CS471: Introduction to Artificial Intelligence

Assignment 1: Python (10 points)

Ben Foltz-Miranda

8/29/2024

For this problem, you are only allowed to use standard python libraries. You may not use third party libraries or call any shell/bash functions.

You are given a list of tuples of the form (<float> x, <float> y, <float> r) (Let’s call these c-tuples). Each c-tuple represents a circle on a rectangular coordinate space, with x and y being the coordinates of the center, and r being the radius. Assume that each c-tuple has a unique radius.

Let a cluster of circles be a group of circles where each circle in the group overlaps with at least one other circle in that group. A path is formed between two circles when they overlap. Define a cluster as a group of n circles, where each circle is reachable from every other circle through the formed paths.

Write a python script that does the following: Return True if the given circles form a cluster and return false if they don’t form a cluster.

Below are some test cases.

**Test case 1:**

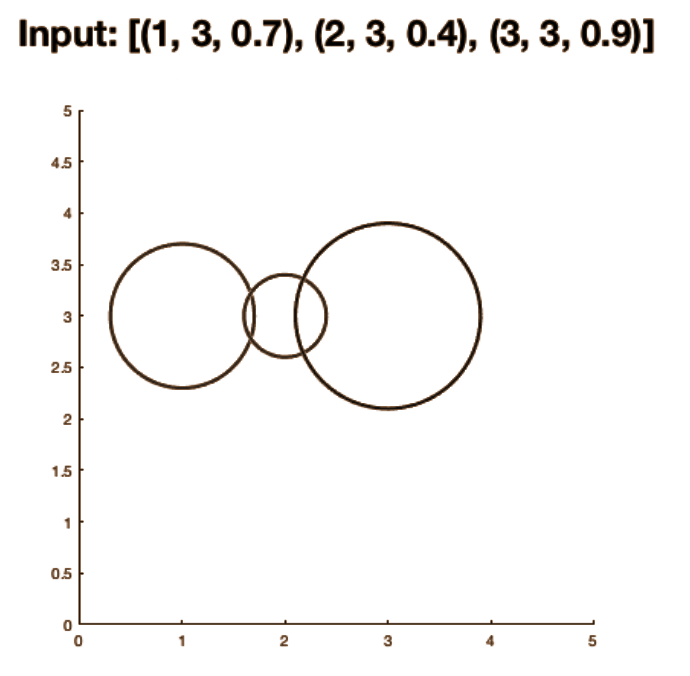


Figure 1: The three circles form the cluster. Output = True

**Test case 2:**

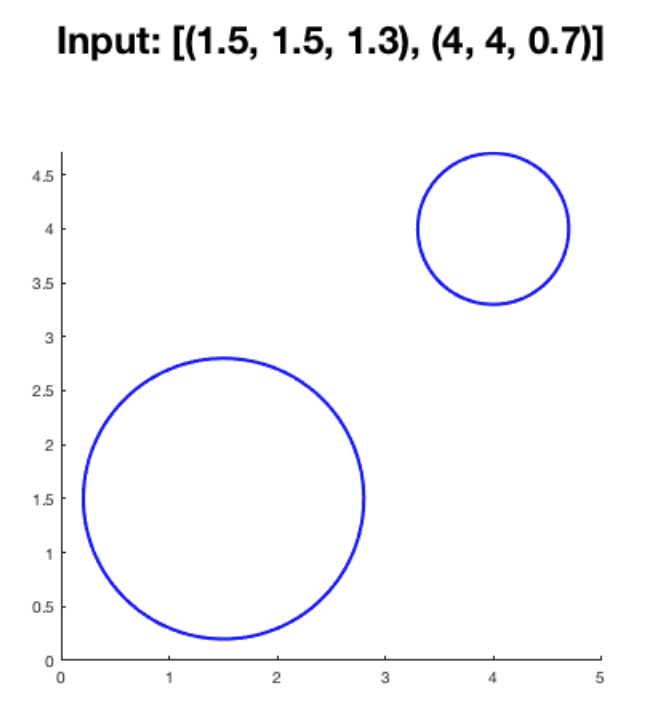


Figure 2: No clusters are found. Output = False

