Лабораторная работа №5

Организация диалога с пользователем в MATLAB. Форматированный вывод.

Цель: Знакомство с особенностями ввода и вывода данных в MATLAB, организацией диалога с пользователем, проверки введенных пользователем данных. Организация форматированного вывода результатов расчета. Закрепление навыков по преобразованию типов данных, организации программ-сценариев и подпрограмм.

Зачастую при работе программы необходимо, чтобы данные вводились пользователем с клавиатуры. Также результаты выполнения программы удобно выводить в консоль.Основу консольного ввода-вывод в MATLAB составляют 3 функции:

- `input` — функция ввода;
- `disp` — функция вывода содержимого указанной переменной;
- `fprintf` — функция форматированного вывода.

Функция ввода `input`

В случае, когда значение некоторой переменной должно задаваться в диалоге с пользователем в ответ на запрос (подсказку) программы, используют функцию `input` . В простейшем случае функция имеет следующий синтаксис

```
a=input('Текст запроса')
```

При запуске программы в консоль будет выведено сообщение, содержащееся в кавычках. Программа будет ждать ответ от пользователя. Переменной `a` будет присвоено значение, введенное пользователем с клавиатуры. Для того, чтобы переменной было присвоено значения какого либо типа данных, необходимо осуществлять ввод с соответствующими атрибутами. Так, например, для ввода матрицы необходимо вводить данные в квадратных скобках, разделяя элементы строки пробелами или запятой, а строки – точкой с запятой. Если ввести в командное окно выражение, MATLAB попытается его вычислить.

```
>> a = input('Введите число a: ');
Введите число a: 12.5
>> b = input('Введите число b: ');
```

```
Введите число b: 2 + 5
>> a
a =
   12.5000
>> b
b =
     7
```

Для того, чтобы присвоить переменной символьное значение, необходимо, чтобы пользователь вводил данные, заключив их в кавычки, иначе программа выведет сообщение об ошибке. Так же в записи функции `input` можно через запятую после текста запроса указать второй параметр `'s'`:

```
>>str = input('Введитестроку: ');
Введитестроку: Hellow world!
Error: Unexpected MATLAB expression.

Введитестроку: 'Hellow world!'
>>str
str =
Hellow world!

>>str = input('Введитестроку: ','s');
Введитестроку: Hellow world!
>>str
str =
Hellowworld!
```

Для того, чтобы ввод значение переменной осуществлялся с новой строки после вывода запроса-подсказки, необходимо в записи функции `input` перед кавычкой, закрывающей текст запроса, поставить оператор перевода строки «`\n`»

```
>> a=input('Введите число:\n');
Введите число:
5
>> a
a =
     5
```

Функция простого вывода `disp`

Функция `disp()` выводит в консоль значение переменной или данные (числовые, символьные), стоящих в скобках. Функции `disp()` имеет единственный аргумент, т.е. с ее помощью можно вывести лишь одну переменную. Вывод функции `disp()` аналогичен тому, если ее аргумент ввести в консоль без символа «`;`» на конце, однако имя переменной выведено не будет.

```
>>disp(str);
Hellow world!
>>str
str =
Hellow world!

>>disp(x)
   12.5000
>>x
x =
12.5000
```

В простейших случаях вывода одной переменной или строки удобно пользоваться функцией `disp`.Для расширения возможностей вывода данных в консоль используют функцию форматированного вывода `fprintf`.

Функция форматированного вывода `fprintf`

Функция форматированного вывода позволяет вывести данные в формате, заданном пользователем. Эта функция соответствует международному стандарту ASCCC и имеет аналоги практически во всех современных языках программирования.

Например, в программе необходимо вывести значение переменной X.

В простейшем случае это можно выполнить, использую функцию `disp`. Проиллюстрируем работу этой функции для различных значений X:

```
1).  >>X=4324;
     >>disp(X)
     4324 %(X отображается как целое число);
2).  >> X=2.45;
     >>disp(X)
```

```
     2.4500    %(X отображается как число записанное с 4 знаками
после запятой);
3).  >>X=2.45999999999999999999999999999;
     >>disp(X)
         2.4600
     >>X=0.45999999999999999999999999999;
     >>disp(X)
     0.4600    % (X отображается аналогично п.п.2 с округлением);
4)   >>X=0.00000000000000000000459999999999999999999999999999;
     >>disp(X)
          4.6000e-22          %(X отображается в показательной
форме).
```

