# 

# EML 4535, EML 5530

# MECHANICAL COMPUTER AIDED DESIGN,

# INTERMEDIATE CAD/CAE

# FALL 2022

**PROJECT 3**

**Please submit your programs to the following G-Mail address:**

[**fiucad1@gmail.com**](mailto:fiucad1@gmail.com)

**Please use the subject like this:**

**Last name – First name - CAD – Project #**

**Example:**

**Tansel Ibrahim – CAD – Project 3**

You are assigned to a basic CAD program development project.

|  |  |  |
| --- | --- | --- |
| Segment | Points | Capability |
| Simple | 50 | Add ellipse drawing capability to your program. User will click the “ellipse” button first. Program should draw the ellipse after the user clicks the mouse twice to select two corners of an imaginary rectangle which touches the ellipse at 4 points. |
| Entry | 60 | Add circle drawing capability to your program. Program should draw a circle when the user clicks at the center and a point on the circle after he clicked at the “circle” button. |
| Mid | 70 | Add a TRIANGLE button. Draw a TRIANGLE when the user clicks three times. The first point the user click will be the first corner of the triangle. Draw a line between two points when the user clicks the mouse for the second time. Draw two lines from the final points of the line to the point where the user clicks the mouse for the third time |
| Upper class | 95 | Add a polygon button. Let the user draw a polygon when the user selects that option and clicks FIVE times at the drawing area. Let the user selects the colors of the lines of the polygon (one out of three colors is fine). |
| Luxury | 110 | Add an erase button for THE LINES. Make the program store the coordinates of the END POINTS OF THE LINES. The line will be erased when the user clicks on a point of the desired line to be erased after he clicked on the erase button. This is extremely difficult. We will compare the programs if anyone completes this option. |

**LIST OF THE EXAMPLE PROGRAM**

# Graphics Program

from tkinter import \* # We read the graphic library

import tkinter as tk

import math

#Initilize the TKInter

root = Tk()

root.title('Mouse based graphics program') # This is the title of the window

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Define the Window sizes and draw 2 frames (plot and control windows) inside a large window

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Window size

window\_width=int(800)

window\_height=int(600)

# Window - setup and draw

canvas = Canvas(root, width =window\_width, height=window\_height)

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Window 1 - Large external frame

x\_and\_y\_offsets=float(0.05) # Space between window and external frame

x\_left\_top\_large\_frame=window\_width \* x\_and\_y\_offsets

y\_left\_top\_large\_frame=window\_height \* x\_and\_y\_offsets

x\_right\_bottom\_large\_frame=window\_width - (window\_width \* x\_and\_y\_offsets)

y\_right\_bottom\_large\_frame=window\_height - (window\_height \* x\_and\_y\_offsets)

#Draw the frame with a rectangle

canvas.create\_rectangle(x\_left\_top\_large\_frame, y\_left\_top\_large\_frame,

x\_right\_bottom\_large\_frame, y\_right\_bottom\_large\_frame,

outline="blue", fill="grey", width='2')

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Window 2 - Plot frame in the large window - A function is used to redraw this window to clean the area

# Plot the "PLOT WINDOW" - Use this function to clean the picture in it - We will erase the curve and both Axes with labels

def draw\_plot\_window(x\_left\_offset, x\_right\_offset, y\_top\_bottom\_offsets):

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

x\_left\_top\_plot\_frame=window\_width \* x\_left\_offset

y\_left\_top\_plot\_frame=window\_height \* y\_top\_bottom\_offsets

x\_right\_bottom\_plot\_frame=window\_width - (window\_width \* x\_right\_offset)

y\_right\_bottom\_plot\_frame=window\_height - (window\_height \* y\_top\_bottom\_offsets)

#Draw the frame with a rectangle

canvas.create\_rectangle(x\_left\_top\_plot\_frame, y\_left\_top\_plot\_frame,

x\_right\_bottom\_plot\_frame, y\_right\_bottom\_plot\_frame,

outline="red", fill="white", width='2')

