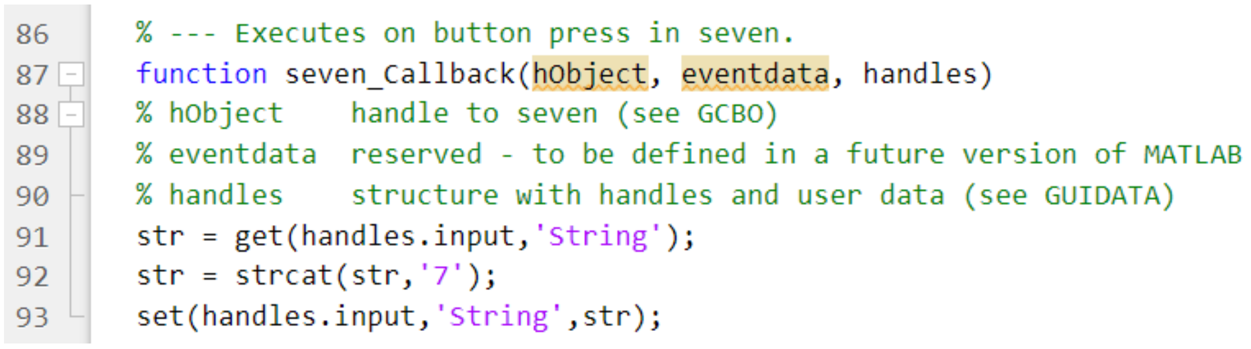
Introduction:

For our GUI project, we decided to build a simple calculator which is capable performing simple mathematical operations, and solving differential and integral operations of algebraic expressions as well as display the solutions' graphs. The virtual calculator consists of 27 buttons. The user interface has been kept as simple as possible for the convenience of users. When it comes to calculus problems, it can solve either a differential or an integral expression at a time. As the calculator offers the function of plotting a derivative/integral's solution, it helps the user visualize the solution much better and understand the implications of the graph.

Methodology:

Input Buttons:

We initially created a framework for the calculator by using GUIDE and placed the required number of push buttons, displays and plotting axes. Based on this framework, we obtained a MATLAB code where slight modifications were introduced under the function code of each of the buttons, display and axis.

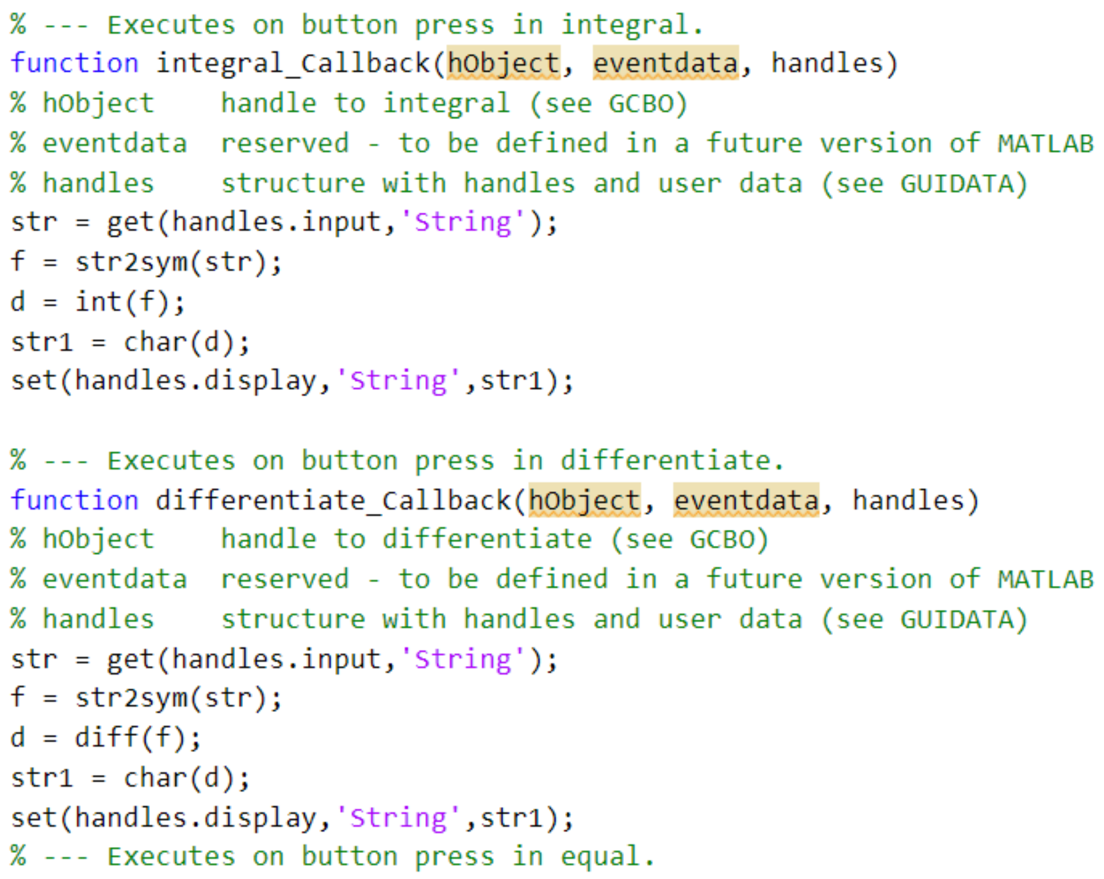


```
% --- Executes on button press in seven.
function seven_Callback(hObject, eventdata, handles)
% hObject    handle to seven (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'7');
set(handles.input,'String',str);
```

Fig:01

The above segment of code was utilised and repeated in taking normal inputs from the numeric and function buttons of the calculator. We can see from the above code that each of the buttons is initially taken as a string in order to be evaluated later on. The above code was basically repeated for all the buttons except the clear (clc), derivative and integral functions.

Derivative/Integral Buttons:



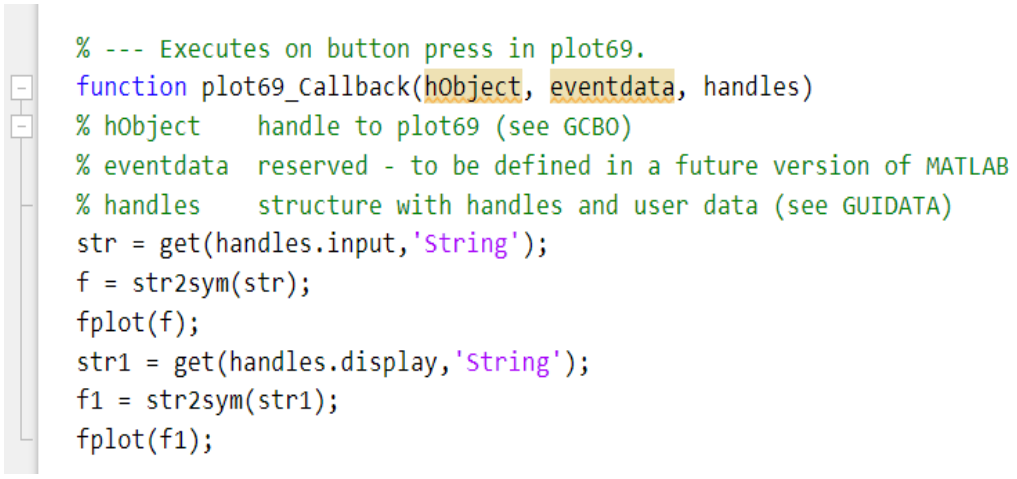
```
% --- Executes on button press in integral.
function integral_Callback(hObject, eventdata, handles)
% hObject    handle to integral (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
f = str2sym(str);
d = int(f);
str1 = char(d);
set(handles.display,'String',str1);

% --- Executes on button press in differentiate.
function differentiate_Callback(hObject, eventdata, handles)
% hObject    handle to differentiate (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
f = str2sym(str);
d = diff(f);
str1 = char(d);
set(handles.display,'String',str1);
% --- Executes on button press in equal.
```

Fig:02

This segment converts the input string into a symbolic function by using the str2sym() function, which is then later differentiated or integrated based on the option being selected. The output symbolic function is then required to be converted back to string, in order to be displayed in the output window.

