МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Основы работы и программирования в системе MathCad

*Рекомендовано в качестве учебного пособия
Редакционно-издательским советом
Томского политехнического университета*

Составитель
Е.А. Кочегурова

Издательство
Томского политехнического университета
2012

УДК 519.6 (075.8)

Основы работы и программирования в системе MathCad: учебное пособие / сост. Е.А. Кочегурова; Томский политехнический университет. – Томск: Изд-во Томского политехнического университета, 2012 – 25 с.

Учебное пособие предназначено для освоения приемов работы в системе проведения математических и инженерных расчетов MathCad. Особое внимание уделено изучению основ программирования собственных задач пользователя.

Оглавление

1. МАТЕМАТИЧЕСКИЙ ПАКЕТ MATHCAD

MathCad – уникальный математический пакет для работы с уравнениями, числами, текстом, и графиками. Особенностями работы в MathCad является:

1. Математические выражения записываются в общепринятом математическом виде $\frac{a}{b}$, $\int_0^1 x^2 dx$, $\sum_i x_i$. Решение квадратного уравнения, например, имеет вид:

$$x = \frac{\sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}.$$

2. Режим «живой математики»: процессы создания и выполнения программы происходят одновременно. При вводе математических выражений, графиков одновременно получаем численное решение и построенный график.

3. Средства символьной математики позволяют получать в аналитическом виде интегралы, производные, аналитические решения уравнений, преобразований и т.д.

4. MathCad – полноценное Windows – приложение: OLE – технология, средства Internet в основной оболочке Mathcad, средства анимации.

Система MathCAD существует в нескольких основных вариантах:

- MathCAD Standard – идеальная система для повседневных технических вычислений. Предназначена для массовой аудитории и широкого использования в учебном процессе;
- MathCAD Professional – промышленный стандарт прикладного использования математики в технических приложениях. Ориентирована на математиков и научных работников, проводящих сложные и трудоемкие расчеты.
- MathCAD Professional Academic – пакет программ для профессионального использования математического аппарата с электронными учебниками и ресурсами.

1. Общие сведения о ППП MATHCAD

Рабочий лист Mathcad позволяет объединять уравнения, текст и графику. Это облегчает наблюдение за наиболее сложными вычислениями и позволяет представить результаты в двух или трехмерных графиках.

Основа Mathcad - полноэкранный числовой и символический калькулятор. Это пустая рабочая область, которая появляется на экране монитора, когда открывается новый Mathcad документ. Чтобы использовать Mathcad как калькулятор, необходимо напечатать арифметическое выражение, а затем - знак равенства:

$$35+\frac{7}{13}=35.538$$

Кроме обычных для Windows- приложений панелей Стандартная и Форматирования, рабочий экран Mathcad содержит математическую панель (палитру). С помощью кнопок математической панели можно выводить на экран отдельные математические панельки, которые включают большинство математических операций.

Арифметическая панель

Панель графики

Панель вычислений

Панель греческого алфавита

Панель отношений и логики

Панель векторов и матриц

Панель программирования

Панель символьных вычислений

Использование математических панелей позволяет просто формировать и вычислять самые разнообразные математические выражения:

$$\frac{\sqrt{\sin(\frac{\pi}{5})+4}}{(\ln(e^5)+3)^2}=0.033$$

Mathcad позволяет работать с матрицами, выполняет операции с комплексными числами и все математические операции которые необходимы в инженерных расчетах. Ниже приведено несколько характерных примеров:

$$\begin{bmatrix} 3 & -1 \\ -5 & 7 \end{bmatrix}^{-1}=\begin{bmatrix} 0.438 & 0.063 \\ 0.313 & 0.188 \end{bmatrix}$$

$$\int_0^{\frac{3}{2}}\frac{1}{1+x^2}\,dx=0.983$$

$$\frac{3+7\cdot i}{8-5\cdot i}=-0.124+0.798\cdot i$$

Иногда необходимо видеть точные ответы, выраженные в виде обыкновенных дробей и радикалов вместо десятичных чисел. В Панели отношений и логики имеется "символический знак равенства" → для получения точных ответов. Например:

$$\frac{\sqrt{\sin\left(\frac{\pi}{5}\right)+4}}{\left(\ln\left(e^{5}\right)+3\right)^{2}} \rightarrow \frac{1}{128}\cdot\sqrt{\sqrt{2}\cdot\sqrt{5-\sqrt{5}}+16}$$

$$\int_{0}^{\frac{3}{2}}\frac{1}{1+x^{2}}dx \rightarrow \operatorname{atan}\left(\frac{3}{2}\right)$$

$$\begin{bmatrix} 3 & -1 \\ -5 & 7 \end{bmatrix}^{-1} \rightarrow \begin{bmatrix} \frac{7}{16} & \frac{1}{16} \\ \frac{5}{16} & \frac{3}{16} \end{bmatrix}$$

$$\frac{3+7i}{8-5i} \rightarrow \frac{-11}{89}+\frac{71}{89}\cdot i.$$

Если необходимо оценить выражение для многих различных операций, требуется определить собственные переменные и функции.