В этих пример на экран выводится только значение переменной X в форме, определяемой внастройкой системы по умолчанию. Эти настройки можно изменить в окне **ComandWindow** меню **Preferences.**

Функция `fprintf()` в отличие от функции `disp()` позволяет в более содержательной форме, например, для `i=5` и `X=0.454999999999` можно организовать вывод сообщения, ограничиваясь двумя знаками после запятой для значения X:

```
>>fprintf('Значение x на %2d шаге цикла равно\t%6.2f\n',i,X)
```

Будет выведено:

```
Значение x на  5 шаге цикла равно  0.45.
```

С помощью функции `fprintf()` можно выводить данные на экран при выполнении программы или в файл, структура которого определяется форматной строкой. Для вывода на экран используется следующая форма записи функции:

```
fprintf(форматная_строка ,A1 ,..., An)
```

Обязательным аргументом функции `fprintf()` является форматная строка. Это строка символов, заключенная в апострофы, которая может содержать два вида объектов:

- обычные символы, выводимые на экран;
- спецификации преобразования, определяющие формат вывода данных.

Символы, выводимые на экран, представляются в кодировке ASCII и, соответственно делятся на печатные и управляющие. Управляющим символам обозначаются символом "\", например `\n` выполнят перевод курсора в начало новой строки и идентична по действию нажатию клавиши Enter. Кроме управляющих предусмотрены специальные символы для отображения на экране символов, имеющих специальное назначение в системе. Перечень управляющих и специальных символов, наиболее часто используемых в функции приведен

в таблице 6.1.

Таблица 6.1. Управляющие и специальные символы

\n	Новая строка – перевод строки после символа, после которого следует данная последовательность	>>fprintf('1\n2\n3\n4\n') 1 2 3 4
\t	Горизонтальная табуляция	fprintf('1\t2\t3') 1 2 3
\f	Пробел после символа, после которого следует данная последовательность	>>fprintf('1\f2\f3\f4') 1 2 3 4
\b	Backspace – удаление символа, после которого следует данная последовательность	>>fprintf('1234\b5678') 1235678
\\	Обратный слеш\	>>fprintf('30%% \\ 70%%\n); 30% \ 70%
%%	Символ %	
"	Одиночная кавычка	>>fprintf('You''re right!') You're right!

Каждая спецификация преобразования начинается с символа «%» и заканчивается символом-спецификатором преобразования, краткий перечень которых приведен в таблице 2. и в документации на систему. Между % и символом-спецификатором преобразования может располагаться форматная строка, состоящая из спецификаторов формата. Спецификаторы формата могут записываются последовательно в порядке, соответствующем их расположению в таблице 3.

Таблица 2. Некоторые важные спецификаторы преобразования

%d или %i	Преобразование в целое число со знаком, дробная часть отбрасывается
%f	Число с фиксированной запятой
%e	Число в экспоненциальной форме
%c	Символ
%s	Строка символов

Таблица 3. Спецификаторы формата

<номер	Номер аргумента, для которого задается	>>fprintf('%1$s =

аргумента>$	форматирование. После номера следует знак «$». Если номер аргумента не задан, форматирование применяется к параметрам в порядке следования.	%3$d\n%2$s = %4$d\n ', 'x', 'y',10,11); % при выводе, сначала будет выведен первый аргумент символьного типа, затем знак равно и третий аргумент, который является целым числом. Будет переведена строка (ENTER). С новой строки будет выведен второй аргумент, являющийся строковым, знак равно и четвертый аргумент, являющийся целым числом, будет переведена строка (ENTER). x = 10 y = 11 >>fprintf('%s = %d\n %s = %d\n', 'x', 10,'y',11); x = 10 y = 11 % форматирование применяется к аргументам по порядку их следования
«-»	Выравнивание выводимого значения по левому краю поля вывода, при необходимости пробелы добавляются не слева, а справа	fprintf('%-5f',1.2) 1.2 % Программа вывела число, используя 3 знака, и добавила справа два пробела
«+»	Всегда печатать знак числа	>>fprintf('%+d', 1,-2, 3) +1-2+3
« »	Добавляет пробел перед значением	fprintf('% d',1,2) 1 2
"0"	Заполнение поля нулями слева числа вместо пробелов	>>fprintf('%010f',1.235) 00000001.235
«#»	Модифицирует заданный формат выводимого числа. Для вещественных чисел %f и %e выводит точку, разделяющую	>>fprintf('%#5.0f',1.256) 1. >>fprintf('%5.0f\n',1.256)