Plotting Function:

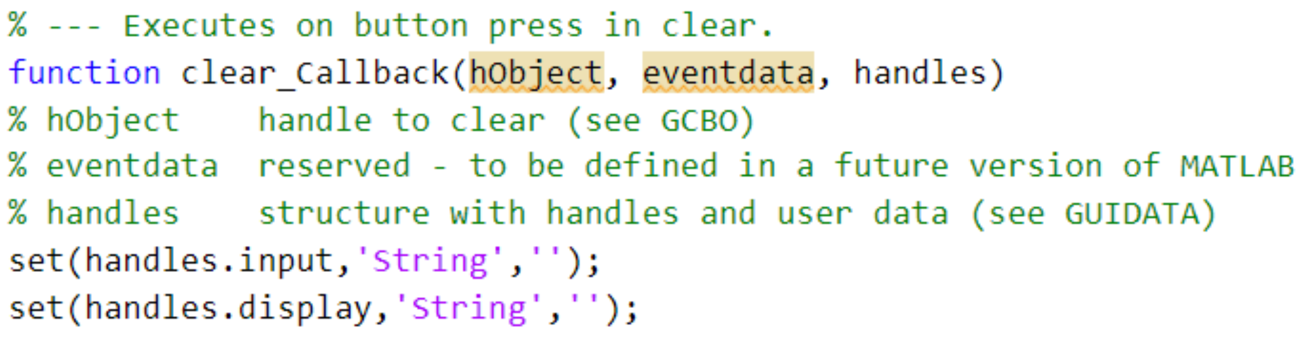


```
% --- Executes on button press in plot69.
function plot69_Callback(hObject, eventdata, handles)
% hObject    handle to plot69 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
f = str2sym(str);
fplot(f);
str1 = get(handles.display,'String');
f1 = str2sym(str1);
fplot(f1);
```

Fig:03

This section allows us to plot the output waveform, where the symbolic function is plotted using the built in fplot() function.

Clear Option:



```
% --- Executes on button press in clear.
function clear_Callback(hObject, eventdata, handles)
% hObject    handle to clear (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.input,'String','');
set(handles.display,'String','');
```

