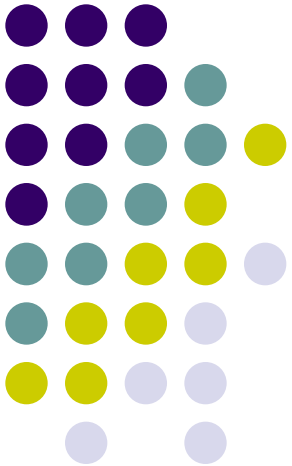
Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Дискретная математика

курс лекций

лекция 8

Алгебраические структуры



Кафедра
«Проектирования и безопасности компьютерных систем»
Гришенцев А. Ю.
www.moveinfo.ru

Санкт-Петербург
2014

Алгебраические структуры

На произвольном множестве *P* возможно задать множество различных алгебраических операций. Множество *P* с заданной на нём алгебраической операцией * обозначается (*P,**). Во многих случаях на множестве *P* целесообразно рассматривать несколько (*N*) различных алгебраических операций $*_1,*_2,\ldots,*_N$. В этом случае используют обозначение $(P,\{*_1,*_2,\ldots,*_N\})$ или $(P, *_1, *_2, \ldots, *_N)$.

Определение. Объект вида $(P,\{*_1,*_2,\ldots,*_N\})$, где *P* – некоторое множество, а $\{*_1,*_2,\ldots,*_N\}$ – множество заданных на P алгебраических операций, называют *алгебраической структурой*.

Пример. $(\mathbb{Q} - \{0\}, \{\cdot, ^{-1}, e=1\})$ – множество рациональных чисел $\mathbb{Q}$ без нуля с умножением, операцией взятия обратного элемента и единичным элементом.

Пример. $(\mathbb{N},\{+,-,e=0\})$ – множество вещественных чисел $\mathbb{R}$ с сложением, вычитанием и нулевым элементом.

Пример. $(A_{MxN},\{\cdot,+\})$ – множество квадратных матриц A_{MxN} размера *MxN* (*M=N*) умножением и сложением.

Ассоциативность и коммутативность

Определение. Пусть на множестве P задана бинарная операция *. Операция * называется *ассоциативной*, если для любых $a,b,c \in P$ выполняется равенство:

$$(a*b)*c = a*(b*c).$$

Если для любых $a,b \in P$ выполняется равенство:

$$a*b = b*c,$$

то операция * называется *коммутативной*.

Замечание. Ассоциативность и коммутативность – это независимые свойства, поскольку существуют операции, обладающие одним из этих свойств, но не другим.

Пример 1. Например, для если для матриц A, B, C существует произведение $(AB)C$ то в силу сочетательного свойства произведения матриц $(AB)C = A(BC)$. В то же время из существования произведения матриц AB вовсе не следует существование произведения BA.

Пример 2. Пусть для множества строк S задана операция конкатенация (склейка или сложение строк) тогда для $s_1,s_2,s_3 \in S$ справедливо ассоциативное свойство $(s_1\ s_2)\ s_3 = s_1\ (s_2\ s_3)$, при этом $s_1\ s_2 \neq s_2\ s_1$.

Пусть: s_1 = «мате», s_2 = «мати», s_3= «ка», тогда

(«мате» «мати») «ка» = «мате» («мати» «ка»), но

«мати» «ка» ≠ «ка» «мати».

Полугруппы

Определение. Алгебраическая структура (P,*), где P – некоторое множество, а * - ассоциативная операция на множестве P, называется *полугруппой*.
Если (P,*) – полугруппа и если операция * коммутативна, то полугруппа P называется *коммутативной полугруппой*.
Определение. Если в полугруппе (P,*) операция * есть операция умножения, то такая полугруппа называется *мультипликативная полугруппа*, если операция * есть операция сложения, то такая полугруппа есть *аддитивная полугруппа*.

Пример 1. ($\mathbb{N}$,+) – аддитивная полугруппа заданная на множестве натуральных чисел $\mathbb{N}$ с операцией сложения.
Пример 2. ($\mathbb{N}$,·) – мультипликативная полугруппа заданная на множестве натуральных чисел $\mathbb{N}$ с операцией умножения.
Пример 3. (S,) – аддитивная полугруппа заданная на множестве строк S с операцией конкатенации (сложения, склейки строк).

Определение. Пусть P – полугруппа относительно операции * и пусть M – замкнутое относительно операции * подмножество множества P ($M \subseteq P$). Тогда M будет полугруппой относительно *, при этом M называется *подполугруппой* полугруппы P.

Единичный элемент

Определение. Элемент $e_L \in P$ называется *левым единичным* (или *левым нейтральным*) элементом алгебраической структуры (P,*), если для любого элемента $p \in P$ имеет место равенство $e_L * p = p$. Аналогично, элемент $e_R \in P$ называется *правым единичным* (или *правым нейтральным*) элементом, если для любого $p \in P$ имеет место равенство $p * e_R = p$.

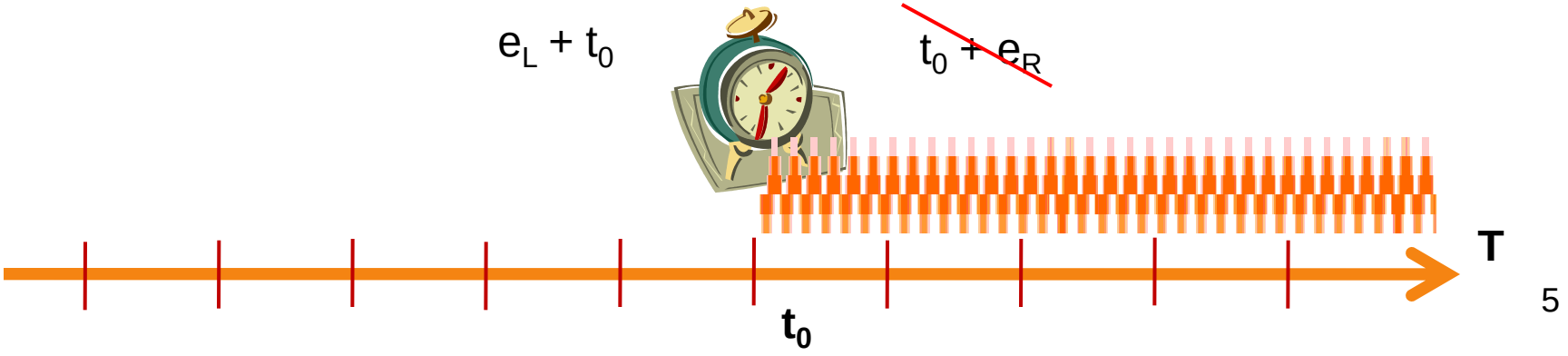
Предположение. Если в алгебраической структуре (P,*) существует левый единичный элемент $e_L \in P$ и правый единичный элемент $e_R \in P$, тогда $e_R = e_L$.