**Test case 3:**

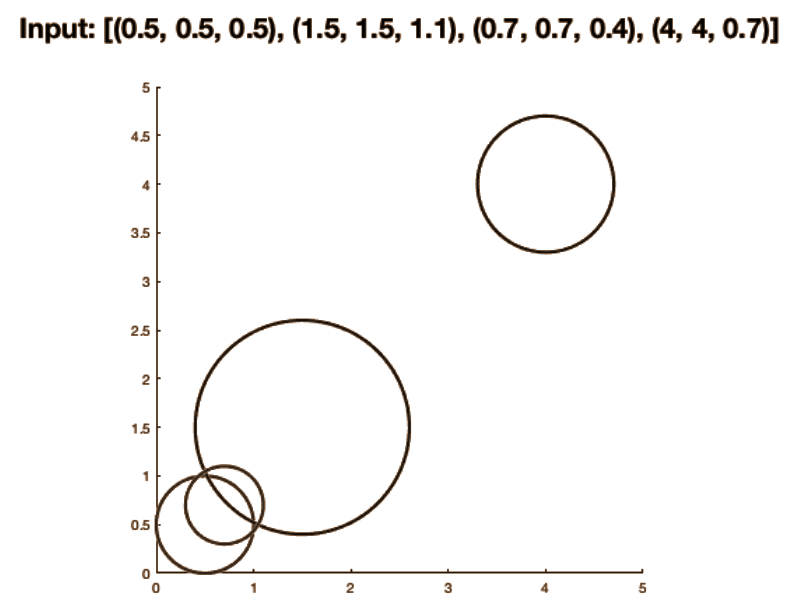


Figure 3: Given circles do not form a cluster. Output = False

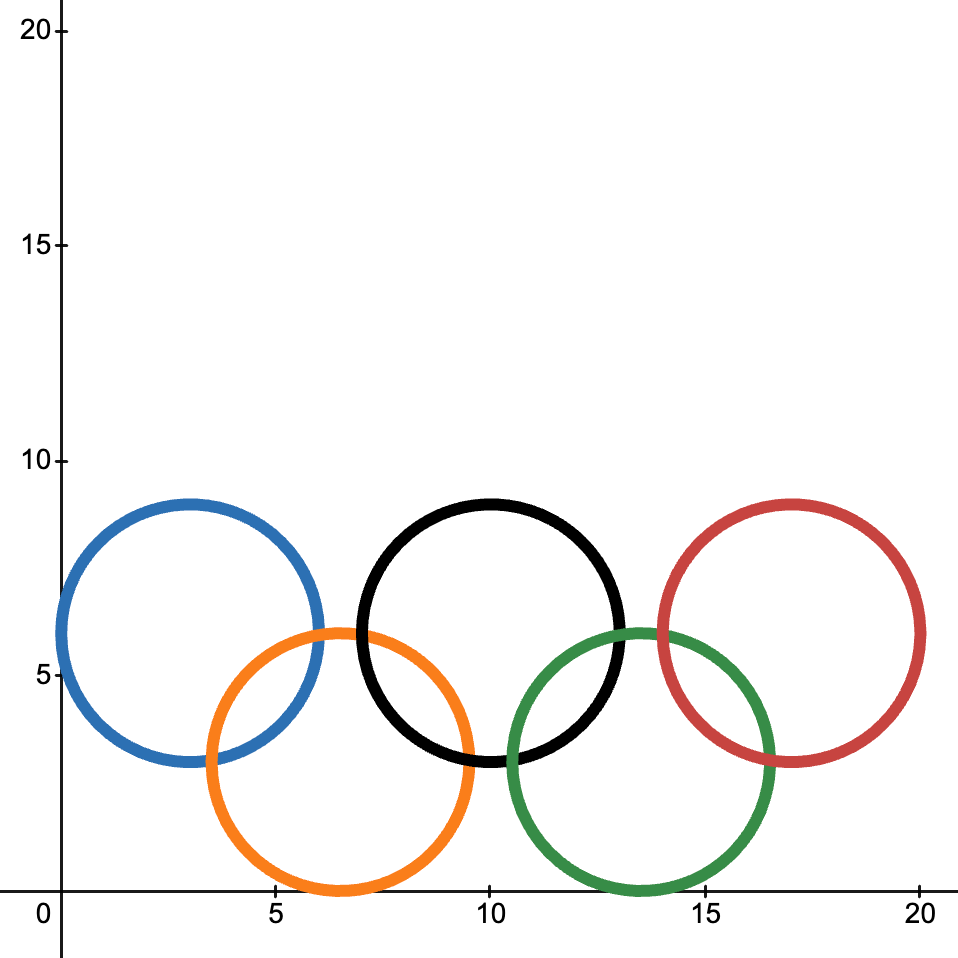
Along with the above three test cases, design a fourth test case of your choice. Share in detail the approach you used to solve the problem.

Hint: Sort the circles based on the radius and determine if the two adjacent circles intersect.

Google Colab Link: [Assignment1.ipynb](https://colab.research.google.com/drive/1HOhvtuwiwEMW24blyv2wk2zf8I1jKyX7?usp=sharing)

**Test case 4:**

**Input: [(3, 6, 3), (6.5, 3, 3), (10, 6, 3), (13.5, 3, 3), (17, 6, 3)]**



I start off by defining a class used to represent a circle called ctuple, the class has 3 attributes, x coordinate, y coordinate, and the radius.

from typing\_extensions import NamedTuple

# Circle class used to represent a circle on a graph

class ctuple(NamedTuple):

x:float

y:float

radius:float

With the ctuple class defined, I created a function to check if two circles overlap by using the distance formula to find the distance between the the center of the two circles and checking if the distance is less than or equal to the sum of the radii

def check\_overlap(circle1, circle2):