Fig:04

Ensuring that the displays are made empty after each operation

Code:

```
function varargout = int_diff_calculator(varargin)
%INT_DIFF_CALCULATOR MATLAB code file for int_diff_calculator.fig
%      INT_DIFF_CALCULATOR, by itself, creates a new
INT_DIFF_CALCULATOR or raises the existing
%      singleton*.
```

```
%
%      H = INT_DIFF_CALCULATOR returns the handle to a new
INT_DIFF_CALCULATOR or the handle to
%      the existing singleton*.
%
%      INT_DIFF_CALCULATOR('Property','Value',...) creates a new
INT_DIFF_CALCULATOR using the
%      given property value pairs. Unrecognized properties are passed
via
%      varargin to int_diff_calculator_OpeningFcn.  This calling
syntax produces a
%      warning when there is an existing singleton*.
%
%      INT_DIFF_CALCULATOR('CALLBACK') and
INT_DIFF_CALCULATOR('CALLBACK',hObject,...) call the
%      local function named CALLBACK in INT_DIFF_CALCULATOR.M with
the given input
%      arguments.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
int_diff_calculator

% Last Modified by GUIDE v2.5 21-Apr-2023 03:51:18

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @int_diff_calculator_OpeningFcn,
...
                   'gui_OutputFcn',  @int_diff_calculator_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{:});
```

```matlab
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before int_diff_calculator is made visible.
function int_diff_calculator_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no display args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   unrecognized PropertyName/PropertyValue pairs from the
%            command line (see VARARGIN)

% Choose default command line display for int_diff_calculator
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes int_diff_calculator wait for user response (see
UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = int_diff_calculator_OutputFcn(hObject,
eventdata, handles)
% varargout  cell array for returning display args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line display from handles structure
varargout{1} = handles.output;


% --- Executes on button press in left_bracket.
function left_bracket_Callback(hObject, eventdata, handles)
% hObject    handle to left_bracket (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'(');
set(handles.input,'String',str);

% --- Executes on button press in seven.
function seven_Callback(hObject, eventdata, handles)
% hObject    handle to seven (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'7');
set(handles.input,'String',str);

% --- Executes on button press in eight.
function eight_Callback(hObject, eventdata, handles)
% hObject    handle to eight (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'8');
set(handles.input,'String',str);

% --- Executes on button press in nine.
function nine_Callback(hObject, eventdata, handles)
% hObject    handle to nine (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'9');
set(handles.input,'String',str);

% --- Executes on button press in four.
function four_Callback(hObject, eventdata, handles)
% hObject    handle to four (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'4');
set(handles.input,'String',str);

% --- Executes on button press in five.
function five_Callback(hObject, eventdata, handles)
% hObject    handle to five (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'5');
set(handles.input,'String',str);
```

```matlab
% --- Executes on button press in six.
function six_Callback(hObject, eventdata, handles)
% hObject    handle to six (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'6');
set(handles.input,'String',str);

% --- Executes on button press in one.
function one_Callback(hObject, eventdata, handles)
% hObject    handle to one (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'1');
set(handles.input,'String',str);

% --- Executes on button press in two.
function two_Callback(hObject, eventdata, handles)
% hObject    handle to two (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'2');
set(handles.input,'String',str);

% --- Executes on button press in three.
function three_Callback(hObject, eventdata, handles)
% hObject    handle to three (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'3');
set(handles.input,'String',str);

% --- Executes on button press in zero.
function zero_Callback(hObject, eventdata, handles)
% hObject    handle to zero (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'0');
set(handles.input,'String',str);
```

```matlab
% --- Executes on button press in dot.
function dot_Callback(hObject, eventdata, handles)
% hObject    handle to dot (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'.');
set(handles.input,'String',str);

% --- Executes on button press in clear.
function clear_Callback(hObject, eventdata, handles)
% hObject    handle to clear (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.input,'String','');
set(handles.display,'String','');
% --- Executes on button press in right_bracket.
function right_bracket_Callback(hObject, eventdata, handles)
% hObject    handle to right_bracket (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,')');
set(handles.input,'String',str);

% --- Executes on button press in sqrt69.
function sqrt69_Callback(hObject, eventdata, handles)
% hObject    handle to sqrt69 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'sqrt');
set(handles.input,'String',str);

% --- Executes on button press in sin69.
function sin69_Callback(hObject, eventdata, handles)
% hObject    handle to sin69 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'sin');
set(handles.input,'String',str);


% --- Executes on button press in cos69.
function cos69_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to cos69 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'cos');
set(handles.input,'String',str);


% --- Executes on button press in tan69.
function tan69_Callback(hObject, eventdata, handles)
% hObject    handle to tan69 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'tan');
set(handles.input,'String',str);


% --- Executes on button press in power.
function power_Callback(hObject, eventdata, handles)
% hObject    handle to power (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'^');
set(handles.input,'String',str);


% --- Executes on button press in x.
function x_Callback(hObject, eventdata, handles)
% hObject    handle to x (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'x');
set(handles.input,'String',str);


% --- Executes on button press in logarithm.
function logarithm_Callback(hObject, eventdata, handles)
% hObject    handle to logarithm (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'log');
set(handles.input,'String',str);
```

```matlab
% --- Executes on button press in inverse.
function inverse_Callback(hObject, eventdata, handles)
% hObject    handle to inverse (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'finverse');
set(handles.input,'String',str);


% --- Executes on button press in pushbutton39.
function pushbutton39_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton39 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in plot69.
function plot69_Callback(hObject, eventdata, handles)
% hObject    handle to plot69 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
f = str2sym(str);
fplot(f);
str1 = get(handles.display,'String');
f1 = str2sym(str1);
fplot(f1);

% --- Executes on button press in plus.
function plus_Callback(hObject, eventdata, handles)
% hObject    handle to plus (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'+');
set(handles.input,'String',str);

% --- Executes on button press in minus.
function minus_Callback(hObject, eventdata, handles)
% hObject    handle to minus (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'-');
set(handles.input,'String',str);

% --- Executes on button press in multiply.
function multiply_Callback(hObject, eventdata, handles)
% hObject    handle to multiply (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'*');
set(handles.input,'String',str);

% --- Executes on button press in division.
function division_Callback(hObject, eventdata, handles)
% hObject    handle to division (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'/');
set(handles.input,'String',str);

% --- Executes on button press in integral.
function integral_Callback(hObject, eventdata, handles)
% hObject    handle to integral (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
f = str2sym(str);
d = int(f);
str1 = char(d);
set(handles.display,'String',str1);

% --- Executes on button press in differentiate.
function differentiate_Callback(hObject, eventdata, handles)
% hObject    handle to differentiate (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
f = str2sym(str);
d = diff(f);
str1 = char(d);
set(handles.display,'String',str1);
% --- Executes on button press in equal.
function equal_Callback(hObject, eventdata, handles)
% hObject    handle to equal (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str1 = eval(str);
set(handles.display,'String',str1);

% --- Executes on button press in increment.
function increment_Callback(hObject, eventdata, handles)
% hObject    handle to increment (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'exp');
set(handles.input,'String',str);


% --- Executes on button press in cont_pi.
function cont_pi_Callback(hObject, eventdata, handles)
% hObject    handle to cont_pi (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
str = get(handles.input,'String');
str = strcat(str,'pi');
set(handles.input,'String',str);
```

