**Министерство образования и науки Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования «КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ» ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ КАФЕДРА ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ**

**Отчет**

по производственной (проектно-технологической) практике

(наименование практики)

Обучающийся Шабаков Ильвар Жомортханович                  
 (ФИО, группа) (подпись)

Руководитель практики на

кафедре вычислительной математики ассистент, Алексеев С.С.

(должность, ФИО)

оценка за практику

(подпись руководителя практики)

Дата сдачи отчета

КАЗАНЬ – 2020 г.

Содержание

[Постановка задачи 3](#_Toc41844592)

[Решение и анализ задачи 4](#_Toc41844593)

[Вывод 5](#_Toc41844594)

[Листинг программы 10](#_Toc41844595)

[Список используемой литературы 24](#_Toc41844596)

# Постановка задачи

Первое задание заключается в построении квадратичного уравнения с определенным условием: указывается ширина графика по X, что представляет собой диапазон значений по X. Программа должна отвечать требованиям: наличие маркеров для графика, цвета и чекбокса Зафиксировать, позволяющего отрисовать новый график поверх старого. Так же на фигуре должно быть меню, отвечающее за взаимодействие с элементами программы

Второе задание представляет собой отображение важных точек следующим способом:  
добавить на форму два чекбокса, один для выбора вершины параболы, другой для корней уравнения. При выборе соответствующего чекбокса точки должны должны отображаться на графике. Для точек сделать выбор цвета и маркера, отдельно от цвета и маркера графика (при этом цвета и маркеры должны быть другими, не как для графика). Также, если у параболы нет корней, то соответствующие элементы должны быть отключены.

# Решение и анализ задачи

mnEx

mnPlot



inputPanel

cEditFun

weightX

nEditFun

aEditFun

bEditFun

text1ex

text8

text7

text6

text2

markerPanel

text9

text10

rbMarkDot

rbMarkStars

rbMarkNo

rbMarkCircle

Line

massive2

btnClc

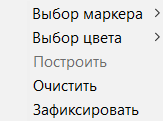
btnGraph

massive1

pmColour

chbxFix

Рис 1. Форма первого задания(некоторые элементы входят в структуру handles)

   
Рис 2. Меню первого задания

menuClear

menuPlot

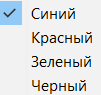
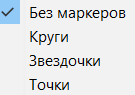
menuColor

menuMarker

menuFix

ex1

ex2



menuColorBlack

menuColorBlue

menuColorGreen

menuColorRed

menuMarkerDot

menuMarkerStars

menuMarkerNone

menuMarkerCirc

Рис 3. Маркеры и цвета первого задания

# 

pmColourDot

text24

pmMarker

text23

xChckBox

text12

text19

hChckBox

Line

Line4

Line3

Line2

btnClc

btnGraph

nEditFun

cEditFun

aEditFun

bEditFun

weightX

text9

text8

text7

text6

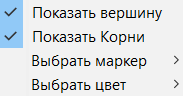
text2

inputPanel

mnEx

mnEx2

Рис 4. Форма второго задания(некоторые элементы входят в структуру handles)

ex2

ex1

Ex2xChbx

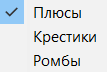
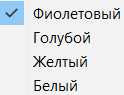
Ex2hChbx

Ex2Color

Ex2Marker

Рис 5. Меню второго задания

menuColorPurple



menuMarkerCross

menuMarkerPlus

menuMarkerRhombus

menuColorBlue2

# 

menuColorWhite

menuColorYellow

# Рис 6. Маркеры и цвета первого задания

При запуске программы программы пользователю предлагается ввести в поле коэффициенты уравнения, после нажатия кнопки на фигуру выводится график, у которого можно сменить маркер и цвет(рис. 7)

