# Лабораторная работа № 3

**Программирование в среде MATLAB**

**Цель работы**: изучение возможностей MATLAB по составлению файлов-сценариев и файлов-функций, в том числе для организации разветвляющихся и циклических программ.

## Рабочее задание

Работа в режиме калькулятора в среде MATLAB не позволяет выполнять циклические алгоритмы вычислений. Такие вычисления удобнее выполнять в виде программ. Для этого следует набирать команды в окне редактирования **Editor**, а затем выполнять их все сразу или частями, в том числе и из командной строки, а также сохранить их в файле с расширением .m и использовать в дальнейшем. Кроме того, в окне редактирования удобнее выполнять отладку программы.

Файлы, которые содержат в себе коды программы MATLAB, называются *М*-файлами. Они бывают двух типов: *файлы-сценарии* (**Script M-Files**) и *файлы-функции* (**Function M-Files**), описывающие функции, определяемые пользователем.

Основные программы, управляющие от начала до конца порядком всего вычислительного процесса, являются файлами-сценариями. Файлы-функции используют для оформления отдельных процедур и функций, при выполнении которых либо внутри файлов-сценариев, либо внутри других файлов-функций необходимо сначала задать значения их входных переменных.

1 Запустите программу MATLAB и настройте имя директории по умолчанию для хранения создаваемых файлов. Для этого выберите команду **Home****Set Path****Add Folder** (для интерфейса с меню – **File****Set Path**…**Add Folder**). В появившемся окне (рисунок 3.1) укажите свою папку, находящуюся на диске С:, нажмите на клавишу Save и закройте окно, подтвердив запрос об использовании указанной папки для хранения создаваемых файлов по умолчанию.

###### Обратите внимание, что имя вашей папки должно содержать лишь латинские буквы или цифры без пробелов!

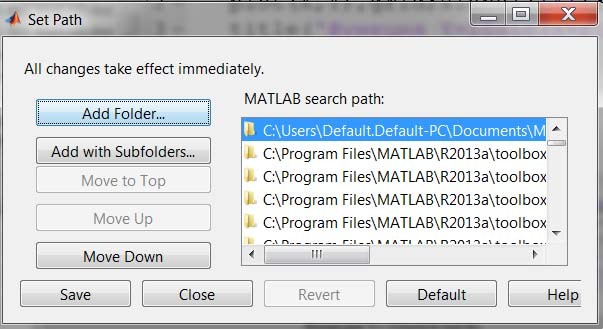


Рисунок 3.1

1. Создайте *М*-файл с помощью команды **Home****New****Script** (для интерфейса с меню – **File****New****M-file**) из основного окна MATLAB. В результате откроется окно редактора *М*-файлов **Editor**, в котором наберите команды для построения графика (рисунок 3.2).

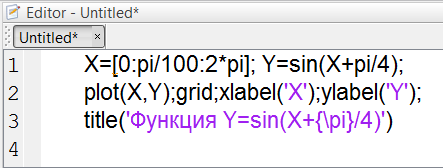


Рисунок 3.2

1. Сохраните *М*-файл в своей папке на диске С: под именем grafik1.m с помощью команды **Editor****Save As** (для интерфейса с меню – **File****Save As** в окне редактора *М*-файлов).

###### Обратите внимание, что имя файла должно содержать лишь латинские буквы или цифры без пробелов!

1. С помощью команды **Editor****Run** (для интерфейса с меню –**Debug****Run** в окне редактора *М*-файлов) или путем нажатия клавиши F5 выполните все команды из окна редактирования. В результате на экране появится графическое окно **Figure l** с графиком функции и заголовком.
2. Увеличьте в программе диапазон изменения аргумента X в два раза и запустите все команды снова. Посмотрите, как изменится график, и сделайте вывод.
3. Выделите с помощью мыши первые две строки программы в окне редактирования *М*-файлов, нажмите правую клавишу мыши и выберите в появившемся контекстном меню команду **Evaluate Selection** (или нажмите клавишу F9). Обратите внимание, что в графическом окне выведется только график без заголовка.

**В отчет:** Включить в отчет по лабораторной работе содержимое окон редактирования *М*-файлов и графики, полученные при выполнении пп. 26.

