# CMPE 343
# Fall 2023
# Programming Homework 1

This assignment is due by 23:55 on Monday 13, November 2023.

You are welcome to ask your HW related questions. You should use only one of these options:

There will be three Q&A Office Hours on the following days:

- CMPE343-HW1-OfficeHour 1: November 1 Wednesday, 07:00-08:00 PM, Zoom ID: https://tedu.zoom.us/j/97784526533
- CMPE343-HW1-OfficeHour 2: November 6 Monday, 07:00-08:00 PM, Zoom ID: https://tedu.zoom.us/j/91551623181
- CMPE343-HW1-OfficeHour 3: November 10 Friday, 07:00-08:00 PM, Zoom ID: https://tedu.zoom.us/j/93969706001

Note: Please make sure that you have read the HW document well before participating. However, no HW related questions will be accepted except from the above options.

## PROGRAMMING TASK

**In this part, you must implement your own graph data structure by taking inspiration from your textbook and use it to help to solve problem. You are not allowed to use any external library or .jar file. Any solutions without using graph data structure are not evaluated!**

**Question 1(25 points):**

You live in a city having *N* stations connected by *M* undirected railway tracks. Each track connects two distinct stations and no two tracks connect the same pair of stations. You can go from any station *A* to any station *B*, if there is a railway track between them. The distance between two stations is the minimum possible number of railway tracks on the path between them. You want to add new tracks in the city. But you do not want to decrease the distance between your home station *X* and TEDU station *Y*. You need to find the number of unordered pairs of two distinct stations that are not connected initially, such that if the new track between these two stations is built, the distance between stations *X* and *Y* won't decrease.

In the input, the first line contains 4 space separated integers, *N*, *M*, *X* and *Y*. *N* denotes the total number of stations, *M* denotes the railway tracks, *X* denotes the home station and *Y* denotes

TEDU station respectively. The next *M* lines represent the two integers *A* and *B*, such that there is a railway track from station *A* to station *B*.

In the sample input, it is given that we have 5 stations and 4 undirected railway tracks between them. And for this case, the home station is 3 and TEDU station is 5. The next lines give the connections between the stations.

**Sample Input:**

```
5 4 3 5
1 2
2 3
3 4
4 5
```

In the output, first line print an integer *K*, denoting the number of stations you add to the railway tracks. And the next *K* lines give the new railway tracks which do not decrease the distance between the stations X and Y denote the home station and TEDU station. If there is not any possible new tracks, your program should print -1.

The output for the above input is as follows. Please check your program with this input as well as the others that you will create. Please note that we may use other input when grading your assignments.

**Sample Output:**

```
5
1 3
1 4
1 5
2 5
2 4
```

**Question 2(25 points):**

You want to visit *M* museums of the city you live by following the *N* undirected roads connect these museums. However, you have a limited time to visit all the museums so you want to visit them in minimum time. You can pick any museum as the starting point of your journey. Let's say your current position is C. Then you can move to any adjacent museum of C, but it costs your time equal to the weight of the museum. You will continue the above moves until you visit all the museums. Calculate the minimum time to visit all the museums in the city when you can choose any museum as your starting point.

In the input, the first line contains 2 space separated integers, *M* and *N*. *M* denotes the number of museums in the city and *N* denotes the number of roads connect these museums. Next N lines contains two space separated integers each, *U* and *V* denoting that there is an undirected road between museum *U* and museum *V*.

In the sample input, it is given that we have 4 museums and 3 undirected roads between them. The next lines give the connections between the museums.

**Sample Input:**

```
4 3
1 2 1
2 3 2
2 4 3
```

In the output, first line print an integer *T*, denoting the minimum time to visit all museums in the city. And the next line gives the track you need to follow to visit all museums. If you cannot visit all the museums in the city by starting any point, you need to print -1.

The output for the above input is as follows. Please check your program with this input as well as the others that you will create. Please note that we may use other input when grading your assignments.