1.2. Особенности создания основных математических выражений

Каждое уравнение Mathcad, текстовый параграф или график на рабочем листе - отдельный объект названный областью. Можно выбирать одиночную область, щелкая указателем мышки на математическое выражение или текст в вашем рабочем листе. Область высвечивается с помощью тонкого прямоугольника. Если переместить курсор на одну из граней этого прямоугольника, курсор изменяется на маленькую руку - который позволяет перемещать данную область. Если нажимать кнопку мыши внутри математической области, видны синие линии выбора, которые показывают выбранную часть математического выражения. Если нажать внутри текстовой области, то будут видны черные рамки в каждом углу и середине каждой линии. Эти рамки позволяют изменять размеры текстовых областей, которые можно создавать в документе.

Палитры математических операторов позволяют формировать математические выражения Mathcad. Место записи математического выражения необходимо отметить левой клавишей мыши (появится красное перекрестие). Затем с помощью клавиатуры и операторов математических панелей записать нужное математическое выражение. Так как Mathcad является полноценным Windows- приложением, то работа в нем не отличается от остальных приложений.

1.3. Определение переменных

Часто необходимо определять переменные, которые можно использовать в последующих вычислениях. Если в рабочей области напечатать двоеточие **[:]** или нажать клавишу присвоения на Арифметической Палитре, на дисплее появиться знак " **:** = ". Оператор присвоения в Mathcad используется для определения переменных и функций. Чтобы впоследствии увидеть чему равняется переменная или функция, необходимо только напечатать имя этой переменной и знак равно "=". Допустим, необходимо определить площадь круга для различных значений радиуса. Пусть переменная **r** задает конкретное значение :

r:=7

При вычислении площади, получим числовой ответ:

$$\pi \cdot r^2 = 153.938$$

Если требуется вычислить площадь круга при другом значении радиуса можно заменить значение **7** на нужное число, и нажать кнопку мыши снаружи области определения. Появится новое значение площади. Это - "ЖИВАЯ МАТЕМАТИКА" в действии.

1.4. Определение функции

Удобно определить функцию площади, например:

$$area(r) := \pi \cdot r^2$$

и затем использовать в разных местах рабочего документа.

Также возможно формировать связанные функции. Например, сторона квадрата с той же самой площадью, как и у круга, радиусом **r**:

$$side(r) := \sqrt{area(r)}$$

Синтаксис, используемый для определения функции в Mathcad, приведён ниже. Пусть имеем функцию

f(x):=x²

Вычислим значение функции **f(3**):

x:=3 f(x)=9

Можно определить другую функцию через **f(x):**

g(y):=f(y)+6

g(x)=15

Когда изменяются какие-либо переменные, Mathcad немедленно повторно вычисляет функцию.

$$f(x) := \frac{\sin(x)}{\frac{x}{a}} \qquad f(10) = -0.218$$

В Mathcad есть возможность производить арифметические действия, используя встроенные функции и математические операторы.

Список встроенных функций Mathcad, находится в меню Вставка (Insert), команда Функция (Function). Равнозначное действие - Функциональная клавиша Вставка функции на инструментальной панели. 
Можно также вести имя любой встроенной функции с клавиатуры.
Например:

$$\log(1347.2)\cdot\sin\left(\frac{3}{5}\cdot\pi\right)=2.976$$

Значение числа отображаемых знаков после запятой можно изменять командой Число (Number) из меню Формата (Format).

1.5. Определение функции в диапазоне

Часто на практике возникает потребность в определении дискретных значений переменных и функций, лежащих в определённом диапазоне. Оператор диапазона "**..** " может быть введён с помощью кнопки Range Variable, которая находится в Палитре матриц: 
Можно теперь использовать переменную диапазона, как и любую другую переменную. Mathcad создает таблицу вывода - вертикальный ряд позиций, которые содержат числа. Выводить данные в виде таблицы можно, если напечатать с клавиатуры **z =** , **f (z) =** и так далее.

$$f(z):=z^2 \qquad z:=-6,-4..\ 2$$

z	f(z)	f(z)+10
-6	36	46
-4	16	26
-2	4	14
0	0	10
2	4	14

В этом примере кроме начального и конечного значения (-6, 2) диапазона для переменной **z** , задано значение, идущее после начального (-4).