	целую и дробную часть, даже если точность равна 0	1
Числовое значение минимальной ширины поля	Преобразованный аргумент будет в поле, размер которого не меньше указанной ширины. Если количество символов преобразованного аргумента меньше указанной ширины, то поле дополняется пробелами слева (если указан "-", то справа). Если количество символов больше, размер поля увеличивается.	>>fprintf('%s = %4d\n%', 'x',10); x = 10 % Система отобразила два символа для записи числа 10 и добавила два пробела слева от числа 10, в итоге для вывода числа отведено поле в 4 знака >>fprintf('%s = %10d\n%', 'x',10); x = 10 % Система отобразила два символа для записи числа 10 и добавила 8 пробелов слева от числа 10, в итоге для вывода числа отведено поле в 10 знаков
"."	Точка отделяет указатель ширины поля от указателя точности	>>fprintf('%s = %10.3f\n%', 'x',10.0535); x = 10.054 % Система выполняет округление до 3 знаков после запятой, т.е. отображает 3 символа дробной части, один символ для точки, два символа для записи целой части и добавляет 4 пробела слева от числа, в итоге для вывода числа отведено поле в 10 знаков
Число, задающее точность	Указывает число символов после запятой. При необходимости выполняется округление.	

Спецификации преобразования применяются последовательно к аргументам `A1 ,..., An` которыми могут быть имя аргумента, числовое или символьное значения. Если число аргументов больше, чем число спецификаторов преобразования, то применение спецификаторов повторяется.

В самом простом случае, спецификация преобразования в форматной строке ставится в том месте выводимого текста, в которое должно быть подставлено значение соответствующей переменной с заданным форматированием.

Так, форматная строка приведенного выше примера (`'Значение x на %2d`

шаге цикла равно\t%6.2f\n',i,X) означает:

Вывести строку символов в следующей последовательности:

'Значение x на'*<поле в 2 знаковые позиции, числовое значение целого первого аргумента (i) со знаком>*'шаге цикла равно'*<отступ вправо на 1 шаг табуляции><поле минимум в 6 знаковых позиций, дробная часть 2 знака, десятичное значение второго* аргумента*(X)><перевод строки и курсор на первую позицию строки>*

Таким образом, общий формат форматной строки, которая задается символом «%», следующий:

```
%[номер_параметра]$[флаги][ширина][.точность][подтип]спецификатор
_преобразования
```

Например: «%3$0-12.5bu».

Из всех параметров форматной строки обязательными является символ « % » и спецификатор преобразования, остальные параметры могут отсутствовать.

Так же следует отметить, что функция fprintf в отличие от disp не производит автоматический перевод строки, поэтому желательно в конце строки форматирования ставить символ перевода на новую строку «\n».

Проверка данных, введенных пользователем

В случае ввода данных пользователем с клавиатуры могут возникать ошибки, связанные с некорректным вводом. Поэтому на нескольких примерах рассмотрим проверку данных, введенных пользователем.

Пример 1. Рассмотрим функцию check_number, которая выполняет проверку на то, что введенное пользователем данное является вещественным числом. Входным параметром функции является текст запроса для ввода данного (текст, подставляемый в функцию input). Выходным параметров функции является корректно введенное вещественное число. Функция будет запрашивать ввести данное до тех пор, пока ввод не будет осуществлен корректно.

Блок-схема данной функции представлена на рисунке 1.