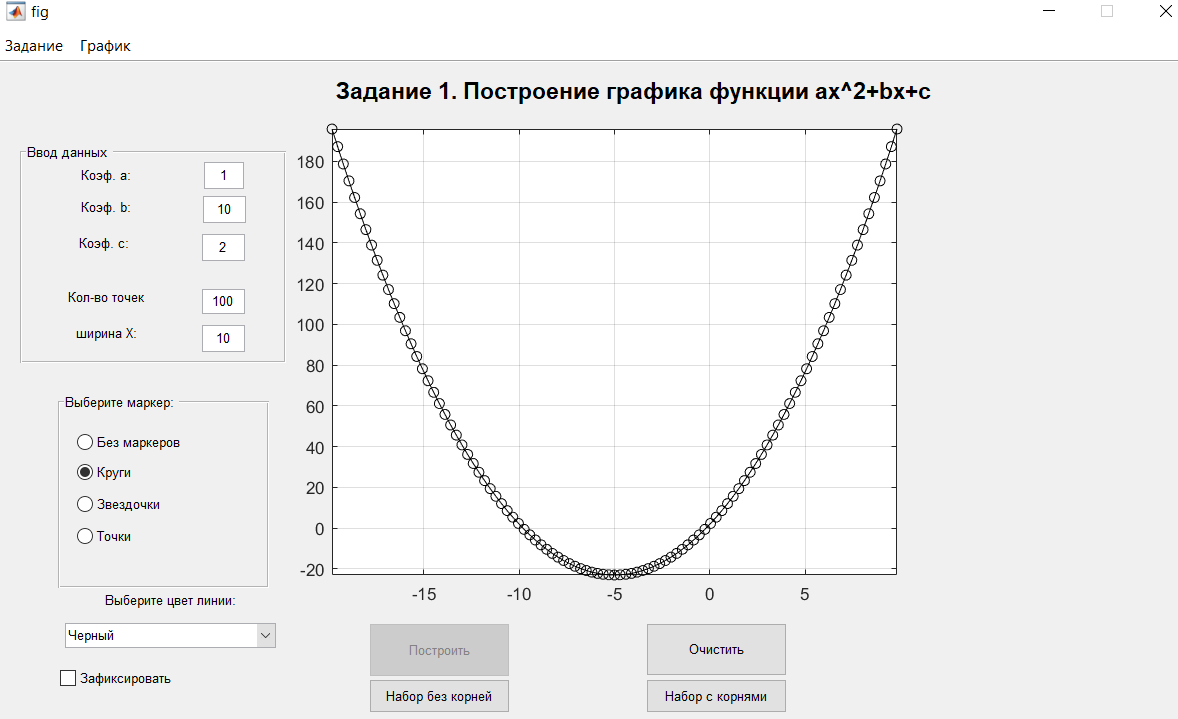


Рис 7. Построение решения квадратичного уравнения

В программе предусмотрено появление ошибки при неправильном вводе данных(рис 8)

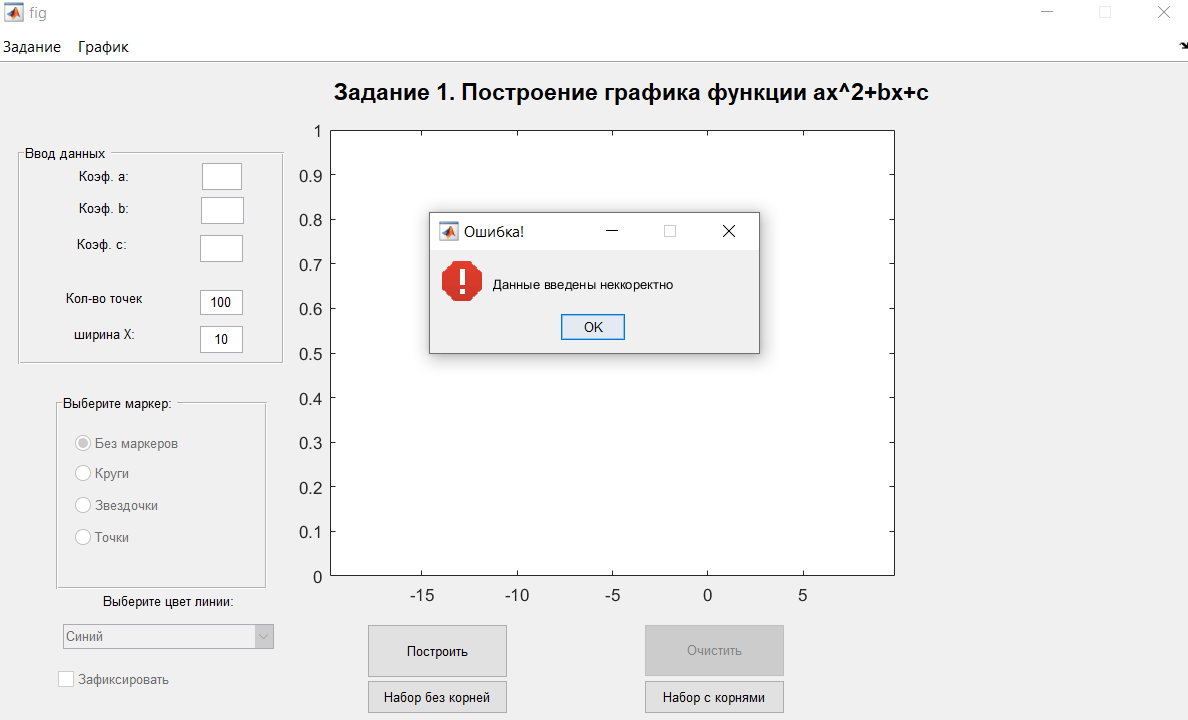


Рис 8. Отлов ошибки

Так же вывод и настройка графика может производиться через меню(все изменения отображаются на форме. При нажатии на чекбокс Зафиксировать происходит наложение нового графика на старый(рис 9)