1.6. Вычисление суммы, интеграла, производной

Сумма, операторы интегралов и производных находятся в Панели Вычисления. Для формирования математического выражения необходимо выбрать кнопку на панели, затем заполнить в появившейся заготовке недостающие элементы (пределы интегрирования и суммирования, переменные и функции и др.). Например:

$$\sum_{n=0}^{10}\frac{1}{n!}=2.7182818$$

$$\int_0^1 \frac{1}{1+x^2}\,dx = 0.785$$

1.7. Определение векторов и матриц

Примеры, описанные выше, включали одиночные числа или скалярные величины. Однако, Mathcad имеет много мощных свойств и функции, для работы с массивами чисел, типа векторов и матриц. Создание вектора или матрицы в Mathcad заключается в выборе размерности массива и вводе его элементов. Сформировать массив можно с помощью клавиши создания массива из панели инструментов: 

Чтобы обращаться и работать с конкретным элементом вектора, необходимо использовать оператор нижнего индекса:  .

На Панели матриц и векторов находятся также некоторые операторы работы с матрицами: транспонирование, получение обратной матрицы, вычисление определителей и др.

2. СОЗДАНИЕ ПРОГРАММЫ - ФУНКЦИИ

1. Каждая программа-функция MathCAD имеет *оригинальное имя*, используя которое осуществляется обращение к этой программе-функции. Через это же имя (*и только через это имя*) “возвращается” в рабочий документ результат выполнения программы-функции.

После имени программы-функции идет *список формальных параметров*, заключенный в круглые скобки. Через формальные параметры "внутрь" программы-функции “передаются” данные необходимые для выполнения вычислений внутри программы . В качестве формальных параметров могут использоваться имена простых переменных, массивов и функций. Формальные параметры отделяются друг от друга запятой.

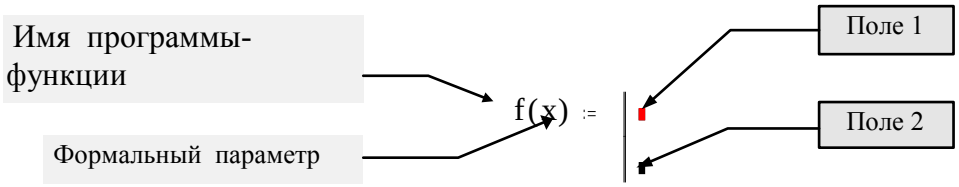
Замечание. Программа-функция может не иметь формальных параметров и тогда данные передаются через имена переменных, определенных выше описания программы-функции. ❖

Тело программы-функции включает любое число операторов локальных операторов присваивания, условных операторов и операторов цикла, а также вызов других программ-функций и функций пользователя.

2.1. Порядок описания программы-функции MathCAD

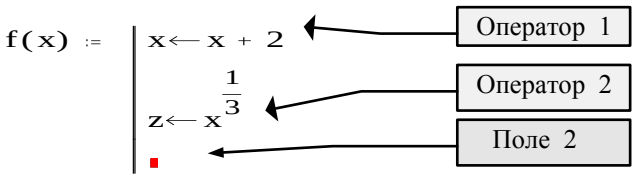
Для ввода в рабочий документ описания программы-функции необходимо выполнить следующие действия:

- ввести имя программы-функции и список формальных параметров, заключенный в круглые скобки ;
- ввести символ ” - на экране отображается как : =;
- открыть наборную панель **Программирования** и щелкнуть кнопкой “Add line” . На экране появится вертикальная черта и вертикальный столбец с двумя полями ввода для ввода операторов, образующих тело программы-функции



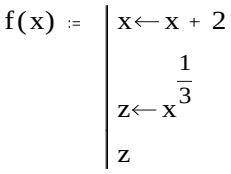
- перейти в поле 1 (щелкнув на нем мышью или нажав клавишу [Tab]) и ввести первый оператор тела программы-функции. Так как самое нижнее поле всегда предназначено для определения возвращаемого программой значения, то поля ввода для дополнительных операторов открываются с помощью щелчка на кнопке “Add line” панели программирования. При этом поле ввода добавляется внизу выделенного к этому моменту оператора. Для

удаления того или иного оператора или поля ввода из тела программы-функции, нужно заключить его в выделяющую рамку и нажать клавишу [Delete];



- заполнить самое нижнее поле ввода (поле 2), введя туда выражение, определяющее возвращаемое через имя программы-функции значение .

В приведенном примере формальным параметром является простая переменная x , тело программы включает два локальных оператора присваивания (см. следующий пункт) и значение переменной z определяет возвращаемый через имя функции результат выполнения программы-функции.



Локальный оператор присваивания. Для задания внутри программы значения какой-либо переменной используется так называемый локальный оператор присваивания, имеющий вид:

< имя - переменной > ← < выражение >

Внимание ! Использование "обычного" оператора присваивания (обозначается : =) в теле программы-функции приводит к синтаксической ошибке.

2.2.Обращение к программе-функции MathCAD

Для выполнения программы-функции необходимо обратиться к имени программы-функции с указанием *списка фактических параметров* (если в описании программы присутствует список формальных параметров), т.е.

< имя - программы > (список фактических параметров)

Фактические параметры указывают при каких конкретных значениях осуществляются вычисления в теле программы. Фактические параметры отделяются друг от друга запятой.

