов существуют?
8. Какие инструкции используются для реализации циклов на Python?
9. Какая инструкция используется для реализации цикла с параметром?
10. Можно ли использовать в теле цикла переменную, являющуюся параметром цикла?
11. Какая инструкция используется для реализации цикла с предусловием?
12. От чего зависит количество повторений тела цикла с предусловием?
13. Какие из конструкций повторения могут привести к зацикливанию?
14. В каком случае в теле цикла используются операторные скобки?

Использование функций для решения прикладных задач. Программирование рекурсивных алгоритмов

1. Что такое подпрограмма?
2. Как подпрограмму можно реализовать в Python?
3. Какие параметры называются формальными?
4. Какие параметры называются фактическими?
5. Что такое переменная?
6. Что такое область действия идентификатора?
7. Что такое локальная переменная? Что такое глобальная переменная?
8. Какова область действия локальных идентификаторов?
9. Какова область действия глобальных идентификаторов?
10. Что такое функция?
11. Перечислите составные части описания функции.
12. Как описывается функция пользователя в программе?
13. Что такое возвращаемое значение функции?
14. Как осуществляется вызов функций из основной программы?
15. Какой процесс называется итеративным?
16. Каким образом реализуются итеративные процессы?
17. Какой алгоритм называется рекурсивным?
18. Какая функция называется рекурсивной?
19. Что должно обязательно присутствовать в теле рекурсивно описанной функции?
20. Перечислите различия между итерацией и рекурсией.
21. Что произойдет, если рекурсивный алгоритм будет вызывать сам себя «бесконечное» число раз?
22. Как предотвратить бесконечное выполнение рекурсивного алгоритма?
23. Верно ли, что решение задачи, реализуемое рекурсивным алгоритмом, можно выразить, используя итерацию?

Строки. Списки. Вложенные списки, матрицы

1. Дайте определение стоки.
2. Дайте определение массива.

3. Каким может быть тип элементов строки?
4. Какова структура строки?
5. Правила использования строковых констант
6. Как осуществляется доступ к элементам строки?
7. Как осуществляется ввод строк?
8. Вывод строк
9. Дайте определение Списка.
10. Какие типы данных не допустимы для компонентов Списка?
11. Правила использования Списка.
12. Как осуществляется доступ к элементам Списка?
13. Как осуществляется ввод Списка?
14. Размещение Списка в памяти ЭВМ.
15. Вывод Списка.
16. Что такое список?
17. Дайте определение массива.
18. Какие типы данных не допустимы для компонентов массива?
19. Может ли левая граница индексов массива быть отрицательной?
20. Какой массив называется двумерным?
21. Как осуществляется доступ к элементам двумерного массива?
22. Как осуществляется ввод двумерного массива?
23. Какие способы ввода двумерного массива вы знаете?
24. Размещение массива в памяти ЭВМ.
25. Вывод двумерного массива.

Множества и строки

1. Дайте определение множества.
2. Как задать множество в программе?
3. Что называется базовым типом для множества?
4. Какие типы данных могут использоваться в качестве базового?
5. Что называется мощностью множества?
6. Что такое пустое множество? Как обозначается пустое множество?
7. Как осуществляется доступ к элементам множества?
8. Какие операции допустимы над множествами?
9. Какие способы помещения элементов во множество вы знаете?
10. Как осуществляется вывод элементов множества?

Использование Множества для решения математических задач

1. Может ли множество содержать несколько одинаковых элементов?
2. Может ли множество содержать элементы разных типов?
3. Какие методы определены для обработки множеств?

Модуль NumPy

1. Основные методы
2. Особенности работы

Работа с графикой

1. Основные методы
2. Особенности работы с графикой

УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ СРС

Перечень основной литературы:

1. Титов А.Н. Решение задач линейной алгебры и прикладной математики в Python. Работа с библиотекой SciPy: учебно-методическое пособие / Титов А.Н., Тазиева Р.Ф. - Казань: Издательство КНИТУ, 2023. - 124 с. - ISBN 978-5-7882-3319-2. - Текст: электронный // Цифровой образовательный ресурс IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/136187.html

2. Титов А. Н. Символьные вычисления в Python. Основы работы с библиотекой SymPy: учебно-методическое пособие / А. Н. Титов, Р. Ф. Тазиева. - Казань: Издательство КНИТУ, 2023. - 100 с. - ISBN 978-5-7882-3321-5. - Текст: электронный // Цифровой образовательный ресурс IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/136191.html

Перечень дополнительной литературы:

1. Борзунов С. В. Алгебра и геометрия с примерами на Python Электронный ресурс / Борзунов С. В., Кургалин С. Д.: учебное пособие для вузов. - Санкт-Петербург: Лань, 2020. - 444 с. - ISBN 978-5-8114-5489-1

2. Дроботун Н.В. Алгоритмизация и программирование. Язык Python: учебное пособие / Дроботун Н.В., Рудков Е.О., Баев Н.А. - Санкт-Петербург: Санкт-Петербургский государственный университет промышленных технологий и дизайна, 2020. - 119 с. - ISBN 978-5-7937-1829-5. - Текст: электронный // Цифровой образовательный ресурс IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/102400.html

3. Сузи Р. А. Язык программирования Python: учебное пособие / Р. А. Сузи. - 4-е изд. - Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2024. - 350 с. - ISBN 978-5-4497-3351-1. - Текст: электронный // Цифровой образовательный ресурс IPR SMART: [сайт]. - URL: https://www.iprbookshop.ru/142310.html

4. Панкратьев Е. В. Введение в компьютерную алгебру Электронный ресурс / Е. В. Панкратьев. - Введение в компьютерную алгебру. - Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. - 324 с. - Книга находится в премиум-версии ЭБС IPR BOOKS. - ISBN 978-5-9556-0099-4

Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины

- https://github.com/topics/python-math-library. На сайте представлены различные репозитории, связанные с математическими библиотеками на Python.
- https://dzen.ru/suite/dbad6997-b99a-42c0-9935-6c3e089535f5. На ресурсе есть подборка материалов по численной математике и алгоритмам на Python.
- https://pythonchik.ru/matematika. На сайте представлены уроки с примерами по математике на Python, включая округление, возведение в степень, базовые математические операции и модуль Math.
- https://lib.dm-centre.ru/lib/document/gpntb/ESVODT/cf829e2c060e920506957c2c096bc695 На сайте можно найти электронный учебник «Математика на Python» авторов Криволапова С. Я. и Хрипуновой М. Б.