# Calculate the distance between the centers of the circles

distance = ((circle1.x - circle2.x) \*\* 2 + (circle1.y - circle2.y) \*\* 2) \*\* 0.5

# Check if the distance is less than or equal to the sum of the radii

return distance <= (circle1.radius + circle2.radius)

Then I created a function that would perform a recursive depth first search to mark the current circle as visited and then would loop through the rest of the circles in the tuple, if the circle has not been marked as visited, it will check if the circles overlap and if they do it will call the function again with the next circle

def dfs(circle, circles, visited):

# Mark the current circle as visted

visited.add(circle)

# Explore the rest of the circles in the circle tuple

for other\_circle in circles:

# Check if the other circle has not been visted but overlaps with the current circle

if other\_circle not in visited and check\_overlap(circle, other\_circle):

# Recursively visit the other circle

dfs(other\_circle, circles, visited)

Lastly I created a function that will take a tuple of circles and determine if the circles are a cluster by starting the depth first search and checking if the length of the visited circles is the same as the number of circles in the tuple.

def check\_cluster(circles):

# If circles is empty, return false

if not circles:

return False

# Initialize a set to keep track of visited circles

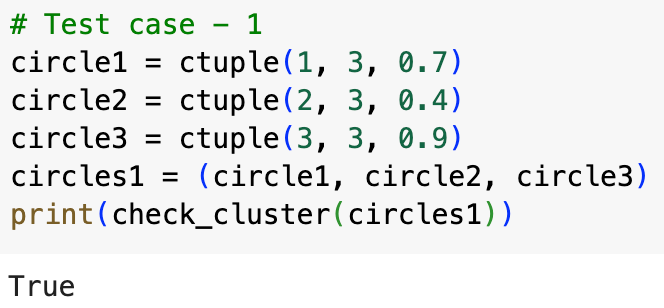
visited = set()

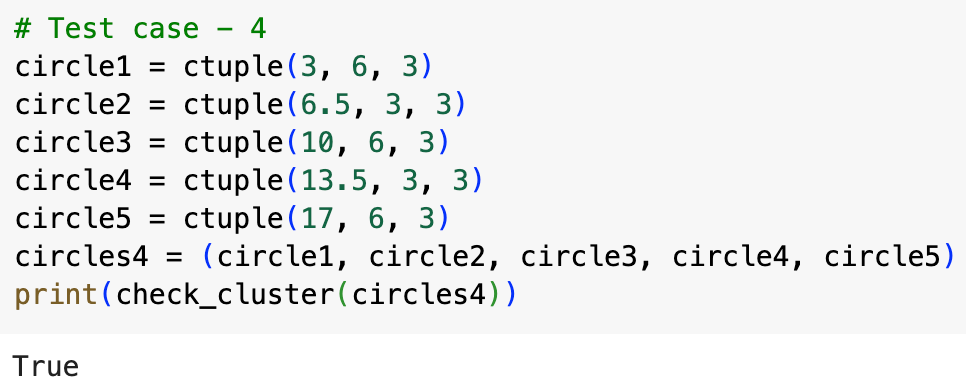
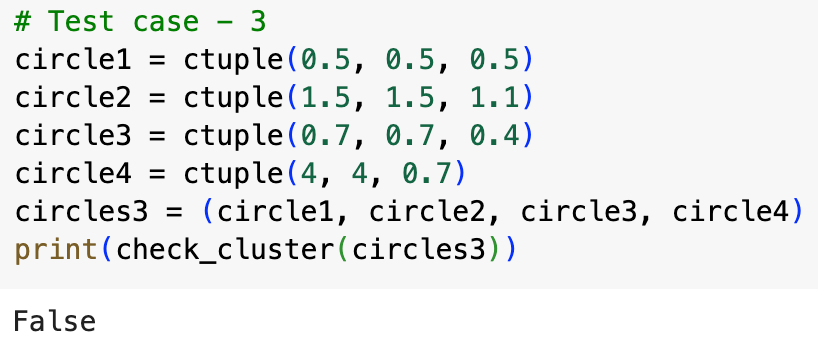
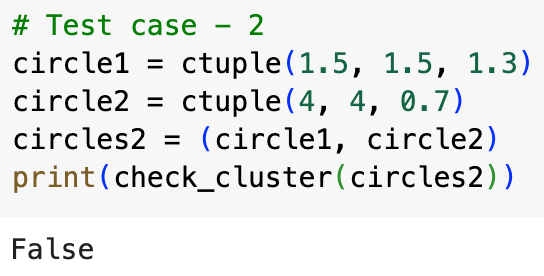
# Start DFS from first circle in tuple

dfs(circles[0], circles, visited)

# Check if all circles have been visited

return len(visited) == len(circles)





**Grading rubric**

Successful testing of test case 1: 2 points

Successful testing of test case 2: 2 points

Successful testing of test case 3: 2 points

Successful testing of test case 4 (designed by you): 2 points

Appropriate commenting and clear formatting of code: 2 points

------------------------------------------------------------------------------------

Total score for this assignment: 10 points