Тема 4.6

Программирование на языке MATLAB

Подпрограммы и функции

Вопросы для изучения

4.26 Подпрограммы
4.27 Функции пользователя. Способы создания и вызова внешних и inline функций.
4.28 Область видимости переменных
4.29 Вложенные функции.
4.29 Способы передачи параметров функциям пользователя

4.26 Подпрограммы

В программировании часто встречаются случаи, когда по ходу выполнения программы приходится выполнять одни и те же вычисления, но при различных исходных данных.

Чтобы исключить повторение одинаковых фрагментов программного кода и сделать тем самым программу проще и понятнее, можно выделить эти повторяющиеся вычисления в самостоятельную часть программы, которая может быть использована многократно по мере необходимости.

Подпрограммой называется автономная часть программы, реализующая определенный алгоритм и допускающая обращение к ней из различных частей общей программы, неограниченное число раз.

В большинстве языков программирования различают подпрограммы-функции и подпрограммы-процедуры. Первые обязаны возвращать числовое значение, результат же действия вторых может быть произвольным.

В MATLAB нет такого деления, здесь слова подпрограмма, функция и процедура являются синонимами.

Преимущества использования подпрограмм:

– нет дублирования кода, что сокращает трудоемкость создания программы;

– более удобный процесс отладки и внесения изменений;

– повышает надежность программы.

В языке MATLAB все подпрограммы можно разделить на стандартные и пользовательские.

Встроенные (стандартные) подпрограммы являются частью языка и могут вызываться по имени без предварительного описания.

Однако в большинстве случаев специфичные для данной программы действия не находят прямых аналогов, и тогда программисту приходится разрабатывать свои нестандартные функции.

Пользовательские (нестандартные) подпрограммы пользователя пишутся самим программистом в соответствии с синтаксисом языка.

В языке MATLAB выделяют два вида функций, создаваемых пользователем, - **встраиваемые(inline)** и **внешние.**

Передача исходных данных в подпрограмму и возврат результата выполнения осуществляются с помощью параметров.

Различают **формальные** и **фактические** параметры.

Параметры, которые указываются при объявлении (описании) подпрограммы, называются **формальными**.

Параметры, которые указываются при вызове подпрограммы, называются **фактическими**.

При вызове процедуры **количество фактических параметров должно обязательно соответствовать числу формальных параметров**. При этом соответствующие параметры не обязательно должны быть одинаково обозначены.

4.27 Функции пользователя. Способы создания и вызова внешних и inline функций

Любая подпрограмма в MATLAB состоит из заголовка и тела функции, в котором располагаются исполняемые операторы.

Структура описания функции

```
%Заголовок
function[выходные формальные параметры]=Имя функции(входные формальные параметры)
%Тело функции
Любые выражения и операторы MATLAB
End % слово end можно допускается не указывать за исключением случая когда используется вложенность функций
```

Во избежание вывода на экран нежелательных промежуточных результатов, необходимо в теле функции все операторы завершать символом " ; ".

В заголовке процедуры, помимо ключевого слова **function** и имени функции, перечисляются **входные и выходные формальные параметры**, которые служат для обмена значениями между процедурой и вызывающей ее программой.

Список входных формальных параметров заключается в круглые скобки, а список выходных параметров в квадратные.

Вызов функции в основной программе осуществляется по ее имени с указанием фактических входных параметров

Имя функции(список входных фактических параметров)

При вызове функции ей при помощи аргументов (формальных параметров) могут быть переданы некоторые значения (фактические параметры), используемые во время выполнения функции.

После завершения работы функции фактические значения присваиваются выходным параметрам.

Допускается также использование подпрограмм, не имеющих аргументов и не возвращающих никаких значений. Действие таких подпрограмм может заключаться, например, в выводе на печать некоторых данных и т.п.

В простейшем случае у функции есть один входной и один выходной формальный параметр:

```
function [y] = func1(x)  %описание функции func1 где y формальный выходной
            % параметр, а x формальный входной параметр
y = x.^2;                %результат вычисления присвоен выходному параметру


>> func1(3)              % вызов функции func1 с фактическим значением входного
      % параметра =3
```

Список входных формальных параметров может состоять из нескольких элементов.

```
function w = func2(x, y, z)
w = x.^2+y.^2+z.^2;

>> func2(1,2,3)
```

При использовании одного выходного параметра квадратные скобки можно не использовать

У функции может быть несколько входных и несколько выходных параметров

```
function [u, v, w] = func3(x, y)
u = x+y;
v = x.*y;
w = x-y;

>> [a,b,c]=func3(2,3)
```

Список как входных, так и выходных формальных параметров может быть пустым. В этом случае получение исходных данных для проведения вычислений и вывод полученных результатов может осуществляться исключительно с помощью операций ввода-вывода, а функция записывается в следующем виде

```
function func4
x = input('Введите x: ');
y = x.^2+1;
disp(y);
>> func4
```

В том случае если функция не содержит операторов управления и ввода-вывода, она может быть определена в качестве встраиваемой функции.

Встраиваемая функция создается с помощью процедуры inline или анонимной командой @ (первый сейчас считается устаревшим способом) непосредственно в командном окне и часто называется inline-функцией.

Имя функции=inline (выражение, список аргументов)

Определяющее функцию выражение и имена аргументов передаются в inline функцию только в виде строковых выражений.

myf = inline('sin(x.^2+y.^2)', 'x', 'y')

список аргументов допускается не указывать.

