CMPT225, Spring 2021
Homework Assignment 4
Due date: Friday, April 2, 2021, 23:59

You need to implement the following class:
Part 1:

- *graph.DenseGraph.java*

You may add more classes to your solution if necessary.

Submit all your files in ***assignment4.zip*** to CourSys. Make sure your zip file can be unzipped using the command "*unzip assignmen4.zip*" in CSIL. All your solutions in the zip file must be under the **src** folder, same as in the provided *hw4-cmpt225-spring21.zip*.
That is, the zip file needs to contain the following file (and additional files if needed)

- src/*graph/DenseGraph.java*

Discussion with others: You may discuss the assignment with your classmates/tutors (or anyone else), but coding must be entirely your own.

References: You may use textbooks, wiki, stack overflow, geeksforgeeks, etc. If you do, specify the references in comments. Posting questions online asking for solutions (e.g. using chegg.com) is prohibited.

Readability: Your code should be readable using the standard Java conventions. Add comments wherever is necessary. If needed, write helper functions or add classes to improve readability.

Compilation: Your code MUST compile in CSIL using javac.
Make sure that your code compiles without warnings/errors.
If the code does not compile in CSIL the grade on that part will be 0 (zero).
Even if you can't solve a problem completely, make sure it compiles.

The assignment will be graded mostly **automatically**, with some exceptions.

Do not add main() to your solutions. The main() method will be in the test files.

Warnings: Warnings during compilation will reduce points.
More importantly, they indicate that something is probably wrong with the code.

Testing: Test your code. Examples of tests are included. Your code will be tested using the provided tests as well as additional tests. You should create more tests to check your solution.

Good luck!

Dense Graph

The class represents a graph in the adjacency matrix representation model. It has one constructor that gets the number of vertices, and the number of vertices does not change.

You need to implement the following methods:

public DenseGraph(**int** n)
creates an empty graph on n nodes.
the "names" of the vertices are 0,1,..,n-1

public int[][] getAdjacencyMatrix()
returns the adjacency matrix representing the graph

public void addEdge(**int** i, **int** j)
adds the edge (i,j) to the graph

public void removeEdge(**int** i, **int** j)
removes the edge (i,j) from the graph

public boolean areAdjacent(**int** i, **int** j)
checks if (i,j) is an edge in the graph

public int degree(**int** i)
returns the degree of i

public Iterator<Integer> neighboursIterator(**int** i)
Returns an iterator that outputs the neighbors of i in the increasing order
Assumption: the graph is not modified during the use of the iterator

public int numberOfVertices()
Returns the number of vertices in the graph

public int numberOfEdges()
Returns the number of edges in the graph

public int distance(**int** i, **int** j)
Returns the distance between i and j in the graph

public static DenseGraph generateRandomGraph(**int** n, **double** p)
Generates a random graph on n vertices such that each edge appears in the graph with probability p independently of all others.
You may use Math.random() to generate a random number between 0 and 1

Good luck!