COSC2430: Programming and Data Structures

Graph based clustering

Introduction

In this homework, you will create a C++ program that receives in input a weighted graph and groups its nodes in the required number of clusters.

Input and Output

The input is a single text file containing:

- The graph, in adjacent list format
- The weight matrix
- The required number of clusters

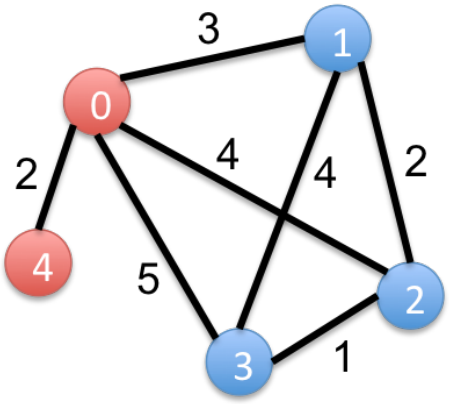
Example 1 of input

```
1 2 3 4
0 2 3
0 1 3
0 1 2
0

0 3 4 5 2
3 0 2 4 -999
4 2 0 1 -999
5 4 1 0 -999
2 -999 -999 -999 0

2
```

-999 represents infinite distance (no connection between two nodes). This input corresponds to the following weighted, undirected graph



The output of your program should be a text file containing the clusters, represented as list of nodes on different rows. As a convention, nodes should appear in numerical order. Clusters should be sorted by numerical order of their first node.

Example 1 of output

0 4
1 2 3

Handling invalid input

Your program should be able to recognize invalid input of the following nature:

- Mismatch between size of adjacency list and weight matrix. You can assume that the adjacency list and the weight matrix will always be present in the input file and contain only numbers, but it is possible that their sizes do not match.
- Request for an invalid number n of clusters (n<1, n>number of nodes in the graph or n missing)

In both cases, the output file should be left empty.

The main C++ program will become the executable to be tested by the TAs. The result should be written on another text file (output file), provided on the command line. The input and output files are specified in the command line, not inside the C++ code.

The general call to the executable (sum_rowcol, in this example) is as follows:

```
cluster "A=<file>;C=<file>"
```

Call example with one input file and another output file.

```
cluster "A=a.txt;C=c.out"
```

Requirements

- **It is NOT allowed to use vector classes or other classes provided in the STL.**
- Your C++ code must be clear, indented and commented.
- Your program will be tested with GNU C++. Therefore, you are encouraged to work on Unix, or at least make sure that your code compiles and runs successfully on the server.
- You can use other C++ compilers, but the TAs cannot provide support or test your programs with other compilers.

Testing for grading:

- Your main cpp program will be automatically compiled. If there is an error in compilation it will not be executed. **Programs that do not compile on the server will receive a score of 0.** You are responsible of the content of your submission folder.
- **The name of your submission folder must be hw3**. The folder must not be nested in another folder.
- Submitted files must be limited to headers and source files (.h and .cpp).
- Your program should not crash, halt unexpectedly, take an unusual amount of time to run (more than 10 seconds) or produce unhandled exceptions.
- Your program will be tested with 10 test cases, going from easy to difficult.

DEADLINE: April 30th, 11:59PM – NO LATE SUBMISSIONS

The due date cannot be changed. No exceptions, unless there are medical or exceptional reasons.

Grading

- A program not submitted by the deadline is zero points.
- A program that compiles but does not work program is worth 10 points. .
- A code with no comments will receive a 5 points penalty.

Plagiarism and cheating: C++ code is individual: it cannot be shared (other than small fragments used in class or in lab examples). Programs will be compared for plagiarism. If the TAs detect that a portion of your C++ code is highly similar to C++ on the Internet or other student the grade will be decreased at least 50%.