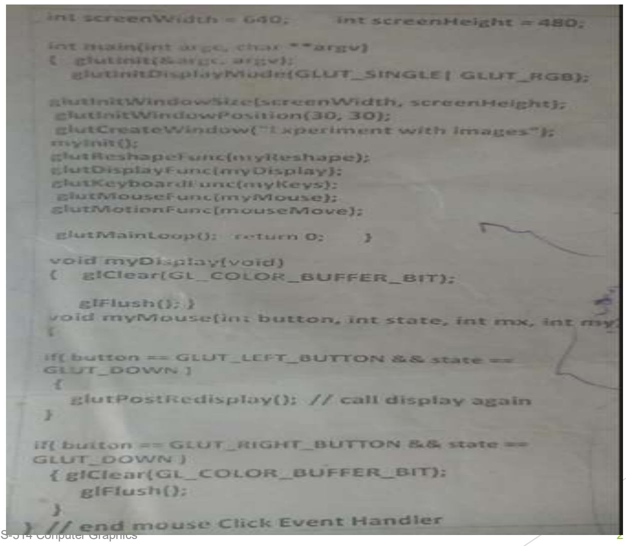
wav.h

Practicing 2D Objects and Application of W-to-V Mapping Adding Text and Image

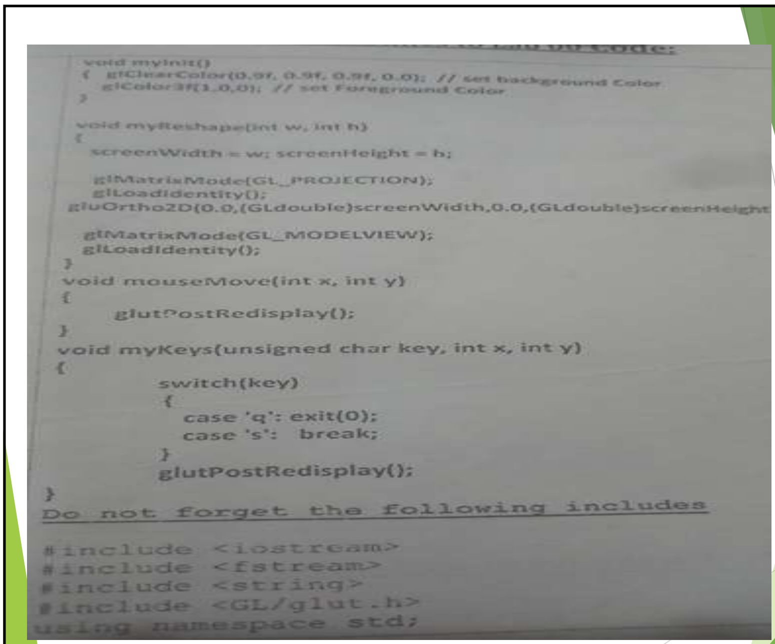
Open GL Lab03

BSCS-514 Conputer Graphics
Instructor Dr. Humera

Step1: Add Reshape, Mouse and Keyboard Handler routine to **Lab 01**

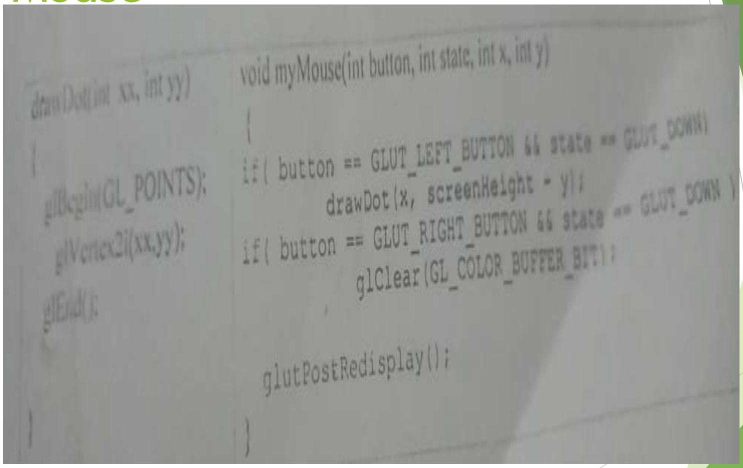


```
int screenWidth = 640;        int screenHeight = 480;

int main(int argc, char **argv)
{   glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE| GLUT_RGB);

glutInitWindowSize(screenWidth, screenHeight);
 glutInitWindowPosition(30, 30);
 glutCreateWindow("Experiment with images");
myInit();
glutReshapeFunc(myReshape);
glutDisplayFunc(myDisplay);
glutKeyboardFunc(myKeys);
 glutMouseFunc(myMouse);
glutMotionFunc(mouseMove);

 glutMainLoop();   return 0;      }

void myDisplay(void)
{   glClear(GL_COLOR_BUFFER_BIT);

    glFlush(); }
void myMouse(int button, int state, int mx, int my
{

if( button == GLUT_LEFT_BUTTON && state ==
GLUT_DOWN )
 {
   glutPostRedisplay();  // call display again
 }

if( button == GLUT_RIGHT_BUTTON && state ==
GLUT_DOWN )
 { glClear(GL_COLOR_BUFFER_BIT);
    glFlush();
  }
} // end mouse Click Event Handler
```



```
void myInit()
{  glClearColor(0.9f, 0.9f, 0.9f, 0.0); // set background Color
   glColor3f(1,0,0);  // set Foreground Color
}

void myReshape(int w, int h)
{
  screenWidth = w; screenHeight = h;

   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
 gluOrtho2D(0.0,(GLdouble)screenWidth,0.0,(GLdouble)screenHeight

   glMatrixMode(GL_MODELVIEW);
   glLoadIdentity();
 }
 void mouseMove(int x, int y)
 {
       glutPostRedisplay();
 }
 void myKeys(unsigned char key, int x, int y)
 {
            switch(key)
             {
               case 'q': exit(0);
               case 's':   break;
             }
            glutPostRedisplay();
}
Do not forget the following includes

#include <iostream>
#include <fstream>
#include <string>
#include <GL/glut.h>
using namespace std;
```