Доказательство. Следует из записи $e_R = e_L * e_R = e_L$.

Определение. Элемент $e \in P$ называется *единичным* (или *нейтральным*) элементом алгебраической структуры (P,*), если для любого элемента $p \in P$ имеет место равенство $e*p=p*e=p$.

Замечание. Если в алгебраической структуре существует единичный элемент, то он является единственным.

Доказательство. Пусть в алгебраической структуре существует два единичных элемента e_1 и e_2, тогда, по определению единичного элемента: $e_1 = e_1 * e_2 = e_2$.



Полугруппа с единичным элементом или моноид

Определение. Если P – есть полугруппа относительно операции * и если для алгебраической структуры $(P,\{*,e\})$ задан единичный элемент e, то P называется *полугруппой с единицей*, или *моноидом*.

Пример. Пусть задана алгебраическая структура $(\Omega,\{\cap,e\})$, где Ω – множества, а $\cap$ – операция пересечения множеств, тогда единичным элементом e будет пустое множество $e = \varnothing$.

Определение. Пусть P – полугруппа с единицей (моноид) относительно операции * и пусть M – замкнутое относительно операции * подмножество множества P ($M \subseteq P$), причём $e \in M$ (где e – единица полугруппы P). Тогда M называется подмоноидом моноида P.

Обратный элемент

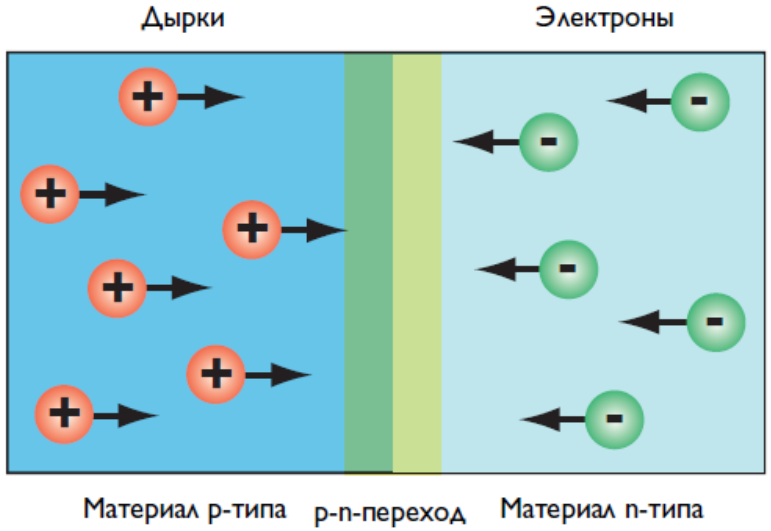
Определение. Для (P,{$*$,e}), где P множество с определённой на нём операцией $*$, и единичным элементом e, если для любого $x \in P$, определён элемент a:

1. $x*a = e$, то такой элемент есть обратный к x элемент справа и обозначается $x_R^{-1} = a$;
2. $a*x = e$, то такой элемент есть обратный к x элемент слева и обозначается $x_L^{-1} = a$;
3. $a*x=x*a=e$, то такой элемент есть обратный к x элемент и обозначается $x^{-1} = a$.

Элемент для которого существует обратный элемент, называют обратимым элементом.

Замечание. Если заданная на множестве P операция $*$ коммутативна и для элемента $x \in P$ определёны правый и левый обратные элементы x_R^{-1} и x_L^{-1}, то из свойства коммутативности $x*x_R^{-1} = x_L^{-1}*x = e \rightarrow x_R^{-1} = x_L^{-1} = x^{-1}$.

Пример. Пусть определена строка s = «*abc*» тогда при заданной коммутативной операции сложения (вычитания) строк *обратная строка* будет $-s$ = -«*abc*», *единичным элементом* e в этом случае будет пустая строка e = «», возможно записать: $e = s+(-s)$ = «*abc*»+(-«*abc*») = «».



Группы

Определение *Группа* есть множество G на котором определена бинарная операция $x*y$ и унарная операция x^{-1} (обратный элемент), определён отмеченный элемент e (единица), т.е. $(G,\{*,^{-1},e\})$. Группа G для $\forall x,y,z \in G$, удовлетворяет следующим аксиомам:

1. $(x*y)*z = x*(y*z)$ 2. $x^{-1}*x = x*x^{-1} = e$ 3. $x*e = e*x = x$.

Группа G коммутативна (или абелева), если выполняется:

4. $x*y = y*x$.

или *Определение.* Полугруппа G с единицей такая, что для любого элемента $x \in G$ ссуществует обратный элемент $x^{-1} \in G$ называется *группой*. Число $|G|$ называется *порядком группы* G.

Замечание

1. Группа обозначается через $(G,\{*,^{-1},e\})$ или просто буквой G. Группа конечна, если она имеет конечное число элементов.
2. Если операция звёздочка эквивалентна умножению $(x*y \leftrightarrow x\cdot y)$ то говорят, что группа мультипликативная. Если операция звёздочка эквивалентна сложению $(x*y \leftrightarrow x+y)$ то говорят, что группа аддитивная.
3. Степень $a^n=a\cdot a\cdot a\cdot\ldots\cdot a$ (n раз). По определению полагают, что $a^0=e$ и $a^{-n}=(a^{-1})^n$.

Определение *Группа конечна*, если число её элементов конечно. *Порядок* конечной группы есть число её элементов.

Определение Непустое множество $H \subseteq G$ есть *подгруппа* группы G, если H есть группа по отношению к операциям, определённым в G. Если $H \neq G$, то H есть *собственная подгруппа* в G.