# Define boundaries and call the plot window drawing function

x\_left\_plot\_frame\_offset=float(0.1) # Space between window and plot frame

x\_right\_plot\_frame\_offset=float(0.3) # Space between window and plot frame

y\_top\_bottom\_plot\_frame\_offsets=float(0.1) # Space between window and plot frame

# Draw the window with above boundaries by using the above function

draw\_plot\_window(x\_left\_plot\_frame\_offset, x\_right\_plot\_frame\_offset, y\_top\_bottom\_plot\_frame\_offsets)

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Window 3 - Control button frame in the window

x\_left\_control\_offset=float(0.72) # Space between window and plot frame

x\_right\_control\_offset=float(0.08) # Space between window and plot frame

y\_top\_bottom\_control\_offsets=float(0.1) # Space between window and plot frame

x\_left\_top\_control\_frame=window\_width \* x\_left\_control\_offset

y\_left\_top\_control\_frame=window\_height \* y\_top\_bottom\_control\_offsets

x\_right\_bottom\_control\_frame=window\_width - (window\_width \* x\_right\_control\_offset)

y\_right\_bottom\_control\_frame=window\_height - (window\_height \* y\_top\_bottom\_control\_offsets)

#Draw the frame with a rectangle

canvas.create\_rectangle(x\_left\_top\_control\_frame, y\_left\_top\_control\_frame,

x\_right\_bottom\_control\_frame, y\_right\_bottom\_control\_frame,

outline="red", fill="white", width='2')

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Set the default values to draw the geometric functions

# Put the range of the x axis in radian

# Define the boundaries of the x variable and display them on the screen

global x\_start, x\_end

x\_start=int(0)

x\_end=float(4\*3.14)

x\_button\_offset= x\_left\_control\_offset + 0.02 # Percentage of the width of the window

y\_button\_offset= y\_top\_bottom\_control\_offsets + 0.02 # Percentage of the height of the window

y\_button\_space= 30 # Exact distance between the buttons and input boxes in the control frame

x\_button\_location=window\_width\*x\_button\_offset

y\_button\_location=window\_height\*(y\_button\_offset+0.02)

canvas.create\_text(x\_button\_location, y\_button\_location,

anchor=W, font=('verdana', 12),fill='blue',

text="Select")

canvas.create\_text(x\_button\_location, y\_button\_location+y\_button\_space,

anchor=W, font=('verdana', 12),fill='blue',

text="option")

# No text box will be used in this program - Following lines are availible if needed with any purpose

#textBox1=Text(root, height=1, width=8)

#textBox1.place(x = x\_button\_location, y = y\_button\_location+y\_button\_space)

##########textBox1.pack()

#textBox1.insert(tk.END, x\_start)

#textBox2=Text(root, height=1, width=8)

#textBox2.place(x = x\_button\_location, y = y\_button\_location+y\_button\_space\*2)

#########textBox1.pack()

#textBox2.insert(tk.END, x\_end)

# Function to clear the plotting area - Redraw that frame

def clear():

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

# Draw the window with above boundaries by using the above function

draw\_plot\_window(x\_left\_plot\_frame\_offset, x\_right\_plot\_frame\_offset, y\_top\_bottom\_plot\_frame\_offsets)

# Function to draw a line on the canvas, Step 1 & 4

def draw\_line ():

canvas.bind('<Button-1>', draw\_line\_1)

# Function to draw a line on the canvas, Step 2

def draw\_line\_1 (event):

global control

control=1

global x,y

x = event.x

y = event.y

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x>x\_left\_top\_plot\_frame and x<x\_right\_bottom\_plot\_frame and y>y\_left\_top\_plot\_frame and y<y\_right\_bottom\_plot\_frame:

global cursor1, cursor2

cursor1=canvas.create\_line(x-5,y,x+5,y, fill = 'red', width = 1)

cursor2=canvas.create\_line(x,y-5,x,y+5, fill = 'red', width = 1)

canvas.bind('<Button-1>', draw\_line\_2)

# Function to draw a line on the canvas, Step 3

def draw\_line\_2(event):

global control

global cursor1, cursor2

global x,y,x1,y1

x1 = event.x

y1 = event.y

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x1>x\_left\_top\_plot\_frame and x1<x\_right\_bottom\_plot\_frame and y1>y\_left\_top\_plot\_frame and y1<y\_right\_bottom\_plot\_frame:

#Erase the cursor which marks the location of first click

canvas.delete(cursor1, cursor2)

if control== 1:

canvas.create\_line(x,y,x1,y1, fill="black", width = 3)

control=0

# Function to draw a circle on the canvas, Step 1 & 4

def draw\_circle ():

canvas.bind('<Button-1>', draw\_circle\_1)

# Function to draw a circle on the canvas, Step 2

def draw\_circle\_1 (event):

global control

control=1

global x,y

x = event.x

y = event.y

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x>x\_left\_top\_plot\_frame and x<x\_right\_bottom\_plot\_frame and y>y\_left\_top\_plot\_frame and y<y\_right\_bottom\_plot\_frame:

global cursor1, cursor2

cursor1=canvas.create\_line(x-5,y,x+5,y, fill = 'red', width = 1)

cursor2=canvas.create\_line(x,y-5,x,y+5, fill = 'red', width = 1)

canvas.bind('<Button-1>', draw\_circle\_2)

# Function to draw a circle on the canvas, Step 3

def draw\_circle\_2(event):

global control

global cursor1, cursor2

global x,y,x1,y1

x1 = event.x

y1 = event.y

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x1>x\_left\_top\_plot\_frame and x1<x\_right\_bottom\_plot\_frame and y1>y\_left\_top\_plot\_frame and y1<y\_right\_bottom\_plot\_frame:

#Erase the cursor which marks the location of first click

canvas.delete(cursor1, cursor2)

if control== 1:

d = math.sqrt((x1 - x) \*\* 2 + (y1 - y) \*\* 2)

canvas.create\_oval(x-d,y+d,x+d,y-d, outline="black", width = 3)

control=0

# Function to draw an ellipse on the canvas, Step 1 & 4

def draw\_ellipse ():

canvas.bind('<Button-1>', draw\_ellipse\_1)

# Function to draw a line on the canvas, Step 2

def draw\_ellipse\_1 (event):

global control

control=1

global x,y

x = event.x

y = event.y

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x>x\_left\_top\_plot\_frame and x<x\_right\_bottom\_plot\_frame and y>y\_left\_top\_plot\_frame and y<y\_right\_bottom\_plot\_frame:

global cursor1, cursor2

cursor1=canvas.create\_line(x-5,y,x+5,y, fill = 'red', width = 1)

cursor2=canvas.create\_line(x,y-5,x,y+5, fill = 'red', width = 1)

canvas.bind('<Button-1>', draw\_ellipse\_2)

# Function to draw a line on the canvas, Step 3

def draw\_ellipse\_2(event):

global control

global cursor1, cursor2

global x,y,x1,y1

x1 = event.x

y1 = event.y

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x1>x\_left\_top\_plot\_frame and x1<x\_right\_bottom\_plot\_frame and y1>y\_left\_top\_plot\_frame and y1<y\_right\_bottom\_plot\_frame:

#Erase the cursor which marks the location of first click

canvas.delete(cursor1, cursor2)

if control== 1:

canvas.create\_oval(x,y,x1,y1, outline="black", width = 3)

control=0

# Function to draw a rectangle on the canvas, Step 1 & 4

def draw\_rectangle ():

canvas.bind('<Button-1>', draw\_rectangle\_1)

# Function to draw a line on the canvas, Step 2

def draw\_rectangle\_1 (event):

global control

control=1

global x,y

x = event.x

y = event.y

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x>x\_left\_top\_plot\_frame and x<x\_right\_bottom\_plot\_frame and y>y\_left\_top\_plot\_frame and y<y\_right\_bottom\_plot\_frame:

global cursor1, cursor2

cursor1=canvas.create\_line(x-5,y,x+5,y, fill = 'red', width = 1)

cursor2=canvas.create\_line(x,y-5,x,y+5, fill = 'red', width = 1)

canvas.bind('<Button-1>', draw\_rectangle\_2)