Очевидно, что между фактическими и формальными параметрами должно быть ***соответствие по количеству, порядку следования и типу***. Последнее соответствие означает:

- если формальным параметром является простая переменная, то в качестве фактического может использоваться константа, переменная, арифметическое выражение;
- если формальным параметром является вектор или матрица, то фактическим должен быть вектор или матрица;
- если формальным параметром является имя встроенной функции или другой программы, то и фактическим параметром должен являться тот же объект.

Замечание. Обращение к программе-функции должно находиться после описания программы-функции и ***к моменту обращения фактические параметры должны быть определены.*** ❖

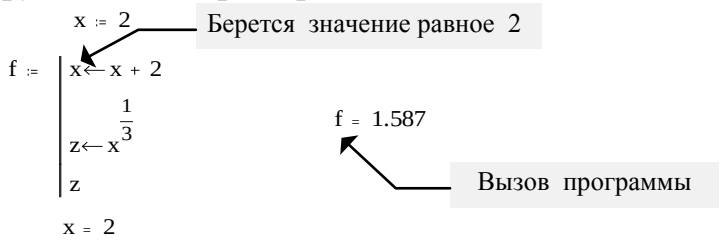
Пример 2.1. Обращение к программе f(x) может иметь следующий вид:

$x := 2 \quad f(x) = 1.587 \quad f(-3.23) = 0.536 + 0.928i$

$z := f(x + 4.5) \quad z = 2.041$

Заметим, что переменная z никак не связана с “локальной” переменной z, используемой внутри тела программы-функции.

Замечание. Передать данные внутрь программы-функции можно используя внутри программы переменные, определенные до описания программы-функции. Например :



Хотя значение переменной x изменилось внутри программы-функции, вне описания программы-функции эта переменная сохранила свое прежнее значение. ❖

Замечание. Имена фактических параметров при вызове программы-функции могут не совпадать с именами ее формальных параметров. ❖

2.3. Элементы программирования в MathCad

2.3.1. Программирование в программе-функции линейных алгоритмов

Напомним, что **п**од линейным алгоритмом понимается вычислительный процесс, в котором необходимые операции выполняются строго последовательно. Операторы, реализующие этот алгоритм в теле программы - функции также размещаются последовательно и выполняются все, начиная с первого оператора и кончая последним.

Пример. Оформим в виде программы-функции вычисление корней квадратного уравнения $ax^2 + bx + c = 0$ по формуле

$$x_{1,2} = \frac{-b \mp (b^2 - 4ac)^{1/2}}{2a}$$

Для этого введем следующее описание программы-функции

$$qq1(a,b,c,sig1) := \left| \begin{array}{l} d1 \leftarrow b^2 - 4 \cdot a \cdot c \\ d2 \leftarrow 2 \cdot a \\ d3 \leftarrow \left(-b + sig1 \cdot \sqrt{d1}\right) \\ \dfrac{d3}{d2} \end{array} \right.$$

Программа qq1 имеет четыре параметра: смысл первых трех понятен, а четвертый определяет знак перед корнем квадратным - задавая Sig1=1, получаем корень x_1; Sig1= - 1, получаем корень x_2. Программа реализует ***линейный алгоритм - все операторы выполняются всегда строго последовательно.*** ■

2.3.2. Программирование в программе-функции разветвляющихся алгоритмов

Напомним, что в разветвляющихся алгоритмах присутствует несколько ветвей вычислительного процесса. Выбор конкретной ветви зависит от выполнения (или невыполнения) заданных условий на значения переменных алгоритма.

Пример. Переменная y задается следующим выражением

$$y(x) = \begin{cases} x^2, & x \le 0; \\ \sqrt{x}, & x > 0. \end{cases}$$

Видно, что алгоритм вычислений содержит две ветви и выбор зависит от значения переменной x.

Для программирования разветвляющихся алгоритмов в MathCAD имеется *условная функция if и условный оператор*. Используя эти конструкции можно "изменить" последовательное выполнение операторов. В этих конструкциях могут использоваться следующие новые понятия.

Выражения отношений

Эти выражения используются для сравнения двух арифметических выражений между собой. Выражение отношений записываются в виде :

< выр. A > < знак отношения > < выр. B> ,

где в качестве знака отношения выступают символы, приведенные в таблице 1. Если заданное отношение выполняется, то выражение отношений принимает значение равное 1 ("истина"), в противном случае - 0 ("ложь").

Знак отношения	Вводимые символы
=	[Ctrl] + [=]
<	[<]
>	[>]
≥	[Ctrl] + [0]
≤	[Ctrl] + [9]
≠	[Ctrl] + [3]

Пример Вычисление выражения отношений

x := 6 x≤2 = 0 ← Результат вычисления выражения отношений

В отличие от языков программирования можно сразу в одном выражении проверять несколько условий путем добавления знаков отношений и арифметических выражений. Эту возможность иллюстрирует следующий пример.

Пример x := 6 2≤x≤8 = 1 ■

Логические операции. Определены две логических операции, которые ставятся между выражениями отношений.