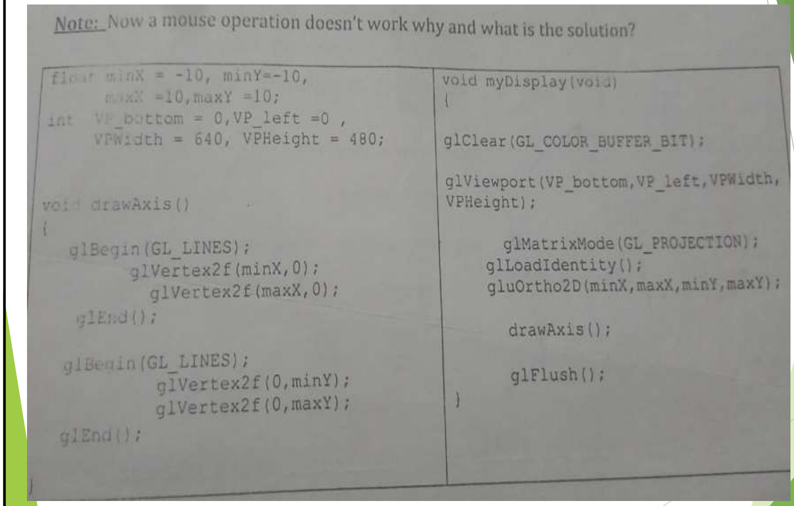
STEP 2: Placing Dots with Mouse



```
drawDot(int  xx, int yy)        void myMouse(int button, int state, int x, int y)
{                               {
   glBegin(GL_POINTS);          if( button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
    glVertex2i(xx,yy);                   drawDot(x, screenHeight - y);
   glEnd();                     if( button == GLUT_RIGHT_BUTTON && state == GLUT_DOWN )
                                           glClear(GL_COLOR_BUFFER_BIT);

                                   glutPostRedisplay();
}                               }
```

Step 03: Free Hand Drawing with Fat Brush

Step 04: Using both ViewPort and World Window

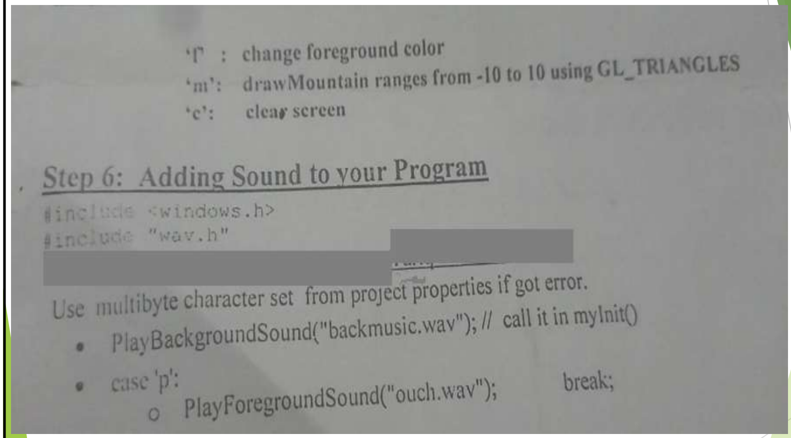


Note: Now a mouse operation doesn't work why and what is the solution?

```
float minX = -10, minY=-10,
      maxX =10,maxY =10;
int   VP_bottom = 0,VP_left =0 ,
      VPWidth = 640, VPHeight = 480;


void drawAxis()
{
   glBegin(GL_LINES);
          glVertex2f(minX,0);
            glVertex2f(maxX,0);
    glEnd();

   glBegin(GL_LINES);
             glVertex2f(0,minY);
             glVertex2f(0,maxY);
   glEnd();

}
```

```
void myDisplay(void)
{

glClear(GL_COLOR_BUFFER_BIT);

glViewport(VP_bottom,VP_left,VPWidth,
VPHeight);

       glMatrixMode(GL_PROJECTION);
     glLoadIdentity();
     gluOrtho2D(minX,maxX,minY,maxY);

       drawAxis();

       glFlush();
 }
```

