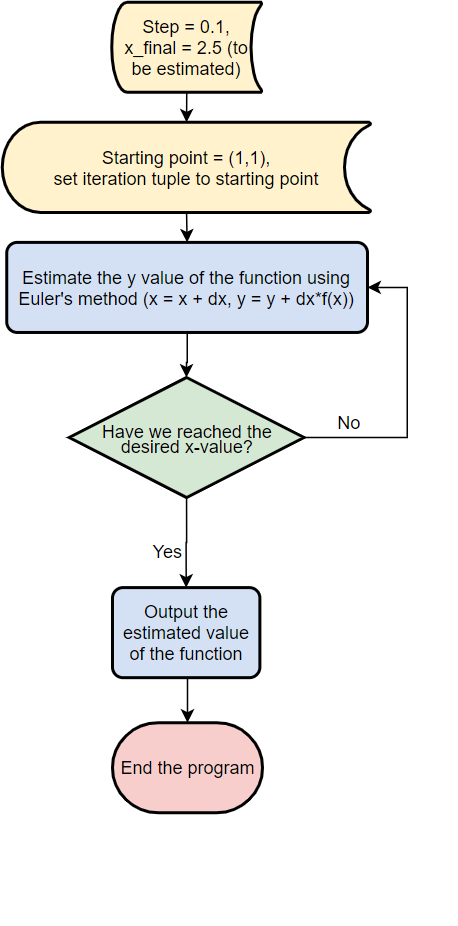


**Report for Programming Lab #2**

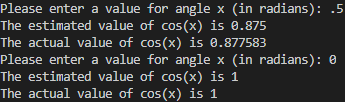
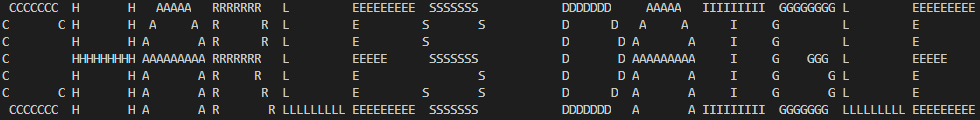
Charles Daigle

**Bala Maheswaran**

**Non-Programming Questions**

1. Below is the flowchart corresponding to the first program (temperature conversion):

**Code Outputs**

1. 
2. 
3. 
4. 

**Results/Discussion**

In this lab, the first three problems used iteration and aggregating each iteration result into three ways of numerically approximating differential equation solutions, area under a curve, and a function’s Taylor expansion. These demonstrated the ability for a computer to perform many iterations of a given task very quickly.

The extra credit problem was solved after a long amount of time spent trying to use a regular three-dimensional array. However, I found much trouble in this – the data was impossible to retrieve or process as is. It was having issues concatenating letters to other letters horizontally – I couldn’t add pre-existing 2d arrays into an array of their own (like I could in Python). I did some research for a few hours on standard C++ libraries that might help us store/retrieve data. I found a library called <vector> that acts very similarly to regular multi-dimensional arrays, but they can be very easily concatenated into new vectors containing vectors. The data structures make it very easy to make a new letter and make any large ASCII word using these letters – all you have to do is declare a new variable with (kind of hilarious looking) type vector<vector<vector<char>>>. Then, you just place into it sequentially the names of variables that represent letters. There is then a function that uses a triply nested for loop to iterate over one of these lists of letter variables, and this unzips each row and prints it horizontally. I really enjoyed solving this problem, as it was very difficult, and I think my solution is extremely clean – other than the structures containing letter data, the actual algorithm is only around 25 lines of code.