CSCI203
Winter 2025

Programming assignment #1

Date : 20/01/2025

Submission Due dates:

- **Design 27/01/2025 at 22:59 pm Dubai time**
- **Full report 10/02/2025 at 22:59 pm Dubai time**

GROUP OF THREE ASSIGNMENT

(Marked out of 100) contributes 10% to your subject mark

STUDENT ID	FAMILY NAME	FIRST NAME	TUTORIAL DAY/TIME	INSTRUCTOR	SIGNATURE

LEARNING OUTCOMES

1. Design an algorithm, select data structures
2. Specify the input/output requirements of an algorithms
3. Use pseudo-code to specify an algorithm
4. Designing and comparing different Brute-Force approaches to solve a geometric problem.
5. Estimate the asymptotic notation of an algorithm
6. Program your algorithm using a programming language: C, C++, Java, Python
7. Empirically analyse the performance of an algorithm as a function of its input size
8. Empirically compare the performance of two algorithms in the solution of a given problem

Late submission is subject to penalties as specified by the subject outline.

UNIVERSITY
OF WOLLONGONG
IN DUBAI

Faculty of Engineering
and Information Sciences

CSCI 203 Winter 2025 programming assignment #1

Guidelines

- This is your first major programming assignment.
- It is a group assignment.
- This is the same group that you formed in tutorials.
- It will be valid for the entire session.
- **Design is due on: 27/01/2025 at 22:59 pm Dubai time**
- **Full report is due on: 10/02/2025 at 22:59 pm Dubai time**

In this assignment you will have to:

— Design algorithms
— Specify algorithms using pseudo-code
— Select appropriate data structures for your algorithms
— Use the asymptotic notations (O , Θ, Ω) to estimate the complexity of your algorithm
— Code your algorithms in Java or C++
— Compare the empirical estimation of your algorithms to the asymptotic estimation that you determined at the design stage.

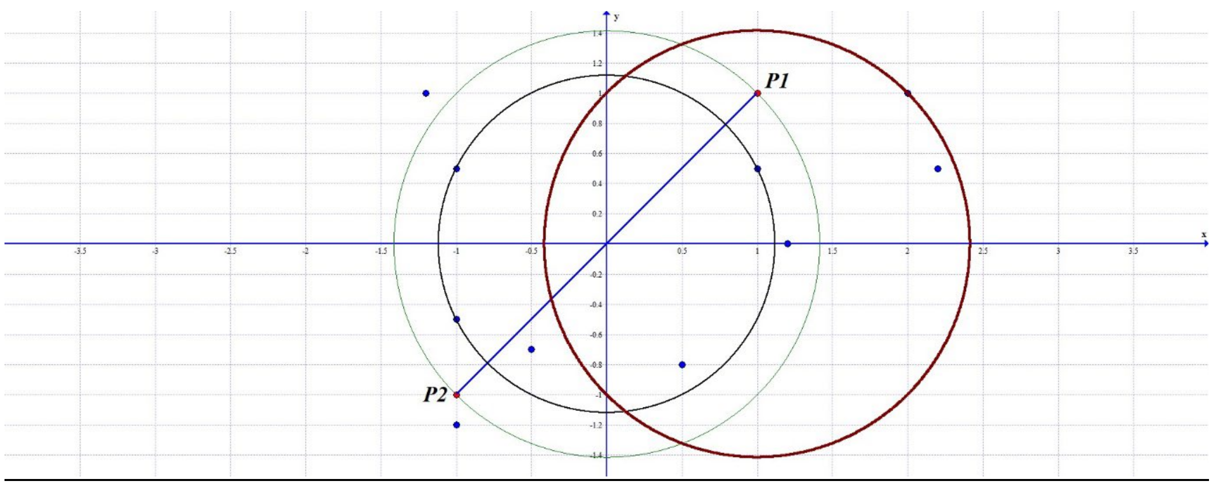
Outline

.