Step 5: Adding Keyboard Interaction



'f' : change foreground color
'm': drawMountain ranges from -10 to 10 using GL_TRIANGLES
'c': clear screen

Step 6: Adding Sound to your Program

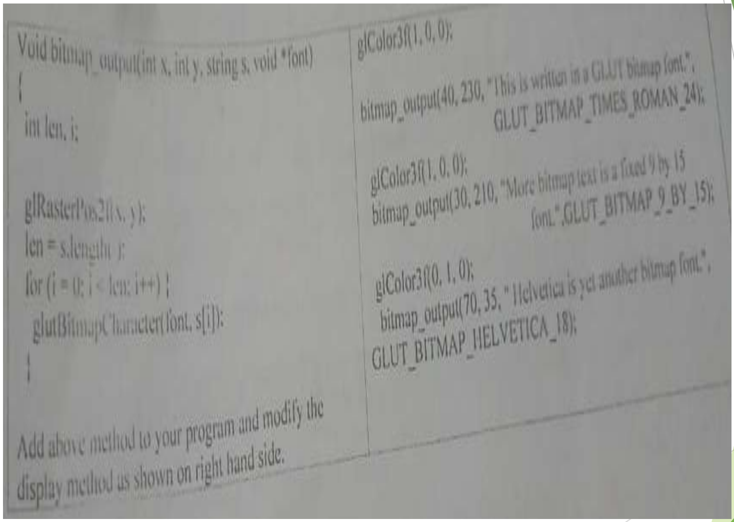
```
#include <windows.h>
#include "wav.h"
```

[redacted]

Use multibyte character set from project properties if got error.

- PlayBackgroundSound("backmusic.wav"); // call it in myInit()
- case 'p':
  - PlayForegroundSound("ouch.wav"); break;

STEP 7: Writing Text to the Screen



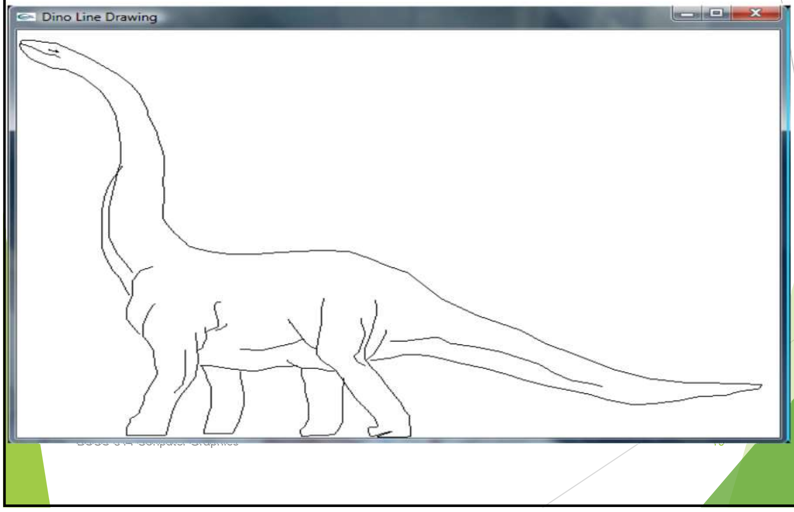
```
Void bitmap_output(int x, int y, string s, void *font)
{
 int len, i;

 glRasterPos2f(x, y);
 len = s.length();
 for (i = 0; i < len; i++) {
  glutBitmapCharacter(font, s[i]);
 }
```

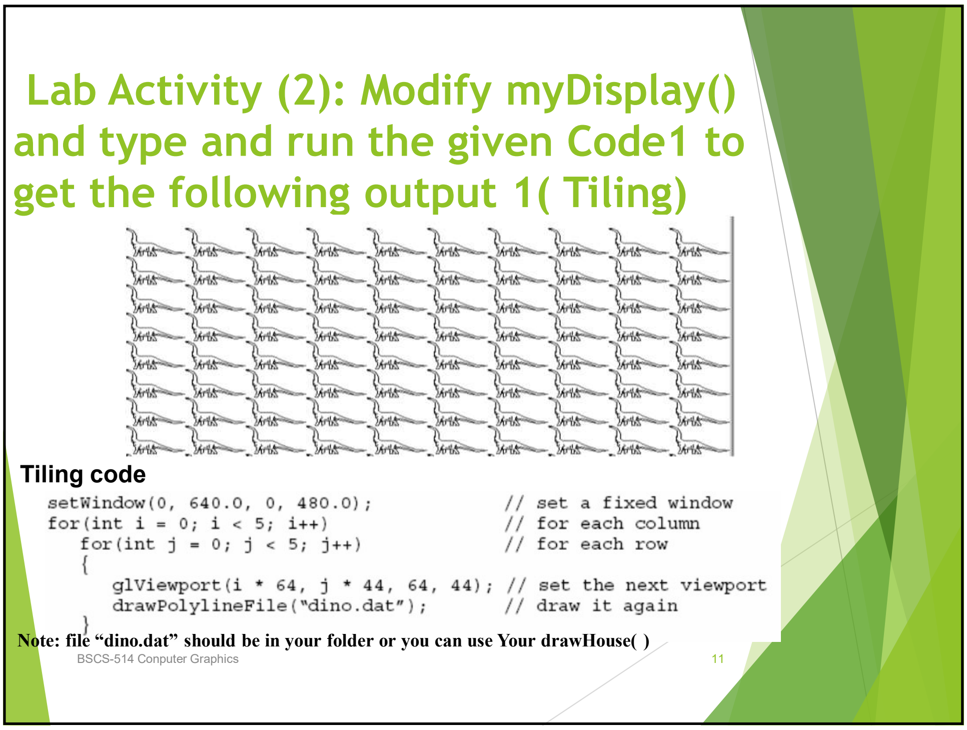
Add above method to your program and modify the display method as shown on right hand side.

```
glColor3f(1, 0, 0);
bitmap_output(40, 230, "This is written in a GLUT bitmap font.",
                          GLUT_BITMAP_TIMES_ROMAN_24);

glColor3f(1, 0, 0);
bitmap_output(30, 210, "More bitmap text is a fixed 9 by 15
                          font.",GLUT_BITMAP_9_BY_15);

glColor3f(0, 1, 0);
bitmap_output(70, 35, "Helvetica is yet another bitmap font.",
GLUT_BITMAP_HELVETICA_18);
```

