Debug

Overall, this project helped introduce me to the Sympy library and the concept of I-frames to connect Html to Python. Near the beginning, I had an issue with getting changes in the CSS to match up with changes in the Html. Later, I found out that this was because the code in ID’s needed to match up with the code in classes. I also need to press ctrl F5 to keep the changes going. After a while, I realized that using the Sympy library would be a lot easier than manually coding every mathematical function, so I converted my code to incorporate this library, along with many elif statements and while loops to determine which function to use. In the end, I came across many argument errors for Sympy functions because I converted outputs to strings by mistake.

Project Design

1. I will create a calculator that will help IMSA students with their math classes. Since I am in differential equations, the program will initially function as a differential equations solver. I will later expand it to solve math operations from other courses, too. To add to the graphics portion of the project, I will create a window with the graphics module, with multiple buttons for the user to calculate values with.
2. To do this project, I will need to use HTML and Python for a good user interface. There will be multiple pages on the website, one for each function. For example, to solve a system of ODE’s (with distinct, real eigenvalues), I will have the user input a matrix and calculate the solutions using eigenvalues and eigenvectors. To get the determinant of an n-size matrix, I can use a while-loop. For a 4x4 matrix, I can get the determinants of the 3x3 matrices (using 2x2 matrices). To find a derivative, I can use if/elif/else statements to find out if I need to use any rules.
3. Based on which page the user goes to, I can call upon the appropriate function.
4. / 5. Here is the code for systems of ODE’s:

#KomoravoluQ4.py

#Sameer Komoravolu

#5.4.2020

import math

#x' = ax + by

#y' = cx + dy

#A = [a b]

# {c d}

'''print(" {a b} ")

print("x' = [c d]x")

'''

#L^2 - aL - dL + ad - bc = 0

def MatrixInput():

matrixSize = (int(input("What is the size of the matrix?")))\*\*2

matrix = []

for i in range(matrixSize):

element = input("Element" + str(i+1) + ": ")

matrix.append(element)

return(matrix)

def Eigen(matrixInput):

a = float(matrix[0])

b = float(matrix[1])

c = float(matrix[2])

d = float(matrix[3])

L1 = ((a + d) - math.sqrt((a + d)\*(a + d) - 4\*(a\*d - b\*c))) / 2

L2 = ((a + d) + math.sqrt((a + d)\*(a + d) - 4\*(a\*d - b\*c))) / 2

if L1 != L2:

x1 = -d/(a - L1)

if x1 < 1 and x1 > -1:

y1 = 1/x1

x1 = 1

else:

y1 = 1

#E1 = x1, y1

x2 = -d/(a - L2)

if x2 < 1 and x2 > -1:

y2 = 1/x2

x2 = 1

else:

y2 = 1

#E2 = x2, y2

eigenList = [L1, L2, x1, x2, y1, y2]

return eigenList

def sysODE\_solver(eigenList):

if E1[0] == 1:

const1 = ""

elif E1[0] == -1:

const1 = "-"

else:

const1 = str(E1[0])

if E1[1] == 1:

const2 = ""

elif E1[1] == -1:

const2 = "-"

else:

const2 = str(E1[1])

if E2[0] == 1:

const3 = ""

elif E2[0] == -1:

const3 = "-"

else:

const3 = str(E2[0])

if E2[1] == 1:

const4 = ""

elif E2[1] == -1:

const4 = "-"

else:

const4 = str(E2[1])

x = const1 + "c1" + "e^" + str(L1) + "t" + "+" + const3 + "c2" + "e^" + str(L2) + "t"

y = const2 + "c1" + "e^" + str(L1) + "t" + "+" + const4 + "c2" + "e^" + str(L2) + "t"

print("x(t) = " + x)

print("y(t) = " + y)

#E1: (a - L1)x + dy = 0

#if x > y:

# x =

matrix = MatrixInput()

print(Eigen(matrix))

Project Proposal

For my 4th quarter project, I want to make a program with an animated user interface, and one that incorporates artificial intelligence. For this project, I will likely need to research machine learning in python. For the user interface, I can model the cases of coronavirus with respect to time. Using the SIR differential equations model, I can input a set of data into the machine and it will predict the next number of cases.