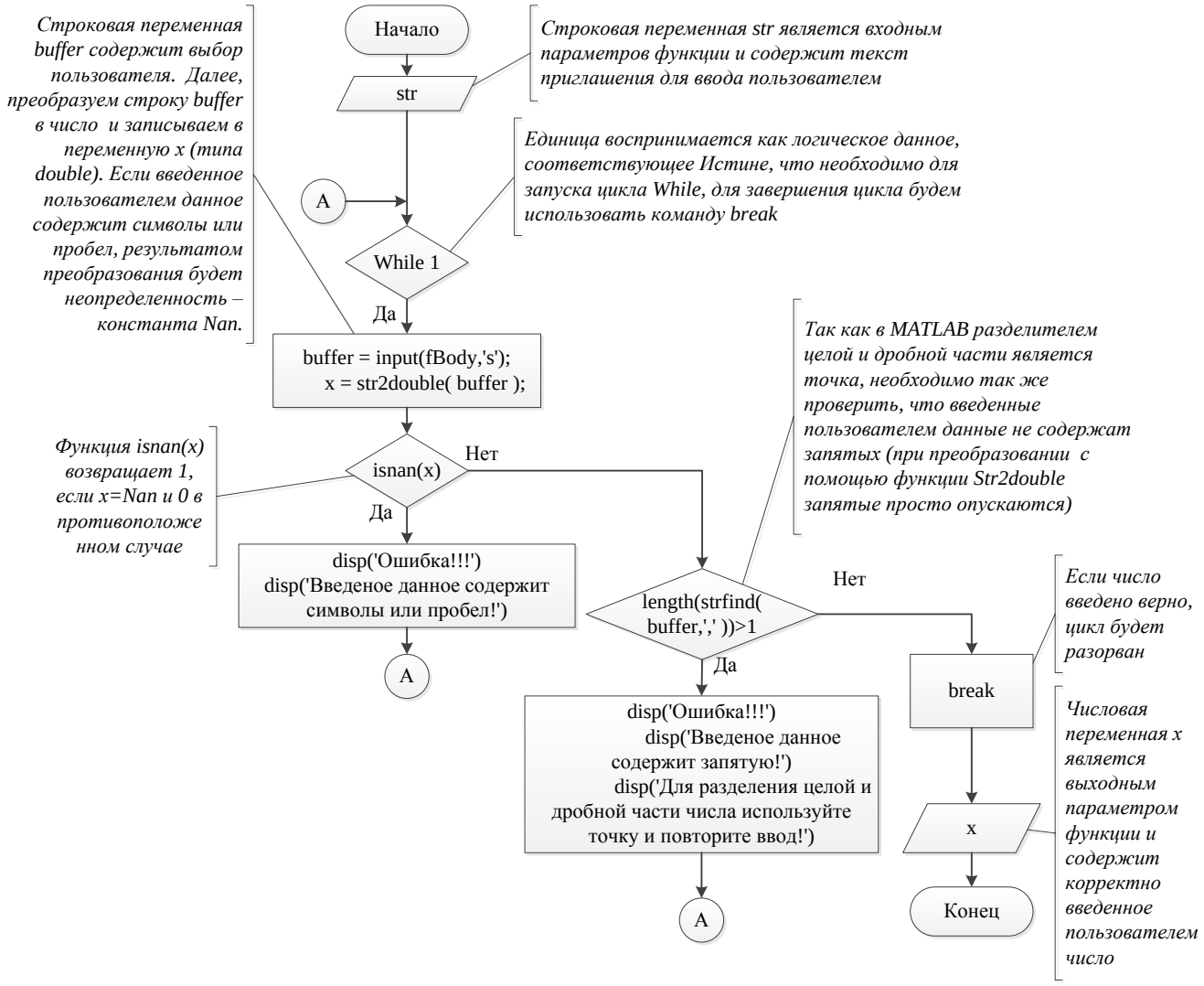

Рисунок 1. Блок-схема алгоритма функции `check_number`

Пример работы данной функции

```
>> a = check_number('Введите число, пожалуйста:\n')
Введите число, пожалуйста:
fgu

Ошибка!!!
Введенное данное содержит символы или пробел!
Повторите ввод
Введите число, пожалуйста:
3,6

Ошибка!!!
```

```
Введенное данное содержит запятую!
Для разделения целой и дробной части числа используйте точку и повторите ввод!
Введите число, пожалуйста:
3.6


a =
    3.6000
```

Пример 2. Рассмотрим функцию `more_less`, которая запрашивает у пользователя два числа, выполняет проверку на то, что введенные пользователем данные являются вещественными числами, проверяет, что первое введенное данное меньше второго. Входных параметров функция не имеет. Выходными параметрами функции являются два корректно введенных вещественных числа, причем первое возвращаемое функцией число больше второго. Функция будет запрашивать ввести данные до тех пор, пока ввод не будет осуществлен корректно. Функцию `more_less` использует ранее описанную функцию `check_number`.

Блок-схема данной функции представлена на рисунке 2.

