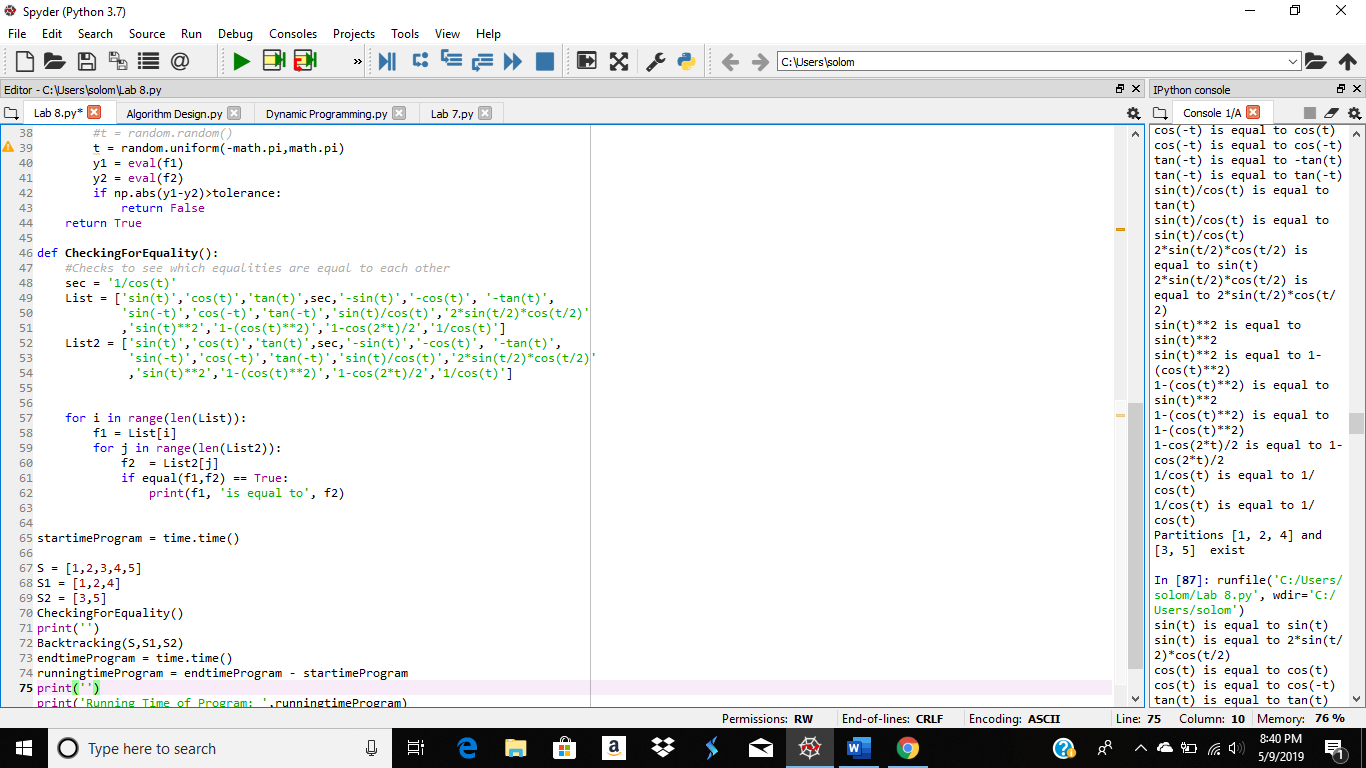
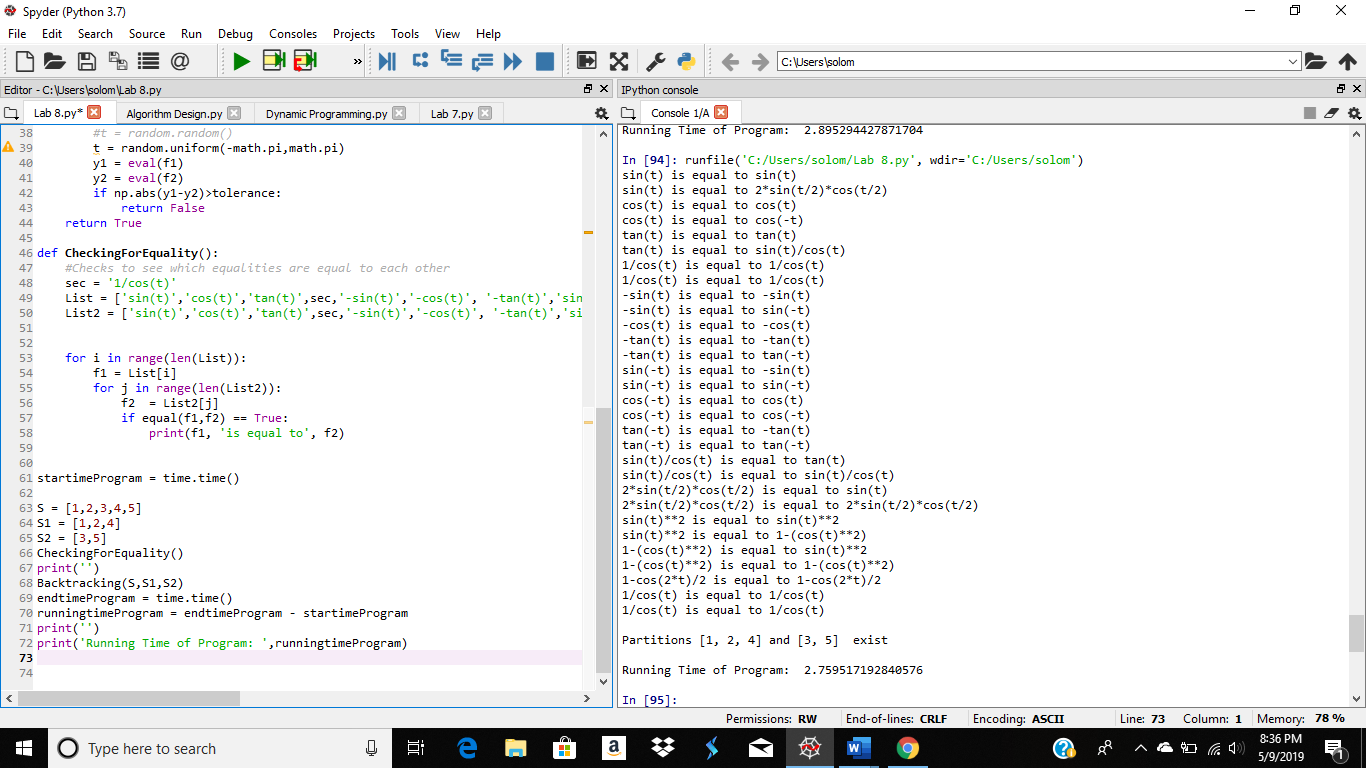
Solomon Davis Lab 8 Report