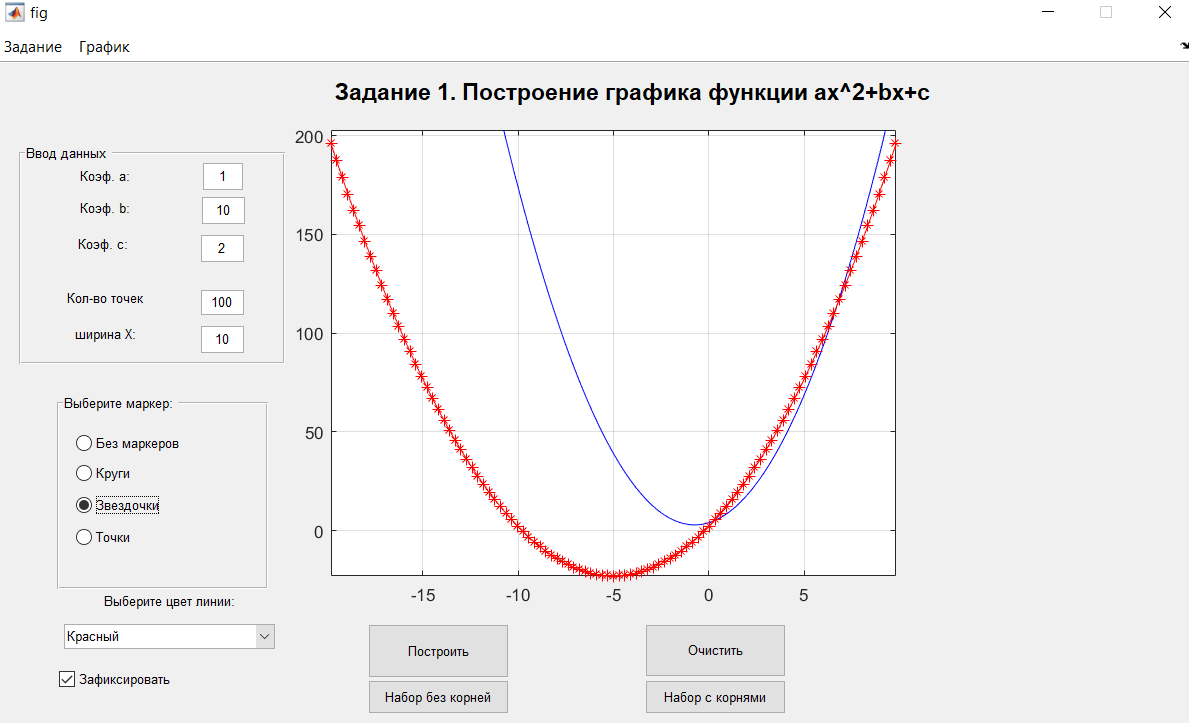


Рис 9. Фиксирование графика

Кнопка Очистить возвращает программу к изначальному состоянию, закрывая доступ к маркерам и цвету для предотвращения ошибок(рис 10)

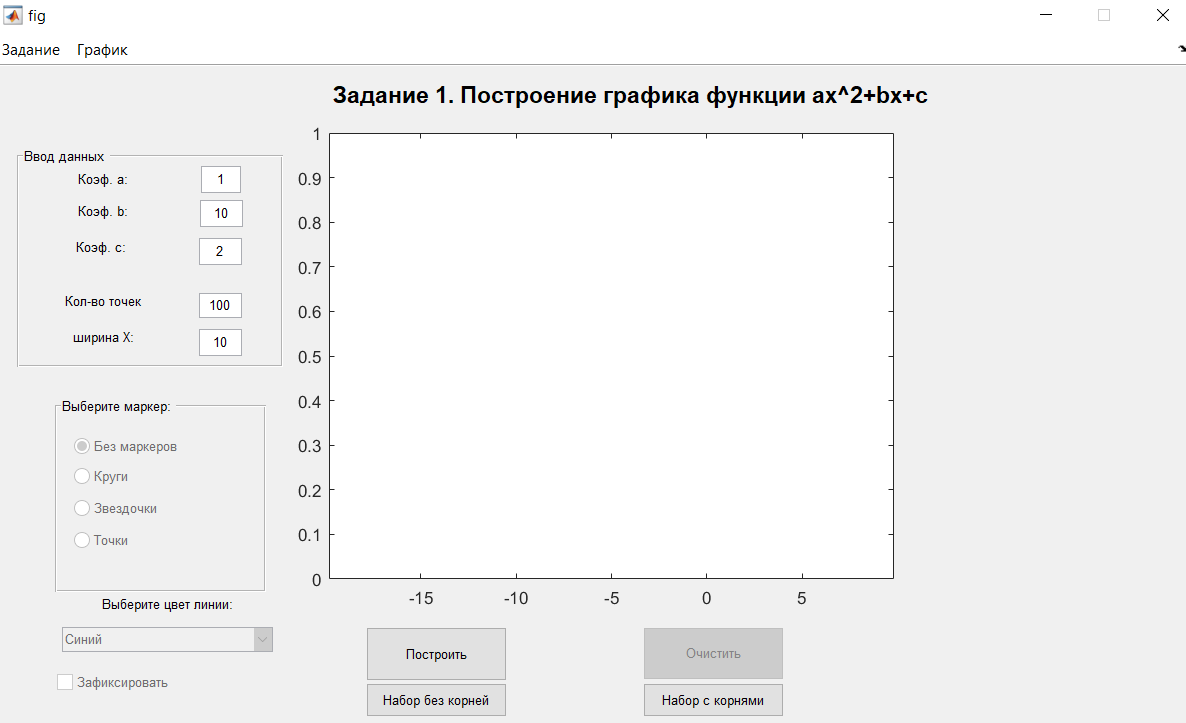


Рис 10. Очищение формы

С помощью меню Задание можно переключать программу между заданиями.

Второе задание представляет собой вывод вершины и корней(при возможности) на график. Так же, как и у графика, у них имеется выбор маркера и цвета(рис 11)

