Министерство образования и науки Российской Федерации
ФГБОУ ВПО "Ивановский государственный университет"
Факультет математики и компьютерных наук
Кафедра алгебры и математической логики

АЛГЕБРАИЧЕСКИЕ ВЫЧИСЛЕНИЯ
В СИСТЕМЕ SAGE

**Методические указания по дисциплинам
"Фундаментальная алгебра" и "Компьютерная алгебра"
для студентов 2 курса
факультета математики и компьютерных наук
(квалификация "Бакалавр")**

Иваново
Издательство "Ивановский государственный университет"
2014

Составитель: кандидат физико-математических наук, профессор
кафедры алгебры и математической логики **Н. И. Яцкин**
Под редакцией заведующего кафедрой прикладной математики
и компьютерных наук **Е. В. Соколова**

В пособии демонстрируются возможности применения одной из современных систем компьютерной алгебры, **SAGE**, при выполнении типовых вычислений с такими объектами абстрактной алгебры как группы, кольца, поля. Издание допускает использование ознакомительного характера в процессе освоения студентами соответствующего теоретического материала в курсе **"Фундаментальная алгебра"**, а также, в следующем семестре, – активное практическое использование при изучении алгебраических алгоритмов (и их программных реализаций) в курсе **"Компьютерная алгебра"**.

Печатается по решению методической комиссии
факультета математики и компьютерных наук
Ивановского государственного университета

Рецензент
кандидат физико-математических наук ***С. И. Хашин*** *(ИвГУ)*

Содержание

Введение

Предлагаемое учебное пособие призвано помочь студентам, изучающим учебную дисциплину "Компьютерная алгебра", в практическом освоении важнейших приемов оперирования с абстрактными *алгебраическими объектами* (такими как группы, кольца, поля), в рамках специально сконструированной для этих целей, современной и постоянно совершенствуемой (усилиями широких слоев математического сообщества), *свободно распространяемой* системы компьютерной алгебры **Sage**. На этой базе предполагается дальнейшее изучение основных точно работающих алгебраических алгоритмов.

Предварительное знакомство с идеологией и языком **Sage** можно осуществить с помощью методического пособия [8], подготовленного студентами факультета математики и компьютерных наук ИвГУ А. Е. Куваевым и А. С. Смоляковым, которое, в частности, может служить детальным руководством по *установке* системы и, кроме того, содержит первоначальные сведения об основах **Sage**-программирования. (Именно благодаря этой книжке автору настоящего пособия удалось установить **Sage** на своем компьютере и написать первую **Sage**-процедуру.)

Автором использовалась новейшая (на момент написания пособия) *версия* 6.0 системы; полный комплект документации доступен на главной странице проекта **Sage** (см. [1, 2, 6]). Весьма интересные методики использования **Sage** при изучении стандартных университетских курсов алгебры представлены в учебных пособиях [3 – 5, 7]. Укажем также те учебники по чисто алгебраическим курсам, к которым были "привязаны" позднейшие **Sage**-разработки: [9 - 11].

Вообще, предполагается, что определения и свойства основных алгебраических структур (объектов), упоминаемых в данном пособии, либо уже известны читателям (например, из курса *фундаментальной алгебры*), либо могут быть оперативно подгружены в их оперативную память (из достаточно богатой учебной литературы или из сети; см., в частности, русскоязычные издания [12 - 22]). Некоторые, наиболее важные, аспекты теории кратко напоминаются и поясняются, однако более полную и строгую теоретическую информацию следует искать в указанных источниках.

Автор, не будучи по образованию специалистом в компьютерных науках, но заинтересованный, тем не менее, во внедрении, современных (основанных на использовании систем компьютерной алгебры) *образовательных технологий* в учебный процесс (в том числе - и по базовым математиче-

ским дисциплинам), полагает важным упомянуть, что весьма полезными в ходе подготовки настоящего пособия были обсуждения предварительных вариантов текста с Е. В. Соколовым и другими коллегами, а также - с наиболее "продвинутыми" студентами-компьютерщиками, и, в частности, - с соавторами пособия [8].

Возможно, читателям будет полезно сразу увидеть

Список примеров (с указанием страниц)

1. Элементы линейной алгебры

1.1. Матрицы, векторные пространства и системы линейных уравнений

Предусмотрено несколько способов ввода *матриц*. Представим простейший и самый естественный из них: матрица задается построчно, как список списков. Обратите внимание на то, что в качестве первого аргумента фигурирует имя *поля* (или *кольца*), над которым рассматривается матрица. **Sage** узнает по имени несколько важнейших колец и полей:

ZZ = $\mathbb{Z}$, кольцо целых чисел;

QQ = $\mathbb{Q}$, поле рациональных чисел;

RR = $\mathbb{R}$, поле действительных чисел;

CC = $\mathbb{C}$, поле комплексных чисел;

другие примеры будут описаны далее, по мере необходимости. Сразу акцентируем внимание на том, что в первых двух примерах кольцо и поле являются *точными* (в том смысле, что имеется принципиальная возможность *абсолютно точного* представления для их элементов); в двух последующих примерах поля являются *приближенными* (для записи их элементов приходится прибегать к аппроксимациям, с той или иной степенью точности).

Пример 1.1.1. Решение системы линейных уравнений.

```
sage: A=matrix(QQ,[[1,4,0,-1,0,7,-9],
                   [2,8,-1,3,9,-13,7],
                   [0,0,2,-3,-4,12,-8],
                   [-1,-4,2,4,8,-31,37]]); A;
```

```
[   1    4    0   -1    0    7   -9]
[   2    8   -1    3    9  -13    7]
[   0    0    2   -3   -4   12   -8]
[  -1   -4    2    4    8  -31   37]
```

Определим *размеры* и *ранг* введенной матрицы.

```
sage: m,n=A.nrows(),A.ncols(); m,n; A.rank()
```

```
(4, 7)
3
```

Далее вводится *вектор-столбец*, "высота" которого совпадает с количеством строк ранее введенной матрицы. **Sage** записывает его в строчку, но не в квадратных (как положено для списков), а в круглых скобках.

```
sage: b=vector(QQ,[3,9,1,4]); b
```

```
(3, 9, 1, 4)
```

Рассмотрим теперь *систему линейных уравнений* (*с.л.у.*) $A \cdot \bar{x} = \bar{b}$, содержащую $m = 4$ уравнений с $n = 7$ неизвестными. Для ее исследования образуем расширенную (*аугментированную*) матрицу системы $B = (A|\bar{b})$.

(Разграничивающая черта в матрице появляется благодаря применению опции **subdivide=True**; разумеется, она не обязательна, ни на что не влияет и используется исключительно в обучающих примерах.)

```
sage: B=A.augment(b,subdivide=True); B; B.rank()
```

```
[   1    4    0   -1    0    7   -9|    3]
[   2    8   -1    3    9  -13    7|    9]
[   0    0    2   -3   -4   12   -8|    1]
[  -1   -4    2    4    8  -31   37|    4]

3
```

Вычислив ранг матрицы B и пользуясь теоремой Кронекера-Капелли, мы можем заключить, что равенство $\mathrm{rank}(A) = \mathrm{rank}(B)$ влечет *совместность* рассматриваемой с.л.у. Далее, поскольку количество уравнений меньше количества неизвестных, эта система является *неопределенной* и, следовательно, в ее *общем решении* должны присутствовать *свободные неизвестные*, в количестве, равном $n - \mathrm{rank}(A)$.

Переходим к практическому отысканию общего решения. Приведем матрицу B к *виду Жордана-Гаусса*. В англоязычной математической литературе этот вид принято называть "*приведенной, эшелонированной по строкам формой*" (**rref = Reduced Row-Echelon Form**).

```
sage: B.rref()
```

```
[  1   4   0   0   2   1  -3|   4]
[  0   0   1   0   1  -3   5|   2]
[  0   0   0   1   2  -6   6|   1]
[  0   0   0   0   0   0   0|   0]
```

Визуально наблюдаем *главные* (*ключевые*) столбцы, содержащие (единичные) *ключевые элементы*. По-английски последние называются **pivots** (загляните в англо-русский словарь: **pivot** = штырь, опорный пункт, ключевой игрок и т. п.; математический словарь добавляет: основной, разрешающий, опорный элемент). В **Sage** предусмотрены методы для вывода перечней номеров ключевых и неключевых (*свободных*) столбцов. Количество ключевых столбцов – это ранг $r = \mathrm{rank}(A) = \mathrm{rank}(B)$, количество неключевых столбцов ($s = n - r$) равно количеству свободных неизвестных. (Не забываем, что всё и всюду в **Sage** нумеруется, начиная с нуля.)

```
sage: princ=B.pivots(); princ; r=len(princ); r
```

```
(0, 2, 3)
3
```

```
sage: free=A.nonpivots(); free; s=len(free); s

(1, 4, 5, 6)
4
```

Sage умеет находить одно *частное* решение с.л.у., называемое *опорным*, принцип получения которого таков: свободные неизвестные приравниваются нулю.

```
sage: z=A.solve_right(b); z

(4, 0, 2, 1, 0, 0, 0)
```

Мы применили "правый" метод решения с.л.у.; есть еще "левый" метод, **.solve_left**, разрешающий с.л.у. $\bar{x}^t \cdot A = \bar{0}^t$, в которой неизвестные записываются не в столбец, но в строку.

Если с.л.у. является *однородной* ($\bar{b} = \bar{0}$), то будет выдано "неинтересное" (*тривиальное*, т. е. нулевое) решение. Для получения *фундаментальной матрицы*, содержащей *базисные* решения однородной с.л.у., применяется следующий метод вычисления так называемого *ядра* (*нуль-пространства*, **kernel**) матрицы.

```
sage: K=A.right_kernel(basis='pivot'); K

Vector space of degree 7 and dimension 4 over
 Rational Field User basis matrix:

[ -4   1   0   0   0   0   0]
[ -2   0 -1 -2   1   0   0]
[ -1   0   3   8   0   1   0]
[  3   0 -5 -6   0   0   1]
```

Найден и записан (по строкам) в матрицу K *базис* в (правом) ядре матрицы A. **Sage** описывает это ядро как *подпространство* размерности 4 в *арифметическом линейном пространстве* $\mathbb{Q}^7$ размерности 7 (размерность объемлющего пространства **Sage** называет "степенью", **degree**). Далее вычисленный выше базис нам понадобится представить как бы "россыпью", в виде *последовательности* векторов-столбцов.

```
sage: N=K.basis(); N
[
(-4, 1, 0, 0, 0, 0, 0),
(-2, 0,-1,-2, 1, 0, 0),
(-1, 0, 3, 6, 0, 1, 0),
( 3, 0,-5,-6, 0, 0, 1)
]
```

Как известно, **о.р.н.** (общее решение неоднородной с.л.у.) складывается из **о.р.о.** (общего решения соответствующей однородной с.л.у.) и произвольного **ч.р.н**. (частного решения исходной неоднородной с.л.у.); **ч.р.н. z** уже

найдено; **о.р.о.** представляется в виде *линейной комбинации* базисных частных решений однородной системы (**ч.р.о.**, которые также уже вычислены; см. последовательность **N**); в качестве коэффициентов линейной комбинации должны фигурировать *переменные* (свободные неизвестные, номера которых нам известны; см. список **free**).