# Function to draw a line on the canvas, Step 3

def draw\_rectangle\_2(event):

global control

global cursor1, cursor2

global x,y,x1,y1

x1 = event.x

y1 = event.y

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x1>x\_left\_top\_plot\_frame and x1<x\_right\_bottom\_plot\_frame and y1>y\_left\_top\_plot\_frame and y1<y\_right\_bottom\_plot\_frame:

#Erase the cursor which marks the location of first click

canvas.delete(cursor1, cursor2)

if control== 1:

canvas.create\_rectangle(x,y,x1,y1, outline="black", width = 3)

control=0

# Function to draw a triangle on the canvas, Step 1 & 4

def draw\_triangle ():

canvas.bind('<Button-1>', draw\_triangle\_1)

# Function to draw a triangle on the canvas, Step 2

def draw\_triangle\_1 (event):

global control

control=1

global x,y

x = event.x

y = event.y

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x>x\_left\_top\_plot\_frame and x<x\_right\_bottom\_plot\_frame and y>y\_left\_top\_plot\_frame and y<y\_right\_bottom\_plot\_frame:

global cursor1, cursor2

cursor1=canvas.create\_line(x-5,y,x+5,y, fill = 'red', width = 1)

cursor2=canvas.create\_line(x,y-5,x,y+5, fill = 'red', width = 1)

canvas.bind('<Button-1>', draw\_triangle\_2)

# Function to draw a triangle on the canvas, Step 3

def draw\_triangle\_2(event):

global control

global cursor1, cursor2

global x,y,x1,y1

x1 = event.x

y1 = event.y

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x1>x\_left\_top\_plot\_frame and x1<x\_right\_bottom\_plot\_frame and y1>y\_left\_top\_plot\_frame and y1<y\_right\_bottom\_plot\_frame:

#Erase the cursor which marks the location of first click

canvas.delete(cursor1, cursor2)

if control== 1:

canvas.create\_line(x,y,x1,y1, fill="black", width = 3)

# control=0

canvas.bind('<Button-1>', draw\_triangle\_3)

# Function to draw a triangle on the canvas, Step 4

def draw\_triangle\_3(event):

global control

global cursor1, cursor2

global x,y,x1,y1,x2,y2

x2 = event.x

y2 = event.y

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x2>x\_left\_top\_plot\_frame and x2<x\_right\_bottom\_plot\_frame and y2>y\_left\_top\_plot\_frame and y2<y\_right\_bottom\_plot\_frame:

#Erase the cursor which marks the location of first click

canvas.delete(cursor1, cursor2)

if control== 1:

canvas.create\_line(x1,y1,x2,y2, fill="black", width = 3)

canvas.create\_line(x2,y2,x,y, fill="black", width = 3)

control=0

def motion\_line\_live\_1(event):

global cursor1, cursor2

global control\_cursor

if control\_cursor>0:

canvas.delete(cursor1, cursor2)

x, y = event.x, event.y

cursor1=canvas.create\_line(x-5,y,x+5,y, fill = 'green', width = 1)

cursor2=canvas.create\_line(x,y-5,x,y+5, fill = 'green', width = 1)

control\_cursor=1

def motion\_line\_live\_2(event):

global x,y

global cursor1, cursor2,cursor3,cursor4

global control\_cursor

global control\_live\_line\_option

# global control\_live\_line

global control\_cursor\_live\_line,line\_mouse

if control\_live\_line\_option>0:

if control\_cursor\_live\_line>0:

# canvas.delete(cursor1, cursor2)

canvas.delete(cursor3, cursor4)

if control\_cursor\_live\_line>0:

canvas.delete(line\_mouse)

xm, ym = event.x, event.y

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if xm>x\_left\_top\_plot\_frame and xm<x\_right\_bottom\_plot\_frame and ym>y\_left\_top\_plot\_frame and ym<y\_right\_bottom\_plot\_frame:

#Erase the cursor which marks the location of first click

cursor3=canvas.create\_line(xm-5,ym,xm+5,ym, fill = 'green', width = 1)

cursor4=canvas.create\_line(xm,ym-5,xm,ym+5, fill = 'green', width = 1)

line\_mouse=canvas.create\_line(x,y,xm,ym, fill="green", width = 1)

control\_cursor\_live\_line=1

else:

return

def draw\_line\_live ():

canvas.bind('<Button-1>', draw\_line\_live\_1)

canvas.bind('<Motion>', motion\_line\_live\_1)