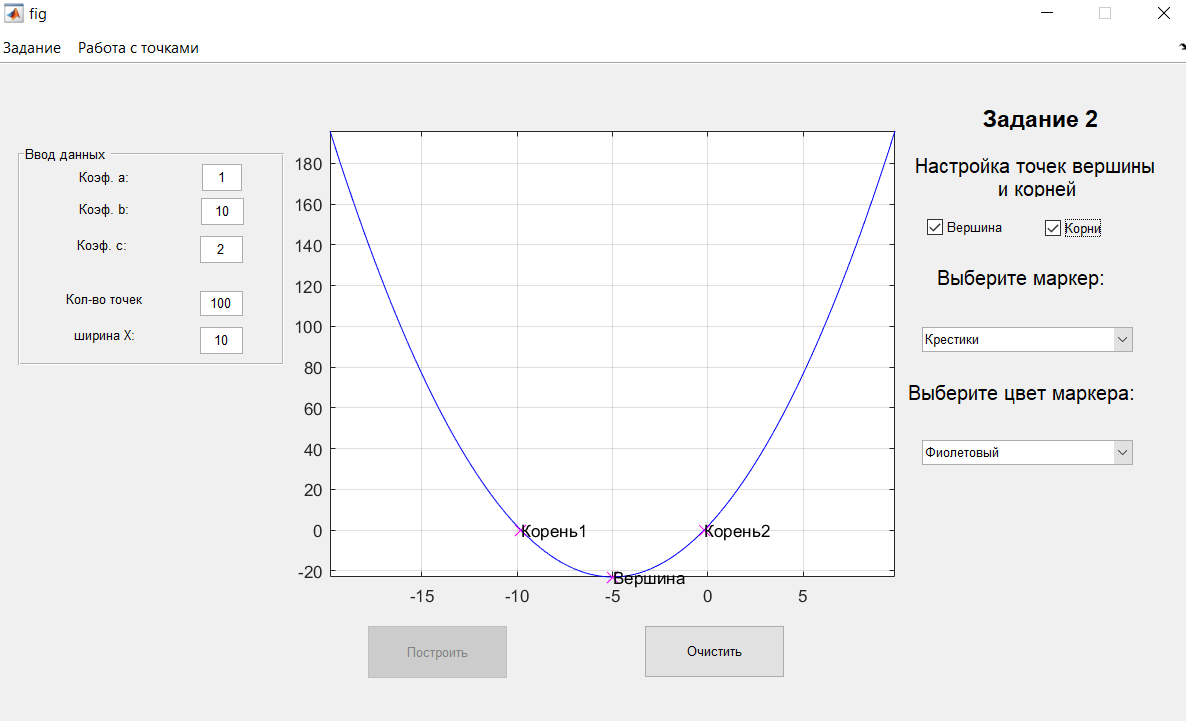


Рис 11. Второе задание

Все то же самое можно проделать и через меню Работа с точками(изменения дублируются на форме)

При нажатии на кнопку Очистить программа возвращается к изначальному состоянию, всплывающие меню блокируются для предотвращения ошибок(рис 12)

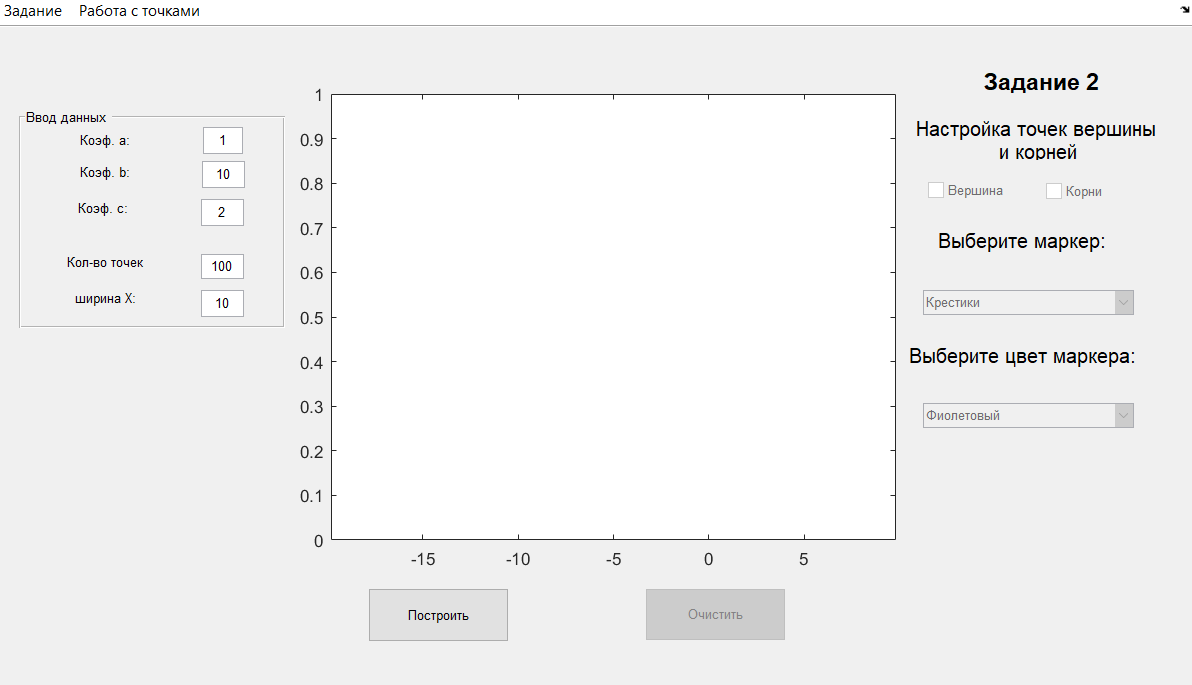


Рис 12. Начальное состояние второго задания

# Вывод

Поставленная задача была реализована в среде Guide. В ходе выполнения были получены различные навыки и понимания принципа программирования функций в матлабе и взаимодействия элементов между собой.

Основная часть программы затруднений не вызвала, так как выполнение подобных задач было рассмотрено на практических занятиях. Некоторые моменты приходилось изучать самостоятельно, к примеру, сохранение переменных командой save для использования их в других функций. Или команда axis, благодаря которой можно было обрезать неиспользуемое пространство двумерной плоскости. Второе задание было схоже с функционалом с первым, почему и не вызвало особых затруднений.

# Листинг программы

function varargout = fig(varargin)

gui\_Singleton = 1;

gui\_State = struct('gui\_Name', mfilename, ...