По правилам **Sage** каждая переменная должна объявляться, поэтому сейчас мы определим *кольцо* **QQx** многочленов над полем **QQ** от семи переменных (неизвестных в рассматриваемой с.л.у); сами эти переменные будут фигурировать как *порождающие элементы* (*генераторы*, **gens**) построенного кольца.

```
sage: QQx=PolynomialRing(QQ,n,'x'); QQx; QQx.gens()

Multivariate Polynomial Ring in x0,x1,x2,x3,x4,x5,x6
 over Rational Field
(x0, x1, x2, x3, x4, x5, x6)
```

Теперь мы создадим (в цикле **for**) список, элементами которого будут векторы – найденные выше базисные **ч.р.о.**, причем они будут рассматриваться уже не над полем **QQ**, но над (более широким) кольцом многочленов **QQx**, хотя ни один из этих векторов пока не содержит ни одной из переменных. Переменные появятся на следующем шаге, как коэффициенты линейной комбинации.

```
sage: f=[vector(QQx,n,N[i]) for i in range(m)]; f

[(-4,1,0,0,0,0,0),(-2,0,-1,-2,1,0,0),(-1,0,3,6,0,1,0),
 (3,0,-5,-6,0,0,1)]
```

Выберем из числа переменных те, которые являются свободными неизвестными и вычислим сумму произведений свободных неизвестных на соответствующие базисные **ч.р.о.**, плюс опорное **ч.р.н.**

```
sage: y=[QQx.gen(free[i]) for i in range(m)]; y

[x1, x4, x5, x6]

sage: x=sum([y[i]*f[i] for i in range(m)])+z; x

(-4*x1-2*x4-x5+3*x6+4, x1, -x4+3*x5-5*x6+2,
 -2*x4+6*x5-6*x6+1, x4, x5, x6)
```

Наконец, **о.р.н. x** найдено; проверку можно выполнить, вычислив (над кольцом многочленов!) *невязк*у.

```
sage: A*x-b
(0, 0, 0, 0)
```

Разбирая алгоритм решения с.л.у., мы уже сталкивались с необходимостью рассмотрения *линейных пространств* (над различными полями). Так

ядро $(m \times n)$-матрицы A является линейным подпространством, размерность которого равна $n - \mathrm{rank}(A)$, в арифметическом линейном пространстве $\mathbb{Q}^n$. По матрице A определяется еще одно линейное подпространство – *образ* этой матрицы, понимаемый как *линейная оболочка* (**span**) векторов-столбцов матрицы. Это подпространство (в примере оно обозначено **W**) располагается (вообще говоря) уже в другом линейном пространстве $\mathbb{Q}^m$. Размерность образа совпадает с рангом матрицы.

Пример 1.1.2. Базис в подпространстве, порожденном столбцами матрицы. (Матрица **A** - из предыдущего примера).

```
sage: V=QQ^4; V; W=V.span(A.columns()); W

Vector space of dimension 4 over Rational Field
Vector space of degree 4 and dimension 3 over
 Rational Field Basis matrix:

[  1   0   0 -31/7]
[  0   1   0  12/7]
[  0   0   1  13/7]

sage: M=W.basis(); M; d=W.dimension(); d
[
( 1, 0, 0,-31/7),
( 0, 1, 0, 12/7),
( 0, 0, 1, 13/7)
]
3
```

Матрицы фиксированного размера над заданным полем сами образуют векторное пространство над этим полем. Его можно явно задать и после этого выполнять в нем все допустимые в векторных пространствах операции. Ниже показано, например, как вызывается *естественный базис* в пространстве матриц заданного размера.

Пример 1.1.3. Линейное пространство матриц заданного размера.

```
sage: W=MatrixSpace(QQ,2,3); B=W.basis()

Full MatrixSpace of 2 by 3 dense matrices over
 Rational Field
[
[1 0 0] [0 1 0] [0 0 1] [0 0 0] [0 0 0] [0 0 0]
[0 0 0],[0 0 0],[0 0 0],[1 0 0],[0 1 0],[0 0 1]
]
```

1.2. Функции от матриц. Преобразования и канонические формы матриц

В линейной алгебре изучаются разнообразные функции, зависящие от матричного аргумента. Некоторые из этих функций определены для произвольных (прямоугольных) матриц; некоторые – лишь для квадратных, причем не обязательно для всех. Некоторые функции принимают скалярные значения (*ранг*, *определитель*, *след* и т. п.); другие сопоставляют матрице новую матрицу (*транспонированную*, *присоединенную*, *обратную* и т. д.). Могут сопоставляться матрицам и более сложные (составные) объекты: *характеристический* и *минимальный* многочлены, *спектр* (как множество ее *собственных значений*, возможно, - с учетом их *кратностей*, *алгебраических* и *геометрических*).

Многие из названных (и неназванных) функций легко реализуемы в компьютерных алгебраических системах (и, в том числе, в **Sage**); реализация других (над неточными полями) весьма трудоемка.

Пример 1.2.1. Спектральные характеристики матрицы.

Начнем с того, что сгенерируем *случайную* (4×4)-матрицу A с элементами из поля рациональных чисел [синтаксис соответствующей команды предусматривает ограничения на числители (**num = numerator**) и знаменатели (**den = denominator**) матричных элементов] и применим к этой матрице упомянутые выше функции.

```
sage: A=random_matrix(QQ,4,4,num_bound=4,den_bound=5);A

[ -1 -3/2   -1    1]
[  2    3    0   -3]
[  0  4/5   -2    3]
[ -2   -1  3/2  2/3]

sage: A.rank(),A.det(),A.trace()

(4, 314/15, 2/3)

sage: A.transpose(),A.adjoint(),A.inverse()
(
[  -1   2    0   -2]  [-151/10 -689/60  -13/4  -72/5]
[-3/2   3  4/5   -1]  [   -1/3    47/6   37/6     8]
[  -1   0   -2  3/2]  [-236/15 -106/15     -2   4/5]
[   1  -3    3  2/3], [  -52/5   -34/5      4  -8/5],

[ -453/628 -689/1256 -195/1256  -108/157]
[   -5/314   235/628   185/628    60/157]
[ -118/157   -53/157   -15/157     6/157]
[  -78/157   -51/157    30/157   -12/157]
)
```

Отметим, что обратную матрицу можно было бы вычислить иначе: `A^(-1)`. Чтобы найти характеристический многочлен $h_A(t) = \det(tE - A)$, мы предварительно объявляем переменную `t`.

```
sage: t=var('t'); h=A.charpoly(t); h; h.roots()

t^4 - 2/3*t^3 - 19/2*t^2 + 163/15*t + 314/15
[]
```

Попытка найти *собственные значения* (*характеристические корни*) приводит к пустому списку: в поле `QQ` их нет. Есть, однако, возможность расширить (алгебраически замкнуть) поле рациональных чисел.

Всякое поле P можно вложить в некоторое *алгебраически замкнутое* поле. Наименьшее из таких полей определено однозначно (с точностью до изоморфизма); оно называется *алгебраическим замыканием* поля P и обозначается $\bar{P}$.

Из начального курса алгебры известно, что алгебраическим замыканием поля действительных чисел является поле комплексных чисел: $\overline{\mathbb{R}} = \mathbb{C}$. Поле рациональных чисел $\mathbb{Q}$ является подполем в $\mathbb{R}$, и поэтому его алгебраическое замыкание $\mathbb{A} = \overline{\mathbb{Q}}$ (именуемое *полем алгебраических чисел*) является подполем в $\mathbb{C}$. В более подробных курсах алгебры доказывается, что поле $\mathbb{A}$ получается присоединением к $\mathbb{Q}$ всех *корней* всевозможных многочленов с *целочисленными* коэффициентами.

В системе `Sage` для поля $\mathbb{A}$ используется обозначение `QQbar`; обозначение `AA` резервируется для подполя $\mathbb{A} \cap \mathbb{R}$ действительных алгебраических чисел. Переход от одного основного кольца (или поля) к другому осуществляется следующим образом.

```
sage: A1=A.change_ring(QQbar); A1

[  -1 -3/2   -1    1]
[   2    3    0   -3]
[   0  4/5   -2    3]
[  -2   -1  3/2  2/3]
```

Матрица, разумеется, не изменилась, но теперь ее элементы рассматриваются в более широком поле `QQbar` (которое система `Sage` относит к категории *приближенных*).

```
sage: A1.charpoly(t).roots()

[(-2.967556346854603?, 1),(-1.093191009111068?, 1),
(2.363707011316169? - 0.9303855331002023?*I, 1),
(2.363707011316169? + 0.9303855331002023?*I, 1)]
```

Мы получили четыре попарно различных (*однократных*) корня: два действительных и два комплексно сопряженных друг другу (мнимую едини-

цу **Sage** кодирует символом **I**); ответ выдан в виде списка пар; первый элемент пары – это корень, второй – кратность корня.

В случае, когда квадратная матрица имеет полный набор попарно различных собственных значений, говорят, что она имеет *простой спектр*. Это условие является достаточным для *диагонализируемости* матрицы.

(Напомним, что квадратная матрица называется диагонализируемой, если она *подобна* некоторой диагональной матрице; отношение подобия $A \approx B$ определяется формулой $B = T^{-1}AT$, где T - некоторая *обратимая* матрица.)

В примере мы можем сделать вывод о диагонализируемости данной матрицы над полем алгебраических чисел (над полем рациональных чисел она не является диагонализируемой). **Sage**-тест может подтвердить данный вывод.

```
sage: A1.is_diagonalizable(); A.is_diagonalizable()

True
Traceback (click to the left of this block for
 traceback)
...
RuntimeError: an eigenvalue of the matrix is not
 contained in Rational Field
```

Обратите внимание на то, что иногда, вместо естественного **False** (или, в других случаях, - вместо пустого ответа), **Sage** запрограммирован на выдачу *сообщения об ошибке*.

Рассмотрим со всеми подробностями задачу о приведении квадратной матрицы к *жордановой нормальной форме* (*ж.н.ф.*).

Пример 1.2.2. Приведение матрицы к жордановой нормальной форме.

```
sage: A=matrix(QQ,[[-9,-5,-3,0,-8,-3,-7,13],
                   [1,0,2,1,2,-1,2,-3],
                   [-6,-6,-5,1,-9,-4,-5,12],
                   [0,1,0,-1,1,-1,1,0],
                   [-1,-1,-1,0,-3,0,-1,2],
                   [1,1,1,1,1,-3,2,-2],
                   [1,0,1,0,0,0,-1,-2],
                   [-6,-5,-2,1,-8,-4,-5,9]]); A

[-9 -5 -3  0 -8 -3 -7 13]
[ 1  0  2  1  2 -1  2 -3]
[-6 -6 -5  1 -9 -4 -5 12]
[ 0  1  0 -1  1 -1  1  0]
[-1 -1 -1  0 -3  0 -1  2]
[ 1  1  1  1  1 -3  2 -2]
```

```
[ 1  0  1  0  0  0 -1 -2]
[-6 -5 -2  1 -8 -4 -5  9]
```

Вычислим и разложим на неприводимые множители (метод `.factor`) характеристический многочлен.

```
sage: t=var('t'); h=A.charpoly(t); h; h.factor()

t^8+13*t^7+70*t^6+196*t^5+280*t^4+112*t^3-224*t^2
 -320*t-128
(t-1)*(t+2)^7
```

Из разложения определяем собственное значение $t_1 = 1$ кратности $m_1 = 1$ и собственное значение $t_2 = 1$ кратности $m_2 = 7$. Сумма (алгебраических) кратностей собственных значений равна размерности пространства (размеру матрицы): $m_1 + m_2 = 8$. Это свидетельствует о том, что данная матрица приводима к *жордановой нормальной форме* (*ж.н.ф.*) над полем рациональных чисел.