CS 2302 - MW 1:30 Spring 2019

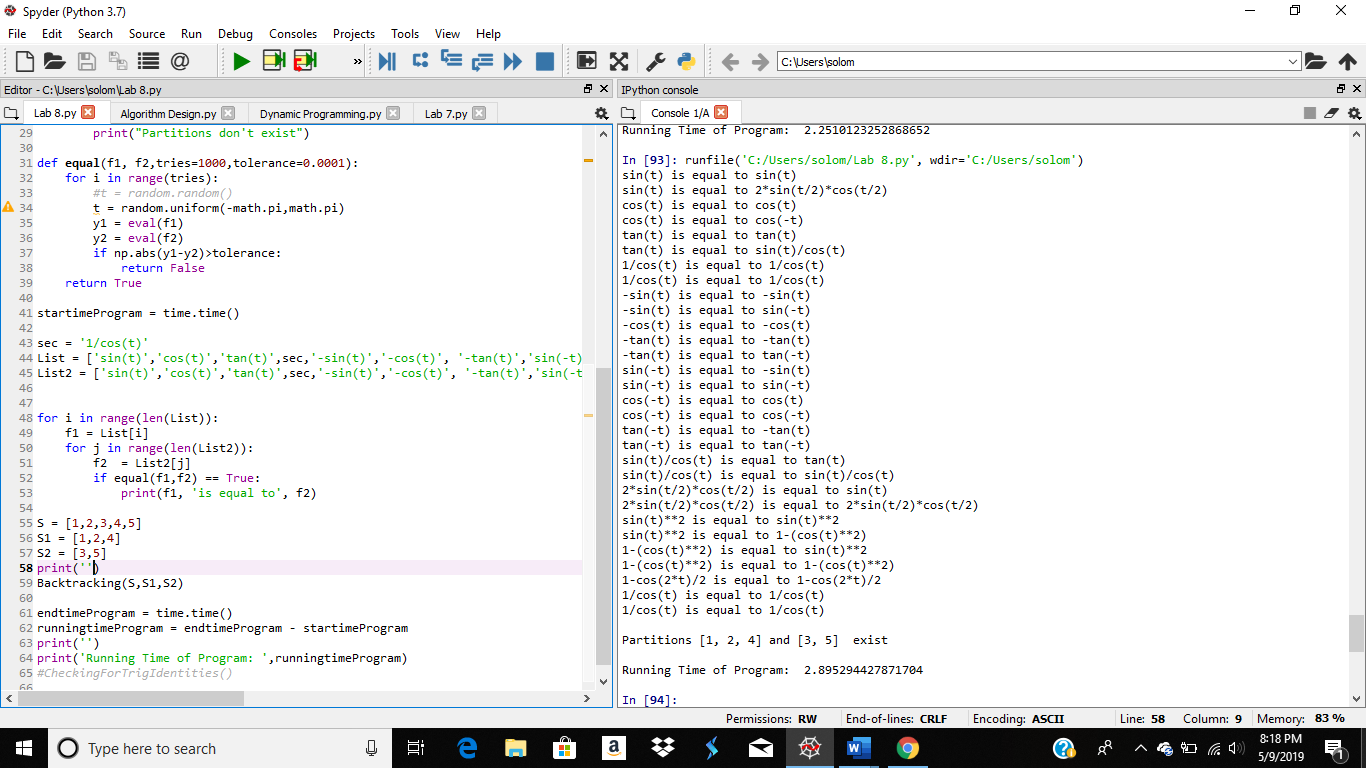
In lab 8 the lab checks to see which equalities are equal to one each other and uses backtracking to see if 2 sets are subsets of a bigger set. To check to see if the equations are equal, I created 2 empty list then I run a 2-dimensional array that compares one equation at a time to all the equations in the list. If an equation is equal to another equation it prints which equations are equal to the screen. To check to see if the sets are subsets of the bigger set, I checked compared each value of each of the smaller set. If the sets are a subset of the bigger set I print “the partitions exist” to the screen. If the sets are not subsets of S I print “the partitions don’t exist” to exist” to the screen.