# Function to draw a line on the canvas, Step 2

def draw\_line\_live\_1 (event):

global control,control\_live\_line\_option

control=1

global x,y

x = event.x

y = event.y

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x>x\_left\_top\_plot\_frame and x<x\_right\_bottom\_plot\_frame and y>y\_left\_top\_plot\_frame and y<y\_right\_bottom\_plot\_frame:

global cursor1, cursor2

cursor1=canvas.create\_line(x-5,y,x+5,y, fill = 'red', width = 1)

cursor2=canvas.create\_line(x,y-5,x,y+5, fill = 'red', width = 1)

canvas.bind('<Button-1>', draw\_line\_2)

control\_live\_line\_option=1

canvas.bind('<Motion>', motion\_line\_live\_2)

# Function to draw a line on the canvas, Step 3

def draw\_line\_live\_2(event):

global control,control\_live\_line\_option

global cursor1, cursor2

global x,y,x1,y1

x1 = event.x

y1 = event.y

# Stop drawing rubber lines

control\_live\_line\_option=0

# Evaluate if the point is at the plot window

global x\_left\_top\_plot\_frame, x\_right\_bottom\_plot\_frame, y\_left\_top\_plot\_frame, y\_right\_bottom\_plot\_frame

if x1>x\_left\_top\_plot\_frame and x1<x\_right\_bottom\_plot\_frame and y1>y\_left\_top\_plot\_frame and y1<y\_right\_bottom\_plot\_frame:

#Erase the cursor which marks the location of first click

canvas.delete(cursor1, cursor2)

if control== 1:

canvas.create\_line(x,y,x1,y1, fill="black", width = 3)

control=0

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* MAIN ACTION \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# To avoid problems at the first run - It will be 1 after the first point in the functions

global cursor\_control

global control\_live\_line

global control\_cursor\_live\_line

global control\_live\_line\_option

control\_cursor=0

control\_live\_line=0

control\_cursor\_live\_line=0

control\_live\_line\_option=0

# Draw the sine wave with defined boundaries first time

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Button to clear the drawing area

line\_button= Button(canvas,text="Clear", command=clear, height=1, width=8, bg="yellow",compound=LEFT)

line\_button.place(x = x\_button\_location, y = y\_button\_location+y\_button\_space\*3)

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Button to draw line

line\_button= Button(canvas,text="Line", command=draw\_line, height=1, width=8, bg="yellow",compound=LEFT)

line\_button.place(x = x\_button\_location, y = y\_button\_location+y\_button\_space\*4)

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Button to draw circle

ellipse\_button= Button(canvas,text="Circle", command=draw\_circle, height=1, width=8, bg="yellow",compound=LEFT)

ellipse\_button.place(x = x\_button\_location, y = y\_button\_location+y\_button\_space\*5)

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Button to draw ellipse

ellipse\_button= Button(canvas,text="Ellipse", command=draw\_ellipse, height=1, width=8, bg="yellow",compound=LEFT)

ellipse\_button.place(x = x\_button\_location, y = y\_button\_location+y\_button\_space\*6)

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Button to draw rectangle

rectangle\_button= Button(canvas,text="Rectangle", command=draw\_rectangle, height=1, width=8, bg="yellow",compound=LEFT)

rectangle\_button.place(x = x\_button\_location, y = y\_button\_location+y\_button\_space\*7)

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Button to draw a triangle

polygon\_button= Button(canvas,text="Triangle", command=draw\_triangle, height=1, width=8, bg="yellow",compound=LEFT)

polygon\_button.place(x = x\_button\_location, y = y\_button\_location+y\_button\_space\*8)

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Button to draw a line - line shape is displayed

polygon\_button= Button(canvas,text="Line - live", command=draw\_line\_live, height=1, width=8, bg="yellow",compound=LEFT)

polygon\_button.place(x = x\_button\_location, y = y\_button\_location+y\_button\_space\*9)

canvas.pack()

root.mainloop()