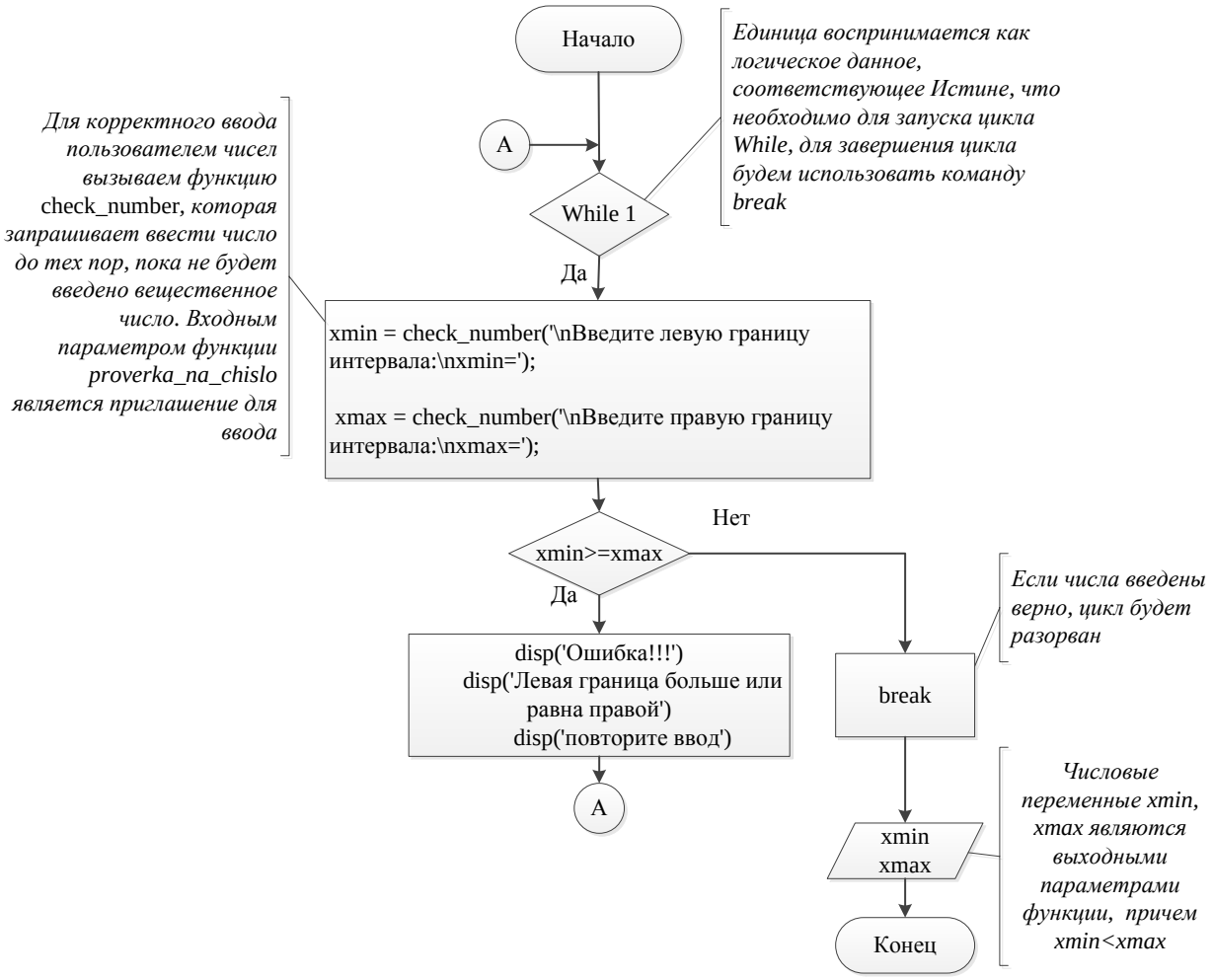

Рисунок 2. Блок-схема алгоритма функции `more_less`

Пример работы данной функции

```
>> [xmin,xmax] = more_less

Введите левую границу интервала:
xmin=5.7

Введите правую границу интервала:
xmax=5.1

Ошибка!!!
Левая граница больше или равна правой
повторите ввод

Введите левую границу интервала:
```

```
xmin=5.1

Введите правую границу интервала:
xmax=5.7

xmin =

    5.1000

xmax =

    5.7000
```

Пример 3. Рассмотрим функцию `check_vector`, которая запрашивает у пользователя вектор и выполняет проверку на то, что пользователем введен вектор чисел. Скаляр воспринимается как вектор размером 1x1. Входным параметром функции является текст запроса для ввода данных (текст, подставляемый в функцию input). Выходным параметров функции является корректно введенный числовой вектор. Функция будет запрашивать ввести данные до тех пор, пока ввод не будет осуществлен корректно.