1. Закройте окно редактирования *М*-файлов и графическое окно. Откройте снова файл **figura1.m** с помощью команды **Home****Open** (для интерфейса с меню – **File****Open**).
2. В редакторе *М*-файлов может быть одновременно открыто несколько файлов. Переход между файлами осуществляется с помощью закладок с именами файлов. Создайте новый *М*-файл под именем **fazagrad.m** как файл-функцию и определите в нем новую *М*-функцию с помощью ключевого слова **function**. Формат заголовка *М*- функции имеет следующий вид:

**Function [выходные переменные] = Имя (входные переменные)**

Определите в окне редактора *М*-файлов **Editor** (рисунок 3.3) функцию под именем **fazagrad**, которая предназначена для перевода фазового сдвига между напряжением и током на участке электрической цепи, измеренного по осциллографу в секундах (**fazaс**), в градусы (**fazag**).

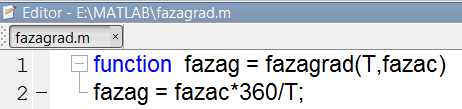


Рисунок 3.3

В качестве входных параметров данной функции выступают измеренные по осциллографу (в секундах) период гармонического сигнала *T* и фазовый сдвиг **fazaс**. Данная функция возвращает свое значение (в градусах) переменной **fazag**.

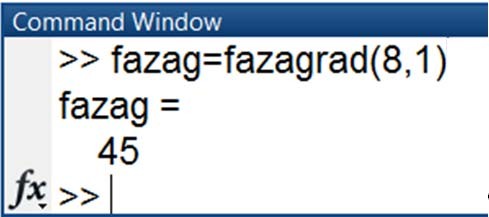
1. Сохраните созданную *М*-функцию с помощью команды **File****Save** и закройте окно редактирования.
2. Обращение к созданной функции **fazag** может быть выполнено или в командном окне, или в окне редактора файла-сценария. На рисунке 3.4 представлен пример обращения к функции **fazagrad** из командного окна.

Рисунок 3.4

**В отчет:** Включить в отчет по лабораторной работе содержимое окна редактирования, полученное при выполнении п. 8, а также результаты обращения к функции для трех различных значений ее аргументов.

1. Как в файлах-сценариях, так и в файлах-функциях для организации разветвляющиеся и циклические вычисления используют такие операторы управления вычислительным процессом, которые начинаются со служебных слов **if**, **while** или **for** и заканчиваются служебным словом **end**. Между указанными служебными словами располагаются другие операторы.

Синтаксис оператора условного перехода приведен на рисунке 3.5.

##### 

##### if условие 1

**<операторы 1>**

##### elseif условие 2

**< операторы 2>**

##### . . .

**else**

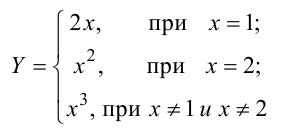
##### <операторы>

##### end

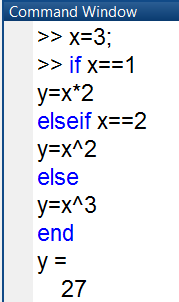
Рисунок 3.5

При выполнении этого оператора сначала проверяется выполнение первого условия. Если оно ложно, то по порядку проверяется условие в каждом из следующих за ним операторов **ElseIf** (их может быть сколько угодно). Как только находится истинное условие, выполняются соответствующие операторы, после чего выполнение блока **If** прекращается и управление передается на оператор, следующий за словом **End**. Если никакое из проверенных условий **ElseIf** не справедливо, то выполняются операторы, следующие за словом **Else**, если они есть. После чего выполнение блока **If…end** прекращается.

Пример программы с применением условного оператора для вычисления значения ступенчатой функции



и результаты ее выполнения приведены на рисунке 3.6.

Рисунок 3.6

1. Циклы с неопределенным количеством повторений позволяет задавать конструкция **While…end**, в которой указывается условие продолжения цикла:

##### 

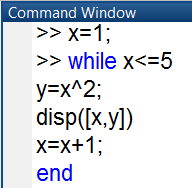
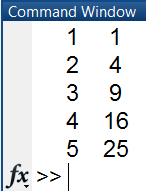
##### While условие

**... [операторы]**

**end**

Цикл повторяется до тех пор, пока условие не станет ложным. После чего все операторы до оператора **end** включительно пропускаются, и программа выполняется дальше.

На рисунке 3.7,*а* приведен пример программы с применением оператора **While**, а на рисунке 3.7,*б* – результаты ее работы.

*а*) *б*)

Рисунок 3.7

1. В MATLAB имеется команда **disp**, которая осуществляет вывод значений указанной переменной или указанного текста в командное окно. Обращение к ней имеет следующий вид:

**disp (<переменная или текст в апострофах>)**

Чтобы вывести значения нескольких переменных в одну строку (например, при создании таблиц данных), нужно создать единый объект, который содержал бы все эти значения. Это можно сделать, объединив соответствующие переменные в вектор, пользуясь операцией создания вектора строки:

**x = [x1 x2 ... ].**

На рисунке 3.8 приведен пример программы, в результате выполнения которой в командное окно в одну строку выводятся значения четырех ранее определенных переменных через пробелы.