I. The problem:

Your given a **set S** of ***N*** **distinct random points** in a Cartesian plane, and <u>you are asked</u> to find **the smallest circle C*** so that the N given points:

- are either inside the circle
- or on its circumference.



II. Review of a circle properties.

From your calculus classes, you should remember that given a circle C of center O (h, k) and radius a, the standard equation of this circle is given below:

Standard equation of a circle

The circle with centre (h, k) and radius $a \geq 0$ has equation

$$(x-h)^2 + (y-k)^2 = a^2.$$

In particular, the circle with centre at the origin $(0, 0)$ and radius a has equation

$$x^2 + y^2 = a^2.$$

III. Algorithmic approaches to the problem.

In this assignment you will experience how two different algorithm implementations result in different levels of efficiency. Two main approaches are used to solve the same problem. Both are considered as an enhancement of the brute force approach. They are:

a) The two-point approach
b) The three -point approach

In both approaches, you will, you will use the same set **S** of N points.
You must name these points as ***P$_1$*, *P$_2$*, *P$_3$*, ….*P$_N$***.
Each point ***P$_i$*,** is described by its cartesian coordinates (X_i, Y_i)
Your task is to design an algorithmic solution to each approach, then compare them in terms of complexity and accuracy of the results.

A. Approach #1: the two-point approach

This approach can be summarized as follows:
For all possible combinations of two points, let us call them P1, and P2.

a) Build a circle C such that:
   - both P_1 (X_1, Y_1) and P_2(X_2, Y_2) are end points of a diameter of C.
   - Such a circle will be specified by a center $C(X_C, Y_C)$ and a radius R_C given by the following equations:

$$\begin{cases} X_C = \dfrac{X_1 + X_2}{2}; & (1) \\ Y_C = \dfrac{Y_1 + Y_2}{2}; & (2) \\ (R_C)^2 = \dfrac{(X_1 - X_2)^2 + (Y_1 - Y_2)^2}{4} & (3) \end{cases}$$

b) Check if the rest of points of S are included inside S (either on its circumference or inside S)
   - A point P_k (X_k, Y_k) is included in the circle C if :

   $(X_k - X_C)^2 + (Y_k - Y_C)^2 \le (R_C)^2$

c) If the circle you drew did not include all the points in the set S, then select two other points P1 and P2, a repeat the procedure.

d) If the circle you drew does include all the points in the set S,
   - This is a possible solution
   - Compare the radius of this circle, with radius of the previous possible solutions you generated so far
   - In a matter of optimization, you keep the characteristics of the “currently “ optimal circle, that is the circle with the smallest radius as the current possible solution.
   - Then select another pair of points and repeat the procedure, until you tried all possible combination of two points.

B. Brute force approach: the three-point approach

This is another brute force approach. Instead of starting with two points that are located on the circle diameter, you will:

a) Select three random points from the set S,
b) Draw a circle that includes these three points on its circumference
c) Check if the rest of the points are either inside the circle you drew or on its circumference
d) If the circle you drew does include all the points in the set S,
   - this is a possible solution
   - compare the radius of this circle, with radius of the previous possible solution.
   - Keep the circle with the smallest radius as the current possible solution
   - Then select another pair of points and repeat the procedure, until you tried all possible combination of three points.
e) and ultimately you will retain the smallest circle that includes all the points.

The next page provides you with hints on the method needed to create a circle that includes three specific points on its circumference.

Hint: this approach #2 can be described as follows:
For all possible combinations of three points in S , let us call them P1, P2, P3

a) Build a circle C such that:
   - both $P_1(X_1, Y_1)$, $P_2(X_2, Y_2)$, and $P_3(X_3, Y_3)$ are on the circumference of C
   - Such a circle will be specified by a center $C(X_C, Y_C)$ and a radius R_C given by the following equations:

$$\begin{cases} X_C = \dfrac{d}{2a}; & (1) \\ Y_C = \dfrac{-e}{2a}; & (2) \\ (R_C)^2 = \dfrac{d^2+e^2}{4a^2} + \dfrac{f}{a} & (3) \end{cases}$$