Предусмотрены специализированные методы: `.eigenvalues` - вычисляет собственные значения; `.eigenspaces_right` – описывает соответствующие *собственные подпространства* (как линейные оболочки; указываются базисные *собственные векторы*). Снова существуют две версии методов: левая и правая; наш выбор, привычный по курсу линейной алгебры, - правая версия. Кроме того, следует обратить внимание на уточнение `'all'`: ищутся *все* собственные подпространства.

```
sage: A.eigenvalues();

[1,-2,-2,-2,-2,-2,-2,-2]

sage: A.eigenspaces_right(format='all')

[
(1, Vector space of degree 8 and dimension 1 over
 Rational Field User basis matrix:
 [1 0 1 0 0 0 0 1]),
(-2, Vector space of degree 8 and dimension 3 over
 Rational Field User basis matrix:
 [ 1  0  0  1  0  0 -1  0]
 [ 0  1  0  1 -1  1  0  0]
 [ 0  0  1  0  0  1  1  1])
]
```

Теперь нам видны *геометрические кратности* собственных значений (т. е. размерности собственных подпространств): $n_1 = 1$, $n_2 = 3$; их сумма $n_1 + n_2 < 8$. Вспоминая критерий диагонализируемости матрицы (сумма геометрических кратностей собственных значения должна равняться размер-

ности пространства), заключаем, что данная матрица не является диагонализируемой. (Можете спросить **Sage**, он скажет то же самое.)

Найдем ж.н.ф. J матрицы A, вместе с матрицей перехода T, реализующей подобие $J = T^{-1}AT$.

```
sage: J,T=A.jordan_form(transformation=True); J,T
```

```
(
[ 1| 0  0  0  0| 0  0| 0]
[--+------------+-----+--]
[ 0|-2  1  0  0| 0  0| 0]
[ 0| 0 -2  1  0| 0  0| 0]   [1  1  2 0 0 -1/3    1 0]
[ 0| 0  0 -2  1| 0  0| 0]   [0  0  1 1 0  1/3    0 0]
[ 0| 0  0  0 -2| 0  0| 0]   [1  0  0 1 0    0    0 1]
[--+------------+-----+--]   [0  1  1 1 1    0    0 0]
[ 0| 0  0  0  0|-2  1| 0]   [0  0  0 0 0 -1/3    0 0]
[ 0| 0  0  0  0| 0 -2| 0]   [0  0  0 1 0  1/3 -2/3 1]
[--+------------+-----+--]   [0 -1 -1 0 0  1/3 -2/3 1]
[ 0| 0  0  0  0| 0  0|-2], [1  0  1 1 0    0    0 1]
)
```

Проверка.

```
sage: (J-T^(-1)*A*T).is_zero()
```

```
True
```

Sage старательно подразделяет матрицу **J** на блоки; *жордановы ящики* очень хорошо видны. Заметим, однако, что (даже и при не очень больших размерах матриц) стремление **Sage** сделать все столбцы одинаковыми по ширине приводит к хаотическому и трудно читаемому разбросу элементов по странице. В таких случаях лучше не выводить матрицы простым указанием их имен, но поступить чуть хитрее: в двойном цикле перебрать все элементы. (Полученный список списков при необходимости легко преобразуется обратно в матрицу.)

```
sage: [[T[i,j] for j in range(T.ncols())]
                for i in range(T.nrows())]
```

```
[[1,1,2,0,0,-1/3,1,0],[0,0,1,1,0,1/3,0,0],
 [1,0,0,1,0,0,0,1],[0,1,1,1,1,0,0,0],
 [0,0,0,0,0,-1/3,0,0],[0,0,0,1,0,1/3,-2/3,1],
 [0,-1,-1,0,0,1/3,-2/3,1],[1,0,1,1,0,0,0,1]]
```

1.3. Матрицы над евклидовыми кольцами. Каноническая форма Смита

Для матриц над полем основным рабочим инструментом служит алгоритм Жордана-Гаусса приведения к виду Жордана-Гаусса (**rref**, см. п. 1.1), основанный на применении к строкам матрицы так называемых *элементарных преобразований* трех типов: (1) перестановка строк; (2) прибавление к некоторой строке другой строки, домноженной на некоторый скаляр; (3) умножение некоторой строки на ненулевой скаляр. Если разрешить аналогичные преобразования над столбцами, то любую прямоугольную матрицу можно привести к так называемому *скелетному* виду: в начале главной диагонали располагаются единицы, в количестве, равном рангу исходной матрицы, остальные элементы равны нулю. Кстати, вы можете легко, пользуясь справочником или "хелпами", получить информацию о **Sage**-методах, реализующих элементарные преобразования всех типов.

При работе с матрицами, элементы которых принадлежат не полю, но, скажем, *коммутативному кольцу* K, возникает одно, но очень важное отличие: в преобразованиях типа (3), над строками или столбцами, скаляр обязан быть *обратимым* элементом кольца K (а в кольце обратимыми могут оказаться не все, а лишь некоторые из ненулевых элементов; они образуют группу, которая обозначается K^* и называется *мультипликативной группой* кольца K). Наиболее простой вид соответствующая теория приобретает в случае так называемых *евклидовых* колец. Евклидовыми называются *целостные* кольца, в которых может быть задан некоторый *алгоритм деления с остатком*, причем для ненулевых элементов кольца должна быть определена некоторая "следящая функция" (обычно именуемая *евклидовой нормой*) такая, что остаток либо равен нулю, либо его норма строго меньше нормы делителя.

Простейшим евклидовым кольцом является кольцо $\mathbb{Z}$ целых чисел (евклидова норма совпадает с функцией "абсолютная величина": $x \mapsto |x|$); в этом кольце всего два обратимых элемента: $\mathbb{Z}^* = \{1, -1\}$.

К классу евклидовых относятся также все кольца $P[x]$ многочленов, *от одной переменной* (**univariate**), с коэффициентами из любого *поля* P; нормой служит функция "степень": $f(x) \mapsto \deg(f(x))$; обратимыми являются многочлены нулевой степени, т. е. ненулевые константы; мультипликативная группа кольца многочленов: $(P[x])^* = P^* = P \setminus \{0\}$.

Кольца многочленов от нескольких переменных (**multivariate**) уже не являются евклидовыми. Ниже будут приведены еще некоторые примеры евклидовых колец, возникающие в теории алгебраических чисел.

Над евклидовым кольцом K любая $(m \times n)$-матрица A с помощью элементарных преобразований (трех типов, над строками и столбцами) может быть приведена к так называемой *канонической форме Смита* S: по главной диагонали, в количестве, равном рангу $r = \mathrm{rank}(A)$ данной матрицы, распо-

лагаются ненулевые элементы $\mu_1, \mu_2, \ldots, \mu_r$ кольца K, причем каждый следующий элемент делится на предыдущий ($\mu_1 | \mu_2 | \ldots | \mu_r$); остальные элементы матрицы S равны нулю. Форма Смита определена по матрице A однозначно, с точностью до *ассоциированности* диагональных элементов (ассоциированными называются элементы, отличающиеся обратимым множителем). Элементы $\mu_1, \mu_2, \ldots, \mu_r$ называются *инвариантными множителями* для A.

Форма Смита (для матриц над евклидовым кольцом) является обобщение скелетного вида (для матриц над полем); в случае поля все инвариантные множители равны единице.

Матрицы A и S связаны соотношением $S = UAV$, где $(m \times m)$-матрица U и $(n \times n)$-матрица V являются *унимодулярными*, т. е. их определители являются обратимыми элементами (принадлежат K^*); в матрице U "накапливаются" все элементарные преобразования над строками, а в матрице V - преобразования над столбцами.

Над кольцом $K = \mathbb{Z}$ можно добиться полной однозначности в определении формы Смита, если потребовать, чтобы инвариантные множители выбирались *положительными*. Аналогично, для матриц над кольцом многочленов $K = P[x]$ форма Смита будет определена однозначно, если выбирать инвариантные множители (в данном случае, это - многочлены) *нормализованными,* т. е. с единичным старшим коэффициентом (в "английском математическом" есть удобный короткий термин: **'monic polynomials'**).

Пример 1.3.1. Приведение целочисленной матрицы к канонической форме Смита.

Вычисления ведутся в (евклидовом) кольце целых чисел.

```
sage: A=matrix(ZZ,[[0,-4,5,-3],
                   [3,6,-12,6],
                   [-9,-6,18,24]]);A

[  0  -4   5  -3]
[  3   6 -12   6]
[ -9  -6  18  24]

sage: S,U,V=A.smith_form(); S,U,V

(
                                [-4 -5 -6 16]
[ 1  0  0  0]  [ 1  0  0]  [-4 -5 -5 13]
[ 0  3  0  0]  [ 0  1  0]  [-3 -4 -4 11]
[ 0  0 12  0], [-6 -1  1], [ 0  0  0  1]
)
```

Форма Смита найдена; инвариантные множители матрицы A: 1, 3, 12. (Кстати, работа на эту тему опубликована английским математиком Генри Смитом в далеком 1861 г.; вот какие древние вещи мы изучаем.)

Далее следует проверка (напомним, что определители матриц перехода U и V должны принадлежать $\mathbb{Z}^*$, т. е. равняться ± 1).

```
sage: U*A*V==S,U.det(),V.det()
```

```
(True, 1, -1)
```

Пример 1.3.2. Приведение полиномиальной матрицы к канонической форме Смита.

Вычисления ведутся в (евклидовом) кольце многочленов над полем рациональных чисел.

```
sage: x=var('x'); QQx.<x>=QQ[]; QQx
```

```
Univariate Polynomial Ring in x over Rational Field
```

```
A=matrix(QQx,[[x^2-2*x+1, -x^2+1, -2*x^2+2*x,
    -x^3+x^2+x-1], [x^3-3*x^2+x+1, -x^4+x^3+2*x^2-x-1,
    -3*x^4+4*x^3+7*x^2-6*x-2, x^2-1], [x^2-1,
    x^4-x^3-4*x^2+x+3, 3*x^4-3*x^3-11*x^2+3*x+8,
    -x^4+2*x^2-1]])
```

Здесь и далее мы применяем искусственный прием "компактификации" визуального представления матриц, описанный в конце примера 1.2.2.

```
m,n=A.nrows(),A.ncols(); m,n
(3,4)
```

```
[[A[i,j] for j in range(n)] for i in range(m)]
```

```
[[x^2-2*x+1, -x^2+1, -2*x^2+2*x, -x^3+x^2+x-1],
 [x^3-3*x^2+x+1, -x^4+x^3+2*x^2-x-1,
  -3*x^4+4*x^3+7*x^2-6*x-2, x^2-1],
 [x^2-1, x^4-x^3-4*x^2+x+3, 3*x^4-3*x^3-11*x^2+3*x+8,
  -x^4+2*x^2-1]]
```

```
S,U,V=A.smith_form()
```

```
SS=[[S[i,j] for j in range(n)] for i in range(m)]; SS
```

```
[[x-1, 0, 0, 0], [0, x^2-1, 0, 0],
 [0, 0, -4*x^3+8*x^2+4*x-8, 0]]
```

Выделим из матрицы `S` список `invar` инвариантных множителей (многочленов), попутно разлагая их на неприводимые множители.

```
r=S.rank()
```

```
invar=[S[i,i].factor() for i in range(r)]; invar
```

```
[x-1, (x-1)*(x+1), (-4)*(x-2)*(x-1)*(x+1)]
```

Наблюдаем, прежде всего, что требование делимости для инвариантных множителей выполняется. В то же время замечаем, что третий из них не является *нормализованным* (старший коэффициент равен -4). По-видимому, алгоритм приведения к форме Смита над кольцом многочленов, встроенный в **Sage**, вообще не стремится к обеспечению нормализованности инвариантных множителей, хотя ее легко достичь, например, с помощью домножения строк на (возможно дробные) константы. Теперь посмотрим на матрицы **U** и **V** и произведем проверку.

```
UU=[[U[i,j] for j in range(m)] for i in range(m))]; UU
```

```
[[1/2*x-1/2, -1/2, 0],
 [1/2*x^3-x-1/2, -1/2*x^2+1/2, -1/2*x^2+1/2*x],
 [x^4-2*x^3-2*x^2+5*x+2, -x^3+2*x^2+x-4,
  -x^3+3*x^2-2*x-2]]
```

```
VV=[[V[i,j] for j in range(n)] for i in range(n)]; VV
```

```
[[1, -1/2*x^3+1/2*x^2+x, -x^3+x^2+2*x+1, x+3/2],
 [0, 1, -1, -1/2], [0, 0, 1, 1/2], [0, 0, 0, 1]]
```

```
S==U*A*V,U.det(),V.det()
```

```
(True, 1, 1)
```

2. Элементы общей (абстрактной) алгебры

2.1. Поля, кольца, идеалы в кольцах, фактор-кольца

С важнейшими примерами числовых полей мы встречались в первой части пособия: **QQ, RR, CC, QQbar, AA** (в обозначениях **Sage**); с конечными полями познакомимся в п. 2.3. В первых двух пунктах второй части наши интересы будут сосредоточены на *ассоциативных коммутативных кольцах с единицей*, а также на том, как посредством их *факторизации* получаются *поля*.