Блок-схема данной функции представлена на рисунке 3.

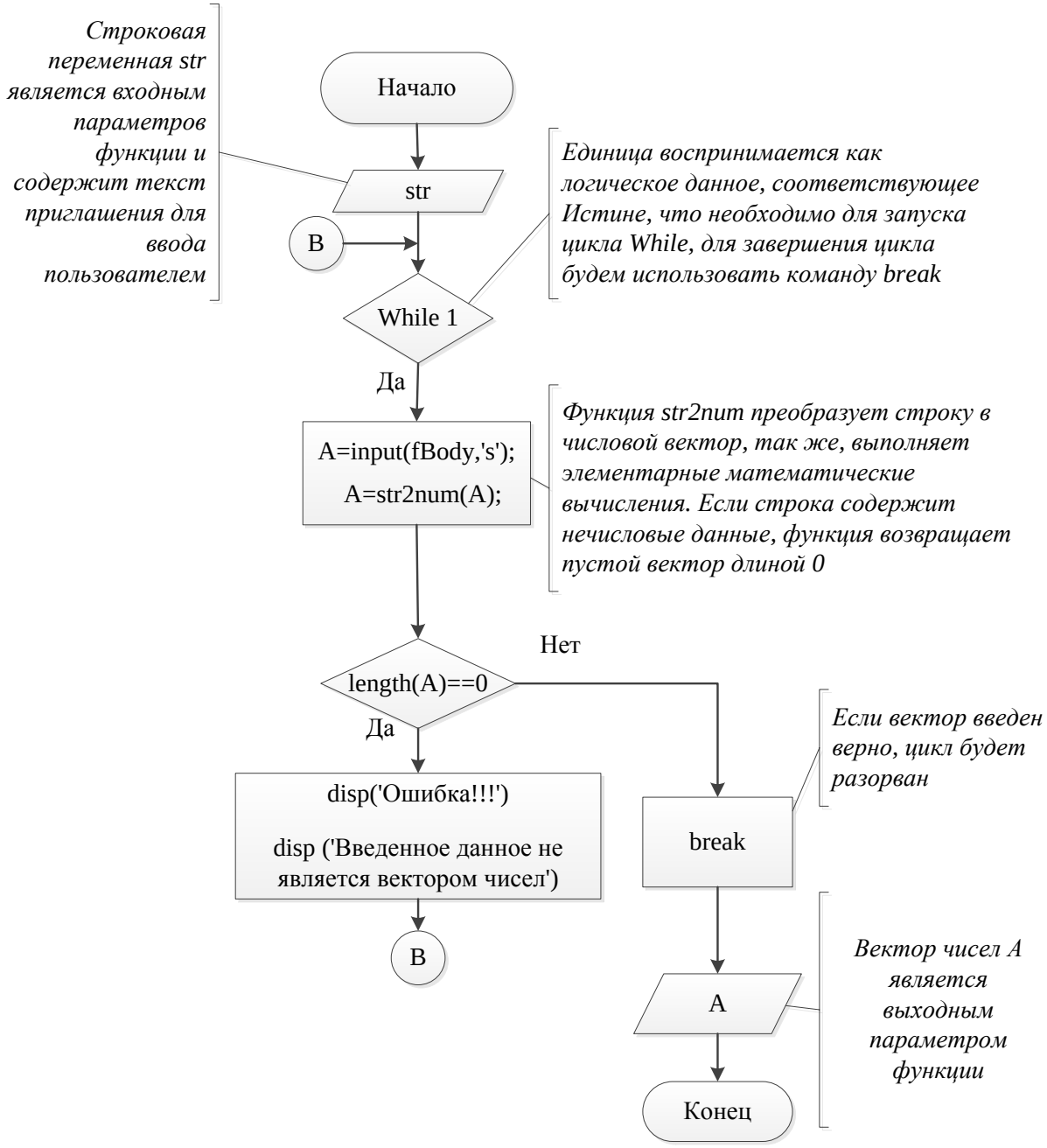

Рисунок 3. Блок-схема алгоритма функции `check_vector`

Пример работы данной функции

```
>>check_vector('Введите вектор:\nX = ')
Введите вектор:
X = 1 3 6 hj 0
Ошибка!!!
Введенное данное не является вектором чисел
Введите вектор:
```

```
X = 1 4 6 9+3 5*2

ans =

     1     4     6    12    10
```