Practicing 2D Objects and Application of W-to-V Mapping

Activity (1): Run the given skeleton code DinoPolyLines which should give following output on display.



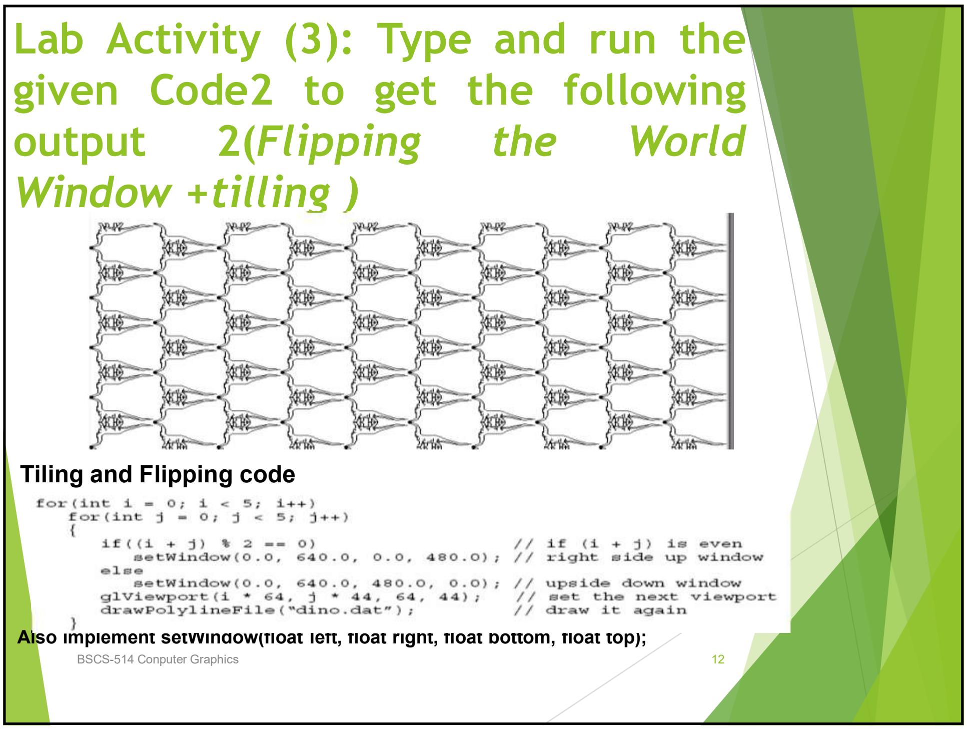


Lab Activity (2): Modify myDisplay() and type and run the given Code1 to get the following output 1(Tiling)

Tiling code

```
setWindow(0, 640.0, 0, 480.0);             // set a fixed window
for(int i = 0; i < 5; i++)                 // for each column
   for(int j = 0; j < 5; j++)              // for each row
   {
      glViewport(i * 64, j * 44, 64, 44); // set the next viewport
      drawPolylineFile("dino.dat");        // draw it again
   }
```

Note: file “dino.dat” should be in your folder or you can use Your drawHouse( )



Lab Activity (3): Type and run the given Code2 to get the following output 2(*Flipping the World Window +tilling*)

Tiling and Flipping code

```
for(int i = 0; i < 5; i++)
   for(int j = 0; j < 5; j++)
   {
      if((i + j) % 2 == 0)                          // if (i + j) is even
         setWindow(0.0, 640.0, 0.0, 480.0);  // right side up window
      else
         setWindow(0.0, 640.0, 480.0, 0.0);  // upside down window
      glViewport(i * 64, j * 44, 64, 44);    // set the next viewport
      drawPolylineFile("dino.dat");          // draw it again
   }
```

Also implement setWindow(float left, float right, float bottom, float top);