S1 and S2 are the smaller sets while S is the larger set.



Output of equations and backtracking results



#Course: CS2302 - Spring 2019

#Author: Solomon Davis

#Lab Number: 8

#Instructor: Olac Fuentes

#Last Modified: May 9, 2019

#Due Date: May 9, 2019

#Description: This Lab will create determine whether an equation from the list

# eaual to other equations on the list. This lab will also decide if a list is

# a subset of a bigger set using backtracking.

import random

import numpy as np

import math

from math import \*

import time

def Backtracking(S,S1,S2):

#Decides if two different sets are a subset of a bigger set.

count = 0

count2 = 0

for i in range(len(S1)):

for j in range(len(S)):

if S1[i] == S[j]:

count +=1

for i in range(len(S2)):

for j in range(len(S)):

if S2[i] == S[j]:

count2 +=1

if count == len(S1) and count2 == len(S2):

print("Partitions",S1,"and",S2," exist")

else:

print("Partitions don't exist")

def equal(f1, f2,tries=1000,tolerance=0.0001):

for i in range(tries):

#t = random.random()

t = random.uniform(-math.pi,math.pi)

y1 = eval(f1)

y2 = eval(f2)

if np.abs(y1-y2)>tolerance:

return False

return True

def CheckingForEquality():

#Checks to see which equalities are equal to each other

sec = '1/cos(t)'

List = ['sin(t)','cos(t)','tan(t)',sec,'-sin(t)','-cos(t)', '-tan(t)','sin(-t)','cos(-t)','tan(-t)','sin(t)/cos(t)','2\*sin(t/2)\*cos(t/2)','sin(t)\*\*2','1-(cos(t)\*\*2)','1-cos(2\*t)/2','1/cos(t)']

List2 = ['sin(t)','cos(t)','tan(t)',sec,'-sin(t)','-cos(t)', '-tan(t)','sin(-t)','cos(-t)','tan(-t)','sin(t)/cos(t)','2\*sin(t/2)\*cos(t/2)','sin(t)\*\*2','1-(cos(t)\*\*2)','1-cos(2\*t)/2','1/cos(t)']

for i in range(len(List)):

f1 = List[i]

for j in range(len(List2)):

f2 = List2[j]

if equal(f1,f2) == True:

print(f1, 'is equal to', f2)

startimeProgram = time.time()

S = [1,2,3,4,5]

S1 = [1,2,4]

S2 = [3,5]

CheckingForEquality()

print('')

Backtracking(S,S1,S2)

endtimeProgram = time.time()

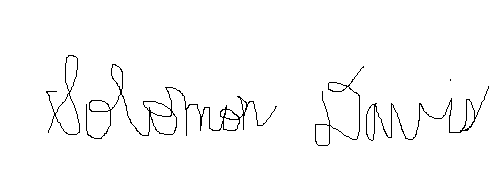
runningtimeProgram = endtimeProgram - startimeProgram

print('')

print('Running Time of Program: ',runningtimeProgram)

Academic Service Certificate:

I certify that this project is entirely my own work. I wrote, debugged, and tested the code being presented, performed the experiments, and wrote the report. I also certify that I did not share my code or report or provided inappropriate assistance to any student in the class.



Solomon Davis