

CMSC 132 Take Home Quiz 2 (20 pts)

Due Date: Monday Nov 14, 5:00 pm (No late submissions accepted)

Overview

For this take home quiz you will practice Dijkstra's algorithm, DFS and BFS. This homework should not take long and should be completed before you start working on Project #6. Project #6 requires you to understand the above algorithms so it is best if you finish the quiz first.

This quiz will be considered a "take home quiz" and you should treat the quiz as an open homework as described in:

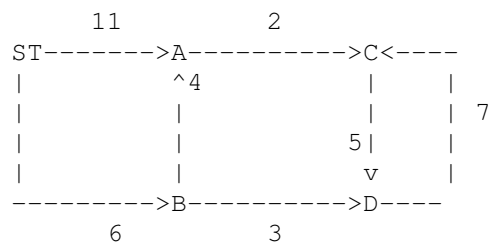
<http://www.cs.umd.edu/class/fall2005/cmsc132/openClosedPolicy.shtml>

We have left at the end of this document an example of how to run Dijkstra's algorithm using ST as the start vertex. When running Dijkstra's, follow the same steps as illustrated by the example. You will get 0 credit if you just write the result graph (the one with costs and predecessors) without the intermediate results.

What you need to do

1. *Running the algorithms*

Using the following graph (which is the same one used in the example below):



- Run DFS with ST as the start vertex and write the traversal. Writing only the traversal is fine.
- Run DFS with A as the start vertex and write the traversal. Writing only the traversal is fine.
- Run BFS with ST as the start vertex and write the traversal. Writing only the traversal is fine.
- Run BFS with A as the start vertex and write the traversal. Writing only the traversal is fine.
- Run Dijkstra's algorithm with A as the start vertex
- Run Dijkstra's algorithm with B as the start vertex
- Run Dijkstra's algorithm with C as the start vertex

2. *Submission*

Create a document named **hw2.txt** or **hw2.doc** with your solutions. Upload your document using the submit server. The name of the quiz is in the submit server is TakeHomeTwo.

Example:

The following example shows how we can run Dijkstra's algorithm over a graph. The start vertex is ST.

% → stands for infinity
 - → stands for no predecessor
 (#) → represents the order the vertices are being processed.
 [x,y] → x represents the cost of reaching the node and y the predecessor.

```

[0,-]  11 [%,-]    2 [%,-]
ST----->A----->C<-----
|           ^4           |       | 7
|           |           |       |
|           |           5|       |
|           |           v       |
----->B----->D-----
          6 [%,-]    3 [%,-]
  
```

```

[0,-]  11 [11,ST]  2 [%,-]
(1)ST----->A----->C<-----
|           ^4           |       | 7
|           |           |       |
|           |           5|       |
|           |           v       |
----->B----->D-----
          6 [6,ST]   3 [%,-]
  
```

```

[0,-]  11 [10,B]   2 [%,-]
(1)ST----->A----->C<-----
|           ^4           |       | 7
|           |           |       |
|           |           5|       |
|           (2)|         v       |
----->B----->D-----
          6 [6,ST]   3 [9,B]
  
```

```

[0,-]  11 [10,B]   2 [16,D]
(1)ST----->A----->C<-----
|           ^4           |       | 7
|           |           |       |
|           |           5|       |
|           (2)|         (3)v     |
----->B----->D-----
          6 [6,ST]   3 [9,B]
  
```

```

[0,-] 11 [10,B]      2 [12,A]
(1) ST----->A----->C<-----
|           (4)^4           |       | 7
|           |               |       |
|           |               5|       |
|           (2)|           (3)v       |
----->B----->D-----
          6 [6,ST]      3 [9,B]

```

```

[0,-] 11 [10,B]      2 [12,A]
(1) ST----->A----->C<-----
|           (4)^4           (5)|       | 7
|           |               |       |
|           |               5|       |
|           (2)|           (3)v       |
----->B----->D-----
          6 [6,ST]      3 [9,B]

```