'gui\_Singleton', gui\_Singleton, ...

'gui\_OpeningFcn', @fig\_OpeningFcn, ...

'gui\_OutputFcn', @fig\_OutputFcn, ...

'gui\_LayoutFcn', [] , ...

'gui\_Callback', []);

if nargin && ischar(varargin{1})

gui\_State.gui\_Callback = str2func(varargin{1});

end

if nargout

[varargout{1:nargout}] = gui\_mainfcn(gui\_State, varargin{:});

else

gui\_mainfcn(gui\_State, varargin{:});

end

function fig\_OpeningFcn(hObject, eventdata, handles, varargin)

handles.output = hObject;

guidata(hObject, handles);

function varargout = fig\_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;

function btnGraph\_Callback(hObject, eventdata, handles)

clc

global a

global b

global c

global D

try

a=str2num(get(handles.aEditFun, 'String'))

b=str2num(get(handles.bEditFun, 'String'))

c=str2num(get(handles.cEditFun, 'String'))

N=str2num(get(handles.nEditFun, 'String'))

weightX=str2num(get(handles.weightX, 'String'))

hFun=-b/(2\*a);

save('h','hFun');

D=b\*b-4\*c;

check=0;

negativeX=0;

positiveX=0;

if D<=0;

h=abs(((hFun+weightX)-(hFun-weightX)))/N;

negativeX=hFun-weightX;

positiveX=hFun+weightX;

x=negativeX:h:positiveX;

set(handles.Ex2xChbx,'Enable','off')

else

x1=(-b-sqrt(D))/(2\*a)

x2=(-b+sqrt(D))/(2\*a)

save('xx','x1','x2');

h=abs(((x2+weightX)-(x1-weightX)))/N;

negativeX=x1-weightX;

positiveX=x2+weightX;

x=negativeX:h:positiveX

check=1;

set(handles.xChckBox,'Enable','on')

set(handles.Ex2xChbx,'Enable','on')

end

set(handles.hChckBox,'Enable','on')

set(handles.Ex2hChbx,'Enable','on')

%выключение корней при фиксации

if check==0

set(handles.xChckBox,'Enable','off')

set(handles.Ex2xChbx,'Enable','off')

end

handles.checkVarH=0;

handles.checkVarX=0;

y=f(x);

handles.Line=plot(x,y);

%при фиксации сброс маркера и цвета

if get(handles.chbxFix,'Value')==1

set(handles.pmColour, 'Value',1)

set(handles.Line, 'Marker','none')

set(handles.rbMarkNo,'Value',1)

end

pmColour\_Callback(hObject, eventdata, handles)

axis([ negativeX positiveX -inf inf])

grid on

set(handles.btnClc,'Enable','on')

set(handles.pmColour,'Enable','on')

set(handles.rbMarkNo,'Enable','on')

set(handles.rbMarkCircle,'Enable','on')

set(handles.rbMarkStars,'Enable','on')

set(handles.rbMarkDot,'Enable','on')

set(handles.chbxFix,'Enable','on')

set(handles.pmColourDot,'Enable','on')

set(handles.pmMarker,'Enable','on')

guidata(gcbo,handles);

set(handles.menuClear,'Enable','on')

set(handles.menuFix,'Enable','on')

set(handles.menuFix,'Enable','on')

set(handles.menuPlot,'Enable','off')

%выключение кнопкпи Построить при отсутствии фиксации

if get(handles.chbxFix, 'Value')==0

set(handles.btnGraph,'Enable','off')

end

%выключение вершины и корней построения при фиксации

if get(handles.chbxFix,'Value')==1 & strcmp(get(handles.Ex2hChbx,'Checked'),'on')

Ex2hChbx\_Callback(handles.Ex2hChbx, eventdata, handles)

end

if get(handles.chbxFix,'Value')==1 & strcmp(get(handles.Ex2xChbx,'Checked'),'on')

Ex2xChbx\_Callback(handles.Ex2xChbx, eventdata, handles)

end

rbMarkNo\_Callback(hObject, eventdata, handles)

set(handles.menuMarker,'Enable','on')

set(handles.menuColor,'Enable','on')

set(handles.Ex2Color,'Enable','on')

set(handles.Ex2Marker,'Enable','on')

catch

errordlg('Данные введены неккоректно', 'Ошибка!')

set(handles.xChckBox,'Enable','off')

set(handles.Ex2xChbx,'Enable','off')

set(handles.hChckBox,'Enable','off')

set(handles.Ex2hChbx,'Enable','off')

end

function btnClc\_Callback(hObject, eventdata, handles)

button=questdlg('Очистить оси?','Очистка','Да','Нет','Да');

if strcmp(button,'Да')

menuColorBlue\_Callback(handles.menuColorBlue, eventdata, handles)

menuMarkerNone\_Callback(handles.menuMarkerNone, eventdata, handles)

cla

clc

set(hObject,'Enable','off')

set(handles.pmColour,'Enable','off')

set(handles.rbMarkNo,'Enable','off')

set(handles.rbMarkCircle,'Enable','off')

set(handles.rbMarkStars,'Enable','off')

set(handles.rbMarkDot,'Enable','off')

set(handles.chbxFix,'Enable','off')

set(handles.rbMarkNo,'Value',1)

set(handles.chbxFix,'Value',0)

grid off

set(handles.pmColourDot,'Enable','off')

set(handles.pmMarker,'Enable','off')

set(handles.Ex2hChbx,'Checked','off')

set(handles.Ex2xChbx,'Checked','off')