или

Имя функции=@ (аргументы) выражение

myf1 = @(x, y) sin(x.^2+y.^2)

У inline-функций может быть только один выходной параметр, совпадающий с именем самой функции.

Получить формулу функции можно при помощи любой из двух процедур класса inline - char(имя функции) или formula(имя функции).

Процедуры disp(имя функции) и display(имя функции) осуществляют вывод на экран дисплея заданного inline-объекта.

Получение имен аргументов inline функции. Это действие осуществляется процедурой argnames(имя функции), работает с функциями созданными с помощью процедуры inline

4.28 Область видимости переменных

Область видимости — это та часть программного кода, в которой может быть использован данная переменная.

Переменная считается видимой если в этом блоке программы или m-файле известны имя и тип этой переменной.

Переменная может быть видимой в пределах блока, файла или во всех исходных файлах, образующих программу. Это зависит от того, на каком уровне объявлена переменная: **на внутреннем (локально),** то есть внутри некоторого блока, или **на внешнем (глобально)**, то есть вне всех блоков.

По умолчанию все переменные, встречающиеся в подпрограмме, являются локальными и действуют только внутри этой подпрограммы. Никакой связи между ними и объектами, вызывающей программы, имеющими, возможно, те же самые имена, нет. Они полностью независимы.

При вызове функций можно использовать идентификаторы переменных, описанные только в вызывающей программе. Такие идентификаторы являются глобальными.

В общем случае объявленные внутри функций переменные, имеют область видимости только в пределах функции в которой объявлены - являются локальными, и за ее пределами уже не доступны (не видны).

Пример

```
function MyFunc()          %главная функция MyFunc
x = 10
disp(x)
MyFunc2()

    function MyFunc2()  %подфункция MyFunc2
    disp(x)
```

В результате на экране будет отображено

```
10
??? Undefined function or variable 'x'.
```

Для того чтобы переменная была видна за пределами функции, в которой объявлена – являлась **глобальной** используют обращение к этой переменной с помощью ключевого слова global, за которым следует имя глобальной переменной.

Пример:

```
function MyFunc ()                %главная функция MyFunc
x = 10
disp(x)
MyFunc2()

    function MyFunc2()   %подфункция MyFunc2
    global x             %глобальное описание переменной x
    disp(x)
```

Ключевое слово global говорит о том, что переменная x уже объявлена ранее и нужно ее использовать внутри текущей функции как глобальную.

4.29 Вложенные функции и подфункции

Matlab допускает два способа описания нескольких функций внутри одного файла:

• основная функция и ее **подфункции**;
• основная функция и **вложенные функции**.

Пользоваться обеими этими способами в одном m-файле нельзя.

М-файл с программой-функцией может содержать описание не одной, а нескольких функций. Имя главной функции должно совпадать с именем файла.

При вызове функции можно будет обратиться только этой функции, имя которой совпадает с именем m-файла. Все остальные функции **называются внутренними (подфункциями)** и могут быть вызваны только из главной функции.

Каждая из них может быть вызвана либо из основной функции, либо из другой подфункции того же самого m-файла.

Например описание функции и подфункций в одном файле:

```
function main                %главная функция main
x = input('Введите x: ');
if x>0  y = f1(x);
else  y = g1(x);
end
disp(y);

function y = f1(x)           %подфункция f1
y = x+1;

function y = g1(x)           %подфункция g1
y = x-1;
```

Все переменные, используемые внутри подфункций f1 и f2, являются **локальными**: их область видимости ограничивается только этими подфункциями.

Все переменные, используемые в основной функции, также являются локальными: их область видимости распространяется только на саму функцию, но не на ее подфункции.

В примере выше переменные с одинаковым именем x в основной функции и двух подфункциях различны.

Функции можно вкладывать внутрь других функций.

Чтобы создать вложенную функцию, нужно разместить ее описание внутри тела другой функции.

В этом случае каждая из функций, описанных в файле должна заканчиваться ключевым словом end.

Все команды после заголовка функции (строка function ...) и до соответствующего ключевого слова end составляют тело главной функции.

Количество уровней вложенности не ограничено. Область видимости переменных главной функции распространяется на все вложенные в нее функции (а также функции, вложенные во вложенные функции, и т.д.).

Во вложенных функциях f1 и g1, есть доступ к переменной x из главной функции main.

Например описание функции с вложенными функциями в одном файле:

```
function main %главная функция main
x = input('Введите x: ');
if x>0   y = f1(x)
else   y = g1(x);
end
disp(y);

    % вложенная функция f1
    function y = f1(x)
    y = x+1;
    end

    % вложенная функция g1
    function y = g1(x)
    y = x-1;
    end
end
```

Тело главной функции

```
function [. . .] = main(. . .)
 a = . . .;
 b = . . .;
    function [. . .] = sub(. . .)
    c = . . .;
        function [. . .] = subsub(. . .)
        end;
    end;
    function [. . .] = sub2 (. . .)
     c = . . .;
    end;
end
```

В примере выше во вложенных функциях sub, subsub и sub2 есть доступ к переменным a и b из главной функции main.

Во вложенной функции subsub есть доступ к переменной c из sub.

В то же время переменные c одинаковым именем c в функции sub и функции sub2 различны (в основной функции main нигде не встречается переменная c).