Chris Hopp

Professor Boris Jeremic

ENG-104A

12 December 2018

Inertia Calculator

I want to start out with the statement that I find coding to be an extremely frustrating task. I find the logic and algorithms behind a computation to be interesting, but often find the syntax cumbersome and illogical. To compound my problems, I have a background in C, C++, and Java, but have never used languages such as Python or MATLAB (which I’ve been told are far more user friendly). For this project, I used C++ as that is the most recent language I have taken a course in and what I am currently most comfortable with.

I initially started out with a program that I felt was a very well thought out organization. My goal was to utilize an abstract base class for a shape that could be expanded to include classes for various geometries. In this project we would be using an object of a rectangle and an object for a circle. These objects would then be used to make up a cross section. This code would have been, what I consider, more professional. It would have allowed for a more modular approach and lent itself to being utilized for more than just the calculations for this project. However, I ran into a number of difficulties in implementation. In hindsight, I should have used arrays of pointers instead of trying to pass by value. Recognizing that the complexity of my program would greatly increase the further I went, I decided to abandon this approach. While not functional, I did include the code labeled as “Inertia Calculator Broken” with my submission.

Instead, I wrote another program that sacrificed elegance and professionalism for functionality. The program still utilizes a user interface for the user to enter information about a cross section and its corresponding shapes to create a cross section object. However, the bulk of the work done by the program is in the constructor for the object itself. There is a great loss of modularity with this approach. I also utilized a number of for loops to iterate through arrays which cut into program efficiency. The function to calculate the normal stress given a moment is very specific to this project in that it calculates “c” as the distance from the x-axis to the centroid. This should calculate the maximum value of “c” measured from the centroid to either the top or bottom of the beam. All of the shapes in our project have a centroid located such that the maximum “c” is always measured from the centroid to the x-axis. With more time I would have calculated a maximum at the top of the beam from shape height and centroid location to compare with the value to the x-axis.

The user is able to enter in any cross section comprised of circle or rectangular shapes. A density parameter is added to allow the user to input both solid shapes and holes. The program then calculates the moments of inertia about each axis for each shape as well as the product of inertia. These are then stored in an array to be summed to calculate the total inertia of the object with parallel axis theorem applied. Maximum and minimum inertia are then calculated using the formula along with stress from a 10kN m moment.

Results are output to an included text file. They were checked against hand calculations and the program functions as designed. I started with 20 cross sections to compensate if I made a mistake in data entry. I added a cross section of a square after shape 12 and 15 for comparison and entered in zeros for the remaining sections.