set(handles.Ex2hChbx,'Enable','off')

set(handles.Ex2xChbx,'Enable','off')

guidata(gcbo,handles);

set(handles.pmColourDot,'Value',1)

set(handles.hChckBox,'Enable','off')

set(handles.hChckBox,'Value',0)

set(handles.xChckBox,'Enable','off')

set(handles.xChckBox,'Value',0)

set(handles.menuMarker,'Enable','off')

set(handles.menuColor,'Enable','off')

set(handles.menuClear,'Enable','off')

set(handles.menuFix,'Enable','off')

set(handles.menuFix,'Enable','off')

set(handles.menuPlot,'Enable','on')

set(handles.btnGraph,'Enable','on')

handles.checkVarH =0;

handles.checkVarX =0;

set(handles.Ex2Color,'Enable','off')

set(handles.Ex2Marker,'Enable','off')

set(handles.menuFix,'Checked','off')

end

function aEditFun\_Callback(hObject, eventdata, handles)

function aEditFun\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

function bEditFun\_Callback(hObject, eventdata, handles)

function bEditFun\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

function cEditFun\_Callback(hObject, eventdata, handles)

function cEditFun\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

function nEditFun\_Callback(hObject, eventdata, handles)

function nEditFun\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

function weightX\_Callback(hObject, eventdata, handles)

function weightX\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

function chbxFix\_Callback(hObject, eventdata, handles)

if get(handles.chbxFix, 'Value')

hold on

set(handles.btnGraph,'Enable','on')

set(handles.menuFix,'Checked','on')

else

hold off

set(handles.btnGraph,'Enable','off')

set(handles.menuFix,'Checked','off')

end

function pmColour\_Callback(hObject, eventdata, handles)

Num=get(handles.pmColour, 'Value');

switch Num

case 1

set(handles.Line, 'Color', 'b');

set(handles.menuColorBlue,'Checked','on')

set(handles.menuColorRed,'Checked','off')

set(handles.menuColorGreen,'Checked','off')

set(handles.menuColorBlack,'Checked','off')

case 2

set(handles.Line, 'Color', 'r');

set(handles.menuColorBlue,'Checked','off')

set(handles.menuColorRed,'Checked','on')

set(handles.menuColorGreen,'Checked','off')

set(handles.menuColorBlack,'Checked','off')

case 3

set(handles.Line, 'Color', 'g');

set(handles.menuColorBlue,'Checked','off')

set(handles.menuColorRed,'Checked','off')

set(handles.menuColorGreen,'Checked','on')

set(handles.menuColorBlack,'Checked','off')

case 4

set(handles.Line, 'Color', 'k');

set(handles.menuColorBlue,'Checked','off')

set(handles.menuColorRed,'Checked','off')

set(handles.menuColorGreen,'Checked','off')

set(handles.menuColorBlack,'Checked','on')

end

function pmColour\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

function rbMarkNo\_Callback(hObject, eventdata, handles)

set(handles.Line, 'Marker','none')

set(handles.menuMarkerNone,'Checked','on')

set(handles.menuMarkerCirc,'Checked','off')

set(handles.menuMarkerStars,'Checked','off')

set(handles.menuMarkerDot,'Checked','off')

function rbMarkCircle\_Callback(hObject, eventdata, handles)

set(handles.Line, 'Marker', 'o')

set(handles.menuMarkerCirc,'Checked','on')

set(handles.menuMarkerNone,'Checked','off')

set(handles.menuMarkerStars,'Checked','off')

set(handles.menuMarkerDot,'Checked','off')

function rbMarkStars\_Callback(hObject, eventdata, handles)

set(handles.Line, 'Marker', '\*')

set(handles.menuMarkerStars,'Checked','on')

set(handles.menuMarkerStars,'Checked','on')

set(handles.menuMarkerCirc,'Checked','off')

set(handles.menuMarkerNone,'Checked','off')

set(handles.menuMarkerDot,'Checked','off')

function rbMarkDot\_Callback(hObject, eventdata, handles)

set(handles.Line, 'Marker', '.')

set(handles.menuMarkerDot,'Checked','on')

set(handles.menuMarkerStars,'Checked','off')

set(handles.menuMarkerCirc,'Checked','off')

set(handles.menuMarkerNone,'Checked','off')

function mnEx\_Callback(hObject, eventdata, handles)

function mnPlot\_Callback(hObject, eventdata, handles)

function menuMarker\_Callback(hObject, eventdata, handles)

function menuColor\_Callback(hObject, eventdata, handles)

function menuPlot\_Callback(hObject, eventdata, handles)

btnGraph\_Callback(handles.btnGraph, eventdata, handles)

function menuClear\_Callback(hObject, eventdata, handles)

btnClc\_Callback(handles.btnClc, eventdata, handles)

function menuFix\_Callback(hObject, eventdata, handles)

if strcmp(get(hObject,'Checked'),'on')

set(hObject,'Checked','off')

set(handles.chbxFix,'Value',0)

hold off

set(handles.btnGraph,'Enable','off')

else

set(hObject,'Checked','on')

set(handles.chbxFix,'Value',1)

hold on

set(handles.btnGraph,'Enable','on')

end

function ex1\_Callback(hObject, eventdata, handles)

set(handles.ex2,'Checked','off')

set(hObject,'Checked','on')

set(handles.markerPanel,'Visible','on')

set(handles.text10,'Visible','on')

set(handles.pmColour,'Visible','on')

set(handles.chbxFix,'Visible','on')

set(handles.mnPlot,'Visible','on')

set(handles.text1ex,'Visible','on')