Идеал J в кольце K определяется как подмножество в K, являющееся подгруппой по сложению и устойчивое относительно умножения на элементы из K (в коммутативном случае не важно, с какой стороны производится умножение и не различаются идеалы левые, правые и двусторонние). Кольцо K разбивается в объединение попарно не пересекающихся *классов смежности* по идеалу J, каждый из которых может быть представлен в виде $x + J$, где x - произвольный *представитель* класса. В реальных вычислениях, как правило, приходится фиксировать некоторый *выделенный* представитель в каждом из классов смежности. С помощью представителей корректным образом определяются алгебраические действия над классами смежности: $(x + J) + (y + J) = (x + y) + J$; $(x + J) \cdot (y + J) = (x \cdot y) + J$; тем самым множество K/J всех классов смежности превращается в кольцо (называемое *фактор-кольцом* (**quotient ring**) кольца K по идеалу J).

Особый и важнейший случай: если идеал J является *максимальным* (т. е. не содержится ни в каком строго более широком идеале, отличном от всего кольца K), то фактор-кольцо K/J оказывается *полем*.

Пример 2.1.1. Кольцо целых чисел и кольца классов вычетов.

В кольце целых чисел $\mathbb{Z}$ **= ZZ** всякий идеал является *главным* (может быть порожден одним элементом): $J = (n) = n\mathbb{Z}$, где n - неотрицательное целое число; этот идеал является *максимальным* тогда и только тогда, когда число n является *простым* натуральным; соответствующее фактор-кольцо обозначается $\mathbb{Z}_n = \mathbb{Z}/n\mathbb{Z}$; в случае простого n оно является *полем*. **Sage** про всё это знает; результат факторизации кодируется следующим образом: $\mathbb{Z}_n$ **= Intgers(n)**.

```
sage: K=ZZ; K; J=K.ideal(8); J;
```

```
Integer Ring
Principal ideal (8) of Integer Ring
```

Определены кольцо K и главный идеал J, порожденный элементом 8. Ниже, с помощью метода **.quotient_ring** вычисляется фактор-кольцо $L = K/J$, далее выводится список элементов L и затем кольцо L тестируется на предмет того, является ли оно полем.

```
sage: L=K.quotient_ring(J); L; L.list(); L.is_field()

Ring of integers modulo 8
[0, 1, 2, 3, 4, 5, 6, 7]
False
```

Обратите внимание на то, что в качестве элементов фактор-кольца **L** фигурируют *канонические представители* классов смежности, а именно – наименьшие неотрицательные *вычеты* по модулю 8.

Доступны таблицы сложения и умножения для конечных колец. (Опция в скобках назначает в качестве *имен* элементов сами эти элементы; по умолчанию мы получили бы восемь первых букв английского алфавита.)

```
sage: mt=L.multiplication_table(names='elements'); mt

*  0 1 2 3 4 5 6 7
 +----------------
0| 0 0 0 0 0 0 0 0
1| 0 1 2 3 4 5 6 7
2| 0 2 4 6 0 2 4 6
3| 0 3 6 1 4 7 2 5
4| 0 4 0 4 0 4 0 4
5| 0 5 2 7 4 1 6 3
6| 0 6 4 2 0 6 4 2
7| 0 7 6 5 4 3 2 1
```

Из таблицы умножения усматриваем, что кольцо **L** не только не является полем, но не является (в отличие от **K**) *целостным* кольцом (например, $4 \cdot 6 = 0$). **Sage** (реагируя на применение теста **.is_integral_domain**) подтверждает это наблюдение:

```
sage: K.is_integral_domain(), L.is_integral_domain()

(True, False)
```

Мы уже говорили о целостных кольцах в п. 1.3; упоминалось там и понятие *мультипликативной группы* кольца, которая, по определению, состоит из *обратимых элементов* кольца, еще они называются *единицами* кольца. И здесь мы (используя тестирующий элементы на обратимость метод **.is_unit**) вычислим список элементов группы $\mathbb{Z}_8^*$ обратимых вычетов по модулю 8

```
sage: U=[k for k in L if k.is_unit()]; U

[1, 3, 5, 7]
```

Целостные кольца в некоторой степени "похожи" на кольцо целых чисел (недоучившиеся алгебре пользователи могут даже путать эти термины). Однако более интересен более узкий класс колец – *факториальные* кольца.

Вот вам и новая возможность для путаницы! Факториальные кольца - совсем не то же самое, что фактор-кольца.

Англоязычные компьютерщики не спутают: *фактор-кольцо* = **quotient ring**, - это некоторое новое кольцо, которое строится по данному кольцу и идеалу в нем; *факториальное кольцо* = **unique factorization domain** (**UFD**) , - это кольцо, в котором может быть развита содержательная *теория делимости*; дословный перевод: *кольцо с однозначным разложением на множители*.

Факториальные кольца "очень похожи" на кольцо целых чисел; в них, как и в $\mathbb{Z}$, имеется "достаточно много" *неразложимых* (*простых*) элементов (каждый из которых не представим в виде произведения двух необратимых элементов), а всякий ненулевой и необратимый элемент - *разлагается в произведение неразложимых*, причем такое разложение определено *однозначно*, с точностью до порядка сомножителей и их *ассоциированности* (напомним: ассоциированными называются элементы, отличающиеся обратимым множителем).

Класс факториальных колец является, однако, трудно обозримым; на практике используются некоторые достаточные условия факториальности. Например, доказано, что факториальными являются *кольца главных идеалов* (*к.г.и.* = **PID, principal ideal domain**), в которых (тоже, - как и в $\mathbb{Z}$) всякий идеал является *главным* (см. определение в начале данного пункта). **Sage** - "в курсе", какие из знакомых ему колец являются к.г.и.; пользователи также должны понимать, что таковыми являются, в частности, все *евклидовы кольца* (*е.к.* = **Euclidian domain**; см. начало п. 1.3).

Следующий пример будет посвящен теме, к сожалению, не доведенной пока разработчиками **Sage** до "рабочего уровня", – вычислениям в кольцах алгебраических чисел.

Пример 2.1.2. Алгебраические расширения поля рациональных чисел.

Можно, присоединив мнимую единицу, расширить поле рациональных чисел $\mathbb{Q}$ = **QQ** до так называемого поля *рациональных гауссовых чисел* (математическое обозначение: $\mathbb{Q}[i]$; на языке **Sage** следует указывать порождающий элемент).

```
sage: GQ.<i>=QQ[I]; GQ; z=3/4+5/3*i; z; z in GQ; z^10

Number Field in I with defining polynomial x^2+1
5/3*I + 3/4
True
-316547951975/859963392*I + 11977223406401/61917364224
```

Кольцо $\mathbb{Z}[i]$ *целых гауссовых чисел* может быть получено как кольцо целых элементов поля $\mathbb{Q}[i]$.

```
sage: GZ=GQ.ring_of_integers(); GZ; GZ.gens(); z in GZ
```

```
Maximal Order in Number Field in I with defining
 polynomial x^2+1
[1, I]
False
```

Над целыми и рациональными гауссовыми числами можно выполнять простейшие алгебраические действия, однако разлагать гауссовы целые числа на простые (неразложимые) множители **Sage** пока не умеет.

Есть над чем работать! Можно надеяться, что энергичным неофитам окажется по силам довести технику **Sage** в этом частном вопросе хотя бы до уровня **Maple**. Почитайте предварительно теорию: в учебниках по алгебре можно найти доказательство *евклидовости* (и, следовательно, *факториальности*) кольца $\mathbb{Z}[i]$ роль *евклидовой нормы* играет квадрат модуля комплексного числа; обратимых элементов в этом кольце четыре: ± 1 и $\pm i$; неразложимые элементы распознаются так: это, во-первых, четыре числа с нормой 2, а именно: $\pm 1 \pm i$, во-вторых, все числа, норма которых равна нечетному простому числу, и, в-третьих, числа, норма которых равна квадрату простого числа, но не любого, а сравнимого с тройкой по модулю 4.

Интересно, что **Sage** все же способен решать задачу разложения на множители, но не "на уровне элементов", а на уровне так называемых *дробных идеалов*. К сожалению, уровень ожидаемой математической подготовки читателей данного пособия не дает нам возможности углубляться в такие "дебри".

Покажем еще *поле рациональных* и *кольцо целых чисел Эйзенштейна*. Поле $\mathbb{Q}[i\sqrt{3}]$, получается присоединением к полю $\mathbb{Q}$ элемента $\sqrt{-3} = i\sqrt{3}$ (являющегося корнем многочлена $x^2 + 3$; по умолчанию **Sage** использует для добавляемого элемента обозначение **a**; поле расширения состоит из всевозможных линейных комбинаций вида **x*1+y*a** с рациональными коэффициентами **x,y**).

```
sage: EQ=QQ[sqrt(-3)]; EQ
```

```
Number Field in a with defining polynomial x^2+3
```

Однако, в какой-то степени неожиданно, оказывается, что подкольцо целых элементов поля $\mathbb{Q}[i\sqrt{3}]$ состоит из линейных комбинаций такого же вида, **x*1+y*a**, однако (вопреки ожиданиям) не обязательно - с целыми **x,y**; эти коэффициенты могут быть *либо оба целыми, либо оба полуцелыми*. **Sage** знает об этом.

```
sage: EZ=EQ.ring_of_integers(); EZ; w,v=EZ.gens(); w,v
```

```
Maximal Order in Number Field in a with defining
 polynomial x^2+3
(1/2*a+1/2, a)
```

Среди порождающих элементов (*генераторов*) кольца оказалось число $\zeta_6 = \frac{1}{2}(1 + a) = \frac{1}{2}(1 + i\sqrt{3})$, которое есть не что иное, как первообразный комплексный корень *шестой* степени из единицы. Почему же такое число, с нецелыми коэффициентами, заслуживает название целого? А по определению! Вспомним, что *алгебраическое число* – это корень многочлена с целыми коэффициентами, а *целое алгебраическое число* – корень *нормализованного* многочлена с целыми коэффициентами. Ниже мы тестируем число $w = \zeta_6$ на принадлежность полю алгебраических чисел, а потом находим для этого числа минимальный аннулирующий многочлен (т. е. многочлен наименьшей степени с целыми или рациональными коэффициентами, корнем которого является w).

```
sage: w=(1+I*sqrt(3))/2; w; w in QQbar; w.minpoly()

1/2*I*sqrt(3)+1/2
True
x^2-x+1
```

Минимальный многочлен имеет целые коэффициенты и является нормализованным. Значит, **w** является целым алгебраическим числом. Повторим испытание для другого числа.

```
sage: w1=(2+I*sqrt(3))/2; w1; w1 in QQbar; w1.minpoly()

1/2*I*sqrt(3)+1
True
x^2-2*x+7/4
```

Выдан нормализованный минимальный многочлен, но – с рациональными коэффициентами. Коэффициенты можно сделать целыми, если домножить многочлен на 4, но тогда он перестанет быть нормализованным. Число **w1** является алгебраическим, но не целым алгебраическим.