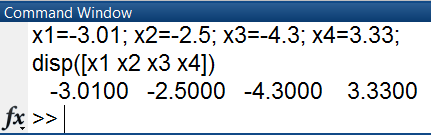


Рисунок 3.8

Для одновременного вывода символьной и цифровой информации в командное окно удобно использовать функцию **sprintf**. На рисунке 3.9 приведен пример соответствующей программы.

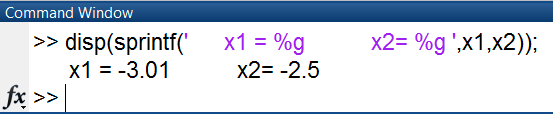


Рисунок 3.9

1. В MATLAB для организации цикла с заданным числом повторений используется оператор **For…end**, имеющий следующий формат:

##### for Имя = <Начальное значение>:<Шаг>:<Конечное значение>

**. . . операторы**

##### end

где **Имя** *–* имя управляющей переменной цикла. Если шаг не указан, то по умолчанию он равен единице.

Конструкция выполняет операторы, следующие за оператором **For**, пока в программе не встретится оператор **end**. Тогда к текущему значению переменной цикла прибавляется значение шага, и полученное значение сравнивается с конечным значением. Если значение переменной цикла больше конечного значения, то выполняется оператор, следующий за оператором **end**, иначе управление снова передается к оператору, находящемуся за оператором **For**.

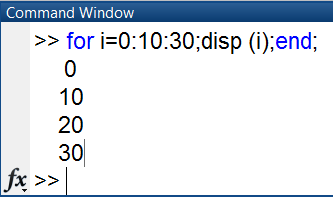
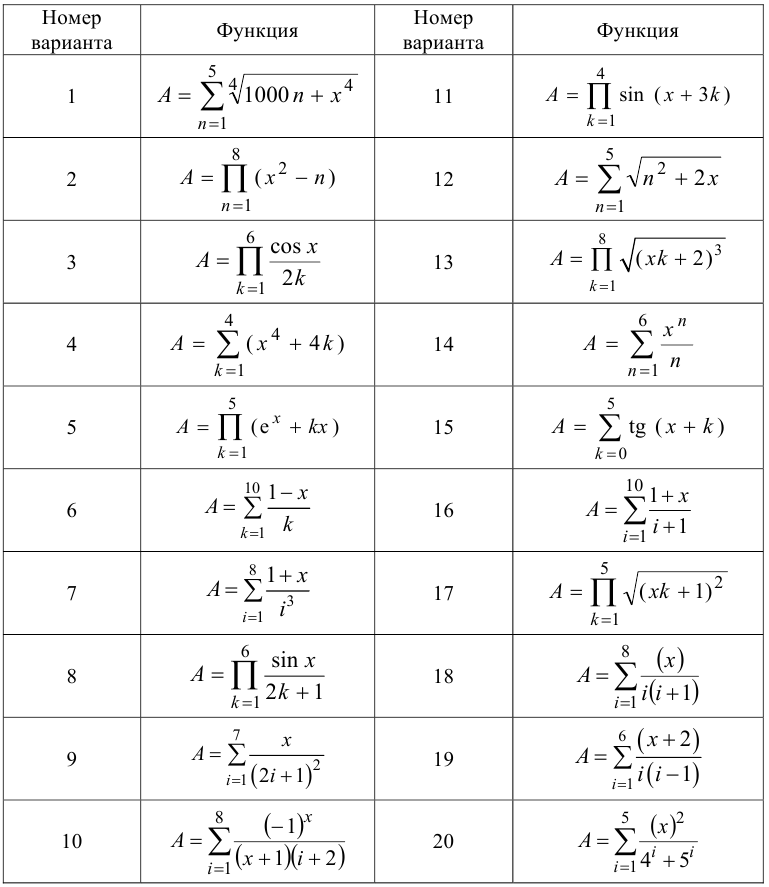
На рисунке 3.10 приведен пример циклической программы, выводящей на дисплей значения переменной *i*, изменяющиеся от 0 до 30 с шагом, равным 10.

Рисунок 3.10

1. Запишите в среде MATLAB программу для вычисления и вывода на экран в виде таблицы значений *x* и функции *A*(*x*) для заданного варианта из таблицы 3.1. Для всех вариантов значения *x* меняются от 10 до 10 с шагом 5. Для всех вариантов составьте две программы: с использованием оператора **For…end** и оператора **While…end**.
2. **В отчет:** Включить в отчет по лабораторной работе содержимое окна редактирования и командного окна, полученное при выполнении пп. 1115.

