Problem Set 2, Programming Part
Minimum Spanning Trees

Due: 11:55pm Thursday, October 25, 2018

---

1 Programming Assignment

Theresa May just watched Shaun of the Dead and is worried that a zombie revolt is imminent. Should this happen, a natural fear is that the zombies will organize clandestine meetings, pool their collective minimal brain power, and ultimately push for "Zexit" (Zombie Exit), causing economic chaos in the UK.

Therefore, May creates the Ministry of Zombie Affairs, to be headed by Simon Pegg. Simon decides that the best way to prevent zombie meetings is by scrapping London's current road network and replacing it with a network with the property that any two places are connected by precisely one path. This way, zombie drivers easily can be stopped using minimal blockades. Unfortunately, nearly all the available funds were spent on billboards warning of Zexit, and so Simon needs to construct a road network whose total length is minimal in order to minimize construction costs. More formally, the problem is described by the following Input and Output.

Input: A weighted graph G of n vertices. Each edge weight corresponds to the cost of constructing a road between two places in London.

Output: The cost (sum of the edge weights) of the minimum-cost solution.

A Java template has been provided containing an empty function `mst`. This function takes an adjacency list `adj` (stored as a three-dimensional integer array) which represents a weighted graph G; the structure of `adj` is explained in the file `MST.java`. The function `mst` returns the sum of the weights of the edges in a minimum spanning tree of G. Your task is to write the body of the `mst` function, which can include calls to helper functions that you write. Your code is not required to check for incorrect inputs or incorrectly-formed input data.

You must use the provided Java template as the basis of your submission. You may not change the name, return type, or parameters of the `mst` function. The `main` function in the template contains code to help you test your implementation by entering test data or reading it from a file. A sample file is also provided (see the comments in `MST.java` for the file format). You may modify the `main` function or any other function, because your submission will be tested using a different `main` function. Only the contents of the `mst` function and associated helper functions (if any) will be marked.

2 Evaluation Criteria

The programming assignment will be marked out of 40, based on a combination of automated testing (using large graphs) and human inspection.

You are to implement Kruskal's algorithm, equipped with the Weighted Quick-Union version of Union-Find (which you also are to implement). If implemented correctly, the running time of your code should be $O(|E| \log |E|)$. The marks for each submission will be based on both the asymptotic worst case running time and the ability of the algorithm to handle inputs of different sizes. The table below shows the expectations associated with different scores.

Score	Description
0 - 15	Submission does not compile or does not conform to the provided template.
16 - 30	The implemented algorithm is not $O(\|E\| \log \|E\|)$ or is substantially inaccurate on the tested inputs.
31 - 40	The implemented algorithm is $O(\|E\| \log \|E\|)$ and gives the correct answer on all tested inputs.

To be properly tested, every submission must compile correctly as submitted and must be based on the provided template. If your submission does not compile for any reason (including even trivial mistakes like typos) or was not based on the template, it will receive at most 15 out of 40. The best way to ensure your submission is correct is to download it from conneX after submitting and test it.

You are not permitted to revise your submission after the due date, and late submissions will not be accepted, so you should ensure that you have submitted the correct version of your code before the due date. conneX will allow you to change your submission before the due date if you notice a mistake. After submitting your assignment, conneX will automatically send you a confirmation email. If you do not receive such an email, your submission was not received. If you have problems with the submission process, send an email to the instructor before the due date.