Логическая операция ИЛИ . Обозначается знаком + и записывается в виде

< логич.выр.1 > + < логич.выр. 2>

Результат операции равен 0, если оба логических выражения равны 0 и равен 1 для всех остальных значений логических выражений.

Логическая операция И . Вводится знаком * (в тексте это точка) и записывается в виде

< логич.выр.1 > . < логич.выр. 2>

Результат равен 1, если оба логических выражения равны 1 и равен 0 для всех остальных значений логических выражений (сравните с логическим оператором ИЛИ).

Логическое выражение. Логическим выражением называется конструкция, составленная из выражений отношений, знаков логических операций и круглых скобок. Значение логического выражения вычисляется слева направо с учетом известного правила о приоритете операций. Список приоритетов (по их убыванию):

* круглые скобки ;
* логическая операция И;
* логическая операция ИЛИ.

Рекомендация: для однозначного вычисления логического выражения используйте круглые скобки.

2.3.3. Условная функция if

Эта функция записывается в виде (символы if вводятся с клавиатуры) :
if (< логич. выраж. > , < ариф.выраж.1 ДА> , < ариф.выраж.2 НЕТ>)

Правило вычисления условной функции if : если логическое выражение равно 1, то функция принимает значение равное значению арифметического выражения 1 ; если логическое выражение равно 0, то функция принимает значение равное значению арифметического выражения 2.

Условная функция используется в арифметических выражениях, стоящих в правой части локального оператора присваивания.

Пример. Реализуем алгоритм вычисления функции

$$y(x)=\begin{cases} x^2, & x \le 0; \\ \sqrt{x}, & x > 0. \end{cases}$$

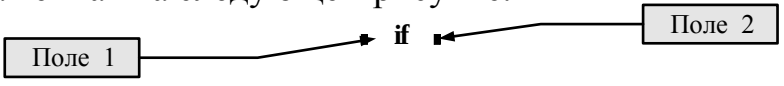
$$y(x) := \left| z \leftarrow \text{if}\left(x \le 0, x^2, \sqrt{x}\right) \right| \\ \left| z \right.$$

Обращение к этой программе-функции в тексте документа

$y(2) = 1.414$ $y(-2) = 4$ ■

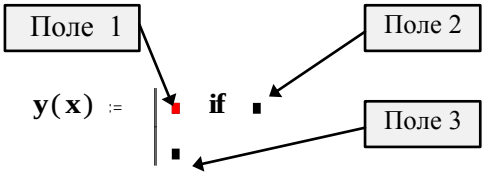
2.3.4. Условный оператор

Этот оператор используется только в теле программы-функции и для его ввода необходимо щелкнуть на кнопке if панели программирования или клавиши [}]. На экране появляется конструкция с двумя полями ввода, изображенная на следующем рисунке.



В поле 2 вводится логическое выражение (в простейшем случае это выражение отношений). В поле 1 вводится выражение (как правило, арифметическое), значение которого используется, если проверяемое логическое выражение принимает значение 1.

Условный оператор может находиться только внутри тела программы-функции. Например :



В поле 3 задается выражение, значение которого используется, если логическое выражение равно 0. Для ввода в поле 3 необходимо :

- заключить это поле в выделяющую рамку;
- щелкнуть на кнопке “otherwise” панели программирования;
- в оставшемся поле введите соответствующее выражение.

Пример. Составим программу-функцию, вычисляющую функцию

$$y(x)=\begin{cases} x^2, & x \le 0; \\ \sqrt{x}, & x > 0. \end{cases}$$

Для этого введем описание следующей программы-функции:

$$y(x) := \begin{cases} x^2 & \text{if } x \le 0 \\ \sqrt{x} & \text{otherwise} \end{cases}$$

Обращение к этой программе-функции имеет вид

$y(2) = 1.414$ $y(-2) = 4$

Таким образом, выражение, стоящее перед словом otherwise выполняется только в том случае, если не выполнено заданное перед этим условием.

В программе можно использовать несколько следующих друг за другом условных операторов с одним выражением перед словом otherwise.

Пример Составим программу-функцию для вычисления переменной z по формуле

$$z(t)=\begin{cases} t^3, t < -3; \\ \ln(t), t > 4; \\ t^2, -3 \le t \le 4. \end{cases}$$

В рабочий документ введем описание следующей программы-функции

$$z(t) := \begin{cases} t^3 & \text{if } t \le -3 \\ t^2 & \text{if } -3 \le t \le 4 \\ \ln(t) & \text{otherwise} \end{cases}$$

Заметим, что функция z(t) получит значение ln(t) только тогда, *когда не выполняется условие записанные в двух вышестоящих строках.*

Обращение к этой программе - функции имеет вид

z(2) = 4 ■

Если в поле 3 ввести оператор без слова otherwise, то этот ***оператор будет выполняться всегда*** *вне зависимости от выполнения выше заданных условных операторов.*

2.3.5. Программирование в программе-функции циклических алгоритмов

Напомним, что циклические алгоритмы (или проще циклы) содержат повторяющиеся вычисления, зависящие от некоторой переменной. Такая переменная называется *параметром цикла*, а сами повторяющиеся вычисления составляют *тело цикла*.