Пример 4. Рассмотрим функцию printTable, которая осуществляет вывод таблицы со значениями математической функций (расчет математической функции выполняется внутри функции printTable). Входными параметрами данной функции являются минимальное значение аргумента, шаг аргумента, максимальное значение аргумента. Выходных параметров функция не имеет.

```
Function printTable(xMin, dx, xMax)
% Вывод таблицы для заданных функций в диапазоне xMin :dx : xMax

% вывод шапки таблицы
fprintf('/--------------------------------\\\n');
fprintf('|   x   |   exp(x)   | normcdf(x) |\n');
fprintf('|-------|------------|------------|\n');
% вывод содержимого таблицы
for x = xMin :dx : xMax
   fprintf('|%7.3f|%12.4f|%12.4f|\n', x, exp(x), normcdf(x));
end
% закроем таблицу
fprintf('\\--------------------------------/\n');

end
```

Пример работы данной функции

```
>>printTable(1,0.1,2)
/--------------------------------\
|   x   |   exp(x)   | normcdf(x) |
|-------|------------|------------|
|  1.000|      2.7183|      0.8413|
|  1.100|      3.0042|      0.8643|
|  1.200|      3.3201|      0.8849|
|  1.300|      3.6693|      0.9032|
|  1.400|      4.0552|      0.9192|
|  1.500|      4.4817|      0.9332|
```

```
|  1.600|      4.9530|      0.9452|
|  1.700|      5.4739|      0.9554|
|  1.800|      6.0496|      0.9641|
|  1.900|      6.6859|      0.9713|
|  2.000|      7.3891|      0.9772|
\-------------------------------/
```

Задание на лабораторную работу №5

Задание №1

Написать функцию, которая запросит у пользователя данное, проверит, удовлетворяет ли введенное данное условию, приведенному в таблице 4 (согласно номеру варианта). Если введенное пользователем значение не удовлетворяет условию, функция выведет сообщение об ошибке, предложит заново ввести данное. Последнее действие должно выполняться, пока не будет осуществлен корректный ввод данного. Входным параметром функции является строка, которая будет выведена при запросе на ввод числа. Выходным параметром является корректное данное (удовлетворяющее условию 1).

Задание №2

Написать функцию, которая запросит два числовых данных, проверит их в соответствии с условием 1 (см. задание 1), для чего использует ранее написанную функцию. Два числа проверит на условие 2 (см. таблицу 4). Если условие ложно, функция выведет сообщение об ошибке и предложит повторить ввод. Последнее действие должно выполняться, пока не будет осуществлен ввод данных, соответствующих условию 2. Входных параметров функция не имеет. Выходными параметрами являются два числа, удовлетворяющих условию 1 и условию 2.

Задание №3

Написать функцию, которая выведет в консоль таблицу вида:

```
/-----------------------\
|  Аргумент |   Функция  |
|-----------------------|
|           |            |
```

```
|           |             |
\-----------------------/
```

Таблица должна быть заполнена данными, поступающими в функцию как аргумент (две переменные, представляющие собой вектора одинаковой длины, элементами которых являются числа). Значения первой переменной функция занесет в первый столбик таблицы, а соответствующие значения второй – во второй столбик таблицы. Пример работы функции приведен ниже.

```
>> x=0:10;
>> y=sin(x);
>>printtable(x,y)
/-----------------------\
|  Аргумент |   Функция  |
|-----------------------|
|     0.0000|      0.0000|
|     1.0000|      0.8415|
|     2.0000|      0.9093|
|     3.0000|      0.1411|
|     4.0000|     -0.7568|
|     5.0000|     -0.9589|
|     6.0000|     -0.2794|
|     7.0000|      0.6570|
|     8.0000|      0.9894|
|     9.0000|      0.4121|
|    10.0000|     -0.5440|
\-----------------------/
```

Задание №4

Написать программу (сценарий), которая запросит у пользователя математическую функцию, запросит интервал для построения графика заданной пользователем функции и проверит введенные значения согласно условию 1 и условию 2 (вызовет ранее написанные функции для заданий 1-2). Далее программа создаст вектор

значений аргумента, рассчитает значения функции для рассчитанных значений аргумента, выведет результат расчета в виде таблицы и построит график функции.