Примеры групп

Пример 1. Множество целых чисел $\mathbb{Z}$ со сложением есть аддитивная абелева группа ($\mathbb{Z}$, {+,-,e=0}).
Пример 2. Множество рациональных чисел $\mathbb{Q}$ без нуля с умножением есть мультипликативная абелева группа ($\mathbb{Q}$ - {0}, {·,$^{-1}$,e=1}).
Пример 3. Множество $\mathbb{Z}_n$ с операцией сложения по модулю n есть аддитивная абелева группа порядка n.
Пример 4. Множество $\mathbb{Z}_n$ с операцией умножения по модулю n не есть группа, ибо не все элементы имеют обратный элемент.
Пример 5. Множество $\mathbb{Z}^*_n$ с операцией умножения по модулю n есть мультипликативная абелева группа порядка $\varphi(n)$ с единицей e=1.
Пример 6. Множество алфавита A={a,b,c,…,z} с операциями сложения (склейки, конкатенации), вычитания и пустой строкой e=λ есть аддитивная не коммутативная группа ($s_1+s_2 \neq s_2 + s_1$, $s_1 \neq \lambda$ & $s_2 \neq \lambda$).

Понятие группы ввёл Эварист Галуа, изучая многочлены в 1830-е годы.

Эварист Галуа (1811 - 1832) — выдающийся французский математик, основатель современной высшей алгебры.

Циклические группы

Определение. Группа G называется *циклической*, если существует элемент $a \in G$, для которого $G=\langle a \rangle$. Элемент a называется *генератором* группы G.

Определение Порядок ord(a) элемента a группы есть порядок циклической подгруппы $\langle a \rangle$, порождённой элементом a. Если подгруппа $\langle a \rangle$ безконечна, то $ord(a) = \infty$.

Утверждение. Если G есть конечная группа и H есть подгруппа в G, то |H| делит |G|. Если $a \in G$, то ord(a) делит |G|.

Утверждение. Каждая подгруппа циклической группы циклична. Если G есть циклическая группа порядка n, то для каждого положительного делителя d для n группа G содержит в точности одну подгруппу порядка d.

Пример. Рассмотрим мультипликативную группу $\mathbb{Z}^*_{17} = \{1,2,3,4,\ldots,16\}$ порядка 16.

Подгруппа	Генераторы	Порядок
{1}	1	1
{1,16}	16	2
{1,4,13,16}	4,13	4
{1,2,4,8,9,13,15,16}	2,8,9,15	8
{1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16}	3,5,6,7,10,11,12,14	16

Генератор 4 для подгруппы {1,4,13,16}:

4 mod(17) = 4; (4*4) mod(17) = 16; (4*4*4) mod(17) = 13; (4*4*4*4) mod(17) = 1.

Кольцо

Определение. *Кольцо* $(R,\{+,\cdot\})$ есть множество R, на котором определены две операции (функции): сумма $x+y$: $R\times R\to R$ и произведение $x\cdot y$: $R\times R\to R$, для $\forall x,y,z\in R$ удовлетворяющие следующим аксиомам:

1. $(x+y)+z = x+(y+z)$
2. $x+y = y+z$
3. $\exists 0\in R\ x+0 = x$
4. $\forall x\in R\ \exists(-x)\in R\ x+(-x) = 0$
5. $(x\cdot y)\cdot z = x\cdot(y\cdot z)$
6. $(x+y)\cdot z = x\cdot z + y\cdot z$, $x\cdot(y+z) = x\cdot y+x\cdot z$.

Кольцо коммутативно, если произведение коммутативно:

7. $x\cdot y = y\cdot x$.

Замечание. Кольцо R конечно, если множество R содержит конечное число элементов. Кольцо есть коммутативная группа по сложению.

Определение. Элемент a кольца R есть *обратимый элемент*, если существует (обратный) элемент $a^{-1} \in R$, для которого $a^{-1}\cdot a=a\cdot a^{-1}=e$.

Утверждение. Множество обратимых элементов кольца R образует мультипликативную группу.

Замечание. Группа обратимых элементов кольца $\mathbb{Z}_n$ есть $\mathbb{Z}^*_n$.

Пример 1. Множество целых чисел $\mathbb{Z}$ со сложением и умножением есть коммутативное кольцо.

Пример 2. Множество $\mathbb{Z}_n$ со сложением и умножением по модулю n есть коммутативное кольцо.

Поле

Определение. *Поле* $(F,\{+,\cdot\})$ есть множество F, на котором определены две операции (функции): сумма $x+y$: $F\times F\to F$ и произведение $x\cdot y$: $F\times F\to F$, для $\forall x,y,z\in R$ удовлетворяющие следующим аксиомам:

1. $(x+y)+z = x+(y+z)$
2. $x+y = y+z$
3. $\exists 0\in F\ x+0 = x$
4. $\forall x\in F\ \exists(-x)\in F\ x+(-x) = 0$
5. $(xy)z = x(yz)$
6. $xy = yx$
7. $\exists e\in F\ \forall x\in F\ x\cdot e = x$
8. $\forall x\in F - \{0\}\ \exists x^{-1}\in F\ x\cdot x^{-1} = e$
9. $(x+y)z = xz + yz$, $x(y+z) = xy + xz$.

Замечание. Поле F *конечно*, если множество F содержит конечное число элементов. Поле есть коммутативная группа по сложению. Поле без нуля есть *коммутативная циклическая группа* по умножению.

Пример 1. Множество рациональных чисел $\mathbb{Q}$ со сложением и умножением образуют поле $(\mathbb{Q},\{+,\cdot\})$.

Пример 2. Множество вещественных чисел $\mathbb{R}$ со сложением и умножением образуют поле $(\mathbb{R},\{+,\cdot\})$.

Пример 3. Множество комплексных чисел $\mathbb{C}$ со сложением и умножением образуют поле $(\mathbb{C},\{+,\cdot\})$.

Многочлены или полиномы

Определение. *Многочленом (полиномом) степени n* от переменной *x* называют выражение вида: $p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$

где $a_n, a_{n-1}, \ldots, a_1, a_0$ – рациональные, действительные или комплексные числа называемые *коэффициентами многочлена*, a_n - *старший коэффициент* ($a_n \neq 0$), каждое слагаемое $a_i x^i$ – ($i=0\ldots n$) называют *членом многочлена*, a_0 – *свободный член*. Степень многочлена есть *n* – целое, не отрицательное число, обозначают $\deg(p(x))$. Многочлен нулевой степени есть $p(x) = a_0$. Многочлен *p(x) нормирован* если его старший коэффициент равен 1.