Нетрудно заметить, что всякое целое число Эйзенштейна можно представить также целочисленной линейной комбинацией несколько иного вида: $u \cdot 1 + v \cdot \zeta_3$, в которой участвует первообразный корень *третьей* степени из единицы $\zeta_3 = \frac{1}{2}(-1 + i\sqrt{3})$. Поэтому математически корректным обозначением кольца целых чисел Эйзенштейна может служить $\mathbb{Z}[\zeta_3]$. Кольцо целых чисел Эйзенштейна также является е.к., норма опять же определяется как квадрат модуля; обратимые элементы образуют циклическую группу шестого порядка, порожденную ζ_6; описание неразложимых элементов не будем здесь приводить, укажем только, что, вместо сравнений по модулю 4, приходится использовать сравнения по модулю 3).

2.2. Кольца многочленов и фактор-кольца для них

С тем, как определяются в системе **Sage** кольца от одной и нескольких переменных, мы знакомились с первых же страниц пособия (см. пример 1.1.1), причем рассматривались кольца многочленов как над полям, так и над другими кольцами. Кольцо $K = P[x]$ многочленов от одной переменной над произвольным полем P является важнейшим примером *евклидова* кольца; в частности, в нем все *идеалы* являются *главными* (**principal**) и справедлива теорема об однозначном (с точностью до ассоциированности) разложении на множители. Напомним, что группа *обратимых* многочленов состоит из ненулевых констант, *ассоциированность*, в данном случае, - это *пропорциональность*; для любых двух многочленов определены (причем, однозначно, если их выбирать нормализованными) *наибольший общий делитель* (НОД = **gcd, greatest common divisor**) и *наименьшее общее кратное* (НОК = **lcm, least common multiple**); *неразложимые* элементы в кольце многочленов принято называть *неприводимыми* (**irreducible**) многочленами. А теперь мы все это покажем средствами **Sage**.

Пример 2.2.1. Теория делимости в кольце многочленов над полем.

```
sage: QQx=PolynomialRing(QQ,n,'x'); QQx

Univariate Polynomial Ring in x over Rational Field
```

Введем два многочлена с целыми коэффициентами (но рассматриваются они над **QQ**); поделим с остатком второй многочлен на первый (**quo = quotient**, *неполное частное*, **rem = remainder**, *остаток*); выполним проверку; затем вычислим НОД и НОК, после чего реализуем *расширенный алгоритм Евклида*: вычисляется не только НОД, но и *линейное представление* для него (в обозначении метода **.xgcd** префикс **x** отсылает к термину **extended**).

```
sage: f= 2*x^5+5*x^3+x^2+2*x+2;
      g=3*x^7+6*x^5-x^3+x^2-2*x+2; f; g

2*x^5+5*x^3+x^2+2*x+2
3*x^7+6*x^5-x^3+x^2-2*x+2

sage: q,r=g.quo_rem(f); q; r; q*f+r==g

3/2*x^2-3/4
-3/2*x^4-1/4*x^3-5/4*x^2-1/2*x+7/2
True
```

```
sage: f.gcd(g); f.lcm(g)

x^2+2
x^10+5/2*x^8+1/2*x^7+2/3*x^6+4/3*x^5-5/6*x^4+2/3*x^3
 -1/6*x^2+1/3

sage: d,u,v=f.xgcd(g); d; u; v; f*u+g*v==d

x^2+2
-183/361*x^4+33/361*x^3-219/361*x^2+75/361*x+154/361
122/361*x^2-22/361*x+207/361
True
```

Далее введем еще один многочлен (над полем `QQ`) и произведем *факторизацию*, т. е. разложим этот многочлен на неприводимые множители; покажем, как извлекаются из *объекта факторизации* отдельные компоненты: пары вида

(неприводимый множитель, его кратность в разложении),

а также сами неприводимые множители.

```
sage:  h=2*x^5+5/2*x^4+3/4*x^3-25/24*x^2-x-1/2; h

2*x^5+5/2*x^4+3/4*x^3-25/24*x^2-x-1/2

sage:  h.is_irreducible()

False

sage: hf=h.factor(); hf

(2)*(x^2+3/2*x+3/4)*(x^3-1/4*x^2-1/3)

sage: hf[0]; hf[1]

(x^2+3/2*x+3/4, 1)
(x^3-1/4*x^2-1/3, 1)

sage: p=hf[1][0]; p; p.is_irreducible()

x^3-1/4*x^2-1/3
True
```

Напомним, что над *алгебраически замкнутым* полем неприводимыми многочленами являются многочлены первой степени и только они.

```
sage: p.change_ring(QQbar).factor()

(x-0.7874995561234334?)*
(x+0.2687497780617167?-0.5924982857827728?*I)*
(x+0.2687497780617167?+0.5924982857827728?*I)
```

Пример 2.2.2. Фактор-кольца кольца многочленов.

А теперь займемся *факторизацией* "в другом смысле". Первый смысл: факторизация – это разложение чего-либо на множители (факторы); второй смысл: факторизация – это построение по некоторому алгебраическому объекту и его подобъекту нового объекта, называемого фактор-объектом. Конкретно, сейчас речь пойдет о вычислении *фактор-кольца* для кольца многочленов по некоторому *идеалу* в этом кольце (аналогичной задачей мы уже занимались в п. 2.1, но – в более простом кольце **ZZ**).

Кольцо многочленов будет рассматриваться то же самое: **QQx**; сначала мы определим в нем три идеала; первые два будут главными по построению, третий, по построению, будет порождаться двумя многочленами, но и он окажется главным: порождающим для него будет НОД двух исходных многочленов; все три идеала будут протестированы на максимальность. Затем мы вычислим три фактор-кольца, и **Sage** подтвердит наши познания: фактор-кольцо по некоторому идеалу является полем тогда и только тогда, когда этот идеал является максимальным, что, для главных идеалов, равносильно неприводимости порождающего идеал многочлена.

```
sage: f=x^2+1; J1=QQx.ideal(f); J1; J1.is_maximal()

Principal ideal (x^2+1) of Univariate Polynomial Ring
 in x over Rational Field
True

sage: g=x^3+1; J2=QQx.ideal(g); J2; J2.is_maximal()

Principal ideal (x^3+1) of Univariate Polynomial Ring
 in x over Rational Field
False

sage: J3=QQx.ideal(x^4-1,x^3-x); J3; J3.is_maximal()

Principal ideal (x^2-1) of Univariate Polynomial Ring
 in x over Rational Field
False
```

Далее мы вычисли три фактор-кольца. В первом из них вы обязаны "опознать" ранее уже встречавшееся нам (см. пример 2.1.2) поле гауссовых чисел **QQ[I]**. Обратите также внимание на метод выявления порождающего элемента (*генератора*) для фактор-кольца; по умолчанию он получает имя **xbar** (но может быть переименован, мы показываем как); через него выражаются все элементы фактор-кольца и, в частности, генерируемые далее *случайные* элементы).

```
sage: K1=QQx.quotient_ring(J1); K1; K1.is_field();
      K1.0; K1.random_element()

Univariate Quotient Polynomial Ring in xbar over
 Rational Field with modulus x^2+1
True
xbar
-16*xbar-1/5

sage: K2.<y>=QQx.quotient_ring(J2); K2; K2.is_field();
      K2.0; K2.random_element()

Univariate Quotient Polynomial Ring in y over Rational
 Field with modulus x^3+1
False
y
-1/2*y^2-1/6*y+3

sage: K3.<z>=QQx.quotient_ring(J3); K3; K3.is_field();
      K3.0; K3.random_element()

Univariate Quotient Polynomial Ring in z over Rational
 Field with modulus x^2-1
False
z
3/5*z+1
```

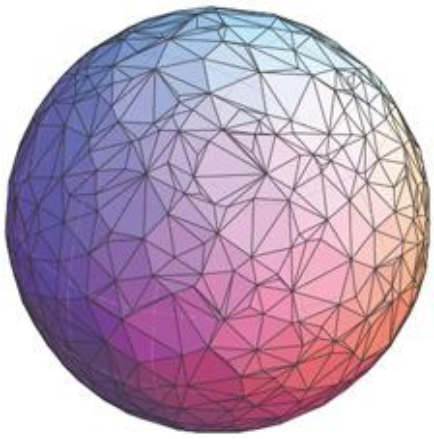
Поясним теперь термин **modulus**, используемый **Sage** при описании фактор-колец. Надо полагать, что к "модулярным вычислениям" в кольцах вычетов читатели уже приучены. Например, в фактор-кольце $\mathbb{Z}_{12} = \mathbb{Z} / (12)$ все вычисления ведутся "по модулю 12", т. е. сложение и умножение вычетов производятся как над обычными целыми числами, с последующей заменой результата на *остаток* от его деления на модуль. Аналогично обстоит дело и

в фактор-кольцах кольца многочленов, но теперь **modulus** - это многочлен, и именно на него надо делить с остатком. **Sage** с этим справляется. Пример приведем в кольце **K2**.

```
sage: a=3/11*y^2-5*y+4/3; a; a in K2;
      b=-2*y^2-1/7; b; b in K2;
      a*b

3/11*y^2-5*y+4/3
True
-2*y^2-1/7
True
-625/231*y^2+97/77*y-214/21
```

В заключение пункта остается посетовать на известную (и, по-видимому, неустранимую) особенность всевозможных учебных пособий: тему приходится "закруглять" именно тогда, когда мы приближаемся к самому интересному и интригующему. Самым интригующим в теме "Многочлены" является теория *идеалов* в кольцах многочленов от *нескольких* переменных, которая, в свою очередь, служит основой для построения алгоритмов решения *нелинейных систем уравнений*. Автор надеется, что читатели данного пособия (на старших курсах, или даже в послеуниверситетской математической жизни) непременно встретятся с понятием *базис Грёбнера* (для полиномиального идеала), научатся такие базисы строить и применять к реальным (и потому - сложным) задачам. С основами теории базисов Грёбнера и с соответствующими возможностями **Sage** можно познакомиться, обратившись к рекомендованной литературе (плюс сетевой поиск).



2.3. Конечные поля

Поле называется *простым*, если оно не содержит собственных подполей. Совсем просто доказывается, что единственными (с точностью до изоморфизма) простыми полями являются: (1) поле рациональных чисел $\mathbb{Q}$; (2) поля $\mathbb{F}_p = \mathbb{Z}_p$ классов вычетов по произвольному простому модулю p.

Произвольное поле F обязано содержать *минимальное* (*простое*) *подполе*. Если для любого натурального n сумма n штук полевых единиц (обозначаемая $n1$) отлична от нуля, то минимальное подполе в поле F изоморфно $\mathbb{Q}$; в данном случае принято говорить, что поле F имеет *нулевую характеристику* ($\mathrm{char}(F) = 0$), при этом оно обязательно бесконечно.

Если среди элементов $n1$ встречаются нулевые, то наименьшее из соответствующих n непременно оказывается простым (переобозначим: $n = p$), тогда минимальное подполе в F будет изоморфно $\mathbb{F}_p$ и, по определению, считается: $\mathrm{char}(F) = p$; само поле F в данной ситуации может быть как конечным, так и бесконечным.

Всякое *конечное* поле F имеет ненулевую характеристику ($\mathrm{char}(F) = p$ для некоторого простого p) и оказывается *линейным пространством* (некоторой *конечной* размерности s) над своим простым подполем. Отсюда немедленно следует, что количество элементов (*порядок*) q конечного поля обязательно является *примарным* числом (т. е. степенью простого числа): $q = p^s$. Доказывается, что для любого примарного числа $q = p^s$ существует *единственное* (с точностью до изоморфизма) поле порядка q; оно называется *полем Галуа* (**`Galois Field`**); наряду с обозначением $\mathbb{F}_q$, используются варианты: $GF(q)$ или $GF(p, s)$.

Практическая конструкция поля Галуа $\mathbb{F}_q$ ($q = p^s$) основана на отыскании некоторого неприводимого многочлена $f(x)$ степени s над простым полем $\mathbb{F}_p$, с последующей *факторизацией* (см. пример 2.2.2) кольца многочленов $K = \mathbb{F}_p[x]$ по главному идеалу $J = (f(x))$.

