I. Structs
   A. Image
      i. Pixel **data – a double array of Pixels that stores the pixel information for the image. Can be accessed with data[rows][cols].
      ii. int rows, cols – two ints that hold the number of rows and cols
      iii. char filename[255] – holds the filename of the image
   B. Point
      i. double val[4] – a double array that holds x, y, z, and h values of coordinates
   C. Line
      i. Point a, b – two points that are the beginning and end of the line
   D. Circle
      i. Point c – the center of the circle
      ii. double r – the radius of the circle
   E. Ellipse
      i. Point c – the center of the ellipse
      ii. double ra, rb, a – ra is the major axis (x), rb is the minor axis (y), and a is the angle of the minor axis relative to the major axis (not implemented)
   F. Polygon – draws a polygon, connecting the last and first vertices
      i. int numVertex – the number of vertices
      ii. Point *vertex – a point array of the vertices
   G. Polyline – draws a polygon, leaving the last and first vertices unconnected, same fields as polygon

II. Methods
   A. Image
      i. constructors and destructors - *Image_create() makes an image where data is size 0, rows and cols are 0, and the filename is a string of nothing.
      *Image_init(int rows, int cols) uses rows and columns to malloc the Pixel **data.
      Image_free(Image *src) frees all the memory
      ii. I/O functions - *Image_read(char *filename) creates an Image from the filename using the ppmIO.h method readPPM. Image_writePPM(Image *src, char *filename) writes the given image to the given filename using the ppmIO.h stuff. The last Image_write is not implemented
      iii. access functions – the Image_get functions return Pixels of either data[i*stuff] or data[r][c]. the Image_set functions set the data[i*stuff] or data[r][c] to a pixel passed into the function
   B. Point
      i. Point_set2D (Point *p, double x, y) – sets a point based on x and y values
      ii. Point_set (Point *p double x, y, z, h) – sets a 3D point with an h value
      iii. Point_draw (Point *p, Image *src, Pixel pix) – draws the point on src using Pixel p
   C. Line
      i. Line_set2D (Line *l, int x0, y0, x1, y1) – sets a line based on an (x0, y0) and an (x1, y1)
      ii. Line_set (Line *l, Point ta, tb) – sets a line based on two Points.
      iii. Line_draw (Line *l, Image *src, Pixel p) – draws the line one src using Pixel p
D. Circle
  i. Circle_set (Circle *c, Point tc, double tr) – sets the circle using center point c and radius r.
  ii. Circle_draw (Circle *c, Image *src, Pixel p) – draws the circle on src using Pixel p
  iii. Circle_drawFill (Circle *c, Image *src, Pixel p) – draws the filled circle on src using Pixel p
  iv. rowFill(Image *src, int row, int x1, int x2, Pixel p) – fills across a row, called in drawFill
  v. Circle_plotPoints(Image *src, int c_y, c_x, y, x, Pixel p) – reflects across the octants

E. Ellipse
  i. Ellipse_set (Ellipse *e, Point tc, double ta, tr) – sets the ellipse using center point c, major axis radius ra, minor axis radius rb, and non-implemented angle between major and minor axes a
  ii. Ellipse_draw (Ellipse *e, Image *src, Pixel p) – draws the ellipse on src using Pixel p
  iii. Ellipse_drawFill (Ellipse *e, Image *src, Pixel p) – draws the filled ellipse on src using Pixel p
  iv. rowFill(Image *src, int row, x1, x2, Pixel p) – fills across a row, called in drawFill
  v. Ellipse_plotPoints(Image *src, int c_y, c_x, y, x, Pixel p) – reflects across the quadrants

F. Polygon
  i. Polygon_create(void), *Polygon_init(int numv, Point *vlist), void
     Polygon_set(Polygon *p, int numv, Point *vlist) – creates polygons, either with null info, or with passed in info
  ii. void Polygon_drawFrame(Polygon *p, Image *src, Pixel c) – draws the frame of a polygon
  iii. void Polygon_drawFill(Polygon *p, Image *src, Pixel c) – draws a filled polygon
  iv. void fillPgon(int cnt, Point p[], Pixel color, Image *im) – is called by drawFill, goes into scanfill.c

G. Polyline
  i. Polyline_create(void), *Polyline_init(int numv, Point *vlist), void
     Polyline_set(Polyline *p, int numv, Point *vlist) – creates polylines, with null info or with info that is passed in
  ii. Polyline_Polyline_drawFrame(Polyline *p, Image *src, Pixel c) – draws a polyline on the image, polyline does not have a drawFill method because there is no guarantee it will be a filled shape

III. Accessing – if there are no accessor methods, just refer to the state variables directly. This is usually how the graphics routines do it.

IV. Storage – the header file for the environment is environment.h, stored in the include directory. The source files for the primitive image types are in the lib directory. These files link to make the library libmygraphics.a. There is one file for each primitive image type, so an image.p, point.c, line.c, circle.c, and ellipse.c.
   C files in src should include both the ppmIO.h and environment.h to successfully use Pixels and image types.

V. Misc. Notes
  A. scanfill.c – a piece of C code mostly written by Professor Maxwell. Does the actual
filling algorithm for the polygon