Замечание. Два многочлена $p(x)$ и $q(x)$ считаются равными, если равны их коэффициенты при одинаковых степенях переменной *x*.

Определение. Пусть даны два многочлена $p(x)$ и $q(x)$:

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0,\ a_n \neq 0,$$

$$q(x) = b_m x^m + b_{m-1} x^{m-1} + \ldots + b_1 x + b_0,\ b_m \neq 0,$$

тогда *суммой многочленов* $p(x)$ и $q(x)$ называют многочлен вида:

$$p(x) + q(x) = c_s x^s + c_{s-1} x^{s-1} + \ldots + c_1 x + c_0,$$

где коэффициенты c_i получены в результате суммы соответствующих коэффициентов многочленов $p(x)$ и $q(x)$ при равных степенях x, т.е. $c_i = a_i + b_i$;

произведением многочленов $p(x)$ и $q(x)$ называют многочлен вида:

$$p(x) \cdot q(x) = d_{n+m} x^{n+m} + d_{n+m-1} x^{n+m-1} + \ldots + d_1 x + d_0,$$

коэффициенты которого получаются в результате суммы коэффициентов при равных степенях x^i в выражении полученном в результате попарного произведения членов многочленов $p(x)$ и $q(x)$. Степень многочлена $p(x) \cdot q(x)$ равна сумме степеней многочленов $p(x)$ и $q(x)$, т.е. $\deg(p(x)) + \deg(q(x))$.

Пример суммы и произведения многочленов

Пример. Пусть даны многочлены:

$$p(x) = 2x^3+x^2+3, \quad q(x) = 3x^2 - x+4,$$

найти их сумму $p(x) + q(x)$ и произведение $p(x) \cdot q(x)$.

Суммой многочленов $p(x)$ и $q(x)$ будет многочлен вида:

$$\begin{array}{r} 2x^3 + x^2 + 0x + 3 \\ + \underline{0x^3 + 3x^2 - x + 4} \\ p(x) + q(x) = 2x^3 + 4x^2 - x + 7. \end{array}$$

Произведением многочленов $p(x)$ и $q(x)$ будет многочлен вида:

$$p(x) \cdot q(x) = (2x^3+x^2+3) \cdot (3x^2 - x+4) =$$

$$= 6x^5 + 3x^4 + 9x^2 - 2x^4 - x^3 - 3x + 8x^3 + 4x^2 + 12 =$$

$$= 6x^5+x^4+ 7x^3+13x^2 - 3x+12.$$

Степень многочленов:

$$\deg(p(x)) = 3,$$

$$\deg(q(x)) = 2,$$

$$\deg(p(x)+q(x)) = max\{\deg(p(x)), \deg(q(x))\} = 3,$$

$$\deg(p(x) \cdot q(x)) = \deg(p(x)) + \deg(q(x)) = 5.$$

Полиномиальные кольца или кольца многочленов

Определение. Если R есть коммутативное кольцо, то *многочлен* (*полином*) переменной x над кольцом R есть выражение вида:

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0,$$

где каждое $a_i \in R$ и $n > 0$.

Определение. Если R есть коммутативное кольцо, то *полиномиальное кольцо* (*кольцо многочленов*) R[x] есть кольцо всех полиномов переменной x с коэффициентами из R. Сложение и умножение полиномов определяется обычным образом. Сложение и умножение коэффициентов выполняется в кольце R.

Замечание. Многочлен $p(x)$ называется *неприводимым* над R, если он не является произведением двух многочленов над R ненулевой степени.

Определение. Пусть R[x] – кольцо многочленов над кольцом R и $p(x) \in R[x]$. отображение φ кольца R[x], при котором каждому многочлену $h(x)$ из R[x] соответствует остаток от деления его на $p(x)$, называют *факторизацией по модулю* $p(x)$.

Определение. Множество образов функции φ, обозначаемое $R[x]/p(x)$ есть кольцо относительно операций сложения и умножения (с последующей факторизацией по модулю $p(x)$). Оно называется факторкольцом кольца многочленов R[x] по модулю $p(x)$. Функция $\varphi: R[x] \to R[x]/p(x)$ является *гомоморфизмом колец*, т.е. для любых p(x), f(x) $\in$ R[x] выполнены:

$\varphi(p(x)+f(x)) = \varphi(p(x)) + \varphi(f(x))$ и $\varphi(p(x)\cdot f(x)) = \varphi(p(x)) \cdot \varphi(f(x))$.

Матрицы. Некоторые свойства

Определение. *Матрицей* **A** размера *nxm* над конечным множеством X называется прямоугольная таблица с *n* строками и *m* столбцами, в каждой ячейке которой записан элемент множества X. Обозначается:

$$\mathbf{A} = \begin{matrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{matrix} = (a_{ij}) \ .$$

Где a_{ij} – есть элемент матрицы **A**, записанный в *i*-й строке и *j*-м столбце, причём *i = 1…n*, *j=1…m*. Если *m=n* то матрицу **A** называют *квадратной матрицей*.

Определение. *Минором* M_j^i любого элемента a_{ij} квадратной матрицы **A** порядка *n*, называется определитель порядка *n-1*, соответствующей матрице полученной из матрицы **A** в результате вычёркивания *i*-ой строки и *j*-го столбца.

Определение. *Определителем* матрицы **A** называют число, равное и обозначаемое:

$$\Delta = |\mathbf{A}| = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{vmatrix} = \sum_{j=1}^{n} (-1)^{i+j} a_{ij} M_j^i$$

Матрицы. Некоторые свойства

На множестве матриц размера *nxm* над аддитивной полугруппой G определена операция сложения. Пусть $\mathbf{A}=(a_{ij})$, $\mathbf{B}=(b_{ij})$, тогда $\mathbf{A}+\mathbf{B}=(a_{ij}+b_{ij})$, где сложение элементов матриц выполняется в полугруппе G.

Матрицу $\mathbf{A}=(a_{ij})$ размера *nxm* над кольцом R можно умножить на матрицу $\mathbf{B}=(b_{ij})$ размера *mxr* над R, причём сложение и умножение элементов матриц выполняется в кольце R: $\mathbf{A}\times\mathbf{B}=(\sum_{k=1}^{m} a_{ik}b_{kj})$

Результатом умножения является матрица размера *nxr* над R.