Классификация циклов. Циклы можно условно разделить на две группы:

- циклы типа арифметической прогрессии;
- итерационные циклы.

Характерной чертой первой группы циклов является то, что количество повторений тела цикла можно определить до начала выполнения программы, реализующей цикл, т.е. априори.

Для итерационных циклов нельзя априори определить количество повторений тела цикла. Это обусловлено тем, что окончание таких циклов определяется не выходом параметра цикла за конечное значение, а более сложными условиями. Это иллюстрирует следующий пример.

Пример Вычислить значение $x=\sqrt{a}$, используя итерационную процедуру

$$x_n=0.5(x_{n--1}+a/x_{n-1}), \qquad n=1,2,3,.., \ x_0=a.$$

В качестве приближенного значения корня квадратного берется такое значение x_n, которое удовлетворяет условию

$$|x_n - x_{n-1}| \le \varepsilon,$$

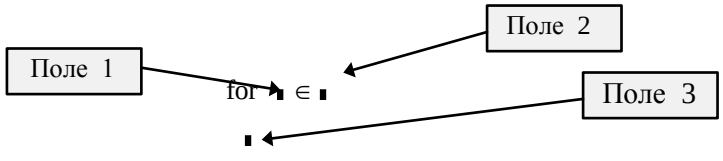
где ε - заданная точность вычисления.

Видно, что задав исходные данные, например, a = 9, $\varepsilon = 10^{-3}$, нельзя, не выполняя итерационные вычисления, определить количество повторений тела цикла. ■

Программирование цикла типа арифметической прогрессии

Для программирования таких циклов используется оператор цикла for. Для ввода такого оператора необходимо выполнить следующие действия:

- щелкнуть на кнопке **for** наборной панели **Программирования**. На экране появятся поля ввода, изображенные на рис.



Структура оператора цикла **for**

- в поле ввода 1 введите имя параметра цикла;
- в поле ввода 2 ввести диапазон значений параметра цикла, используя для этого дискретный аргумент ;
- в поле ввода 3 вводятся операторы, составляющие тело цикла. Если одной строки недостаточно, то дополнительные поля ввода (дополнительные строки) создаются щелчком на кнопке “Add line” в панели программирования и тогда слева от тела цикла появляется вертикальная черта.

Пример Для x меняющего от -2 до 2 с шагом 0.5 вычислить значение $f(x) = e^{-x} \cdot \mathrm{Cos}(2x)$ и сформировать из этих значений вектор y, т.е. $y_1 = f(-2)$, $y_2 = f(-1.5)$ и т.д.

В этом примере количество повторений определяется по формуле $\left[\frac{x_k - x_0}{d}\right] + 1$, где x_k , x_0 - конечное и начальное значение параметра цикла, d - шаг его изменения. Подставив значения, получаем (2 - (-2))/0.5+1=9.

Описание программы-функции имеет вид

```
form_tab(x0,xk,d) := | i ← 1
                     | for x ∈ x0, x0 + d .. xk
                     |     | z ← exp((-x))·cos(2·x)
                     |     | y_i ← z
                     |     | i ← i + 1
                     | y
```

В этом варианте описания программы-функции формальные параметры используются для задания диапазона изменения параметра цикла (переменная x). Для изменения индекса y элемента массива y вводится переменная i целого типа внутри программы-функции. Обращение к описанной программе-функции может иметь вид

```
z := form_tab(-2,2,0.5)
```
■

Замечание. Если значение индексов y элементов массива меняется начиная с 1 (как в этом примере), то начальное значение индекса необходимо установить равным 1 (для этого обратиться к пункту **MATH** команде **Built-in Variables,** а затем в поле ввода **Origin** ввести ***значение 1*** (вместо установленного по умолчанию значения 0). ❖

Пример. Немного изменим условия примера 3.5 , а именно : значения x, для которых вычисляется функция y(x) задается вектором x, имеющим n проекций. Для каждой проекции вектора x вычислить значение функции

$f(x)=e^{-x}$ Cos **(** 2x) и записать это значение в соответствующую проекцию вектора y.

Описание программы-функции, решающую эту задачу имеет вид :

```
form_tab1(n,x) := | for i ∈ 1..n
                  |     z ← exp(-x_i)·cos(2·x_i)
                  |     y_i ← z
                  | y
```

Здесь формальным параметром являются : n - число элементов вектора x ; x - вектор, состоящий из n элементов.