**Sample Output:**

```
7
4 2 1 2 3
```

## WHAT TO HAND IN

- **You need to upload your code into VPL on LMS for each question.** If you do not upload your code into VPL on LMS, your homework will **not be graded.**
- The Java sources should be WELL DOCUMENTED as comments, as part of your grade will be based on the level of your comments.
- You need to upload **maximum-3 pages** PDF report document that explains your own answers for programming task in a clearly readable PA report format (refer to **PA REPORT FORMAT** section).

## PA REPORT FORMAT

A programming assignment report is a self-description of a programming assignment and your solution. The report must not be hand-written. You may use a word processor or the on-line editor of your choice and prepare as a PDF document. The report must be grammatically correct and use complete English sentences. Each report should include the following sections, in the order given:

**Information (%2.5)**: This section includes your ID, name, section, assignment number information properly.

**Problem Statement and Code Design (%15)**: Include a brief summary of the problem and/or your sub-tasks to be completed in this assignment. You should show your modular design rationale by creating a structure chart that indicates your top-down, stepwise refinement of the problem solution. You may create the structure chart using available graphical tools like MS PowerPoint, SmartDraw etc.

**Implementation and Functionality (%20)**: Since you have modular source code, you should describe each sub-module (program) in this section. Each sub-module should include names and types of any input/output parameters as well as the pseudocode algorithm that used for completing its task. By this way, you give meaning to each chart boxes from the previous section.

**Testing (%7.5)**: You should provide a tester class that is able to identify key test points of your program. This class should be able to generate additional (apart from the given sample input/output) test data for the purpose of being clear on what aspects of the solution are being tested with each set. This section should also include a description of any program *bugs* that is, tests which has incorrect results. You should write these to describe your tests, summarize your results, and argue that they cover all types of program behavior.

**Final Assessments (%5)**: In this final section, you should briefly answer the following questions:

- What were the trouble points in completing this assignment?
- Which parts were the most challenging for you?
- What did you like about the assignment? What did you learn from it?

## GRADING:

- Codes ( %50: %25 for Q1 and %25 for Q2)
  - Available test cases evaluation on VPL: %15
  - Hidden test cases evaluation: %15
  - Approach to the problem: %20
- Report ( %50: %25 for Q1 and %25 for Q2)
  - Information: %2.5
  - Problem Statement and Code design: %15
  - Implementation, Functionality: %20
  - Testing: %7.5
  - Final Assessments: %5

## IMPORTANT

IMPORTANT NOTES: Do not start your homework before reading these notes!!!

1. This assignment is due by 23:55 on Monday, November 13th.

2. You should upload your homework to LMS before the deadline. No hardcopy submission is needed. You should upload your codes into VPL and your report into submission place on LMS.

3. The standard rules about late homework submissions apply (20 points will be deducted for each late day). Please see the course syllabus for further discussion of the late homework policy as well as academic integrity.

4. You ARE NOT ALLOWED to modify the given method names. However, if necessary, you may define additional data members and member functions.

5. Your classes' name MUST BE as shown in the homework description.

6. The submissions that do not obey these rules will not be graded.

7. To increase the efficiency of the grading process as well as the readability of your code, you have to follow the following instructions about the format and general layout of your program.

8. Do not forget to write down your id, name, section, assignment number or any other information relevant to your program in the beginning of your Java files. Example:

```
//-------------------------------------------------
// Title: Scheduler tester class
// Author: Name/Surname
// ID: 2100000000
```

```
// Section: 1
// Assignment: 1
// Description: This class tests the …
//-------------------------------------------------------
```

9. Since your codes will be checked without your observation, you should report everything about your implementation. Add detailed comments to your classes, functions, declarations etc. Make sure that you explain each function in the beginning of your function structure. Example:

```
void setVariable(char varName, int varValue)
//-----------------------------------------------------------
// Summary: Assigns a value to the variable whose
// name is given.
// Precondition: varName is a char and varValue is an
// integer
// Postcondition: The value of the variable is set.
//-----------------------------------------------------------
{
      // Body of the function
}
```

10. Indentation, indentation, indentation...

11. This homework will be graded by your TAs, Bedrettin Çetinkaya, Deniz Merve Gündüz. Thus, you may ask them your homework related questions through HW forum on LMS course page. You are also welcome to ask your course instructors Ulaş Güleç for help.