Пример. Сложение матриц

$$\mathbf{A}=\begin{pmatrix}1 & 1\\ 2 & 0\end{pmatrix};\ \mathbf{B}=\begin{pmatrix}1 & -1\\ 0 & 5\end{pmatrix};\quad \mathbf{A}+\mathbf{B}=\begin{pmatrix}1 & 1\\ 2 & 0\end{pmatrix}+\begin{pmatrix}1 & -1\\ 0 & 5\end{pmatrix}=\begin{pmatrix}2 & 0\\ 2 & 5\end{pmatrix}$$

Пример 1. Умножение матриц

$$\mathbf{A}=\begin{pmatrix}1 & 1\\ 2 & 0\end{pmatrix};\ \mathbf{B}=\begin{pmatrix}1 & -1\\ 0 & 5\end{pmatrix};\quad \mathbf{A}\times\mathbf{B}=\begin{pmatrix}1 & 1\\ 2 & 0\end{pmatrix}\times\begin{pmatrix}1 & -1\\ 0 & 5\end{pmatrix}=\begin{pmatrix}1\cdot1+1\cdot0 & 1\cdot(-1)+1\cdot5\\ 2\cdot1+0\cdot0 & 2\cdot(-1)+0\cdot5\end{pmatrix}=\begin{pmatrix}1 & 4\\ 2 & -2\end{pmatrix}$$

Пример 2. Умножение матриц

$$\mathbf{A}=(1\quad 2);\ \mathbf{B}=\begin{pmatrix}3 & -1\\ 0 & 4\end{pmatrix};\quad \mathbf{A}\times\mathbf{B}=(1\quad 2)\times\begin{pmatrix}3 & -1\\ 0 & 4\end{pmatrix}=(1\cdot3+2\cdot0\quad 1\cdot(-1)+2\cdot4)=(3\quad 7)$$

Матрицы. Некоторые свойства

Для множества $M_R(n)$ всех квадратных матриц порядка n над кольцом R. Умножение матриц из $M_R(n)$ является ассоциативной внутренней операцией, выполнены также законы дистрибутивности, поэтому $M_R(n)$ – *кольцо* относительно заданных операций сложения и умножения.

Матрица $\mathbf{A} \in M_R(n)$ называется:

1. *верхнетреугольной*, если $a_{ij} = 0$, при $i>j$, где 0 – ноль кольца R;
2. *нижнетреугольной*, если $a_{ij} = 0$, при $i<j$, где 0 – ноль кольца R;
3. *диагональной*, если **A** является одновременно верхнетреугольной и нижнетреугольной.

Пример.

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 2 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 5 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 7 & 5 & 0 & 0 \\ 9 & 2 & 1 & 0 \\ 0 & 2 & 1 & 4 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 4 \end{pmatrix}$$

Матрицы. Некоторые свойства

Определение. Пусть P – поле. *Единичной матрицей* $\mathbf{E}=(e_{ij})$ из множества $M_R(n)$ называется матрица, у которой $e_{11}=e_{22}=\ldots=e_{nn}=1$ (где *1* единица поля P), а остальные элементы равны 0 (0 поля P). Для любой матрицы $\mathbf{A} \in M_R(n)$ выполняется: $\mathbf{A}\cdot\mathbf{E}=\mathbf{E}\cdot\mathbf{A}=\mathbf{A}$.

Определение. Матрицы $\mathbf{A},\mathbf{B} \in M_R(n)$ называются *взаимно обратными*, если $\mathbf{A}\cdot\mathbf{B}=\mathbf{B}\cdot\mathbf{A}=\mathbf{E}$, при этом матрицу **A** называют обратной к матрице **B** (обозначают $\mathbf{A}=\mathbf{B}^{-1}$) и наоборот.

Замечание. Матрица **A** называется обратимой, если у неё имеется обратная матрица.

Замечание. Матрица $\mathbf{A} \in M_R(n)$ обратима если её определитель не равен нулю $\det(\mathbf{A})\neq 0$, и наоборот: $\det(\mathbf{A})\neq 0 \rightarrow (\exists \mathbf{A}^{-1})\ \mathbf{A}^{-1}=\mathbf{A}$.

Пример. Множество $M_R(n)$ есть *кольцо с единицей* E относительно операций сложения и умножения. Множество всех обратимых матриц из $M_R(n)$ образует относительно умножения *группу невырожденных матриц*.

Матрицы. Некоторые свойства

Определение. *Кронекерово* или *тензорное произведение квадратных матриц* над полем P (обозначается $\otimes$) есть операция $\mathbf{A} \otimes \mathbf{B} \in M_P(rn)$, над квадратными матрицами $\mathbf{A}=(a_{ij})$, $\mathbf{B}=(b_{ij})$, $\mathbf{A} \in M_P(r)$, $\mathbf{B} \in M_P(n)$, получаемая в результате замены каждого элемента a_{ij} матрицы **A** на матрицу $a_{ij} \cdot \mathbf{B}$, т.е.: $\mathbf{A} \otimes \mathbf{B} = (a_{ij} \cdot \mathbf{B})$.

Замечание. Для тензорного умножения справедливы следующие свойства:

1. ассоциативность $\mathbf{A}\otimes(\mathbf{B}\otimes\mathbf{C})=(\mathbf{A}\otimes\mathbf{B})\otimes\mathbf{C}$;
2. дистрибутивность $(\mathbf{A}+\mathbf{B})\otimes\mathbf{C}=\mathbf{A}\otimes\mathbf{C}+\mathbf{B}\otimes\mathbf{C}$, $\mathbf{C}\otimes(\mathbf{A}+\mathbf{B})\otimes\mathbf{C}=\mathbf{C}\otimes\mathbf{A}+\mathbf{C}\otimes\mathbf{B}$;
3. для $\mathbf{A},\mathbf{C} \in M_P(n)$ и $\mathbf{B},\mathbf{D} \in M_P(r)$: $(\mathbf{A}\otimes\mathbf{B})\cdot(\mathbf{C}\otimes\mathbf{D})=\mathbf{A}\cdot\mathbf{C}\otimes\mathbf{B}\cdot\mathbf{D}$;
   из последнего свойства следует, что матрица $\mathbf{A}\otimes\mathbf{B}$ обратима, если обратимы матрицы **A** и **B**, при этом выполнено:
4. $(\mathbf{A}\otimes\mathbf{B})^{-1}= \mathbf{A}^{-1}\otimes\mathbf{B}^{-1}$.