set(handles.massive1,'Visible','on')

set(handles.massive2,'Visible','on')

set(handles.hChckBox,'Visible','off')

set(handles.xChckBox,'Visible','off')

set(handles.text12,'Visible','off')

set(handles.pmColourDot,'Visible','off')

set(handles.pmMarker,'Visible','off')

set(handles.text19,'Visible','off')

set(handles.mnEx2,'Visible','off')

set(handles.text23,'Visible','off')

set(handles.text12,'Visible','off')

set(handles.text24,'Visible','off')

function ex2\_Callback(hObject, eventdata, handles)

set(handles.ex1,'Checked','off')

set(hObject,'Checked','on')

set(handles.markerPanel,'Visible','off')

set(handles.text10,'Visible','off')

set(handles.pmColour,'Visible','off')

set(handles.chbxFix,'Visible','off')

set(handles.mnPlot,'Visible','off')

set(handles.text1ex,'Visible','off')

set(handles.massive1,'Visible','off')

set(handles.massive2,'Visible','off')

set(handles.hChckBox,'Visible','on')

set(handles.xChckBox,'Visible','on')

set(handles.text12,'Visible','on')

set(handles.text23,'Visible','on')

set(handles.text24,'Visible','on')

set(handles.pmColourDot,'Visible','on')

set(handles.pmMarker,'Visible','on')

set(handles.text19,'Visible','on')

set(handles.mnEx2,'Visible','on')

guidata(gcbo,handles);

function menuColorBlue\_Callback(hObject, eventdata, handles)

set(handles.pmColour,'Value',1)

pmColour\_Callback(hObject, eventdata, handles)

function menuColorRed\_Callback(hObject, eventdata, handles)

set(handles.pmColour,'Value',2)

pmColour\_Callback(hObject, eventdata, handles)

function menuColorGreen\_Callback(hObject, eventdata, handles)

set(handles.pmColour,'Value',3)

pmColour\_Callback(hObject, eventdata, handles)

function menuColorBlack\_Callback(hObject, eventdata, handles)

set(handles.pmColour,'Value',4)

pmColour\_Callback(hObject, eventdata, handles)

function menuMarkerNone\_Callback(hObject, eventdata, handles)

rbMarkNo\_Callback(hObject, eventdata, handles)

set(handles.rbMarkNo,'Value',1)

function menuMarkerCirc\_Callback(hObject, eventdata, handles)

rbMarkCircle\_Callback(hObject, eventdata, handles)

set(handles.rbMarkCircle,'Value',1)

function menuMarkerStars\_Callback(hObject, eventdata, handles)

rbMarkStars\_Callback(hObject, eventdata, handles)

set(handles.rbMarkStars,'Value',1)

function menuMarkerDot\_Callback(hObject, eventdata, handles)

rbMarkDot\_Callback(hObject, eventdata, handles)

set(handles.rbMarkDot,'Value',1)

function hChckBox\_Callback(hObject, eventdata, handles)

if get(hObject, 'Value')==1

set(handles.Ex2hChbx,'Checked','on')

handles.checkVarH=1;

hold on

h=load('h');

handles.Line2=plot(h.hFun,f(h.hFun),'MarkerSize',10)

handles.texth=text(h.hFun,f(h.hFun),'Вершина')

guidata(gcbo,handles);

set(handles.hChckBox,'Enable','on')

set(handles.pmColourDot,'Enable','on')

set(handles.pmMarker,'Enable','on')

pmMarker\_Callback(hObject, eventdata, handles)

pmColourDot\_Callback(hObject, eventdata, handles)

else

handles.checkVarH=0;

guidata(gcbo,handles);

delete(handles.Line2);

delete(handles.texth);

set(handles.Ex2hChbx,'Checked','off')

end

function xChckBox\_Callback(hObject, eventdata, handles)

if get(hObject, 'Value')==1

handles.checkVarX=1;

set(handles.Ex2xChbx,'Checked','on')

hold on

xx=load('xx')

handles.Line3=plot(xx.x1,f(xx.x1),'MarkerSize',10)

handles.texts1=text(xx.x1,f(xx.x1),'Корень1')

handles.Line4=plot(xx.x2,f(xx.x2),'MarkerSize',10)

handles.texts2=text(xx.x2,f(xx.x2),'Корень2')

guidata(gcbo,handles);

set(handles.xChckBox,'Enable','on')

set(handles.pmColourDot,'Enable','on')

set(handles.pmMarker,'Enable','on')

pmMarker\_Callback(hObject, eventdata, handles)

pmColourDot\_Callback(hObject, eventdata, handles)

else

handles.checkVarX=0;

set(handles.Ex2xChbx,'Checked','off')

guidata(gcbo,handles);

delete(handles.Line3);

delete(handles.Line4);

delete(handles.texts1);

delete(handles.texts2);

end

function pmColourDot\_Callback(hObject, eventdata, handles)