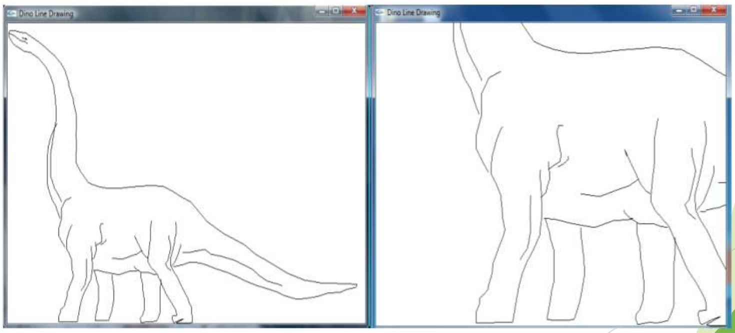
Lab Activity (4): Zooming in and Zooming out on a figure

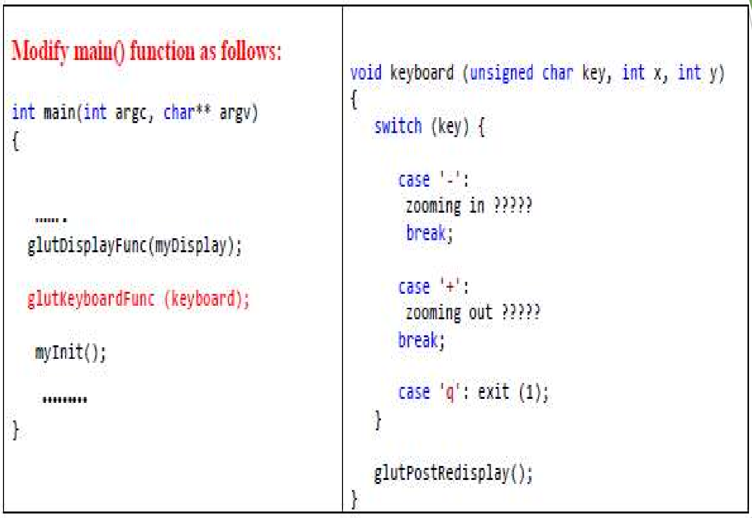
As the window increases in size, the image in the viewport decreases in size and vice versa

Zoom in (To get a closer view)

Keep the Viewport fixed and make the window smaller. When the window is smaller, the portion inside becomes more enlarged.

Zoom Out(To get a view from a greater distance)





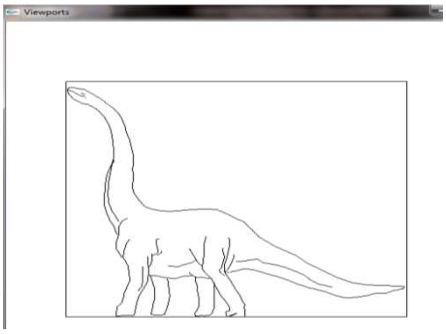
Modify main() function as follows:

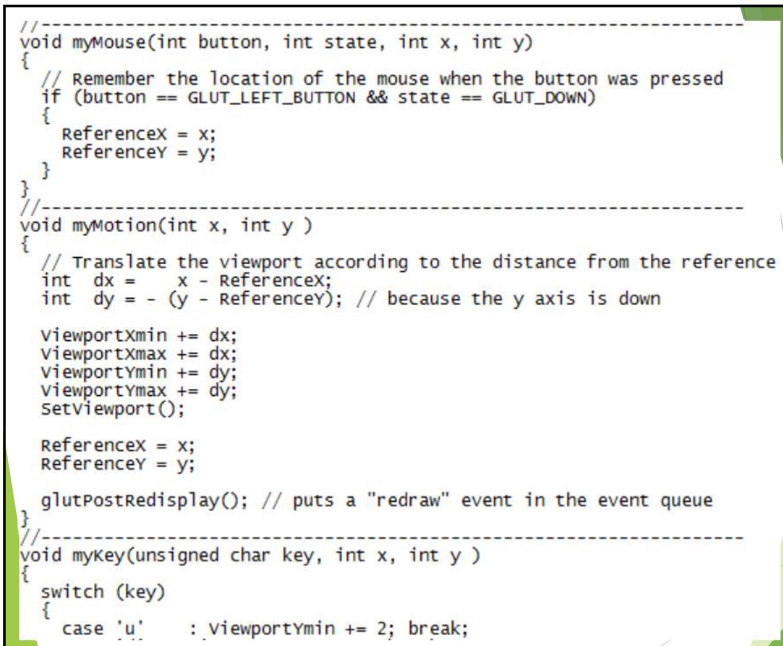
```
int main(int argc, char** argv)
{

  ......
 glutDisplayFunc(myDisplay);

 glutKeyboardFunc (keyboard);

  myInit();

   ........
}
```

```
void keyboard (unsigned char key, int x, int y)
{
   switch (key) {

      case '-':
       zooming in ?????
       break;

      case '+':
       zooming out ?????
      break;

      case 'q': exit (1);
   }

   glutPostRedisplay();
}
```

Lab Activity (5): Make a Viewport Dynamic.(Output is attached) i.e Roaming a.k.a Panning (To move the view in a specific direction

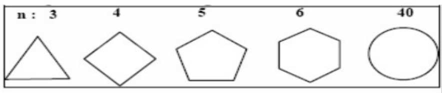
1. Keep VP Parameters variable like VP_LEFT, VP_RIGHT ……
2. Refer to following code.



