CS 2040: Assignment 3

***Up to two days o*verdue submissions:** 20% penalty per day

---

Rewrite function dijkstras_algorithm (see attached files) to use a priority queue.

```
#include <queue>
// Declare the priority queue (as an example)
priority_queue<Edge, vector<Edge>, Compare_Edges> pQ;
```

When inserting edges into the priority queue, the weight is replaced by the total distance from the source vertex to the destination vertex. The source vertex, however, remains unchanged, because it is the predecessor in the shortest path. Test your program with at least 3 different graphs. You may use the provided test files. Compare the performance and results of the program with and without the use of a priority queue. Be sure to highlight the differences in terms of efficiency and the accuracy of the results.

Note: Some source files, which may or may not be complete, are attached as a starting point. You may complete the assignment by using these files.

Marking Scheme:

- **Working program [50%]** A working program which satisfies all of the requirements automatically receives 50% of the total assignment mark. Each element of non-compliance will be penalized with respect to its severity.
- **Program Structure [25%]** A program which follows principles of Object-Oriented Design and structured programming rules (procedural, modular, uses parameters) to perfection automatically receives 25% of the total assignment mark. Marks are deducted depending on severity and number of occurrences of non-compliant elements.
- **Internal documentation [15%]:** Documentation should be complete and in a standard format. Every non trivial part of the code should have a *clear* comment that explains it. In addition, every method or function, including the main program should have an explicative comment header. This header includes: module name, author, date of creation and purpose. A description of parameters and method output is mandatory. Marks are deducted according to the absence of these elements.
- **Program Style [10%]** Style refers to Occam's razor principle. Code that is needlessly tricky, obscure, or difficult to read will be marked accordingly. Program text indentation is also an element of style and must be present. Significant constant, variable and structure names must be used. Marks are deducted on the basis of the frequency of these errors.

- Your submission for each programming question includes: (a) the source code files (.CPP and .h files) (b) output the results to a PDF (either automatically by C++ libraries or manually by copying from terminal), and (c) **screenshots** of testing runs (**all in one PDF**).