Num=get(handles.pmColourDot, 'Value');

switch Num

case 1

set(handles.menuColorPurple,'Checked','on')

set(handles.menuColorBlue2,'Checked','off')

set(handles.menuColorYellow,'Checked','off')

set(handles.menuColorWhite,'Checked','off')

if handles.checkVarH==1

set(handles.Line2, 'Color', 'm');

end

if handles.checkVarX ==1

set(handles.Line3, 'Color', 'm');

set(handles.Line4, 'Color', 'm');

end

case 2

set(handles.menuColorPurple,'Checked','off')

set(handles.menuColorBlue2,'Checked','on')

set(handles.menuColorYellow,'Checked','off')

set(handles.menuColorWhite,'Checked','off')

if handles.checkVarH==1

set(handles.Line2, 'Color', 'c');

end

if handles.checkVarX ==1

set(handles.Line3, 'Color', 'c');

set(handles.Line4, 'Color', 'c');

end

case 3

set(handles.menuColorPurple,'Checked','off')

set(handles.menuColorBlue2,'Checked','off')

set(handles.menuColorYellow,'Checked','on')

set(handles.menuColorWhite,'Checked','off')

if handles.checkVarH==1

set(handles.Line2, 'Color', 'y');

end

if handles.checkVarX ==1

set(handles.Line3, 'Color', 'y');

set(handles.Line4, 'Color', 'y');

end

case 4

set(handles.menuColorPurple,'Checked','off')

set(handles.menuColorBlue2,'Checked','off')

set(handles.menuColorYellow,'Checked','off')

set(handles.menuColorWhite,'Checked','on')

if handles.checkVarH==1

set(handles.Line2, 'Color', 'w');

end

if handles.checkVarX ==1

set(handles.Line3, 'Color', 'w');

set(handles.Line4, 'Color', 'w');

end

end

function pmColourDot\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

function mnEx2\_Callback(hObject, eventdata, handles)

function Ex2hChbx\_Callback(hObject, eventdata, handles)

if strcmp(get(hObject,'Checked'),'on')

set(hObject,'Checked','off')

set(handles.hChckBox, 'Value',0)

hChckBox\_Callback(handles.hChckBox, eventdata, handles)

else

set(hObject,'Checked','on')

set(handles.hChckBox, 'Value',1)

hChckBox\_Callback(handles.hChckBox, eventdata, handles)

end

function Ex2xChbx\_Callback(hObject, eventdata, handles)

if strcmp(get(hObject,'Checked'),'on')

set(hObject,'Checked','off')

set(handles.xChckBox, 'Value',0)

xChckBox\_Callback(handles.xChckBox, eventdata, handles)

else

set(hObject,'Checked','on')

set(handles.xChckBox, 'Value',1)

xChckBox\_Callback(handles.xChckBox, eventdata, handles)

end

function Ex2Marker\_Callback(hObject, eventdata, handles)

function Ex2Color\_Callback(hObject, eventdata, handles)

function pmMarker\_Callback(hObject, eventdata, handles)

Num=get(handles.pmMarker, 'Value');

switch Num

case 1

set(handles.menuMarkerPlus,'Checked','on')

set(handles.menuMarkerCross,'Checked','off')

set(handles.menuMarkerRhombus,'Checked','off')

if handles.checkVarH==1

set(handles.Line2,'Marker','+')

end

if handles.checkVarX ==1

set(handles.Line3, 'Marker','+')

set(handles.Line4, 'Marker','+')

end

case 2

set(handles.menuMarkerPlus,'Checked','off')

set(handles.menuMarkerCross,'Checked','on')

set(handles.menuMarkerRhombus,'Checked','off')

if handles.checkVarH==1

set(handles.Line2, 'Marker', 'x')

end

if handles.checkVarX==1

set(handles.Line3, 'Marker','x')

set(handles.Line4, 'Marker','x')

end

case 3

set(handles.menuMarkerPlus,'Checked','off')

set(handles.menuMarkerCross,'Checked','off')

set(handles.menuMarkerRhombus,'Checked','on')

if handles.checkVarH==1

set(handles.Line2, 'Marker', 'd')

end

if handles.checkVarX==1

set(handles.Line3, 'Marker','d')

set(handles.Line4, 'Marker','d')

end

end

function pmMarker\_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))

set(hObject,'BackgroundColor','white');

end

function y=f(x)

global a

global b

global c

y=a.\*x.^2+b.\*x+c

function massive1\_Callback(hObject, eventdata, handles)

set(handles.aEditFun,'String',2)

set(handles.bEditFun,'String',3)

set(handles.cEditFun,'String',4)

function massive2\_Callback(hObject, eventdata, handles)

set(handles.aEditFun,'String',1)

set(handles.bEditFun,'String',10)

set(handles.cEditFun,'String',2)

function menuColorPurple\_Callback(hObject, eventdata, handles)

set(handles.pmColourDot,'Value',1)

pmColourDot\_Callback(hObject, eventdata, handles)

function menuColorBlue2\_Callback(hObject, eventdata, handles)

set(handles.pmColourDot,'Value',2)

pmColourDot\_Callback(hObject, eventdata, handles)

function menuColorYellow\_Callback(hObject, eventdata, handles)

set(handles.pmColourDot,'Value',3)

pmColourDot\_Callback(hObject, eventdata, handles)

function menuColorWhite\_Callback(hObject, eventdata, handles)

set(handles.pmColourDot,'Value',4)

pmColourDot\_Callback(hObject, eventdata, handles)

function menuMarkerPlus\_Callback(hObject, eventdata, handles)

set(handles.pmMarker,'Value',1)

pmMarker\_Callback(hObject, eventdata, handles)

function menuMarkerCross\_Callback(hObject, eventdata, handles)

set(handles.pmMarker,'Value',2)

pmMarker\_Callback(hObject, eventdata, handles)

function menuMarkerRhombus\_Callback(hObject, eventdata, handles)

set(handles.pmMarker,'Value',3)

pmMarker\_Callback(hObject, eventdata, handles)

# Список используемой литературы

1. И. Ануфриев. MATLAB 7.0. Наиболее полное руководство, 2015. – 615 с.
2. https://hub.exponenta.ru/ [интернет - ресурс]- Сообщество Экспонента