Results:

The following figure shows simple algebraic calculation using our calculator.

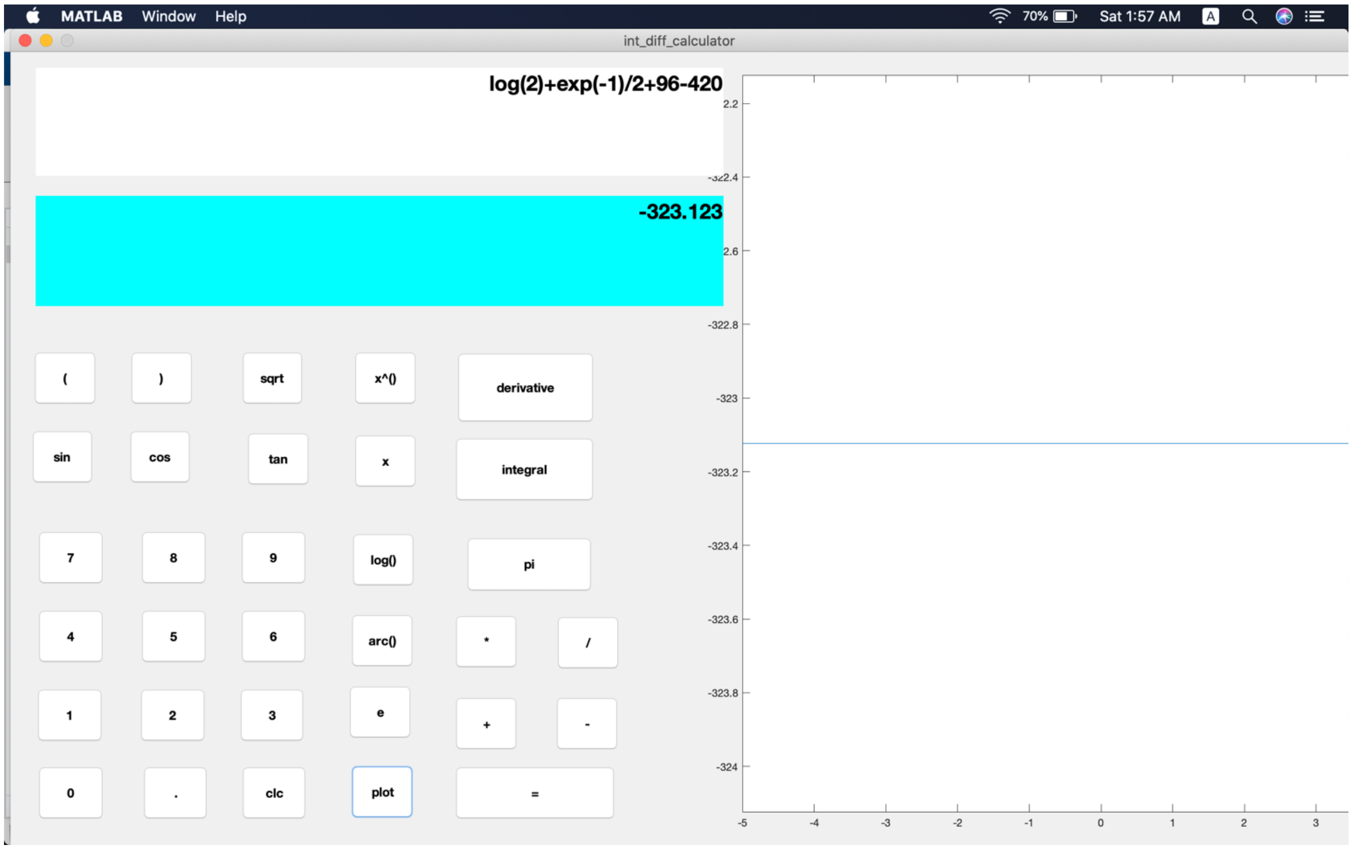
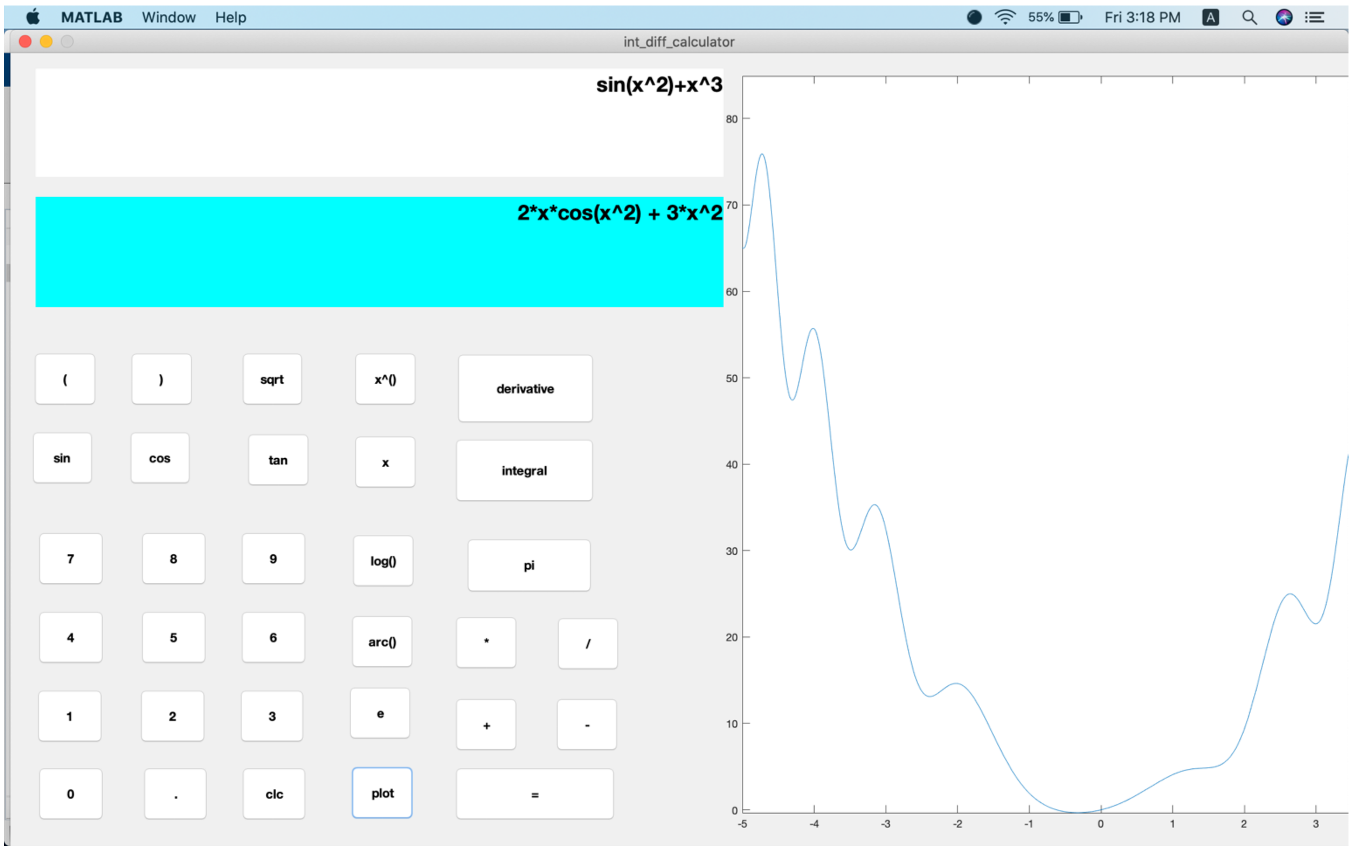