Таблица 3.1



## В отчет: Организация цикла с заранее неизвестным числом повторений. Создать function-файл GeomProgression для вычисления в цикле суммы бесконечной геометрической прогрессии:

## (1)

## с заданной точностью ε=10-5 . После выхода из цикла вычислить точное значение суммы и погрешность ее вычисления (вектор двух значений). *q*-номер варианта

17 Работа с М-файлами включает в себя разработку алгоритма и написание программ — основной программы (script-файла) и подпрограмм (function -файлов);

Рассмотрим пример решения СЛАУ:

АХ =В, (1).

где:

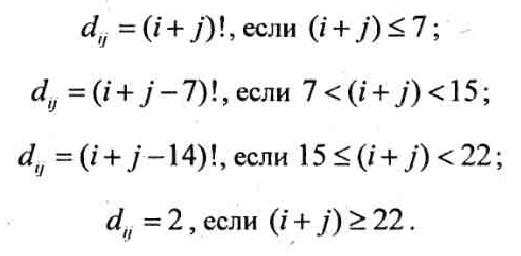
*Х* — вектор-столбец неизвестных корней уравнения;

*В* — вектор-столбец свободных членов длиной n (порядок СЛАУ), причем:

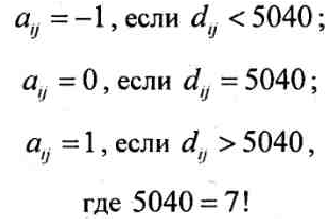
* порядок системы n имеет произвольное значение, вводимое с клавиатуры;
* элементы вектора *В* — натуральные числа: ;

*А* — квадратная матрица порядка *n*, элементы которой вычисляются согласно следующему алгоритму:

1. Сначала находятся элементы , вспомогательной матрицы *D* (символ «!» означает факториал):



2. Затем определяются элементы , матрицы А:



Для решения системы (1) требуется разработать две программы (два М-‘ файла):

* основную программу (script-файл), реализующую:

- ввод произвольного значения порядка системы n с клавиатуры;

- решение СЛАУ;

- вывод решения — вектора Х в окно **Command Window**;

* подпрограмму (function-файл) для вычисления факториала.

Для хранения script-файла и function-файла создадим собственную папку Step\_1 и сохраним путь к ней

В рассматриваемом примере function-файлу присвоим имя **fact**, такое же, как имя функции в заголовке function

function y=fact(x)

y=1;

if x>0

for i=1:x;

y=y\*i;

end

end

script - файл назовем test\_1:

clc

echo on

echo off

a=[];

b=[];

d=[];

n=input('n=')

if n<0

error('ошибка - отрицательный порядок системы n')

end

for i=1:n

for j=1:n

if (i+j)<=7

d(i,j)=fact(i+j);

elseif ((i+j)>7) & ((i+j)<15)

d(i,j)=fact(i+j-7);

elseif ((i+j)>=15) & ((i+j)<22)

d(i,j)=fact(i+j-14);

else

d(i,j)=2

end

if d(i,j)<5040

a(i,j)=-1;

elseif d(i,j)==5040

a(i,j)=0;

else

a(i,j)=1;

end

end

end

disp('Вывод матрицы a')

disp(a)

for i=1:n

b(i)=i;

end

b=b';

disp('Вывод сводных членов-вектора b')

disp(b)

x=a\b;

disp('Вывод порядка системы')

disp(n)

disp('Вывод уравнения')

disp(x)

Результат работы программы:

echo off

n=5

n =

5

Вывод матрицы a

-1 -1 -1 -1 -1

-1 -1 -1 -1 0

-1 -1 -1 0 -1

-1 -1 0 -1 -1

-1 0 -1 -1 -1

Вывод сводных членов-вектора b

1

2

3

4

5

Вывод порядка системы

5

Вывод уравнения

-11

4

3

2

1

## Контрольные вопросы

1. Как задается место хранения рабочих файлов в MATLAB?
2. Чем отличаются файлы-сценарии и файлы-функции в среде

MATLAB?

1. Как создаются, открываются, сохраняются и запускаются на исполнение *М*-файлы в среде MATLAB?
2. Как выполнить несколько строк из окна редактирования в среде MATLAB?
3. Какие операторы управления вычислительным процессом существуют в среде MATLAB и как они работают?
4. Поясните назначение и формат оператора if.
5. Поясните назначение и формат оператора switch.
6. Поясните назначение и формат оператора for.
7. Поясните назначение и формат оператора while.