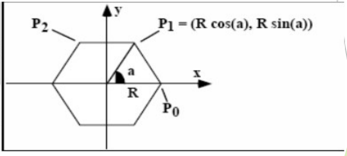


```
//---------------------------------------------------------------------
void myMouse(int button, int state, int x, int y)
{
  // Remember the location of the mouse when the button was pressed
  if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
  {
    ReferenceX = x;
    ReferenceY = y;
  }
}
//---------------------------------------------------------------------
void myMotion(int x, int y )
{
  // Translate the viewport according to the distance from the reference
  int  dx =    x - ReferenceX;
  int  dy = - (y - ReferenceY); // because the y axis is down

  ViewportXmin += dx;
  ViewportXmax += dx;
  ViewportYmin += dy;
  ViewportYmax += dy;
  SetViewport();

  ReferenceX = x;
  ReferenceY = y;

  glutPostRedisplay(); // puts a "redraw" event in the event queue
}
//---------------------------------------------------------------------
void myKey(unsigned char key, int x, int y )
{
  switch (key)
  {
    case 'u'   : ViewportYmin += 2; break;
```

Drawing a Parameterized Regular Polygon:



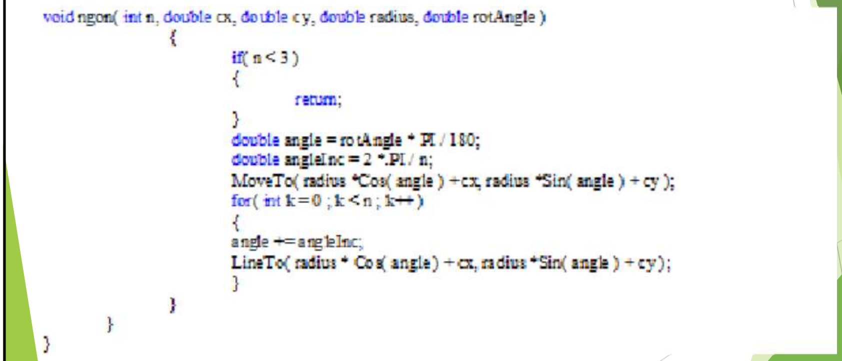
$P_i = (R\,cos(2\pi i / n), R\,sin(2\pi i / n))$, for $i = 0, \ldots, n-1$



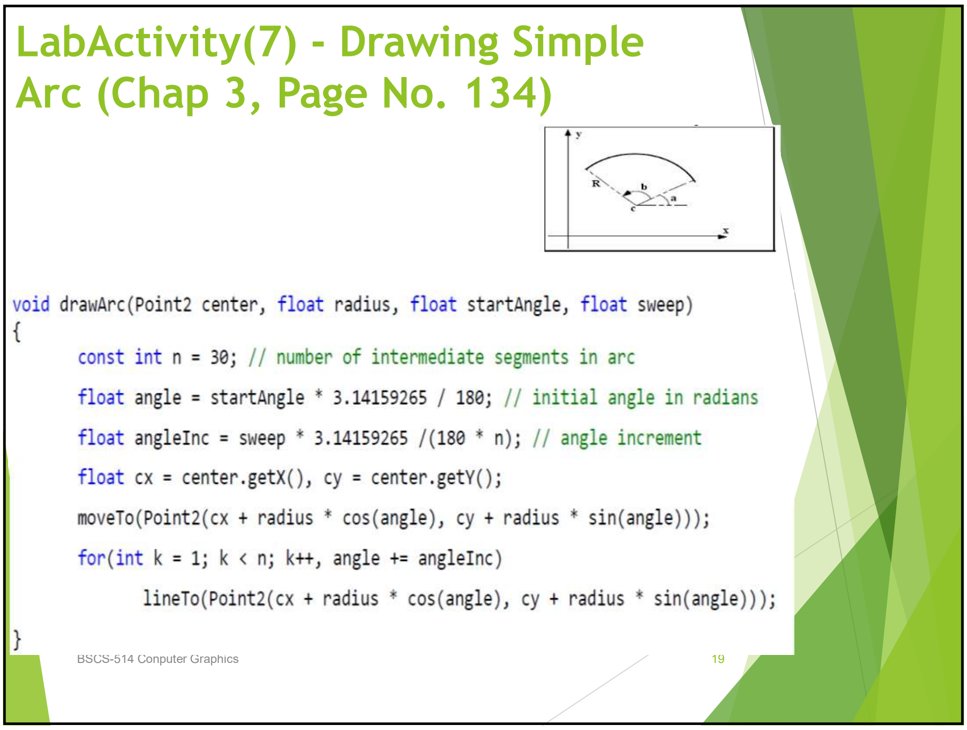
Lab Activity (6): Modify Display Function as follows

