COS 212—Algorithms and Complexity
Run Dijkstra's shortest path algorithm on a graph.
Practical 2 Term 4 12 September 2016

Due before 23h59 on 18 September 2016.

Implement Dijkstra's algorithm to find the shortest path to all other nodes in an edge weighted digraph with non-negative weights.

---

1. In your home directory, inside your `surname,firstname` directory ensure that you do not already have a directory called `24practical`. If you do have one rename it to `24practical-old` before doing anything else. Next, copy one of your old `ddpractical` directories to a new directory called `24practical` and work inside it.

2. Your data must consist of triplets of the form *from-vertex to-vertex edge-weight*, conforming to `.dot` notation augmented to take a weight on each edge. The input data from last week's practical must be extended to have a weight on each edge. Each weight is a positive integer in brackets, preceded by `label = <weight>`. The weight appears after the definition of each edge and before its terminating semicolon. This example is available in the notes file in `Dijkstra.dot`. Note that all whitespace in the `.dot` file is optional.

```
digraph Dijkstra {
  start -> A [label=9];
  start -> B [label=14];
  start -> C [label=15];
  A -> E [label=24];
  B -> E [label=18];
  B -> D [label = 30];
  B -> C [ label=5];
  D -> F [label = 11];
  F -> E [label = 6];
  E -> D [label = 2];
  C -> D [label = 20];
  C -> goal [label = 44];
  D -> goal[label=16];
  E -> goal[label=19];}
```

   A pictorial output in pdf is made from this data using the `dot` visualization program, e.g.

```
dot -Tpdf Dijkstra.dot > Dijkstra.pdf
```

3. The program must be given the `start` vertex. Use the input which may vary to build a graph represented with adjacency lists in your program. See Section 4.4 of Sedgewick and Wayne for hints. You are free to use their APIs.