Фактор-кольцо K/J является, благодаря неприводимости $f(x)$, полем; оно одержит ровно q элементов (именно столько многочленов степени, не превышающей $s - 1$, существует над полем $\mathbb{F}_p$; эти многочлены принимаются в качестве *канонических представителей* классов смежности по идеалу J).

Для обеспечения результативности и корректности описанной конструкции требуется предварительно убедиться в том, что

(1) над полем $\mathbb{F}_p$ существуют неприводимые многочлены произвольной степени s;

(2) при различных выборах неприводимого многочлена одной и той же степени s при факторизации получаются изоморфные поля.

Эти и все другие подробности теории читателям следует изучить по рекомендуемой учебной литературе. Здесь же мы можем подвести предварительный итог.

Конструкция

$$F = \mathbb{F}_p[x]/(f(x)) \qquad (*)$$

при произвольном выборе неприводимого многочлена $f(x)$ степени s приводит к полю F, изоморфному $\mathbb{F}_{p^s}$.

Однако это еще не все трудности, которые необходимо преодолеть. Есть, по крайней мере, еще две серьезные проблемы.

Первая такова: откуда брать неприводимые многочлены? Не отвлекаясь на (вообще говоря, важные) детали сформулируем критерий.

Нормализованный многочлен $f(x)$ степени s над полем $\mathbb{F}_p$ неприводим тогда и только тогда, когда выполнено "условие делимости":

$$f(x) \mid x^{p^s-1} - 1. \qquad (**)$$

Вторая проблема: как представлять элементы построенного поля? Дело в том, что данное выше их описание с помощью многочленов степени не выше $s-1$ не всегда оказывается "удачным". При хорошем описании все ненулевые элементы поля должны представляться в виде *степеней* одного из них (*генератора*). В данном случае хотелось бы, чтобы генератором послужил многочлен первой степени x. Но это не всегда получается. (Какой-либо генератор существует всегда, но другие генераторы менее удобны.)

К счастью, имеется выход. Назовем неприводимый многочлен $f(x)$ *примитивным*, если при факторизации **(*)** одночлен x является генератором поля. Доказывается, что для любого s существуют примитивные многочлены степени s и формулируется следующий критерий примитивности.

Нормализованный многочлен $f(x)$ степени s над полем $\mathbb{F}_p$ является примитивным тогда и только тогда, когда, помимо условия делимости (**), выполнено следующее "условие неделимости": для любого натурального показателя $k < p^s - 1$ справедливо:

$$f(x) \nmid x^k - 1. \qquad (***)$$

Известно, что **(***)** достаточно проверять для $k \mid p^s - 1$. Это известно не только математикам, но и их изобретению – системе **Sage**. В частности, в **Sage** реализован алгоритм вычисления так называемых примитивных *полиномов Конвея*, которые используются при "стандартном" задании системой полей Галуа.)

И, наконец, еще один ключевой момент в теории конечных полей.

Мультипликативная группа $\mathbb{F}_q^* = \mathbb{F}_q \setminus \{0\}$ поля Галуа является циклической группой порядка $q-1$; в случае примитивности многочлена $f(x)$ в качестве порождающего элемента этой группы можно принять класс смежности одночлена x, т. е. всякий элемент $y \in \mathbb{F}_q^*$ представляется в виде

$$y = x^k, \qquad (****)$$

где k - целое число (оно определено однозначно, если предполагать, что $0 \le k \le q-2$).

Формула **(****)** называется *степенным представлением* для (ненулевых) элементов конечного поля.

Теперь пора переходить к примерам. Начнем не с полей, а с многочленов, которые при построении полей используются.

Пример 2.3.1. Полиномы Конвея.

Строгое определение полиномов Конвея (**Conway**) вы можете найти, например, в Википедии. Здесь мы покажем только способ извлечения нужного многочлена из "хранилища" **Sage**, выявим для него "родительское кольцо" (метод **.parent**), после чего применим методы тестирования на неприводимость и примитивность.

```
sage: f=conway_polynomial(2,4); f; f.parent();
      f.is_irreducible(),f.is_primitive()

x^4+x+1
Univariate Polynomial Ring in x over Finite Field of
 size 2 (using NTL)
(True, True)

sage: f=conway_polynomial(5,3); f; f.parent();
  f.is_irreducible(),f.is_primitive()

x^3 + 3*x + 3
Univariate Polynomial Ring in x over Finite Field of
 size 5
(True, True)
```

Пример 2.3.2. Поле Галуа порядка 8.

Зададим поле из восьми элементов **F8** = $\mathbb{F}_8 = \mathbb{F}_2[x]/(f(x))$ [см. схему **(*)**] и выведем список его элементов; все они выражаются через генератор, который в строго математическом смысле является *классом смежности* $x + (f(x))$, однако, по нашему выбору (в угловых скобках), обозначатся про-

сто **x**. Можно тут же запросить используемый системой многочлен $f(x)$ и убедиться в том, что он совпадает с соответствующим полиномом Конвея.

```
sage: F8.<x>=GF(2^3); L=F8.list(); L;
      f=F8.polynomial(); f;
      f==conway_polynomial(2,3)

[0, x, x^2, x+1, x^2+x, x^2+x+1, x^2+1, 1]
x^3+x+1
True
```

Обратите внимание на порядок элементов в списке.

Естественным можно было бы считать такой порядок в множестве многочленов, при котором они группируются по возрастанию степени, а внутри групп – упорядочиваются *лексикографически,* с привлечением обычного порядка `[0,1,2,…,p-1]` в поле $\mathbb{F}_p$. По такому принципу в нашем примере получилось бы: `[0, 1, x, x+1, x^2, x^2+1, x^2+x, x^2+x+1]`.

Чем мотивирован порядок, принятый **Sage**, будет объяснено далее.

А сейчас мы (в продолжение замечания о "модулярных вычислениях" в конце примера 2.2.2) подчеркнем следующее обстоятельство: вычисления в конечных полях также являются модулярными, роль **modulus** играет тот самый многочлен $f(x)$, с помощью которого определялось поле.

Но в данном случае ситуация еще интереснее, поскольку мы фактически работаем *с двумя модулями*: во-первых, коэффициенты всех задействованных многочленов рассматриваются по модулю простого числа p, а, во-вторых, сами многочлены приводятся по модулю (неприводимого и примитивного) многочлена $f(x)$.

Доступны полевые таблицы сложения и умножения. По умолчанию элементам поля присвоены буквенные обозначения: восемь первых букв латинского алфавита кодируют восемь элементов поля, причем именно в том порядке, в котором они занесены выше в список **L**.

Представим здесь таблицу умножения для $\mathbb{F}_8$ и сравним ее с полученной ранее, в примере 2.1.1, таблицей умножения для $\mathbb{Z}_8$ (в которой, для единообразия, сменим кодировку элементов на буквенную).

Попробуйте понять, в чем принципиальное различие между этими таблицами?

```
sage: F8.multiplication_table(),
      Z8.multiplication_table()
```

```
(
*  a b c d e f g h     *  a b c d e f g h
 +---------------      +---------------
a| a a a a a a a a     a| a a a a a a a a
b| a c d e f g h b     b| a b c d e f g h
c| a d e f g h b c     c| a c e g a c e g
d| a e f g h b c d     d| a d g b e h c f
e| a f g h b c d e     e| a e a e a e a e
f| a g h b c d e f     f| a f c h e b g d
g| a h b c d e f g     g| a g e c a g e c
h| a b c d e f g h ,   h| a h g f e d c b
)
```

Читателям следует четко осознать, что $\mathbb{Z}_q$ (кольцо), при *составных* q, отнюдь не идентично с $\mathbb{F}_q$ (полем); тогда как для простых значений порядка ($q = p$) можно считать, что $\mathbb{F}_p = \mathbb{Z}_p$.

Далее поупражняемся в арифметических действиях над элементами поля.

```
sage: x^3, x^1000, 1/x, (1+x^3+x^94)/(1+x^4+x^80)

(x+1, x^2+1, x^2+1, x^2+x+1)
```

И, наконец, вернемся к якобы "неестественной" нумерации элементов поля $\mathbb{F}_q$. На самом деле она достаточно естественна: в качестве номера элемента берется *логарифм* этого это элемента по *основанию*, равному *генератору* поля. Точнее, благодаря степенному представлению **(****)**, всякий элемент $y \in \mathbb{F}_q^*$ представляется в виде $y = x^k$, где x - генератор. Число k естественно назвать логарифмом элемента y по основанию x; используется обычное обозначение: $k = \log_x y$. Логарифм определен однозначно по модулю $q - 1$, и – совсем однозначно, если ограничиться значениями от 0 до $q - 2$. Чтобы не исключать из нумерации нулевой элемент, производится некоторая модификация: номер единичного элемента увеличивается на период, т. е. становится равным $q - 1$, а номер 0 закрепляется за нулевым элементом.

Приведем, без подробных комментариев, следующий

Пример 2.3.3. Поле Галуа порядка 81.

```
sage: F81.<x>=GF(3^4); L=F81.list(); L;
      f=F81.polynomial(); f;
```

```
[0, x, x^2, x^3, x^3+1, x^3+x+1, x^3+x^2+x+1,
 2*x^3+x^2+x+1, x^2+x+2, x^3+x^2+2*x, 2*x^3+2*x^2+1,
 x^3+x+2, x^3+x^2+2*x+1, 2*x^3+2*x^2+x+1, x^3+x^2+x+2,
 2*x^3+x^2+2*x+1, 2*x^2+x+2, 2*x^3+x^2+2*x, 2*x^2+2,
 2*x^3+2*x, 2*x^3+2*x^2+2, x^3+2*x+2, x^3+2*x^2+2*x+1,
 2*x^2+x+1, 2*x^3+x^2+x, x^2+2, x^3+2*x, x^3+2*x^2+1,
 x+1, x^2+x, x^3+x^2, 2*x^3+1, 2*x^3+x+2,
 2*x^3+x^2+2*x+2, 2*x^2+2*x+2, 2*x^3+2*x^2+2*x,
 x^3+2*x^2+2, 2*x+1, 2*x^2+x, 2*x^3+x^2, 2, 2*x, 2*x^2,
 2*x^3, 2*x^3+2, 2*x^3+2*x+2, 2*x^3+2*x^2+2*x+2,
 x^3+2*x^2+2*x+2, 2*x^2+2*x+1, 2*x^3+2*x^2+x,
 x^3+x^2+2, 2*x^3+2*x+1, 2*x^3+2*x^2+x+2,
 x^3+x^2+2*x+2, 2*x^3+2*x^2+2*x+1, x^3+2*x^2+x+2,
 x^2+2*x+1, x^3+2*x^2+x, x^2+1, x^3+x, x^3+x^2+1,
 2*x^3+x+1, 2*x^3+x^2+x+2, x^2+2*x+2, x^3+2*x^2+2*x,
 2*x^2+1, 2*x^3+x, 2*x^3+x^2+2, 2*x+2, 2*x^2+2*x,
 2*x^3+2*x^2, x^3+2, x^3+2*x+1, x^3+2*x^2+x+1, x^2+x+1,
 x^3+x^2+x, 2*x^3+x^2+1, x+2, x^2+2*x, x^3+2*x^2, 1]

x^4 + 2*x^3 + 2

sage: y=2*x^3+x^2+2; y; k=y.log(x); k; y==L[k]
2*x^3 + x^2 + 2
67
True

sage: (L[11]*L[68]).log(x)
79
```

Надо полагать, что читатели обратили внимание, на, может быть, не совсем привычный синтаксис метода `.log`: в скобках, после знака логарифма, указывается *основание* логарифмов, *аргумент* ставится перед точкой и знаком функции. Важно также, что степенное представление делает легко осуществимым *умножение* элементов: при умножении показатели складываются по модулю p. Разумеется, за это приходится платить усложнением алгоритма *сложения*. Что касается операции *обращения* элементов, то, в силу периодичности значений степеней, ее можно заменить вычислением подходящей степени: `x^(q-1)=1` влечет `x^(-1)=x^(q-2)`, и, более общим образом, `L[k]^(-1)=L[q-k-1]`.

2.4. Группы

До сих пор почти все рассматривавшиеся нами **Sage**-объекты и операции относились к *коммутативной алгебре*. (Единственным исключением можно считать алгебру квадратных матриц заданного размера.) Теперь же мы обращаемся к изучению *групп*, которые могут быть как коммутативными, так и некоммутативными. Группа является, в принципе, более простым и фундаментальным объектом, нежели кольцо или поле, поскольку на ней задана всего лишь одна основная алгебраическая операция (сложение или умножение), а не две взаимодействующих операции (как в случае колец). С другой стороны, теория групп значительно богаче важными примерами и значительно глубже развита. В связи с этим "групповая" техника в **Sage** также является более сложной, продвинутой. Мы сможем здесь остановиться лишь на простейших типах и примерах (в основном, *конечных*) групп.

Пример 2.4.1. Группы перестановок.

Перестановки мы понимаем как биекции конечного множества. В учебниках принята *двустрочная* запись перестановок, явно указывающая, какой элемент куда переходит.

Перестановки, будучи *отображениями* и умножаются как отображения, т. е. произведение перестановок – это *композиция*. В большинстве современных учебников по алгебре принято выполнять композицию справа налево, и это правило распространяется на перестановки. Однако сохраняется и противоположная традиция. Во многих классических руководствах учат перемножать перестановки слева направо (хотя в соседних параграфах другие отображения перемножаются в противоположном порядке). Этот "разнобой" был привнесен и в **Sage**, о чем сожалеет, в частности, один из активных пропагандистов системы R. Beezer (см. [4, 5]). Пока с этим ничего поделать нельзя, просто читатели должны помнить, что в данном пособии произведение перестановок выполняется не так, как, скажем, в [13] или [20], а – в противоположном порядке.

Для кодирования перестановок в системах компьютерной алгебры двустрочная запись, как правило, не используется, ввиду ее избыточности. Гораздо более экономной оказывается задание перестановки ее разложением на *независимые циклы*. Именно его применяет **Sage**, причем в двух версиях: с привлечением строчного типа (**`string`**) и в виде списка кортежей (**`tuples`**). Мы пойдем по второму пути; подскажем только тем пользователям, который захотят использовать циклы длины единица (делать это без

особой нужды, разумеется, не стоит), что в круглых скобках, после единственного элемента кортежа единичной длины, полагается запятая.

По записи перестановки с помощью разложения на циклы *степень* перестановки однозначно не определяется, поэтому ввод перестановок обычно предваряется заданием объемлющей группы, которая имеет стандартное имя: *симметрическая*.

```
sage: S6=SymmetricGroup(6); S6

Symmetric group of order 6! as a permutation group

sage: epsilon=S6([()]); epsilon; epsilon in S6;
      phi=S6([(1,),(4,2,6),(5,3)]); phi; phi in S6;
      psi=S6([(6,2,3),(5,4)]); psi; psi in S6

()
True
(2,6,4)(3,5)
True
(2,6)(4,5)
True
```

Sage аккуратен: он опознал, тождественную перестановку, убрал ненужный *тривиальный* цикл, во всех нетривиальных циклах переделал запись так, чтобы цикл начинался с минимального по номеру элемента, и (внимание!) "ликвидировал" характерные для списков квадратные скобки и запятые и тем самым фактически перешел от *списочной* к *строчной* записи перестановок. Далее мы производим перемножение элементов и наблюдаем некоммутативность.

```
sage: alpha=phi*psi; alpha; beta=psi*phi; beta;
      alpha==beta

(3,4,6,5)
(2,4,3,5)
False
```

А теперь мы поработаем с группой S_4 перестановок степени 4: выведем список всех ее элементов, определим *порядок* (мощность) группы, протестируем группу на *цикличность* и коммутативность, вычислим *порядки* всех ее элементов, найдем *порождающие элементы* (**generators**). [Всякий элемент группы может быть представлен в виде произведения генераторов и элементов, обратных к генераторам.]

```
sage: S4=SymmetricGroup(4); S4; S4.order();
      LS4=S4.list(); LS4

Symmetric group of order 4! as a permutation group
24
[(),(3,4),(2,3),(2,3,4),(2,4,3),(2,4),(1,2),(1,2)(3,4),
 (1,2,3),(1,2,3,4),(1,2,4,3),(1,2,4),(1,3,2),(1,3,4,2),
 (1,3),(1,3,4),(1,3)(2,4),(1,3,2,4),(1,4,3,2),(1,4,2),
 (1,4,3),(1,4),(1,4,2,3),(1,4)(2,3)]

sage: S4.is_cyclic(); S4.is_abelian();
      [k.order() for k in S4]

False
False
[1,2,2,3,3,2,2,2,3,4,4,3,3,4,2,3,2,4,4,3,3,2,4,2]

sage: a,b=S4.gens(); a,b

((1,2,3,4),(1,2))
```

Перейдем к исследованию некоторых известных подгрупп в группе S_4 и начнем с *диэдральной* группы D_4, являющейся группой симметрий квадрата. Убедимся, что она действительно является подгруппой в группе S_4 (нециклической, некоммутативной и *ненормальной*), определим генераторы.

```
sage: D4=DihedralGroup(4); D4; D4.order();
      LD4=D4.list(); LD4

Dihedral group of order 8 as a permutation group
8
[(),(2,4),(1,2)(3,4),(1,2,3,4),(1,3),(1,3)(2,4),
 (1,4,3,2),(1,4)(2,3)]

sage: D4.is_subgroup(S4), D4.is_cyclic(),
      D4.is_abelian(), D4.is_normal(S4); D4.gens()

(True,False,False,False)
[(1,2,3,4),(1,4)(2,3)]
```

Итак, диэдральная группа D_4 является подгруппой, хотя и не нормальной, индекса 24/8 = 3 в симметрической группе S_4. Следовательно, мы имеем два (равномощных, но различных) фактор-множества: $S_4 \,/\, D_4$ и $D_4 \setminus S_4$; пер-

вое из них состоит из классов смежности (= cosets) вида xD_4, в количестве, равном индексу, а второе – из классов смежности вида D_4x, в таком же количестве. Здесь снова "в товарищах согласья нет": которые из этих классов *левые*, а которые – *правые*? Обычно левыми называются классы первого типа (в них представитель x располагается слева от подгруппы, в правых классах - наоборот). Sage снова "путает" ориентацию (меняет на противоположную).

R. Beezer [4] пишет: "Не проблема – просто помните об этом!" (Надо не забывать еще и про то, что умножение перестановок в Sage также производится иначе, и либо смириться, либо - пытаться учесть сразу два фактора.)

Попробуйте разобраться в следующих двух списках.

```
sage: S4.cosets(D4,side='right');
      S4.cosets(D4,side='left')
 [
 [(),(2,4),(1,2)(3,4),(1,2,3,4),(1,3),(1,3)(2,4),
  (1,4,3,2),(1,4)(2,3)],
 [(3,4),(2,3,4),(1,2),(1,2,4),(1,4,3),(1,4,2,3),
  (1,3,2),(1,3,2,4)],
 [(2,3),(2,4,3),(1,3,4,2),(1,3,4),(1,2,3),
  (1,2,4,3),(1,4,2),(1,4)]
]

[
 [(),(2,4),(1,2)(3,4),(1,2,3,4),(1,3),(1,3)(2,4),
  (1,4,3,2),(1,4)(2,3)],
 [(3,4),(2,4,3),(1,2),(1,2,3),(1,3,4),(1,3,2,4),
  (1,4,2),(1,4,2,3)],
 [(2,3),(2,3,4),(1,2,4,3),(1,2,4),(1,3,2),
  (1,3,4,2), (1,4,3), (1,4)]
]
```

Для групп невысокого порядка Sage в состоянии вывести списком все их подгруппы (все нормальные подгруппы). В группе D_4 имеется десять различных подгрупп, среди них – шесть нормальных.

```
sage: D4sub=D4.subgroups(); D4sub; len(D4sub)

[
 Permutation Group with generators [()],
 Permutation  Group with generators [(1,3)(2,4)],
```

```
 Permutation Group with generators [(2,4)],
 Permutation Group with generators [(1,3)],
 Permutation Group with generators [(1,2)(3,4)],
 Permutation Group with generators [(1,4)(2,3)],
 Permutation Group with generators [(2,4),(1,3)(2,4)],
 Permutation Group with generators [(1,2,3,4),
  (1,3)(2,4)],
 Permutation Group with generators [(1,2)(3,4),
  (1,3)(2,4)],
 Permutation Group with generators [(2,4),(1,2,3,4),
  (1,3)(2,4)]
]
10
```

```
sage: D4subn=D4.normal_subgroups(); D4subn; len(D4subn)
```

```
[
 Subgroup of (Dihedral group of order 8 as a
  permutation group) generated by [()],
 Subgroup of (Dihedral group of order 8 as a
  permutation group) generated by [(1,3)(2,4)],
 Subgroup of (Dihedral group of order 8 as a
  permutation group) generated by [(1,2)(3,4),
   (1,3)(2,4)],
 Subgroup of (Dihedral group of order 8 as a
  permutation group) generated by [(2,4),(1,3)(2,4)],
 Subgroup of (Dihedral group of order 8 as a
  permutation group) generated by [(1,2,3,4),
   (1,3)(2,4)],
 Subgroup of (Dihedral group of order 8 as a
  permutation group) generated by [(1,2,3,4),
   (1,4)(2,3)]
]
6
```

Повторим наши вычисления для подгруппы D_4 применительно к подгруппе четных перестановок A_4. Эта подгруппа является нормальной и потому левое и правое фактор-множества в данном случае совпадают. На фактор-множестве (здесь оно одно) корректно определяется структура группы. Так получается *фактор-группа* S_4 / A_4. (И мы знаем, что это за группа! Но знает ли про это **Sage**?)

```
sage: A4=AlternatingGroup(4); A4; A4.order();
      LA4=A4.list(); LA4;
      A4.is_subgroup(S4), A4.is_cyclic(),
      A4.is_abelian(), A4.is_normal(S4);
      A4.gens();
      H=S4.quotient(A4); H; H.order(); H.is_ cyclic ()

Alternating group of order 4!/2 as a permutation group
12
[(),(2,3,4),(2,4,3),(1,2)(3,4),(1,2,3),(1,2,4),(1,3,2),
 (1,3,4),(1,3)(2,4),(1,4,2),(1,4,3),(1,4)(2,3)]
(True, False, False, True)
[(2,3,4),(1,2,3)]
Permutation Group with generators [(1,2)]
2
True
```

Да, знает. Получилась циклическая фактор-группа второго порядка, изоморфная $\mathbb{Z}^* = \{1, -1\}$, но здесь она реализована и описывается как группа S_2перестановок степени два. Предлагаем читателям опознать и исследовать еще одну ("именную") подгруппу в S_4 (в учебниках по теории групп она неизменно фигурирует, поскольку служит для демонстрации *нетранзитивности* отношения "быть нормальной подгруппой").

```
sage: V4=PermutationGroup(['(1,2)(3,4)','(1,3)(2,4)'])
```

Пример 2.4.2. Матричные группы над конечными полями.

