```
//These are my notes from Monday. I will create a test file to actually
work the program.

/*
      shape.h
      Stephen Kim (dskim@iupui.edu)
      version 5.0
*/
#ifndef SHAPE_H
#define SHAPE_H
#define  _USE_MATH_DEFINES  // M_PI
#include <stdio.h>
#include <math.h>
#define deg2rad(x)      ((x) * M_PI / 180.0)
#define PI M_PI

typedef struct Point {
    double x, y;
} Point;

struct multipoint {
    int npoints;
    Point *vertex;
};

typedef struct multipoint Polyline;
typedef struct multipoint Polygon;

typedef struct Circle {
    Point center;
    double radius;
} Circle;

typedef struct Arc {
    Point center;
    double radius;
    double angle[3];        // first, middle, last angles in degree
} Arc;


/*********************************************************************
***
    Functions developed by Nathan Christian
**********************************************************************
***/
void print_help (void); // source code is in geodraw.c
void high_low(Point *low, Point *high, Point *t_low, Point *t_high);

/*********************************************************************
***
    shape base function prototypes
**********************************************************************
***/
double distance(Point a1, Point b1);
```

```
int    convert(Point *p, int n, int *x, int *y);
char  *fget_line(char *stream, int max, FILE *fp);
void multipoint_clean(struct multipoint *mp);  //multipoing is already
defined.

/*************************************************************************
***
    arc function prototypes
**************************************************************************
***/
// arc io functions
int arc_draw(FILE *fp, Arc *arc, char *attr);
int arc_sprintf(char *stream, Arc *arc);
int arc_sscanf (char *stream, Arc *arc);

// arc transform functions
int arc_move(Arc *arc, double theta, double distance2, struct Point
delta);
int arc_rotate(Arc *arc, double angle);

// arc property functions
double arc_length(Arc *arc);
double arc_area(Arc *arc);
void arc_boundary(Arc *arc, Point *lower, Point *upper);

// arc manipulation functions
Arc *arc_duplicate(Arc *arc);

/*************************************************************************
***
    polyline function prototypes
**************************************************************************
***/
// polyline io functions
int polyline_draw(FILE *fp, Polyline *polyline, char *attr);
int polyline_sprintf(char *stream, Polyline *polyline);
int polyline_sscanf (char *stream, Polyline *polyline);

// polyline transform functions
int polyline_move(Polyline *polyline, double theta,struct Point delta);
int polyline_rotate(Polyline *polyline, double angle);

// polyline property functions
double polyline_length(Polyline *polyline);
double polyline_area(Polyline *polyline);
void polyline_boundary(Polyline *polyline, Point *lower, Point *upper);

// polyline manipulation functions
Polyline *polyline_duplicate(Polyline *polyline);

/*************************************************************************
***
    polygon function prototypes
```

```
************************************************************************
***/
// polygon io functions
int polygon_draw(FILE *fp, Polygon *polygon, char *attr);
int polygon_sprintf(char *stream, Polygon *polygon);
int polygon_sscanf (char *stream, Polygon *polygon);

// polygon transform functions
int polygon_move(Polygon *polygon, double theta, struct Point delta);
int polygon_rotate(Polygon *polygon, double angle);

// polygon property functions
double polygon_length(Polygon *polygon);
double polygon_area(Polygon *polygon);
void polygon_boundary(Polygon *polygon, Point *lower, Point *upper);

// polygon manipulation functions
Polygon *polygon_duplicate(Polygon *polygon);

// polygon creation functions
Polygon *create_random_polygon(int n);
Polygon *create_random_convex_polygon(int n);
Polygon *create_random_isogonal(int n);
Polygon *create_random_isotoxal(int n);
Polygon *create_random_rectilinear(int n);

// polygon unique property functions
Point *polygon_centroid(Polygon *polygon);
int    polygon_is_simple(Polygon *polygon);
int    polygon_is_convex(Polygon *polygon);
int    polygon_is_concave(Polygon *polygon);
int    polygon_is_equiangular(Polygon *polygon);
int    polygon_is_equilateral(Polygon *polygon);
int    polygon_is_tangential(Polygon *polygon);
int    polygon_is_isogonal(Polygon *polygon);
int    polygon_is_isotoxal(Polygon *polygon);
int    polygon_is_rectilinear(Polygon *polygon);

/***********************************************************************
***
    circle function prototypes
************************************************************************
***/
// circle io functions
int circle_draw(FILE *fp, Circle *circle, char *attr);
int circle_sprintf(char *stream, Circle *circle);
int circle_sscanf(char *stream, Circle *circle);

// circle transform functions
int circle_move(Circle *circle, double theta, double distance2, struct
Point delta);
int circle_rotate(Circle *circle, double angle);

// circle property functions
```

```
double circle_length(Circle *circle);
double circle_area(Circle *circle);
void circle_boundary(Circle *circle, Point *lower, Point *upper);

// circle manipulation functions
Circle *circle_duplicate(Circle *circle);


#ifdef SHAPE_LOG

#define SHAPE_LOG_FUNCTION() fprintf(stderr, "%s()\n", __FUNCTION__)

#else

#define SHAPE_LOG_FUNCTION() (0)

#endif          // #ifdef SHAPE_LOG

#endif          // #ifndef SHAPE_H
```