Обращение к описанной программе-функции можно осуществить с помощью конструкций:

```
m := 5      i := 1..5      z_i := √i
y := form_tab1(m,z)
```

Формирование фактического параметра - вектор z

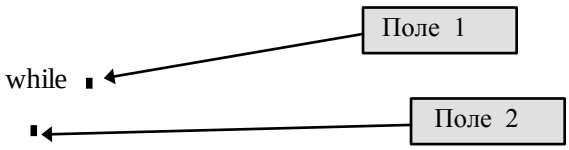
$$z = \begin{bmatrix} 1 \\ 1.414 \\ 1.732 \\ 2 \\ 2.236 \end{bmatrix} \qquad y = \begin{bmatrix} 1.516 \\ 3.84 \\ 5.169 \\ 3.696 \\ 1.702 \end{bmatrix}$$

Сформированный вектор y

Программирование итерационных циклов

Для программирования таких циклов используется оператор цикла while. Для ввода этого оператора необходимо выполнить следующие действия:

- щелкнуть на кнопке while панели **Программирования.** На экране появляются элементы, показанные на рис.



Структура оператора цикла while

- в поле 1 ввести условие выполнения цикла;
- в поле 2 ввести операторы тела цикла. В теле цикла должны присутствовать операторы делающие условие цикла ложным иначе цикл будет продолжаться бесконечно.

Оператор цикла while выполняется следующим образом: обнаружив оператор while, MathCAD проверяет указанное условие. Если оно истинно, то выполняется тело цикла и снова проверяется условие. Если оно ложно, то цикл заканчивается.

Пример. Составим программу-функцию, реализующую итерационную процедуру приближенного вычисления корня квадратного.

```
sqroot(a, eps) := | xc ← 1.0e10
                  | xn ← a
                  | while abs(xn - xc) > eps
                  |     xc ← xn
                  |     xn ← (xc + a/xc)/2
                  | xn
```

Как видно из текста программы-функции нет необходимости хранить в памяти все приближенные решения x_0, x_1, x_2 , ... , и т.д. Достаточно хранить предыдущее (“старое”) значение xc и последующее (“новое”) значение xn.

Обращение к описанной программе будет иметь вид

sqroot(9, 0.0001) = 3 sqroot(25, 0.0001) = 5

sqroot(123, 0.0001) = 11.091 ■

К сожалению организация итерационного цикла с помощью оператора while , без дополнительных средств контроля может привести к зацикливанию. Например, задав при обращении к программе eps < 0 получаем зацикливание.

Поэтому в MathCAD имеется специальный оператор break, который позволяет выйти из цикла или приостановить исполнение программы при выполнении заданного в операторе break условии. Для ввода оператора break необходимо щелкнуть на кнопке **break** панели **Программирования** (нельзя вводить этот оператор с клавиатуры по символам). Оператор **break** используется в левом поле ввода условного оператор **if**, а в правом размещается условие, при выполнении которого происходит прекращение работы цикла или программы, в нижнем поле - оператор, выполняемый если условие не выполнено. Поэтому первоначально вводится оператор **if ,** а затем заполняются поля этого оператора.

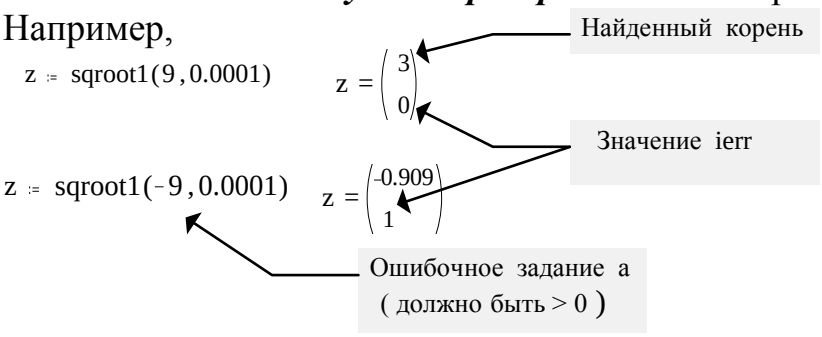
Следующий пример показывает написание не зацикливающей программы с оператором break.

Пример. Составим программу-функцию, реализующую итерационную процедуру вычисления корня квадратного (см. пример 2.9) без зацикливания. Описание такой программы-функции имеет вид :

```
sqroot1(a, eps) := | xc ← a
                   | for i ∈ 1 .. 1000
                   |     xn ← (xc + a/xc) / 2
                   |     break if |xn - xc| ≤ eps
                   |     xc ← xn
                   |     ierr ← 1 if i = 1000
                   |            0 otherwise
                   |     x0 ← xn
                   |     x1 ← ierr
                   | x
```

В этой программе число повторений тела ограничено 1000. Если за это число итераций приближенное значение корня с заданной точностью не найдено, то параметр **ierr** получает значение 1, что говорит об ошибке вычислительного процесса (если были выполнены 1000 итераций) . ***Так как через имя программы передается значение только одной переменной, то для передачи двух значений xn, ierr используется вектор, проекции которого формируются внутри программы.***

Значение ierr нужно проверять после обращения к программе sqroot1. Например,



■

2.3.6. Возможные использования условного оператора IF

Условный оператор if может использоваться для реализации достаточно сложных разветвляющихся алгоритмов в теле операторов цикла. Поэтому рассмотрим различное заполнение поля 1 и поля 3 этого оператора (см. рисунок).