Дополнительное задание: Программа должна запросить шаг построения графика, проверить, что введенное значение является вещественным числом. Если шаг, заданный пользователем, больше или равен разности границ интервала для построения графика, программа выведет сообщение об ошибке и попросит ввести шаг заново. Последнее действие должно выполняться, пока шаг не станет меньшим разности границ интервала.*

Таблица 4. Варианты заданий на лабораторную работу №5

№ Вар-та	Условие для задания №1	Условие для задания №2
1	число является натуральным	Первое из возвращаемых функцией чисел меньше второго не более, чем в 2 раза
2	число делится на 5 только с остатком	Первое из возвращаемых функцией чисел меньше второго
3	число является целым отрицательным	Первое из возвращаемых функцией чисел больше второго и делится на него только с остатком
4	число является положительным	Первое из возвращаемых функцией чисел как минимум в 2 раза меньше второго
5	число является отрицательным	Первое из возвращаемых функцией чисел больше второго
6	число является целым четным	Первое из возвращаемых функцией чисел больше второго максимум в 3 раза
7	число является целым нечетным	Первое из возвращаемых функцией чисел отрицательное, второе - положительное
8	число делится на 3 без остатка	Первое из возвращаемых функцией чисел больше второго
9	число больше 5, но меньше 25	Первое из возвращаемых функцией чисел больше второго
10	число целое и заканчивается на 3	Первое из возвращаемых функцией чисел меньше второго
11	число является целым четным	Первое из возвращаемых функцией чисел положительное, второе - отрицательное
12	число по модулю не больше 100	Первое из возвращаемых функцией чисел больше второго минимум в 2 раза
13	число является целым, но	Первое из возвращаемых функцией

	больше 10	чисел не равно второму и делится на него без остатка
14	число делится на 9 без остатка	Первое из возвращаемых функцией чисел меньше второго
15	число больше 5, но меньше 20	Первое из возвращаемых функцией чисел больше второго
16	число целое и заканчивается на 2	Первое из возвращаемых функцией чисел больше второго максимум в 4 раза
17	число является целым четным, меньшим 50 по модулю	Первое из возвращаемых функцией чисел больше второго минимум в 1,5 раза
18	число по модулю не больше 10	Первое из возвращаемых функцией чисел больше второго
19	число является целым нечетным	Первое из возвращаемых функцией чисел больше второго и делится на него только без остатка
20	число является целым нечетным	Первое из возвращаемых функцией чисел отрицательное, второе - положительное
21	число делится на 6 без остатка	Первое из возвращаемых функцией чисел меньше второго
22	число больше 10, но меньше 50	Первое из возвращаемых функцией чисел больше второго
23	число целое и заканчивается на 1	Первое из возвращаемых функцией чисел положительное, второе - отрицательное
24	число является целым четным	Первое из возвращаемых функцией чисел не равно второму и делится на него без остатка
25	число по модулю не больше 50	Первое из возвращаемых функцией чисел больше второго максимум в 3 раза

Контрольные вопросы

1. Для чего предназначены функции disp, input, fprintf?
2. Приведите примеры ввода значения для переменной строкового и числового форматов с помощью функции input.
3. Приведите пример ввода значения для переменной, представляющей собой вектор, с помощью функции input.
4. Напишите синтаксис функции fprintf и приведите пример ее вызова.
5. Как задать диапазон значений в MATLAB.

6. Что означает специальный символ /n? Для чего онпредназначен.
7. Для чего предназначен символ % в синтаксисе функции fprintf?
8. Что означает спецификатор преобразования в синтаксисе функции fprintf? Для чего он предназначен? Приведите примеры.
9. В чем отличие вывода с помощью функции disp и указания переменной без оператора «;»?
10. Как с помощью функции fprintf вывести переменную X, являющейся вещественным числом, с 2 знаками после запятой после текста «X=»? Напишите синтаксис функции в данном случае.

Требования к содержанию отчета

Отчет по лабораторной работе оформляется в любом текстовом редакторе и предоставляется в электронном виде. Отчет должен состоять из следующих разделов:

1. Титульный лист. На титульном листе необходимо указать номер и название лабораторной работы, номер варианта, ФИО и группу исполнителя, ФИО преподавателя.
2. Цель работы.
3. Задание на лабораторную работу в соответствии с номером варианта.
4. Ход работы:
   - листинги программы и всей подпрограмм;
   - блок-схемы алгоритмов программ и подпрограмм*;
   - результаты работы программы и подпрограмм с демонстрацией ошибки.
5. Выводы по работе.

   К отчету прилагаются:
   - файл с текстом отчета;
   - m-файлы с текстом программы и подпрограмм

   Файл отчета необходимо разместить в личном кабинете.