$$\begin{cases} a = \begin{vmatrix} x_i & y_i & 1 \\ xj & y_j & 1 \\ x_k & y_k & 1 \end{vmatrix} \quad d = \begin{vmatrix} (x_i)^2+(y_i)^2 & y_i & 1 \\ (x_j)^2+(y_j)^2 & y_j & 1 \\ (x_k)^2+(y_k)^2 & y_k & 1 \end{vmatrix} \\ e = \begin{vmatrix} (x_i)^2+(y_i)^2 & x_i & 1 \\ (x_j)^2+(y_j)^2 & x_j & 1 \\ (x_k)^2+(y_k)^2 & x_k & 1 \end{vmatrix} \quad f = \begin{vmatrix} (x_i)^2+(y_i)^2 & x_i & y_i \\ (x_j)^2+(y_j)^2 & x_j & y_j \\ (x_k)^2+(y_k)^2 & x_k & y_k \end{vmatrix} \end{cases}$$

Recall that the determinant of a 3-by-3 matrix is defined as follows:

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} =$$

$$a_{11} \ (a_{33} \times a_{22} - a_{32} \times a_{23}) - a_{12} \ (a_{33} \times a_{21} - a_{31} \times a_{23}) + a_{13} \times (a_{32} \times a_{21} - a_{31} \times a_{22})$$

b) Then you will check if this circle includes all the other points is S
c) If not, then you should select three other points in S, and repeat the process
d) If the circle does include all the other points in S, therefore this circle is a possible solution to the problem.
e) You will have to compare it to the current solution to the problem.
f) IF the circle you just generated has a smaller radius, then it becomes the “current solution to the problem, you will make the circle you generated as the current solution. Otherwise the current solution will not be modified
g) You will select three other points in S and repeat the procedure until you used all possible combinations of three points from S

IV. The Design of your solutions

Before you jump right in and start coding you must design your algorithm. Given that your input consists of N points in the cartesian plan, you need to determine :

a. The type of data structures your algorithm will use, and their size as a function of N.
b. For each of the two approaches, you will determine
    - o The theoretical asymptotic complexity of the approach
    - o The memory requirements of the approach.

V. Your Task and Input

Your task includes:

a) You will have to design an algorithm for each approach to solve this problem,
b) Then you will have to implement the algorithms in Java or C++.
c) You will be provided with six data sets that you will use to run both approaches.
d) They will be posted on Moodle under the folder relative to this programming assignment.
e) They are respectively :

File name	**Number of points**
Circles_10.txt	10
Circles_20.txt	20
Circles_100.txt	100
Circles_400.txt	400
Circles_800.txt	800
Circles_1000.txt	1000
Circles_2000.txt	2000
Circles_4000.txt	4000
Circles_20000.txt	20000
Circles_40000.txt	40000
Circles_100000.txt	100000
Circles_400000.txt	400000
Circles_800000.txt	800000

f) Each of the file is of the following format:
- An integer n on the first line. It will represent the number of points you will have to read.
- Followed by n lines, where each line consists of a pair of integers *x y* representing the x and y coordinates of a given point.
- As an example, a dataset of 5 points will be of the form:

```
  i. 5            // the number of points in this file
 ii. -120 13            //  point (-120,13)
iii. 14    -170   //  point (14, -170)
 iv. 16 17        //point (16, 17)
  v. 234  36      //point  (234, 36)
 vi. -10  -350    //point  (10  -350)
```

VI. The Required Output.

For each dataset, you will produce two types of outputs:

- The smallest circle enclosing the N points .
- Performance measures in terms execution times of the two approaches.

A. The smallest circle

Your output will use the following format:

Table 1.