Группа обратимых матриц над конечным полем сама является конечной и может быть успешно исследована средствами **Sage**. Рассмотрим общую линейную (**general linear**) группу $GL(2, \mathbb{F}_2)$ над полем из двух элементов.

```
sage: G=GL(2,2); G; L=G.list(); L; G.order();
      G.is_abelian();

General Linear Group of degree 2 over Finite Field of
 size 2
(
[0 1]  [0 1]  [1 0]  [1 0]  [1 1]  [1 1]
[1 0], [1 1], [0 1], [1 1], [0 1], [1 0]
)
```

```
6
False
```

Получилась неабелева группа порядка 6. Доступна таблица умножения, которая для групп называется *таблицей Кэли*.

```
sage: G.cayley_table(); g=G.gens(); g
```

```
*  a b c d e f
 +------------
a| c e a f b d
b| d f b e a c
c| a b c d e f
d| b a d c f e
e| f d e b c a
f| e c f a d b
```

Известно, что единственной, с точностью до изоморфизма, неабелевой группой шестого порядка является группа перестановок S_3. Результат $GL(2,\mathbb{F}_2) \cong S_3$ можно получить средствами **Sage**, если предварительно задать группу матриц **G** с помощью порождающих элементов (метод **.gens**), а потом представить заданную генераторами группу как группу перестановок (метод **.as_permutation_group**).

```
sage: g=G.gens(); g
```

```
(
[1 1]  [0 1]
[0 1], [1 0]
)
```

Найдены две матрицы, являющиеся генераторами для **G**, Теперь мы сменим описание группы, она будет задаваться через генераторы; придется изменить имя, но мы тут же убедимся в том, что имя **H** относится к той же самой группе

```
sage: H=MatrixGroup(g[0],g[1]); H; H==G;
```

```
Matrix group over Finite Field of size 2 with 2
 generators
(
[1 1]  [0 1]
[0 1], [1 0]
```

```
)
True
```

А теперь снова сменим описание: конечную матричную группу представим с помощью изоморфной ей группы перестановок.

```
sage: K=H.as_permutation_group(); K;
      K==SymmetricGroup(3)

Permutation Group with generators [(2,3),(1,2)]
True
```

И в заключение – толика информации о работе с бесконечными группами. (Это возможно! Но только, если группа является *конечнопорожденной* (имеет конечное число генераторов). Между генераторами допускается наличие конечного числа *соотношений* (называемых *определяющими*).

Пример 2.4.3. Свободные группы конечного ранга.

Свободная группа $F(X)$ над конечным алфавитом $X = \{x_1, \dots, x_n\}$ определяется как множество *редуцированных слов*, каждое из которых является конечной последовательностью (списком) символов двух типов: *букв* x_i и формально *обратных* к ним элементов x_i^{-1}; редуцированность проявляется в том, что два взаимно обратных символа не могут быть соседними. Умножение слов определяется как их *конкатенация* (слияние), с последующей *редукцией* (сокращением всего, что сокращается); каждое слово оказывается произведением своих букв (в степенях ± 1); роль нейтрального элемента (единицы) играет *пустое слово*. Мощность алфавита называется *рангом* свободной группы. Две свободные группы *изоморфны* тогда и только тогда, когда имеют одинаковые ранги.

В свободной группе совсем нет определяющих соотношений. Всякая группа может быть представлена как фактор-группа свободной группы, что на практике сводится к добавлению определяющих соотношений, которым должны удовлетворять порождающие элементы (генераторы, они же – буквы исходного алфавита).

Определим свободную группу ранга 3 с генераторами по умолчанию:

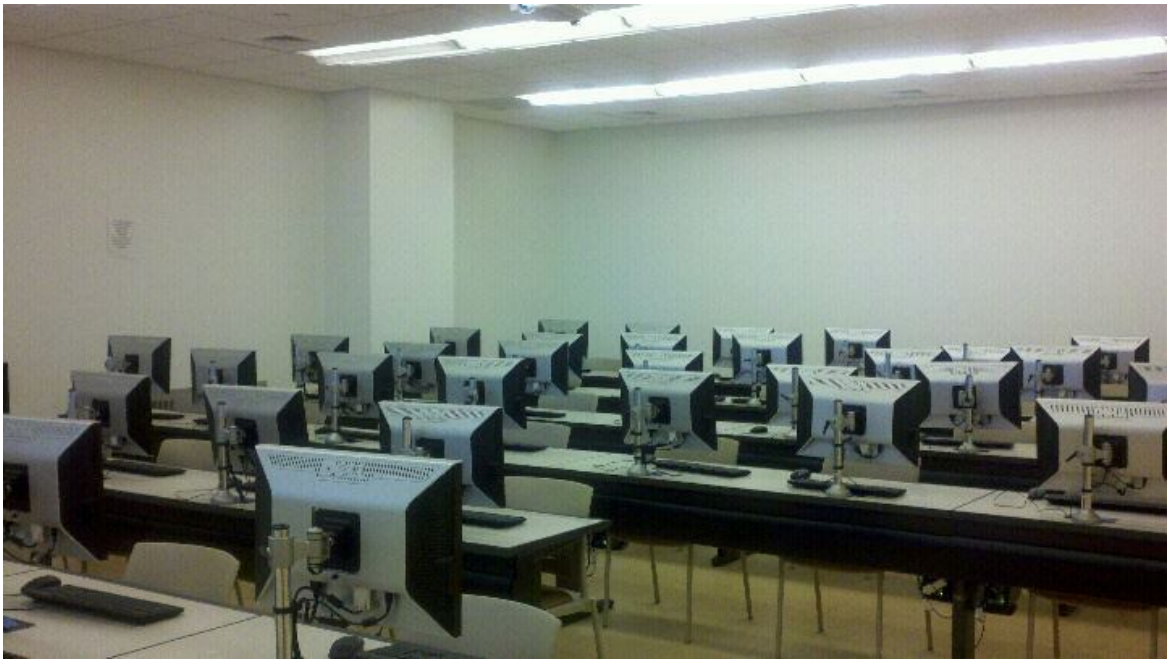
```
sage: FreeGroup(3)

Free Group on generators {x0, x1, x2}
```

А теперь определим свободную группу ранга 3 с заданными именами генераторов (a, b, c). Покажем далее, как вводятся и перемножаются элемен-

ты (исходные слова: $u = aabaa^{-1}a^{-1}cb$, $v = b^{-1}c^{-1}a^{-1}ccca$; их произведение: $uv = aaba^{-1}a^{-1}a^{-1}ccca$).

```
sage: F.<a,b,c>=FreeGroup(); F;
      u=F([1,1,2,-1,-1,3,2]); u;
      v=F([-2,-3,-1,3,3,3,1]); v; u*v
Free Group on generators {a, b, c}
a^2*b*a^-2*c*b
b^-1*c^-1*a^-1*c^3*a
a^2*b*a^-3*c^3*a
```



Интернет-ресурсы и литература

Главная страница SAGE (open source mathematical software)
Tutorials and References

1. *The Sage Development Team.* **SAGE Reference. v6.0.**
http://www.sagemath.org/doc/reference/
2. *The Sage Development Team.* **Sage Tutorial in Russian.** Выпуск 6.0.
http://www.sagemath.org/pdf/ru/tutorial/SageTutorial_ru.pdf

Tutorials (on SAGE and its applications in algebra)

3. *Beezer R.* **Sage Primer for Linear Algebra.** 2013.
http://linear.pugetsound.edu/download/fcla-3.20-sage-6.0-primer.pdf
4. *Beezer R.* **Sage for Abstract Algebra.** A Supplement to "Abstract Algebra, Theory and Applications". 2013.
http://abstract.pugetsound.edu/download/aata-20130816-sage-5.11.pdf
5. *Beezer R.* **Group Theory and SAGE: A Primer**. 2009.
http://buzzard.ups.edu/sage/sage-group-theory-primer.pdf
6. *Kosan T.* **SAGE For Newbies.** 2007.
http://sage.math.washington.edu/home/tkosan/newbies_book/sage_for_newbies_v1.23.pdf

Учебные пособия по SAGE (на русском языке)

7. *Зобнин А. И., Соколова О. В.* **Компьютерная алгебра в системе SAGE.** М.: изд-во МГТУ им. Баумана, 2011. 53 с.
http://halgebra.math.msu.su/practicum/practicum.pdf
8. *Куваев А. Е., Смоляков А. С.* **Система компьютерной алгебры SAGE: установка и основы программирования.** Иваново: изд-во "Ивановский государственный университет", 2013. 35 с.
http://lib.ivanovo.ac.ru:81/elib/dl/matematika/metod/kuvaev_2014.pdf/

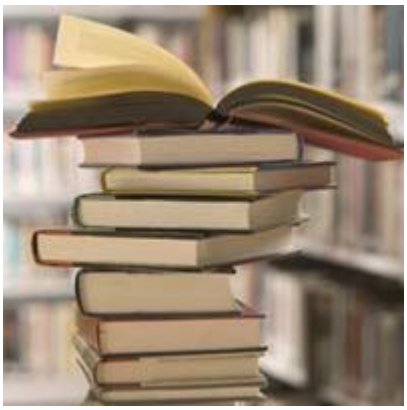
Textbooks in Algebra

9. *Beezer R.* **A First Course in Linear Algebra**. 2013.
http://linear.ups.edu/download/fcla-3.20-tablet.pdf
http://linear.pugetsound.edu/
10. *Beezer R.* **Exercise and Solution Manual** for "A First Course in Linear Algebra". 2013.
http://linear.pugetsound.edu/download/fcla-3.20-solution-manual.pdf
11. *Judeson T.* **Abstract Algebra. Theory and Applications.** 2013.
http://abstract.ups.edu/download/aata-20130816.pdf

Учебники и учебные пособия по алгебре (на русском языке)

12. *Акритас А.* **Основы компьютерной алгебры с приложениями.** М.: Мир, 1994. 544 с.
13. *Винберг Э. Б.* **Курс алгебры.** М.: Факториал-Пресс, 2001. 544 с.

14. *Глухов М. М., Елизаров В. П., Нечаев А. А.* **Алгебра.** Т. 1, 2. М.: Гелиос АРВ, 2003. 336 + 415 с.
15. *Горюшкин А. П., Горюшкин В. А.* **Элементы абстрактной и компьютерной алгебры.** Петропавловск-Камчатский, 2011. 510 с.
16. *Воеводин В. В., Воеводин Вл. В.* **Энциклопедия линейной алгебры.** Электронная система ЛИНЕАЛ. СПб: БХВ-Петербург., 2006. 544 с.
17. *Кокс Д., Литтл Дж., О'Ши Д.* **Идеалы, многообразия, алгоритмы.** М.: Мир, 2000. 688 с.
18. *Ноден П., Китте К.* **Алгебраическая алгоритмика.** М.: Мир, 1999. 720 с.
19. *Панкратьев Е. В.* **Элементы компьютерной алгебры.** М.: Бином, 2007. 248 с.
20. *Яцкин Н. И.* **Алгебра: Теоремы и алгоритмы.** Иваново: изд-во "Ивановский государственный университет", 2006. 506 с.
21. *Яцкин Н. И.* **Линейная алгебра: Теоремы и алгоритмы.** Иваново: изд-во "Ивановский государственный университет", 2008. 608 с.
22. *Яцкин Н. И.* **Жордановы (и частично жордановы) базисы для линейных операторов над полем рациональных чисел.** Иваново: изд-во "Ивановский государственный университет", 2010. 180 с.



Составитель
ЯЦКИН Николай Иванович

АЛГЕБРАИЧЕСКИЕ ВЫЧИСЛЕНИЯ В СИСТЕМЕ SAGE

Методические указания по дисциплинам "Фундаментальная алгебра" и "Компьютерная алгебра" для студентов 2 курса факультета математики и компьютерных наук (квалификация "Бакалавр")

Под редакцией Е. В. Соколова
Печатается в авторской редакции

Директор издательства Л. В. Михеева

Подписано к печати **.**. 2014 г.
Формат 60×64 1/16. Бумага писчая. Печать плоская.
Усл. печ. л. *,**. Уч.-изд. л. *,*. Тираж ** экз.

Издательство "Ивановский государственный университет"
153025 Иванов, ул. Ермака, 39
(4930) 93-43-41 E-mail publisher@ivanovo.ac.ru