Тензорная степень матрицы определяется индуктивно: $\mathbf{A}^{-1}=\mathbf{A}$, $\mathbf{A}^n=\mathbf{A}\otimes\mathbf{A}^{n-1}$, $n=2,3,\ldots$

Замечание. Тензорное умножение матриц не коммутативно.

Замечание. Множество всех квадратных матриц образует моноид относительно операции $\otimes$, в котором единица это матрица размеров 1*x*1, содержащая единицу поля P. Максимальная подгруппа I_G моноида G тривиальна.

Пример. Тензорное произведение матриц:

$$\mathbf{A}=\begin{pmatrix}1 & 1\\ 2 & 0\end{pmatrix};\ \mathbf{B}=\begin{pmatrix}1 & -1\\ 0 & 5\end{pmatrix};\quad \mathbf{A}\otimes\mathbf{B}=\begin{pmatrix}1 & 1\\ 2 & 0\end{pmatrix}\otimes\begin{pmatrix}1 & -1\\ 0 & 5\end{pmatrix}=\begin{pmatrix}1 & -1 & 1 & -1\\ 0 & 5 & 0 & 5\\ 2 & -2 & 0 & 0\\ 0 & 10 & 0 & 0\end{pmatrix}$$

Векторные пространства

Определение. Линейное *векторное пространство* над полем F есть множество V элементов любой природы (векторов) $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$, . . . , на котором определены две (линейные) операции: сложение $\boldsymbol{x} + \boldsymbol{y}$: $V \quad V \to V$ и
скалярное умножение $a \cdot \boldsymbol{x}$: $F \quad V \to V$ вектора $\boldsymbol{x}$ на элемент (скаляр) из F (при этом $a \cdot \boldsymbol{x} = \boldsymbol{x} \cdot a$), эти операции (сложение и умножение) $\forall \boldsymbol{x},\boldsymbol{y},\boldsymbol{z} \in V$, $\forall a,b,c \in F$ удовлетворяет следующим аксиомам:

1. $\boldsymbol{x} + \boldsymbol{y} = \boldsymbol{y} + \boldsymbol{x}$
2. $\boldsymbol{x} + (\boldsymbol{y} + \boldsymbol{z}) = (\boldsymbol{x} + \boldsymbol{y}) + \boldsymbol{z}$
3. $\exists \boldsymbol{0} \in V \; \boldsymbol{x} + (-\boldsymbol{x}) = \boldsymbol{0}$
4. $a(\boldsymbol{x} + \boldsymbol{y}) = a\boldsymbol{x} + a\boldsymbol{y}$
5. $(a + b)\boldsymbol{x} = a\boldsymbol{x} + b\boldsymbol{x}$
6. $(a \cdot b)\boldsymbol{x} = a(b\boldsymbol{x})$
7. $e \cdot \boldsymbol{x} = \boldsymbol{x}, \; e \in F$

Вектор $\boldsymbol{0}$ из V есть нулевой вектор. Вектор $-\boldsymbol{x}$ из V противоположен (обратен по сложению) вектору $\boldsymbol{x}$.

Замечание. Множество V есть аддитивная абелева группа.

Пример. Множество всех векторов из пространства $\mathbb{R}^n$ с операциями суммы двух векторов $\boldsymbol{x} + \boldsymbol{y}$ и умножением вектора на вещественное число $a\boldsymbol{x}$ есть линейное векторное пространство:

$\boldsymbol{x} + \boldsymbol{y} = (x_1+y_1, x_2+y_2, \ldots, x_n+y_n)$

$a\boldsymbol{x} = (ax_1, ax_2, \ldots, ax_n)$.

Векторные пространства

Определение. Пусть $S = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n\}$ есть конечное подмножество векторного пространства V над полем F.

1. *Линейная комбинация* векторов из S есть выражение вида $a_1\boldsymbol{v}_1 + a_2\boldsymbol{v}_2 + \ldots + a_n\boldsymbol{v}_n$, где каждое $a_i \in F$.
2. *Оболочка* для множества S (обозначается $<S>$), есть множество всех линейных комбинаций векторов из S. Заметим, что оболочка для S есть подпространство пространства V.
3. Если U есть подпространство в V, то U натянуто на S (S стягивает U), если $<S>=U$.
4. Множество векторов S *линейно зависимо* над F, если в F существуют скаляры $a_1, a_2, \ldots, a_n$, не все нули, для которых $a_1\boldsymbol{v}_1 + a_2\boldsymbol{v}_2 + \ldots + a_n\boldsymbol{v}_n = 0$. Если таких скаляров не существует, то множество векторов S *линейно независимо* над F.
5. Если векторное пространство V натянуто на линейно независимую систему векторов S (или система S стягивает пространство V), то система S называется *базисом* для V.

Утверждение. Пусть V есть векторное пространство.

1. Если V имеет конечное стягивающее множество векторов то V имеет базис.
2. Если V имеет базис, то все базисы имеют одинаковое число векторов.

Стандартный базис для V есть $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_n\}$, где $\boldsymbol{e}_i$ есть вектор с единицей e в i-ой координате и нулями в остальных.

Векторные пространства

Пример. *Задание*.

Проверить линейную зависимость векторов: $\boldsymbol{x} = \{2,3,4\}$, $\boldsymbol{y} = \{0,1,1\}$, $\boldsymbol{z} = \{1,1,1.5\}$

Решение.

Для линейной зависимости векторов $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ необходимо выполнение условия $a\boldsymbol{x} + b\boldsymbol{y} + c\boldsymbol{z} = 0$, где a, b, c – скалярные коэффициенты

1. Проверка через решение системы уравнений. Если система имеет решение ($a \neq 0 \vee b \neq 0 \vee c \neq 0$) то вектора линейно зависимы

$$\begin{cases} 2a + 0b + 1c = 0 \\ 3a + 1b + 1c = 0 \\ 4a + 1b + 1{,}5c = 0 \\ \left[\begin{array}{l} a \neq 0 \\ b \neq 0 \\ c \neq 0 \end{array}\right. \end{cases}, \Rightarrow \; a = 1; \; b = -1; \; c = -2.$$

2. Проверка через вычисление определителя матрицы, если определитель равен нулю то вектора линейно зависимы

$$\begin{vmatrix} 2 & 3 & 4 \\ 0 & 1 & 1 \\ 1 & 1 & 1.5 \end{vmatrix} = 2\begin{vmatrix} 1 & 1 \\ 1 & 1.5 \end{vmatrix} - 3\begin{vmatrix} 0 & 1 \\ 1 & 1.5 \end{vmatrix} + 4\begin{vmatrix} 0 & 1 \\ 1 & 1 \end{vmatrix} = 0$$

Вектора $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ линейно зависимы.