- myDisplay()
  {
  ……………..
  ngon(6, 20, 20, 10, 0);
  ……………………………… }

"If the no of sides of an n-gon is large, the n-gon approximates a CIRCLE in an appearance"



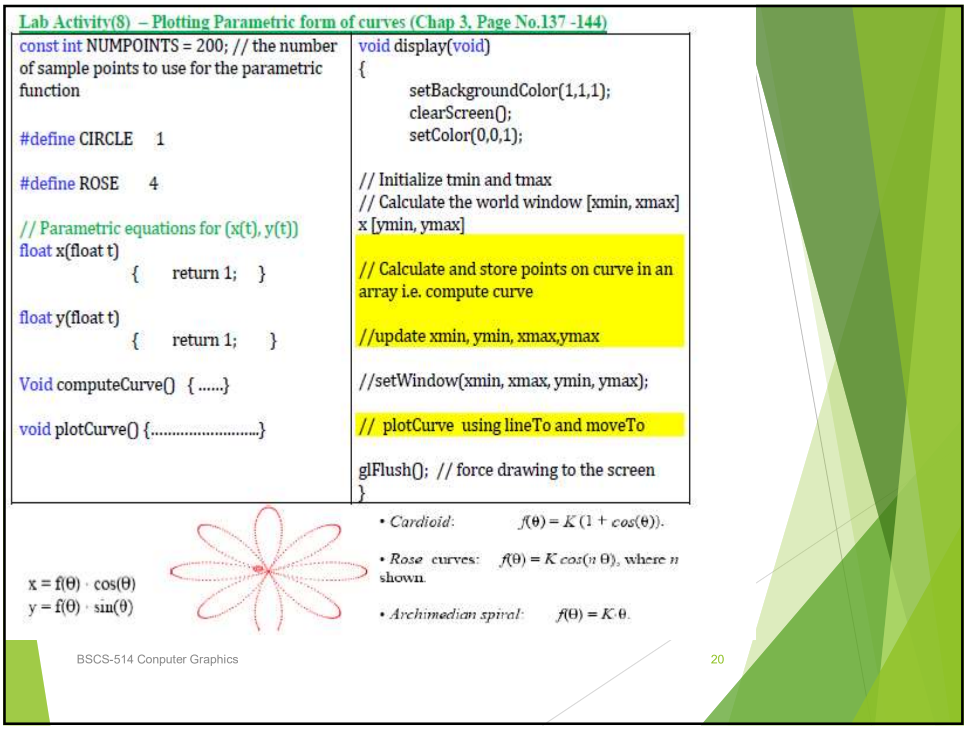
```
void ngon( int n, double cx, double cy, double radius, double rotAngle )
                  {
                         if( n < 3 )
                         {
                                 return;
                         }
                         double angle = rotAngle * PI / 180;
                         double angleInc = 2 *.PI / n;
                         MoveTo( radius *Cos( angle ) +cx, radius *Sin( angle ) + cy );
                         for( int k = 0 ; k < n ; k++ )
                         {
                         angle += angleInc;
                         LineTo( radius * Cos( angle ) + cx, radius *Sin( angle ) + cy );
                         }
                  }
         }
}
```



LabActivity(7) - Drawing Simple Arc (Chap 3, Page No. 134)

```
void drawArc(Point2 center, float radius, float startAngle, float sweep)
{
      const int n = 30; // number of intermediate segments in arc
      float angle = startAngle * 3.14159265 / 180; // initial angle in radians
      float angleInc = sweep * 3.14159265 /(180 * n); // angle increment
      float cx = center.getX(), cy = center.getY();
      moveTo(Point2(cx + radius * cos(angle), cy + radius * sin(angle)));
      for(int k = 1; k < n; k++, angle += angleInc)
            lineTo(Point2(cx + radius * cos(angle), cy + radius * sin(angle)));
}
```

BSCS-514 Conputer Graphics 19



Lab Activity(8) – Plotting Parametric form of curves (Chap 3, Page No.137 -144)