Вариант 1. В поле 1 находится локальный оператор присваивания (формирование единичной матрицы)

```
I(n) := | for i ∈ 0..n − 1
        |   for j ∈ 0..n − 1
        |     m_{i,j} ← 1  if i = j
        |     m_{i,j} ← 0  otherwise
        | m
```

Вариант 2. В поле 1 находятся несколько операторов

```
S(n,a,b) := | j ← 0
            | for k ∈ 1..n
            |   if (mod(k,a) = 0) + (mod(k,b) = 0)
            |     | v_j ← k
            |     | j ← j + 1
            | v
```

2.3.7. Дополнительные операторы программирования циклов в пакете MathCad

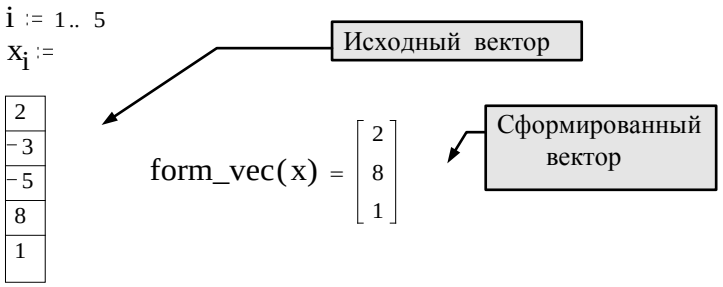
Оператор continue. Обычно используется для продолжения выполнения цикла путем возврата в начало тела цикла. Следующий пример поясняет работу этого оператора.

Пример. Составим программу-функцию, формирующую новый вектор из положительных проекций исходного вектора.

```
form_vec(v) := | i ← 0
               | j ← 0
               | while i < last(v)
               |     | i ← i + 1
               |     | continue  if v_i ≤ 0
               |     | j ← j + 1
               |     | w_j ← v_i
               | w
```

В теле программы-функции используется функция last(v), определяющая индекс последнего элемента массива v (см. замечание 2.5).

Обращение к этой программе функции имеет вид



Если очередной элемент v_i не больше нуля, то пропускаются все нижележащие операторы тела цикла (в нашем случае - два оператора, формирующие очередную проекцию вектора w) и тело цикла повторяется при новом значении параметра цикла i.

Оператор return

Прерывает выполнение программы-функции и возвращает значение операнда, стоящего за ним. Следующий пример поясняет работу этого оператора.

Пример. Составим программу-функцию, находящую первую положительную проекцию исходного вектора. Возможны два варианта.

Вариант А

```
pol(v) := | i ← 1
          | while v_i ≤ 0
          |   i ← i + 1
          | v_i
```

Вариант В

```
pol1(v) := for i ∈ 1.. last(v)
             return v_i if v_i > 0
```

Вариант В представляется более простым и "элегантным".

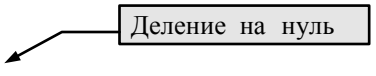
Оператор on error

Этот оператор является обработчиком возникающих при выполнении тех или иных вычислений ошибок и записывается в виде :

< выражение 1 > **on error** < выражение 2 >

и выполняется < выражение 1 >, если при выполнении < выражение 2 > возникает ошибка. Если ошибка не возникает, то выполняется < выражение 2 >.

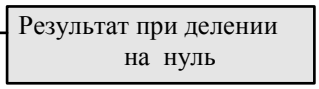
Пример . Используем оператор **on error** для предотвращения появления ошибки "деление на нуль" при вычислении функции angl(x,y).



$$angl(x,y) := \frac{x}{y} \qquad angl(2,0) =$$

$$angl(x,y) := 0 \quad \text{on error } angl(x,y)$$

$$angl(2,0) = 0$$



Функция error. Используется для вывода диагностических сообщений при возникновении в вычислениях ошибки и записывается в виде

error ("< диагностическое сообщение пользователя >")

Функция используется в левом поле условного оператора **if,** как показано в следующем примере.

Пример. Программирование вывода диагностического сообщения при попытке спроектировать вектор v на нулевой вектор w.

$$proj(v,w) := \begin{cases} error(\text{" You cannot project onto the 0 vector"}) & \text{if } |w| = 0 \\ \frac{w}{|w|} \cdot (v \cdot w) & \text{otherwise} \end{cases}$$

Учебное издание

Основы работы и программирования в системе MathCad

Учебное пособие

Составитель
КОЧЕГУРОВА ЕЛЕНА АЛЕКСЕЕВНА

Научный редактор
кандидат технических наук, доцент кафедры АиКС ИК
А.А.Пономарев