24-780 B—Engineering Computation

Assigned: Mon. Oct 18, 2021
Due: Tues. Oct. 26, 2021, 11:59pm

Problem Set 6: Shape

Implement an algorithm that calculates some properties for a geometric shape and incorporate it into existing code. That's it, no more.

I expect this assignment to take about 4-6 hours only, so that you can give a big push on the development of your individual project (the demo).

Task 1: Familiarize yourself with the existing system

Much of the code you'll need to manage, visualize, read/write a shape is already included in the attached Visual Studio project (you can import the same code on Xcode). Note that the *ViewManager* class has been created to carry the burden of running the interface such that the main() function does almost nothing.

In particular, please note the following:

- The input files should now have the extension ".shape" rather than ".gear" (feel free to change this in *ViewManager* so the file loading stays simple)
- The code you receive is adapted from last year's assignment so the features associated with "spline" are turned off or repurposed for other users.
- A shape is "closed" when it is drawn and may be filled in or not (add "F" key event in *ViewManager* to toggle between the two modes). HINT: You will need a member variable in *Shape* to keep track of this. It is up to you if you want to have an "outline" when your shape is filled in.
- You can add points graphically through the interface, but you can also just edit the files.

I've attached some shape files for your use and testing, but feel free to create simpler, more predictable shapes (e.g., rectangles, circles, triangles, etc.)

Feel free to go through the files given to "make them your own" by adding comments, editing variables names, etc.

Task 2: Implement calcProperties() function

We want to add a function to the provided code so that we can calculate some section properties of a shape. In particular, we want to calculate the area, the moments of inertia about the orthogonal neutral axes (X and Y), and the centroid coordinates. We'll discuss in lecture how to approach this problem, so don't worry about it too much right now.

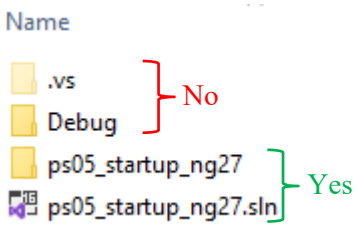
Deliverables

5 files (zipped together):

ViewManager.h, ViewManager.cpp
Shape.h, Shape.cpp
ps06_shape_andrewID.cpp

Upload the zip file to the class Canvas page before the deadline (Tuesday, Oct. 26, 11:59pm).

Alternatively, if you are using Visual Studio, it may be easier to submit your entire solution rather than a collection of files. To do this, create a *zip file* of the whole project (the .sln file and the associated folder), being careful NOT to include the hidden folder called ".vs". This folder is used only to manage the IDE and is typically huge (100MB). Erasing or omitting it will just force Visual Studio to rebuild it when needed. The Debug folder should also be kept out of the zip file to avoid including executable files that some firewalls may disallow. *The name of the project should include your AndrewID*



Learning Objectives

Use of classes and objects in C++.

Mouse input in GUIs

Adapting existing code for new objectives

Increasing understanding of graphical data representation.

Implementing algorithms developed by others.