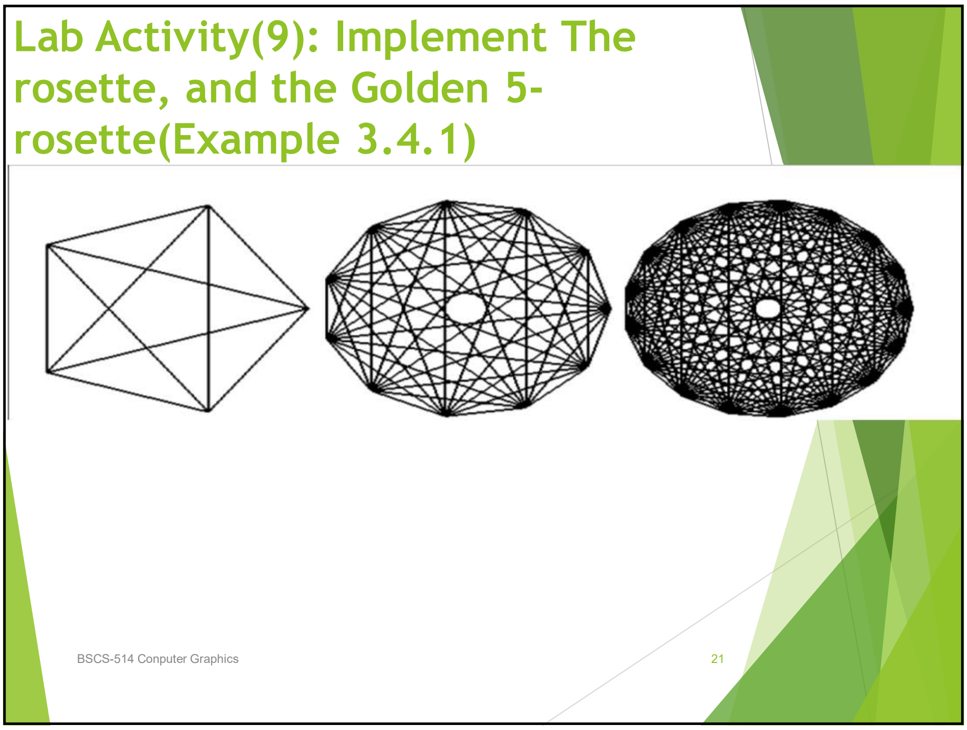
```
const int NUMPOINTS = 200; // the number
of sample points to use for the parametric
function

#define CIRCLE    1

#define ROSE     4

// Parametric equations for (x(t), y(t))
float x(float t)
              {     return 1;    }

float y(float t)
              {     return 1;      }

Void computeCurve()  { ......}

void plotCurve() {..........................}
```

```
void display(void)
{
        setBackgroundColor(1,1,1);
        clearScreen();
        setColor(0,0,1);

// Initialize tmin and tmax
// Calculate the world window [xmin, xmax]
x [ymin, ymax]

// Calculate and store points on curve in an
array i.e. compute curve

//update xmin, ymin, xmax,ymax

//setWindow(xmin, xmax, ymin, ymax);

//  plotCurve  using lineTo and moveTo

glFlush();  // force drawing to the screen
}
```

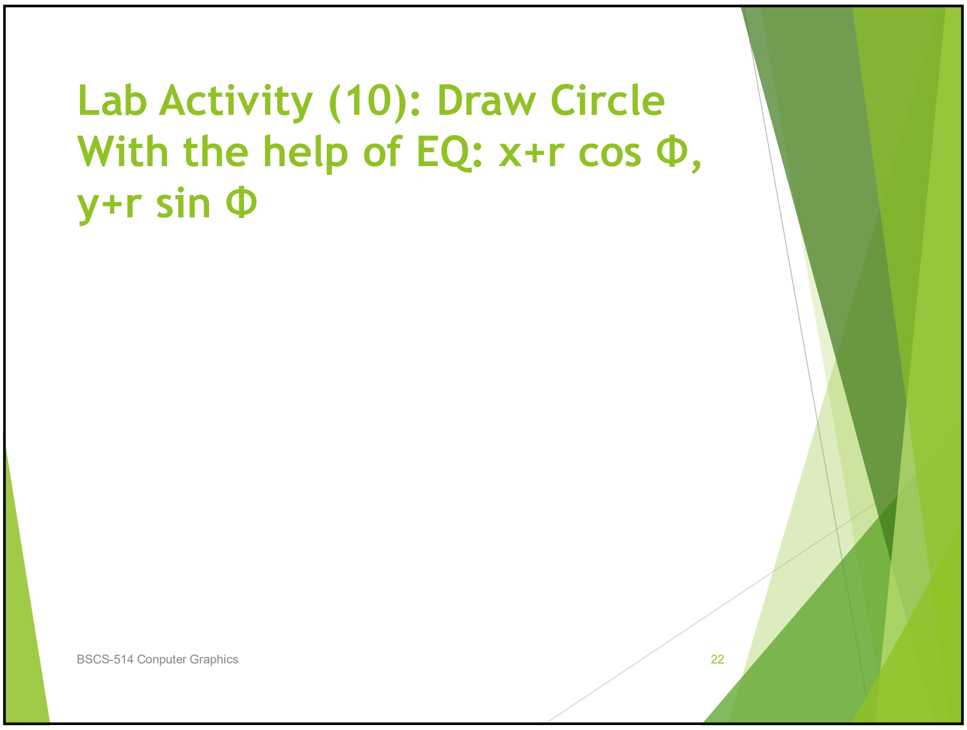
$x = f(\theta) \cdot \cos(\theta)$
$y = f(\theta) \cdot \sin(\theta)$

- *Cardioid:* $f(\theta) = K\,(1 + cos(\theta))$.
- *Rose* curves: $f(\theta) = K\,cos(n\,\theta)$, where n shown.
- *Archimedian spiral:* $f(\theta) = K \cdot \theta$.

BSCS-514 Conputer Graphics 20



Lab Activity(9): Implement The rosette, and the Golden 5-rosette(Example 3.4.1)



Lab Activity (10): Draw Circle With the help of EQ: x+r cos Φ, y+r sin Φ