Реализация алгоритма Решето Эратосфена

```
// листинг C++, реализация алгоритма
// отбора простых чисел решето Эратосфена
#include <iostream>
#include <stdlib.h>
// -------------------------------------------------
static const int N = 100;
int main(int argc, char* argv[]) {
  unsigned int i;
  unsigned int* a = new unsigned int[N];
  // заполняем память единицами
  memset(a, 1, N* sizeof(unsigned int));
  for(i = 2; i < N; i++) {
    if(a[i]) {
      for(unsigned int j = i; j* i < N; j++)
        a[i*j] = 0; // если составное число
    }
  }
  // печатаем список простых чисел от 1 до N
  for(i = 1; i < N; i++) {
    if(a[i]) {
      std::cout << " " << i << std::endl;
    }
  }
  if(a)
    delete[]a;
  // ожидаем нажатия клавиши ввод
  system("pause");
  return 0;
}
// -------------------------------------------------
```

```
 Результаты отработки
программы, список
простых чисел в порядке
возрастания:
 1
 2
 3
 5
 7
 11
 13
 17
 19
 23
 29
 31
 37
 41
 43
 47
 53
 59
 61
 67
 71
 73
 79
 83
 89
 97
```

Алгоритм Эвклида

// листинг C++ кода проверки алгоритма Евклида

```
//-------------------------------------------
#include <iostream>
#include <math.h>
// Функция реализующая алгоритм Евклида
// Поиск наибольшего общего делителя (НОД)
// Greatest common divisor
int gcd(int a, int b) {
  int c;
  while (b) {
    c = a % b;
    a = b;
    b = c;
  }
  return abs(a);
}

using namespace std; // пространство имён std
// главная функция программы
int main(int argc, char* argv[]) {
  int ix, iy;
  cout << "Enter two integers:  " << endl;
  cout << "x = ";
  cin >> ix;
  cout << "y = ";
  cin >> iy;
  cout << "Greatest common divisor:  "
       << gcd(ix,iy) << endl;
  // ожидаем нажатия клавиши ввод
  system("pause");
  return 0;
}
//-------------------------------------------
```

Пример вывода результатов работы программы

```
Enter two integers:
x = 512
y = 224
Greatest common divisor:  32
```

В самом простом случае *алгоритм Евклида* применяется к паре положительных целых чисел и формирует новую пару, которая состоит из меньшего числа и разницы между большим и меньшим числом. Процесс повторяется, пока числа не станут равными. Найденное число и есть наибольший общий делитель исходной пары.

НОД взаимно простых чисел равен единице.

Класс данных, для систем линейных уравнений

```
// файл data.h, шаблонный класс data, C++
#include <vector>
using namespace std;        // пространство имён std
typedef unsigned int uint; // беззнаковый целый
template<class Type>
class data {
public: // открытые свойства и методы
  data(uint ms) {   // конструктор
    try {
      msize = ms;
      vecB.assign(msize, (Type)0);
      vecX.assign(msize, (Type)0);
      matA.resize(msize);
      for (uint i=0; i<msize; i++) matA.at(i).assign(msize, (Type)0);
    } catch (...) {
      msize = 0;
      throw;
    }
  }
  ~data() {} // деструктор
  // методы для манипулирования данными
  uint size() {return (msize);}
  void savA(uint j, uint i, Type val) {matA.at(j).at(i) = val;}
  void savB(uint j, Type val) {vecB.at(j) = val;}
  Type readA(uint j, uint i) {return (matA.at(j).at(i));}
  Type readB(uint j) {return (vecB.at(j));}
  Type readX(uint j) {return (vecB.at(j));}
protected: // защищённые свойства и методы
  typedef vector<Type> vType; // тип вектор типа Type
  typedef vector<Type>::iterator pvType; // итератор для вектора типа Type
  vector<vType> matA; // матрица коэффициентов A
  vType vecB; // вектор правых частей B
  vType vecX; // вектор искомый X
  uint msize; // размер матриц и векторов
};
```

Данный класс реализует хранилище данных для систем линейных уравнений (матрица коэффицентов `matA`, вектор правых частей `vecB`, и вектор искомых значений `vecX`. Векторы и матрицы инициализируются начальным значением 0. В результате того, что класс реализован как шаблон он может работать со следующими типами данных: `float, double, long double, complex` (при перегрузке определённых функций и операторов).

Разделение в отдельные классы данных (класс **`data`**) и алгоритмов (класс **`algorithm`**) обработки данных реализует один из фундаментальных принципов разработки программного обеЗпечения: **разделение данных и алгоритмов обработки данных**.