		Center coordinates	Center coordinates	Radius	Radius	Surface	Surface
File name	**Number of points**	Approach 1	Approach 2	Approach 1	Approach 2	Approach 1	Approach 2
Circles_10.txt	10						
Circles_20.txt	20						
Circles_100.txt	100						
Circles_400.txt	400						
Circles_800.txt	800						
Circles_1000.txt	1000						
Circles_2000.txt	2000						
Circles_4000.txt	4000						
Circles_20000.txt	20000						
Circles_40000.txt	40000						
Circles_100000.txt	100000						
Circles_400000.txt	400000						
Circles_800000.txt	800000						

B. The performance measures.

For the data sets , your program will compute the execution time (milliseconds) for each approach. The results will be used to produce the following summary table.

Table 2

File name Circles_10.txt	**Number of points** 10	Execution Time			Asymptotic Complexity as a function of n		
		Bruteforce1	Brute force2	Brute force1/bruteforce 2	Bruteforce1	Brute force2	Brute force1/bruteforce 2
Circles_20.txt	20						
Circles_100.txt	100						
Circles_400.txt	400						
Circles_800.txt	800						
Circles_1000.txt	1000						
Circles_2000.txt	2000						
Circles_4000.txt	4000						
Circles_20000.txt	20000						
Circles_40000.txt	40000						
Circles_100000.txt	100000						
Circles_400000.txt	400000						
Circles_800000.txt	800000						

C. Analysis of the results

You are expected to analyse these results of the different runs and provide answers to the following questions

i. Using Table1, you must
   a. Analyse the empirical results you obtained while running the two approaches for the different datasets.
   b. You must discuss whether the accuracy of the results of two approaches are consistent or not.
   c. Explain why or why not.
   d. Answers such as " we see that two approaches provide different results" is not an acceptable answer.

ii. Using Table 2, you must:
   a. Compare the execution times of different datasets within one approach as a function of the input size, and the theoretical complexity that you had predicted in your design. How?
   b. You will analyse how the execution time (within one approach) varies as the number of points vary from 10 to 1,000,000 .
   c. Compare this variation in execution times to the theoretical asymptotic estimation. For example ,
      - if I estimated my algorithm to be Θ (n),then I will expect that the running times of my algorithm for a size 100 and a size 1000 to be close to ratio of 1000/100=10.
      - If the algorithm estimation is Θ (n^2), then I will expect that the running times of my algorithm for a size 100 and a size 100 to be close to ratio of 100^2/ $(1000)^2$=100

iii. Comparisons of execution times across the two approaches.

You will compare the executions of the two approaches for each input size and determine if it is in concordance with the asymptotic estimations.

- First, for each dataset, you will compare the execution times of the two-point approach to that of the three-points approach by computing their ratio.
- You will compare these ratios to the ratios of their asymptotic estimations.
- For example, assume that the two-point approach is Θ (n^5) and the three-point approach is Θ (n^2), then we will compare the ratio of the measured execution times to $n^5/n^2= n^3$

At that point, you are expected to comment on the results of your observations.

Reminders

- Answers such as : we see that the execution times grows as the input size grows or it is obvious that the execution time in the brute force approach 2 is higher than in the brute force 1 will not be accepted and results in a zero for this part of the assignment.
- **This is an individual assignment. BEWARE OF PLAGIARISM. BEWARE OF PLAGIARISM. BEWARE OF PLAGIARISM**

VII. Marking of the assignment:

Task	Weight (100 marks)
Algorithm design	**36%**
The pseudo code of your algorithms,	18%
The data structures you selected to use and their sizes	8%
The expected asymptotic complexity of your algorithms.	10%
Programming of the algorithms	**25%**
The Java / C++/Python code/ results of the runs	20%
Screen shots of your test runs.	5%
Analysis of the results	**24%**
Analysis of the results of Table1	8%
Analysis of the results of Table 2	8%
Comparing the execution times of the two approaches	8%
Code Review	**15%**
Questions and presentations from the lab assistant or the lecturer	

VIII. Submission of the work

Further details about the submission will be posted on Moodle under the folder "First Programming Assignment"

Late submission and Plagiarism

- **Beware of plagiarism.**
- **Late submission is subject to penalties as specified by the subject outline.**