Fig:05

In the next figure, we can see finding out the derivative or integral of a given function and generating the graph using plot button.



sin(x^2)+x^3

2*x*cos(x^2) + 3*x^2

Fig:06

Working Principle:

After running the int_diff_calculator.m file we will get the fig file window where we will see two display panels for input and output, many push buttons for giving inputs, a blank window for showing graph of different functions.
We can do simple algebraic calculation using the push buttons such as addition, subtraction, multiplication, division and the digit buttons.
We have also used some predefined functions such as log, exponential, sin, cos, tan, pi, square root, and power. If we want to use these functions, we have to tap the function and use the left and right bracket to pass input to the function. One thing to note is that, while using trigonometric function, passed angle argument will be considered as radian value. So, to use degree, user have to manually change the angle from radian to degree.
Now, Here comes the interesting part of this special calculator. We can write function using variable x to find it's derivative or integral. To write a function using x as function variable and if we want to make expression with our built in functions, we can use those

built in functions and use brackets to form expression. Then if we tap on derivative or integral, we will get our desired result I the output display panel.
The last functionality of our designed calculator is that it can generate plots of function. After getting an expression in the output display panel as derivative or integral, we can just tap plot to generate the graph of the output function.
Finally, if we are done using the calculator, we can just tap clc button to clear both of our display panel and use it for another task.

Limitations:

- Plotting is done automatically and not possible set range of the output function
- Not possible to apply definite integral, or obtain value of derivative
- Is a very simplistic version of an actual calculator
- Impossible to edit a written expression. Have to clear everything to correct the mistake
- Need to use multiplication sign after each character to mean multiplication (Just placing characters side by side won't mean multiplication which we are used to)
- Only expressions generated through integral or derivative can be plotted

Discussion:

The process of making this calculator has familiarized us with the ins and outs of MATLAB GUI functionalities. At times, our team somewhat struggled when it came to writing the lines of code. A lot of time had to be allocated to the thorough debugging of code. Despite the limitations of our calculator, we believe that it serves its primary purposes well enough. The experience of working on this project equips us with the necessary knowledge and skills for accomplishing even bigger projects in future.