Метод Гаусса с выделением ведущего элемента

```
// листинг C++ кода (файл algorithm.h), шаблонный класс algorithm, C++
#define DEBUG true
#include <iostream>
#include <math.h>
#include "data.h"
// класс некоторых действий линейной алгебры
template<class Type> class algorithm : public data<Type> {
public: // открытые свойства и методы
  algorithm(uint ms) : data<Type>(ms) {} // конструктор
  bigelem() { // формирование системы с ведущим элементом
    uint k, i, j, kmax;
    Type maxval;
    for(k = 0; k < size(); k++) {
      maxval = matA.at(k).at(k); // проход по диагонали
      kmax = k;
      for(j = k + 1; j < size(); j++) { // больший элемент в столбце
        if(abs(matA.at(j).at(k)) > abs(maxval)) {
          maxval = matA.at(j).at(k);
          kmax = j;
        }
      }
      if(kmax != k) { // если требуется перестановка
        for(i = 0; i < size(); i++) { // перестановка строк матрицы A
          maxval = matA.at(k).at(i);
          matA.at(k).at(i) = matA.at(kmax).at(i);
          matA.at(kmax).at(i) = maxval;
        }
        maxval = vecB.at(k); // перестановка элементов вектора B
        vecB.at(k) = vecB.at(kmax);
        vecB.at(kmax) = maxval;
      }
    }
  }
```

```
  gauss() {  // метод исключения Гаусса
    int k, i, j;
    Type temp;
    for(k = 1; k < size(); k++) { // прямой ход
      for(i = k; i < size(); i++) {
        temp = matA.at(i).at(k - 1) / matA.at(k - 1).at(k - 1);
        for(j = 0; j < size(); j++) {
          matA.at(i).at(j) =
               matA.at(i).at(j) - temp * matA.at(k - 1).at(j);
        }
        vecB.at(i) = vecB.at(i) - temp * vecB.at(k - 1);
      }
#if (DEBUG == true)
      cout << "Forward stroke, cycle: " << k << endl;
      prnls();
#endif
    }
#if (DEBUG == true)
    cout<<"Reverse"<<endl;
#endif
    for(k = size() - 1; k >= 0; k--) { // обратный ход
      temp = vecB.at(k);
      for(j = size() - 1; j > k; j--) {
        temp -= matA.at(k).at(j) * vecX.at(j);
      }
      vecX.at(k) = temp / matA.at(k).at(k);
#if (DEBUG == true)
      cout<<"X["<<k<<"] = "<<vecX.at(k)<<endl;
#endif
    }
  }
  void prnls() { // вывод системы в поток экрана
    int j, i;
    for(j = 0; j < size(); j++) {
      for(i = 0; i < size(); i++) {
        if(matA.at(j).at(i) < 0)
          cout<<matA.at(j).at(i)<<"*X["<<i <<"]";
        else
          cout<<"+"<<matA.at(j).at(i)<<"*X[" <<i<<"]";
      }
      cout<<"="<<vecB.at(j)<<endl;
    }
    return;
  }
};
```

Данный класс реализует **метод последовательного исключения Гаусса с выделением ведущего элемента** (`bigelem()` и `gauss()`), или без него (`gauss()`). Реализация несколько «облегчена» для сокращения размера листинга. При этом листинг содержит отладочный код который будет полезен для изучения алгоритма. В данной реализации массивы `matA` и `vecB` изменяются в ходе вычислений, что необходимо учитывать при использовании класса. У такого подхода есть положительная сторона – экономия памяти и отрицательная – изменение исходных данных. Отметим, что реализации алгоритма отсутствует проверка возможности возникновения исключительных ситуаций.

Пример решения системы линейных уравнений м. Гаусса

// листинг C++ кода проверки классов data и algorithm

```
#include <iostream>
#include <stdlib.h>
#include "algorithm.h"
using namespace std;
int main(int argc, char* argv[]) {
  cout << "Size of the system of equations:  ";
  int sz;
  float val;
  cin>>sz;  // порядок системы
  // объект класса шаблонного algorithm
  algorithm<float> *la = new algorithm<float>(sz);
  // ввод системы линейных уравнений
  cout<<"Enter lines of system of linear equations:  "<<endl;
  for (int j = 0; j < sz; j++) {
    cout<<"A["<<j<<",1]+...+A["<<j<<", "
        <<(sz-1)<<"]=B["<<j<<"]"<<endl;
    for (int i = 0; i < sz; i++) {
      cin>>val;
      la->savA(j, i, val);
    }
    cin>>val;
    la->savB(j, val);
  }
  cout<<"Linear system:"<<endl;
  la->prnls();
  la->bigelem();
  cout<<"Linear system:"<<endl;
  la->prnls();
  la->gauss();
  delete la;
  system("pause");
  return 0;
}
```

Пример вывода результатов работы программы

```
Size of the system of equations:
3
A[0,1]+...+A[0, 2]=B[0]
1 2 3 4
A[1,1]+...+A[1, 2]=B[1]
5 8 9 7
A[2,1]+...+A[2, 2]=B[2]
7 2 3 5
Linear system:
+1*X[0]+2*X[1]+3*X[2]=4
+5*X[0]+8*X[1]+9*X[2]=7
+7*X[0]+2*X[1]+3*X[2]=5
Linear system:
+7*X[0]+2*X[1]+3*X[2]=5
+5*X[0]+8*X[1]+9*X[2]=7
+1*X[0]+2*X[1]+3*X[2]=4
Forward stroke, cycle: 1
+7*X[0]+2*X[1]+3*X[2]=5
-1.19209e-07*X[0]+6.57143*X[1]+6.85714*X[2]=3.42857
-4.47035e-08*X[0]+1.71429*X[1]+2.57143*X[2]=3.28571
Forward stroke, cycle: 2
+7*X[0]+2*X[1]+3*X[2]=5
-1.19209e-07*X[0]+6.57143*X[1]+6.85714*X[2]=3.42857
-1.36054e-08*X[0]-9.07027e-08*X[1]+0.782608*X[2]=2.3913
Reverse
X[2] = 3.05556
X[1] = -2.66667
X[0] = 0.166667
Press any key for continue . . .
```

Выделение ведущего элемента позволяет повысить точность решения и устойчивость метода Гаусса.

Список литературы

1. Набебин А. А. Дискретная математика. – М.: Научный мир, 2010. 512 с.: ил.
2. Коробейников А. Г, Гатчин Ю. А. Математические основы криптологии. Учебное пособие. СПб: СПб ГУ ИТМО, 2004. – 106 с.: ил.
3. Федоровский К. Ю. Алгебра. Введение в теорию групп. Курс лекций по дисциплине «Алгебра». Учебное пособие. М.: МГТУ им. Н.Э. Баумана, 2012. – 56 с.: ил.
4. Фомичёв В. М. Методы дискретной математики в криптологии. – М.: Диалог-МИФИ, 2010. – 424 с.
5. Андерсон Дж. А. Дискретная математика и комбинаторика. : Пер с англ. – М. Издательский дом «Вильямс», 2004. – 960 с.: ил. – Паралл. тит. англ.
6. Прата С. Язык программирования C++. Лекции и упражнения, 6-е изд.: Пер. с англ. – М.: ООО «И.Д. Вильямс», 2013. – 1248 с.: ил. – Парал. тит. англ.
7. Либерти Д., Джонс Б. Освой самостоятельно C++ за 21 день, 5-е издание.: Пер. с англ. – М.: издательский дом «Вильямс», 2012. – 768 с.: ил.

Билибин Иван Яковлевич (1876-1942). Илья Муромец и Святогор. 1900-е гг.