/\*

\*\*\* FILE NAME : Cos.c

\*\*\* DESCRIPTION : This program draws a cosine wave for y=A+Bcos(x)

\*\*\* DATE : Jan. 2015

\*\*\* WRITTEN By : Farheen Sultana

\*/

/\*

The program draws a Cosine wave for the function y= A+Bcos(x) on a window of size 480\*480.

The variables A and B are the scaling factors.

A is the center line for graph, it represents the horizontal stretch from center line.

B is the Maximum deviation of graph from center line on Y-Axis, i.e., it represents the vertical displacement of graph from center line.

As the center here is at 240,240:

The Maximum Vertical Displacement needed is 240 ==> B= 240

The Maximum Horizontal Stretch needed is 240 ==> A = 240

So, A= 240 and B = 240.

\*/

#include <stdio.h> // standard C libraries

#include <stdlib.h>

#include <math.h>

#include <time.h>

#include <string.h>

#include <GL/glut.h> // GLUT library

#include "cs\_graphics\_setup.h" // Header for CS4250/5250/6250 courses

//@@\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*@@

// Constants

#define WINDOW\_XS 480 // Window size

#define WINDOW\_YS 480

#define WINDOW\_NAME "Cosine Wave" // Window name

//@@\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*@@

// Structure

typedef struct pt2d

{

GLint x, y;

}My2DPoint;

//@@\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*@@

// Function prototypes

void display\_func(void);

void keyboard\_func(unsigned char c, int x, int y);

void mouse\_func(int button, int state, int x, int y);

//@@\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*@@

// Global Variables

My2DPoint XAxis1, XAxis2; //Coordinates of X-axis

My2DPoint YAxis1, YAxis2; //Coordinates of Y-axis

float DegToRad = 3.141592 \* 1.0 / 180.0; // Convert Degrees to Radian Degree = Radian \* 3.14/180

float LineX1 = 0.0, LineY1 = 0.0; // Coordinates of Smaller Line Segments used to draw cosine curve

float LineX2 = 0.0, LineY2 = 0.0; // Coordinates of Smaller Line Segments used to draw cosine curve

float A1 = 240.0, B1 = 240.0; // A and B are Scaling Factors

float x; // X is the pixel X-Coordinate

int flag = 1; //flag variable used to change the curve from solid to dashed upon right mouse click

float set\_angle = 0.0; // variable to adjust axis (480) to angle (360)

//@@\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*@@

int main(int argc, char \*\*argv)

{

glutInit(&argc, argv);

init\_setup(WINDOW\_XS, WINDOW\_YS, WINDOW\_NAME);

// initializing x-axis

XAxis1.x = 0;

XAxis1.y = 240;

XAxis2.x = 480;

XAxis2.y = 240;

// initializing y-axis

YAxis1.x = 240;

YAxis1.y = 0;

YAxis2.x = 240;

YAxis2.y = 480;

glutDisplayFunc(display\_func); // call back for display event

glutKeyboardFunc(keyboard\_func); // call back for keyboard event

glutMouseFunc(mouse\_func); // call back for mouse event

glutMainLoop();

return 1;

} // end of main()

//@@\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*@@

void display\_func(void)

{

glClearColor(1.0, 1.0, 1.0, 1.0); // background color (white)

glClear(GL\_COLOR\_BUFFER\_BIT); // clearing the buffer not to keep the color

//Draw X-axis

glColor3f(1,0,0);

glBegin(GL\_LINES);

glVertex2i(XAxis1.x, XAxis1.y);

glVertex2i(XAxis2.x, XAxis2.y);

glEnd();

//Draw y-axis

glBegin(GL\_LINES);

glVertex2i(YAxis1.x, YAxis1.y);

glVertex2i(YAxis2.x, YAxis2.y);

glEnd();

//DRAW SOLID CURVE

if (flag == 1)

{

glColor3f(0, 0, 1);

glBegin(GL\_LINES);

set\_angle = 0.0;

for (x = 0.0; x < 480.0; x += 1.3333)

{

set\_angle = set\_angle + 0.3333;

LineX1 = x;

LineY1 = A1 + B1\* cos((x-set\_angle)\*DegToRad);

LineX2 = x + 1.3333;

LineY2 = A1 + B1\* cos((x + 1.3333 - set\_angle)\*DegToRad);

glVertex2f(LineX1, WINDOW\_YS - LineY1);

glVertex2f(LineX2, WINDOW\_YS - LineY2);

}

glEnd();

}

//DRAW DASHED CURVE UPON RIGHT MOUSE CLICK

if ( flag == 2)

{

glColor3f(0, 1, 0);

glBegin(GL\_LINES);

set\_angle = 0.0;

for (x = 0.0; x < 480.0; x += 1.3333)

{

set\_angle = set\_angle + 0.9999;

LineX1 = x;

LineY1 = A1 + B1\* cos((x - set\_angle)\*DegToRad);

LineX2 = x + 1.3333;

LineY2 = A1 + B1\* cos((x + 1.3333 - set\_angle)\*DegToRad);

glVertex2f(LineX1, WINDOW\_YS - LineY1);

glVertex2f(LineX2, WINDOW\_YS - LineY2);

x = x + 2.6666;

}

glEnd();

}

glFlush();

} // end of display\_func()

//@@\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*@@

void keyboard\_func(unsigned char c, int x, int y)

{

switch(c)

{

case 'q' :

case 'Q' :

exit(1);

break;

} // end of switch

} // end of keyboard\_func()

//@@\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*@@

void mouse\_func(int button, int state, int x, int y)

{

if (button == GLUT\_RIGHT\_BUTTON && state == GLUT\_DOWN)

{

if (flag == 1)

flag = 2;

else

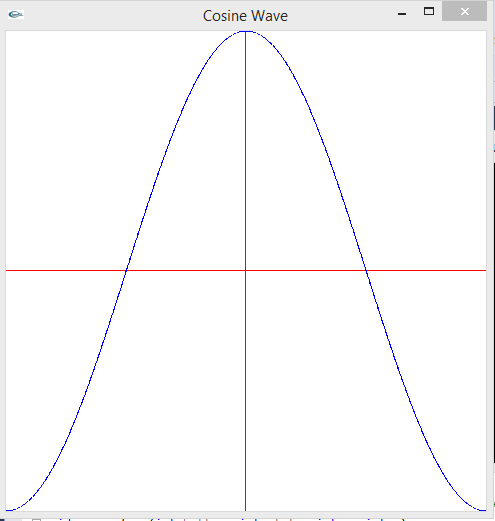
flag = 1;

glutPostRedisplay();

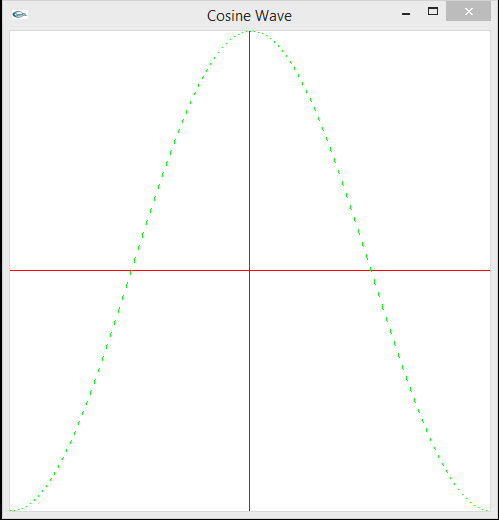
}

} // end of mouse\_func()

Output:



Right Mouse Click:



Right Click Again:

