Spencer Whitehead
1001837401

Program should be able to be run through powershell by simply typing “py <path to dvrouting.py> <path to network info file>”. All of the output is cleanly printed out to the console, and will start with a prompt of whether the user would like to run the program without intervention or not. The program will then read in the file and assign the dv’s to each router, print out the initial dv table, and print out the actions of each node for each step of the iteration. At the start of each iteration once the dv table is printed out, if the user is running with intervention enabled they will have the option to change any link on the table as they please before continuing. Once the dv table has reached a stable state it will notify the user, and if they are running without intervention it will print out the time to reach the current state in milliseconds, otherwise it will give the user the option to continue if they would like to continue changing links.

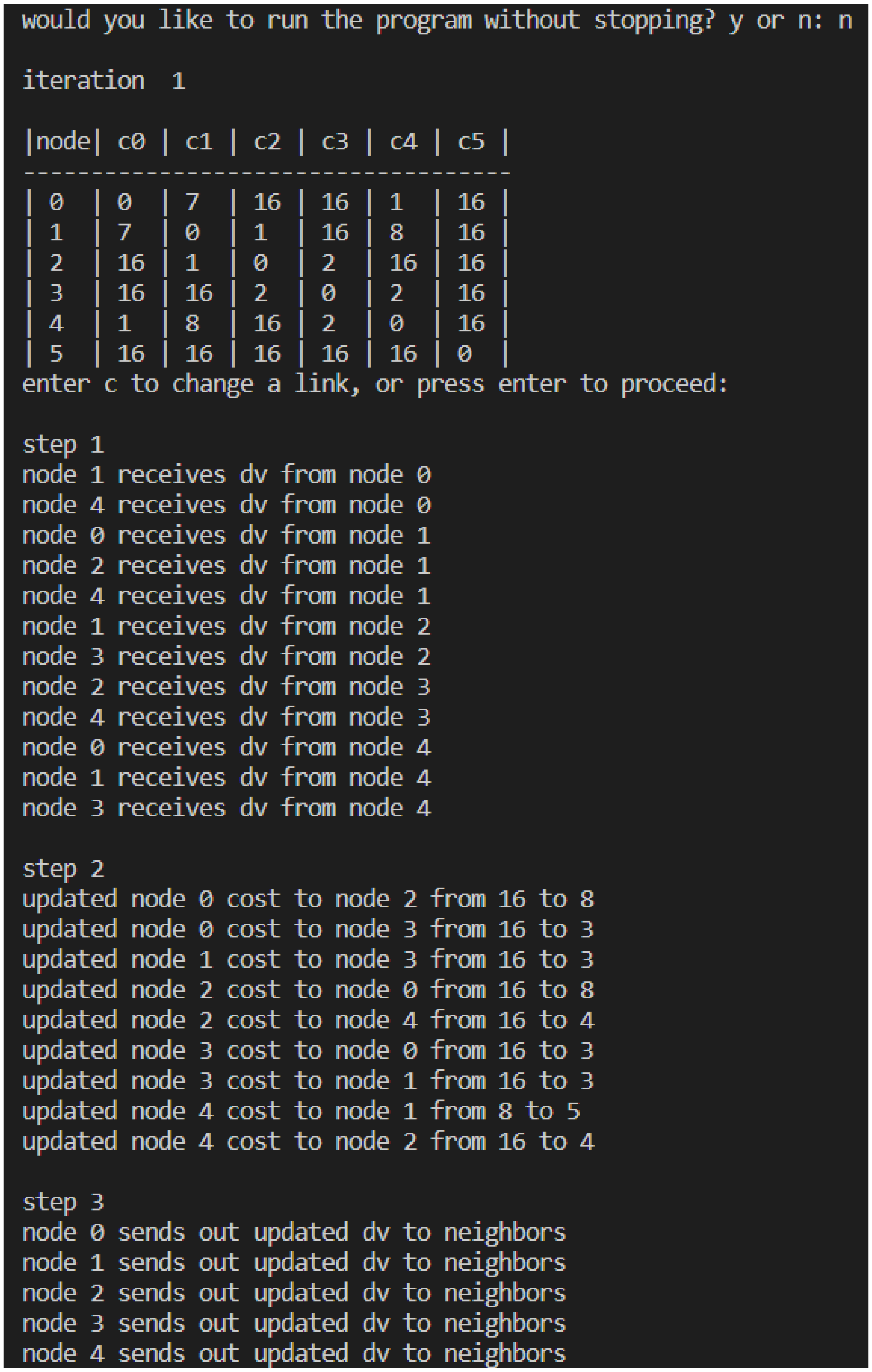
An important thing to note with the program is the names of the nodes have been changed from 1-6 to 0-5. This is because the nodes share their info using a shared buffer, and calculations are made much easier by being able to access a node's info in the shared buffer by simply using their assigned number as their index.

Another assumption made is that when the user establishes a link between two nodes, they become neighbors. This is because establishing them as neighbors is the only way it would reasonably make sense for the cost between two nodes to not be equal to the cost of traveling to other nodes to reach it. The same applies for setting the cost between two nodes to 16, since that is equivalent to infinity it will sever a direct connection between the two nodes if it exists.

One other thing of note is that the dv table does not dynamically adjust to the number of active nodes, and will always show six nodes. This means if not all nodes are active, some of them will show all their costs (except for themselves) as 16. However, this allows the user to see the number of nodes available for if they choose to change the links, and take one of these inactive nodes and make them active.

The following is a series of screenshots demonstrating the program, reading from a file containing the example lines from the project description:
With intervention:

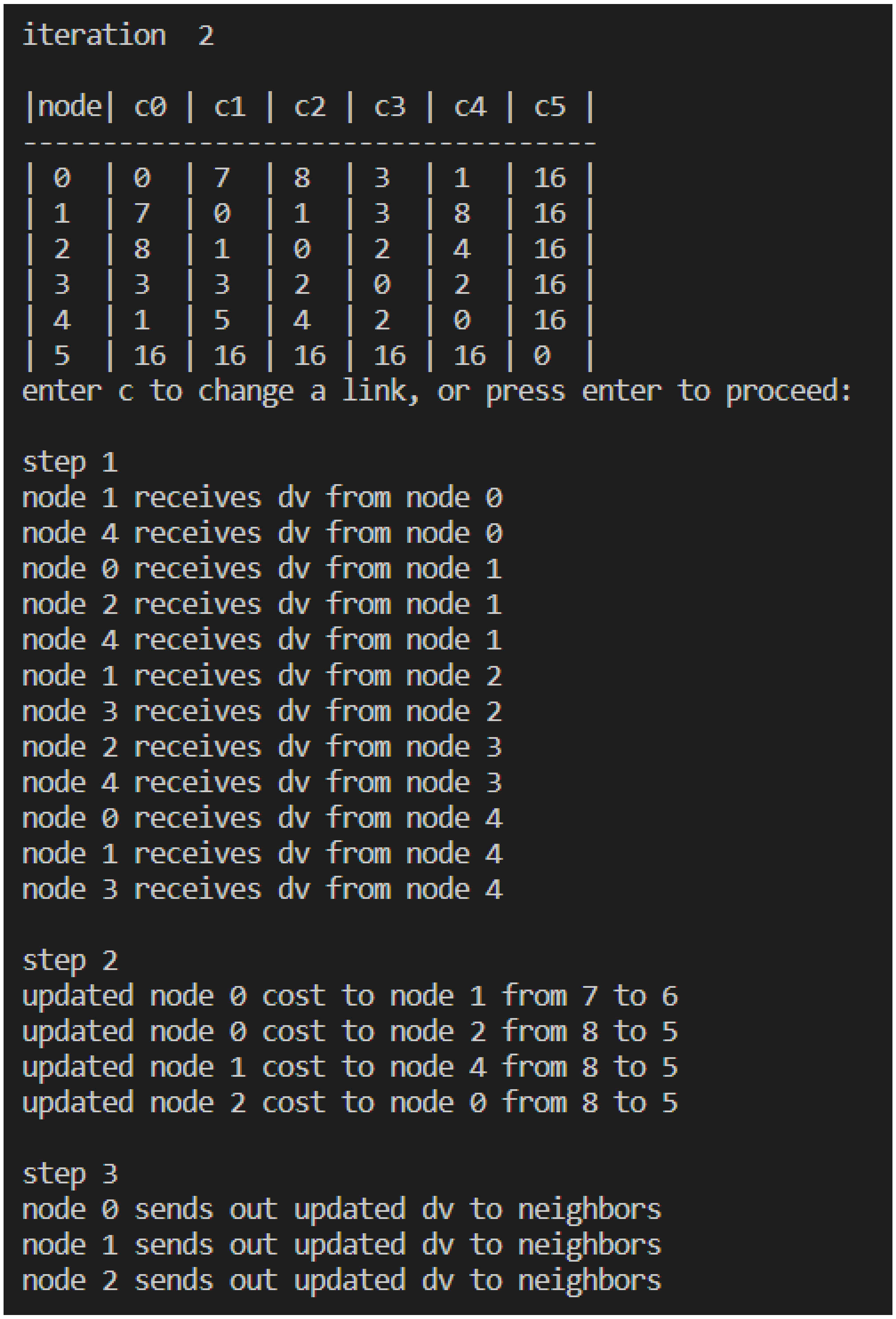
Iteration 1:



```
would you like to run the program without stopping? y or n: n

iteration  1

|node| c0 | c1 | c2 | c3 | c4 | c5 |
------------------------------------
| 0  | 0  | 7  | 16 | 16 | 1  | 16 |
| 1  | 7  | 0  | 1  | 16 | 8  | 16 |
| 2  | 16 | 1  | 0  | 2  | 16 | 16 |
| 3  | 16 | 16 | 2  | 0  | 2  | 16 |
| 4  | 1  | 8  | 16 | 2  | 0  | 16 |
| 5  | 16 | 16 | 16 | 16 | 16 | 0  |
enter c to change a link, or press enter to proceed:

step 1
node 1 receives dv from node 0
node 4 receives dv from node 0
node 0 receives dv from node 1
node 2 receives dv from node 1
node 4 receives dv from node 1
node 1 receives dv from node 2
node 3 receives dv from node 2
node 2 receives dv from node 3
node 4 receives dv from node 3
node 0 receives dv from node 4
node 1 receives dv from node 4
node 3 receives dv from node 4

step 2
updated node 0 cost to node 2 from 16 to 8
updated node 0 cost to node 3 from 16 to 3
updated node 1 cost to node 3 from 16 to 3
updated node 2 cost to node 0 from 16 to 8
updated node 2 cost to node 4 from 16 to 4
updated node 3 cost to node 0 from 16 to 3
updated node 3 cost to node 1 from 16 to 3
updated node 4 cost to node 1 from 8 to 5
updated node 4 cost to node 2 from 16 to 4

step 3
node 0 sends out updated dv to neighbors
node 1 sends out updated dv to neighbors
node 2 sends out updated dv to neighbors
node 3 sends out updated dv to neighbors
node 4 sends out updated dv to neighbors
```

Since all nodes were just initialized, each node sends its dv to all its neighbors

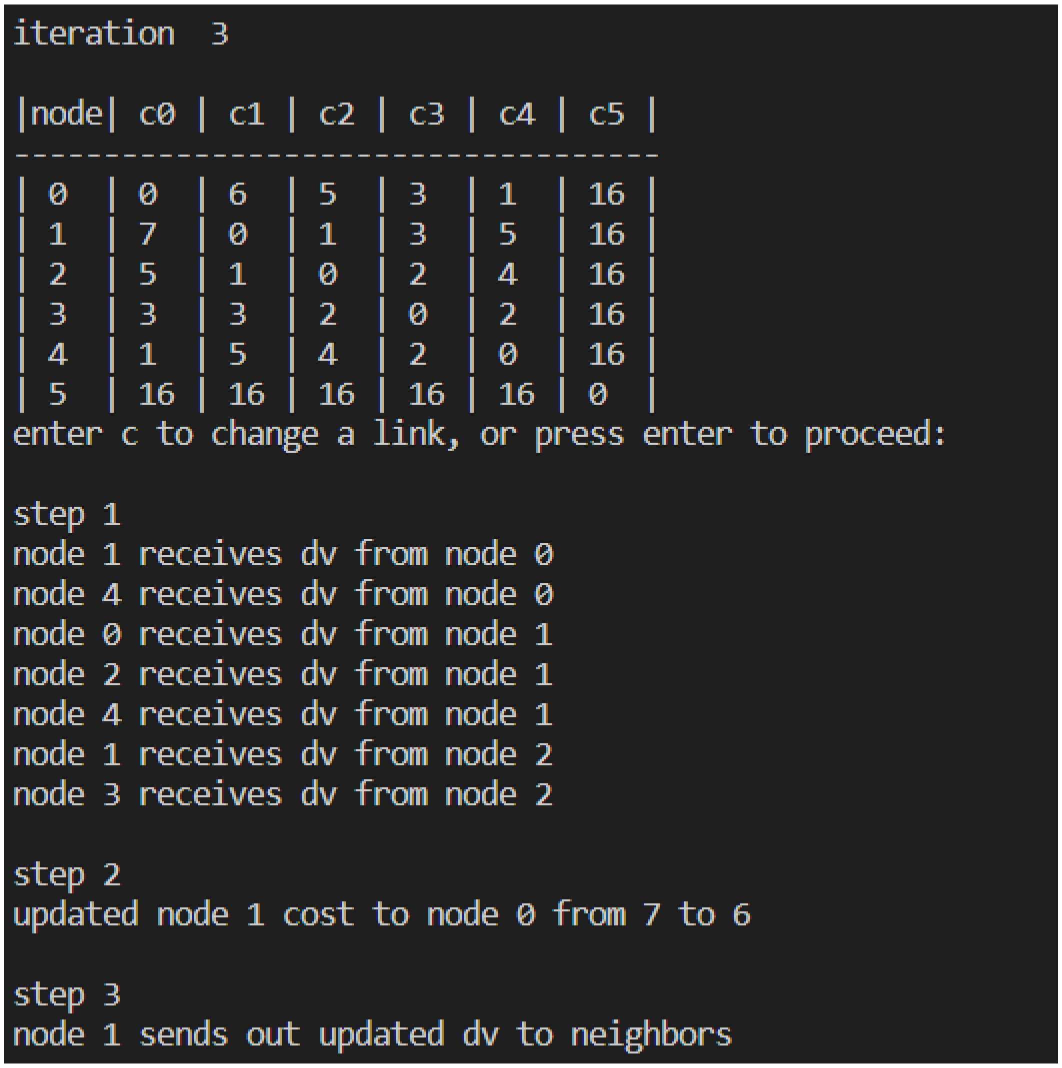
Iteration 2:



```
iteration  2

|node| c0 | c1 | c2 | c3 | c4 | c5 |
------------------------------------
| 0  | 0  | 7  | 8  | 3  | 1  | 16 |
| 1  | 7  | 0  | 1  | 3  | 8  | 16 |
| 2  | 8  | 1  | 0  | 2  | 4  | 16 |
| 3  | 3  | 3  | 2  | 0  | 2  | 16 |
| 4  | 1  | 5  | 4  | 2  | 0  | 16 |
| 5  | 16 | 16 | 16 | 16 | 16 | 0  |
enter c to change a link, or press enter to proceed:

step 1
node 1 receives dv from node 0
node 4 receives dv from node 0
node 0 receives dv from node 1
node 2 receives dv from node 1
node 4 receives dv from node 1
node 1 receives dv from node 2
node 3 receives dv from node 2
node 2 receives dv from node 3
node 4 receives dv from node 3
node 0 receives dv from node 4
node 1 receives dv from node 4
node 3 receives dv from node 4

step 2
updated node 0 cost to node 1 from 7 to 6
updated node 0 cost to node 2 from 8 to 5
updated node 1 cost to node 4 from 8 to 5
updated node 2 cost to node 0 from 8 to 5

step 3
node 0 sends out updated dv to neighbors
node 1 sends out updated dv to neighbors
node 2 sends out updated dv to neighbors
```

Every node in the previous iteration changed one of their values and sent it out to their neighbors, and they all are received in this iteration.

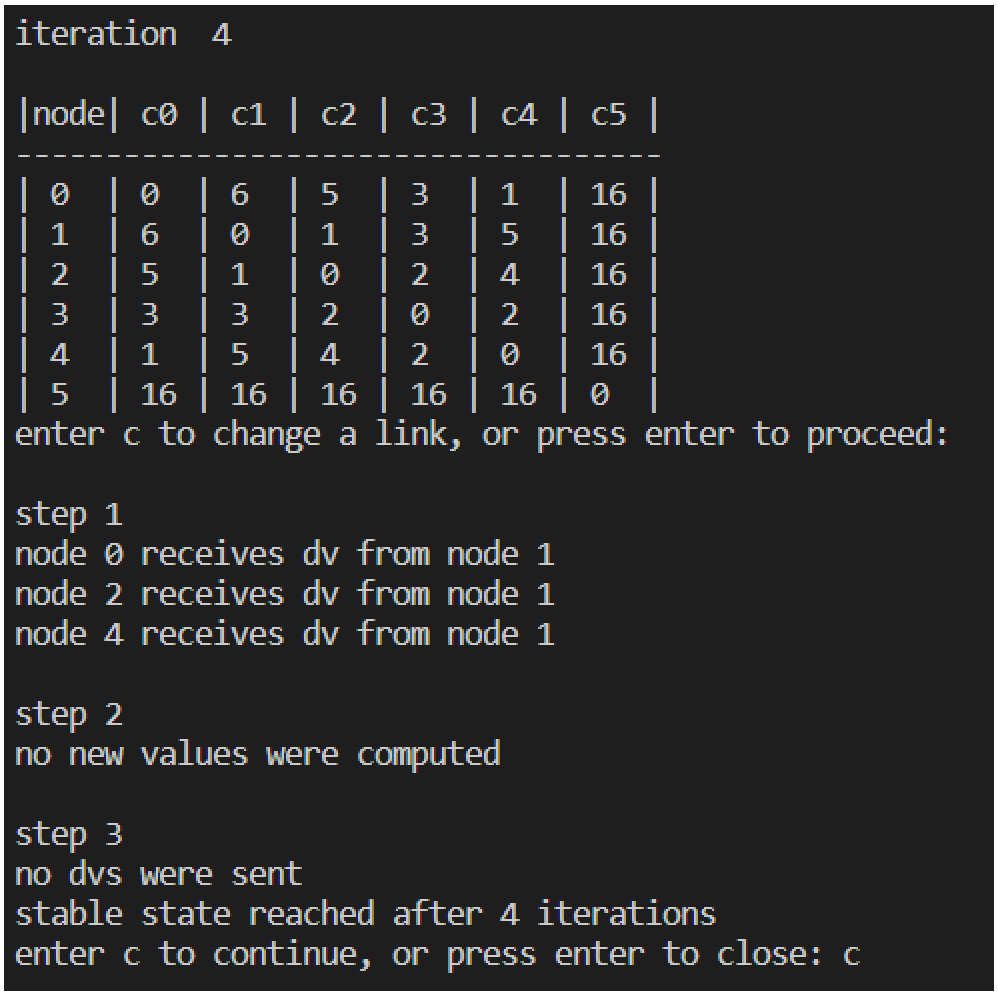
Iteration 3:



```
iteration  3

|node| c0 | c1 | c2 | c3 | c4 | c5 |
-------------------------------------
| 0  | 0  | 6  | 5  | 3  | 1  | 16 |
| 1  | 7  | 0  | 1  | 3  | 5  | 16 |
| 2  | 5  | 1  | 0  | 2  | 4  | 16 |
| 3  | 3  | 3  | 2  | 0  | 2  | 16 |
| 4  | 1  | 5  | 4  | 2  | 0  | 16 |
| 5  | 16 | 16 | 16 | 16 | 16 | 0  |
enter c to change a link, or press enter to proceed:

step 1
node 1 receives dv from node 0
node 4 receives dv from node 0
node 0 receives dv from node 1
node 2 receives dv from node 1
node 4 receives dv from node 1
node 1 receives dv from node 2
node 3 receives dv from node 2

step 2
updated node 1 cost to node 0 from 7 to 6

step 3
node 1 sends out updated dv to neighbors
```

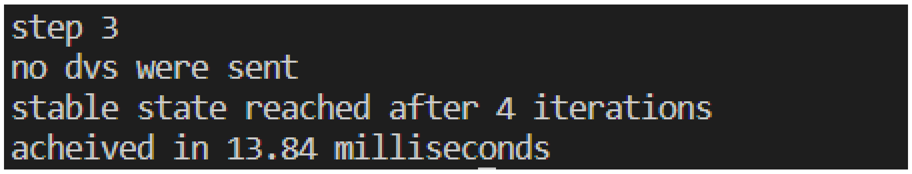
Only one node updated this iteration, so the dv table is close to a stable state.

Iteration 4:



```
iteration  4

|node| c0 | c1 | c2 | c3 | c4 | c5 |
------------------------------------
| 0  | 0  | 6  | 5  | 3  | 1  | 16 |
| 1  | 6  | 0  | 1  | 3  | 5  | 16 |
| 2  | 5  | 1  | 0  | 2  | 4  | 16 |
| 3  | 3  | 3  | 2  | 0  | 2  | 16 |
| 4  | 1  | 5  | 4  | 2  | 0  | 16 |
| 5  | 16 | 16 | 16 | 16 | 16 | 0  |
enter c to change a link, or press enter to proceed:

step 1
node 0 receives dv from node 1
node 2 receives dv from node 1
node 4 receives dv from node 1

step 2
no new values were computed

step 3
no dvs were sent
stable state reached after 4 iterations
enter c to continue, or press enter to close: c
```

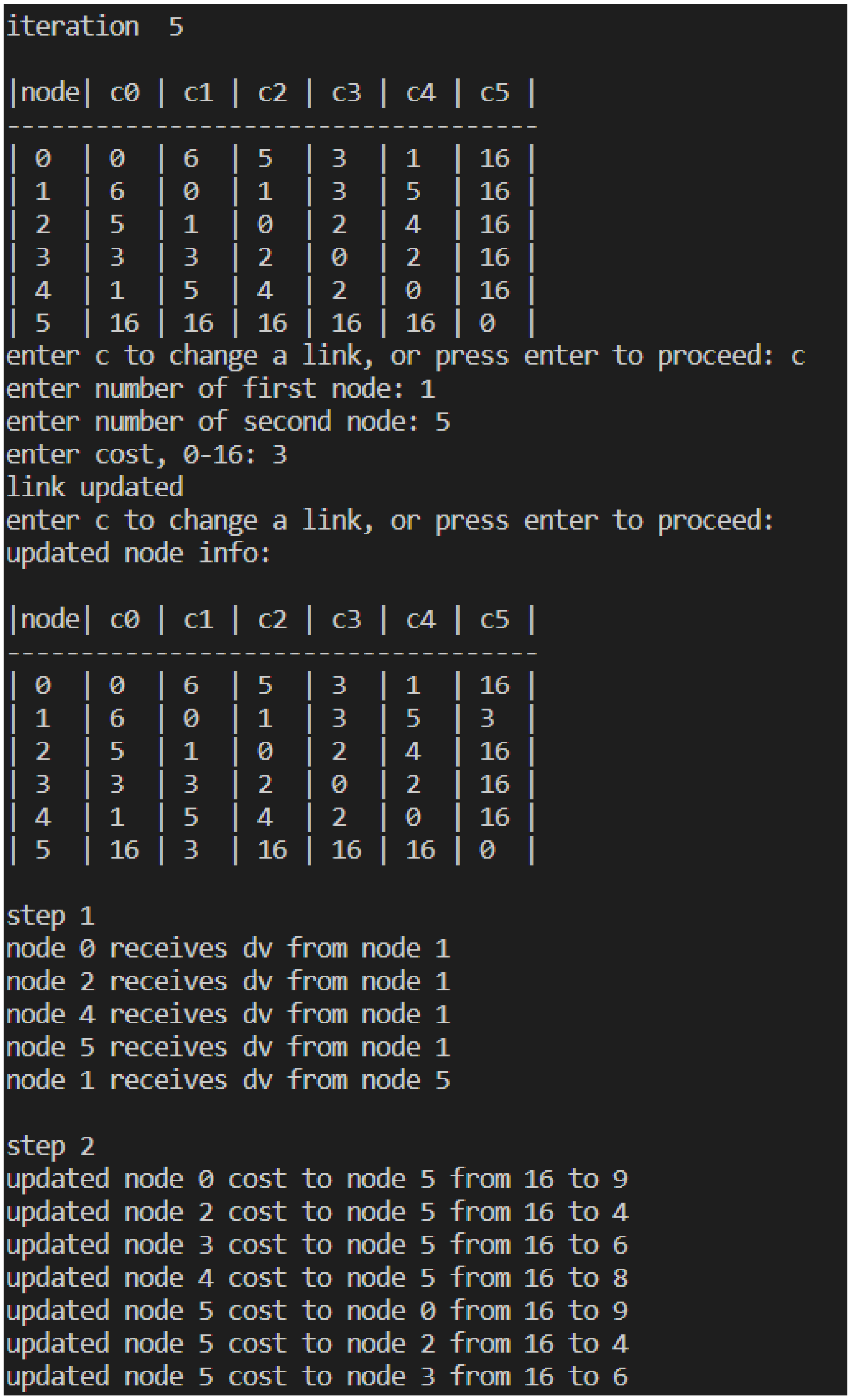
End of iteration 4 without stopping enabled:



```
step 3
no dvs were sent
stable state reached after 4 iterations
acheived in 13.84 milliseconds
```

A stable state is reached this iteration, but the process will be continued to add a connection to the sixth node.

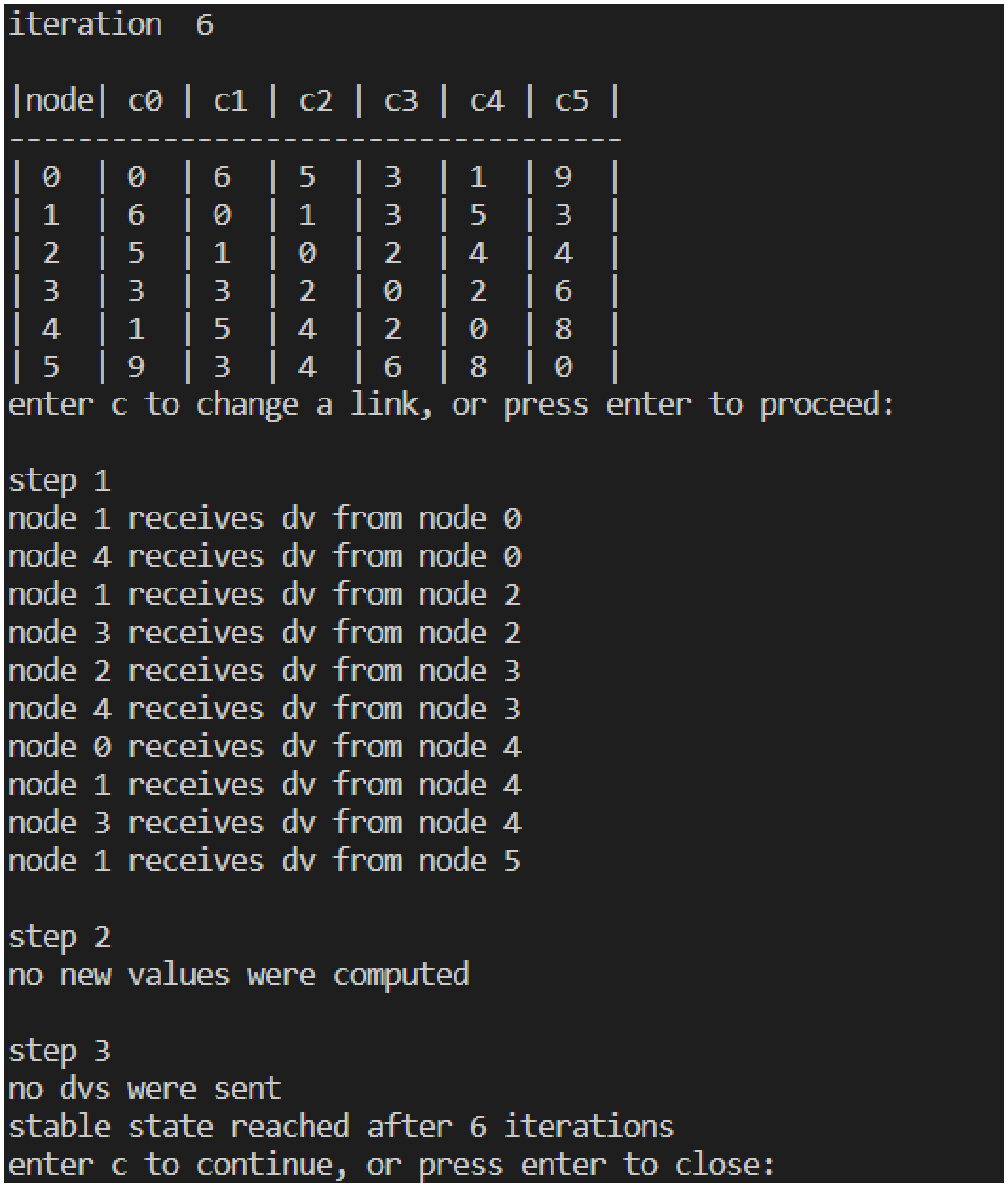
Iteration 5, with adding connection equivalent to file line “2 6 3”:



```
iteration  5

|node| c0 | c1 | c2 | c3 | c4 | c5 |
------------------------------------
| 0  | 0  | 6  | 5  | 3  | 1  | 16 |
| 1  | 6  | 0  | 1  | 3  | 5  | 16 |
| 2  | 5  | 1  | 0  | 2  | 4  | 16 |
| 3  | 3  | 3  | 2  | 0  | 2  | 16 |
| 4  | 1  | 5  | 4  | 2  | 0  | 16 |
| 5  | 16 | 16 | 16 | 16 | 16 | 0  |
enter c to change a link, or press enter to proceed: c
enter number of first node: 1
enter number of second node: 5
enter cost, 0-16: 3
link updated
enter c to change a link, or press enter to proceed:
updated node info:

|node| c0 | c1 | c2 | c3 | c4 | c5 |
------------------------------------
| 0  | 0  | 6  | 5  | 3  | 1  | 16 |
| 1  | 6  | 0  | 1  | 3  | 5  | 3  |
| 2  | 5  | 1  | 0  | 2  | 4  | 16 |
| 3  | 3  | 3  | 2  | 0  | 2  | 16 |
| 4  | 1  | 5  | 4  | 2  | 0  | 16 |
| 5  | 16 | 3  | 16 | 16 | 16 | 0  |

step 1
node 0 receives dv from node 1
node 2 receives dv from node 1
node 4 receives dv from node 1
node 5 receives dv from node 1
node 1 receives dv from node 5

step 2
updated node 0 cost to node 5 from 16 to 9
updated node 2 cost to node 5 from 16 to 4
updated node 3 cost to node 5 from 16 to 6
updated node 4 cost to node 5 from 16 to 8
updated node 5 cost to node 0 from 16 to 9
updated node 5 cost to node 2 from 16 to 4
updated node 5 cost to node 3 from 16 to 6
```

Each node now updates their values to change their cost to the sixth node.

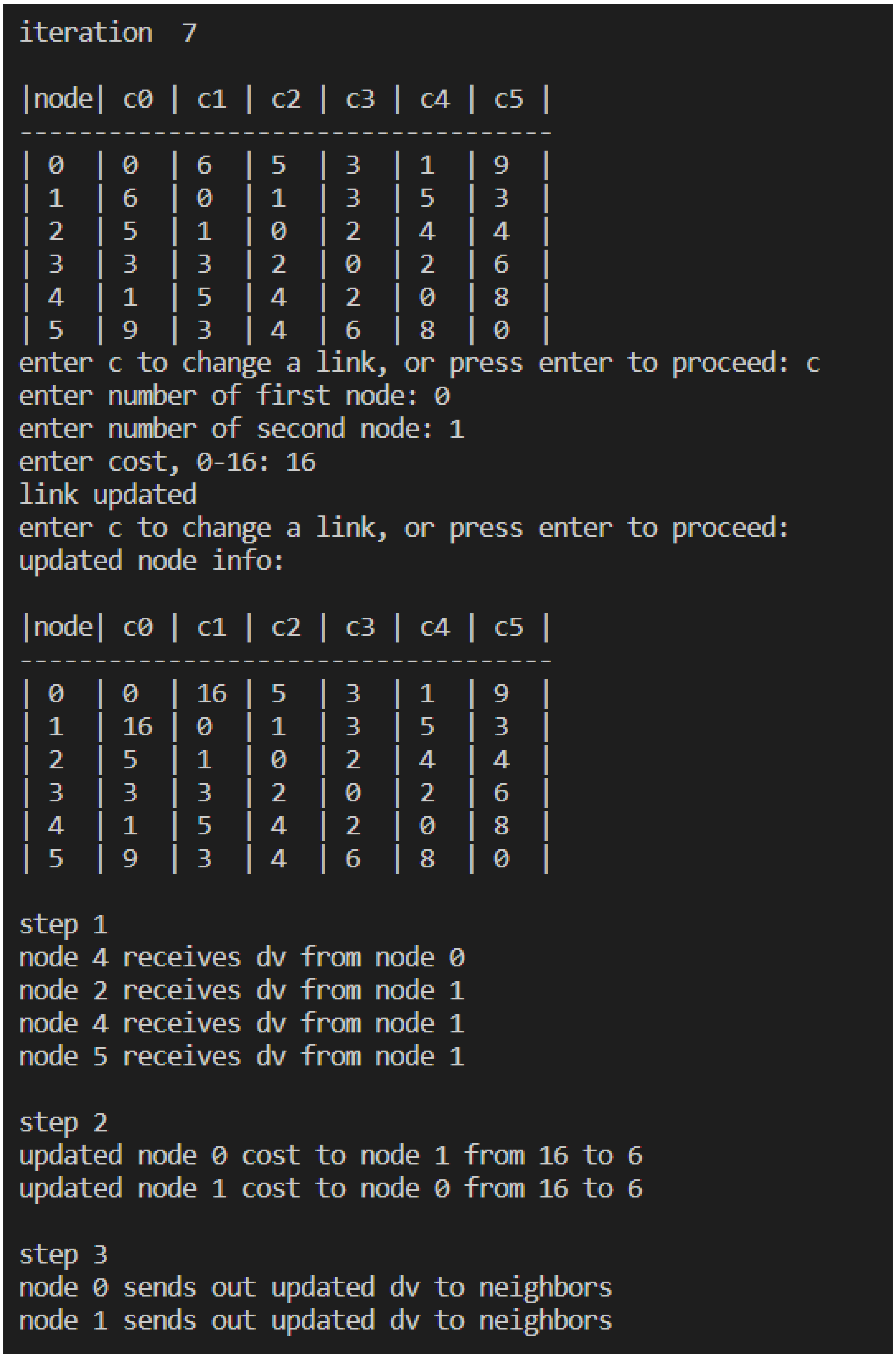
Iteration 6:



```
iteration  6

|node| c0 | c1 | c2 | c3 | c4 | c5 |
------------------------------------
| 0  | 0  | 6  | 5  | 3  | 1  | 9  |
| 1  | 6  | 0  | 1  | 3  | 5  | 3  |
| 2  | 5  | 1  | 0  | 2  | 4  | 4  |
| 3  | 3  | 3  | 2  | 0  | 2  | 6  |
| 4  | 1  | 5  | 4  | 2  | 0  | 8  |
| 5  | 9  | 3  | 4  | 6  | 8  | 0  |
enter c to change a link, or press enter to proceed:

step 1
node 1 receives dv from node 0
node 4 receives dv from node 0
node 1 receives dv from node 2
node 3 receives dv from node 2
node 2 receives dv from node 3
node 4 receives dv from node 3
node 0 receives dv from node 4
node 1 receives dv from node 4
node 3 receives dv from node 4
node 1 receives dv from node 5

step 2
no new values were computed

step 3
no dvs were sent
stable state reached after 6 iterations
enter c to continue, or press enter to close:
```

A stable state is reached with the sixth node, and the process will continue now to simulate a line failure.

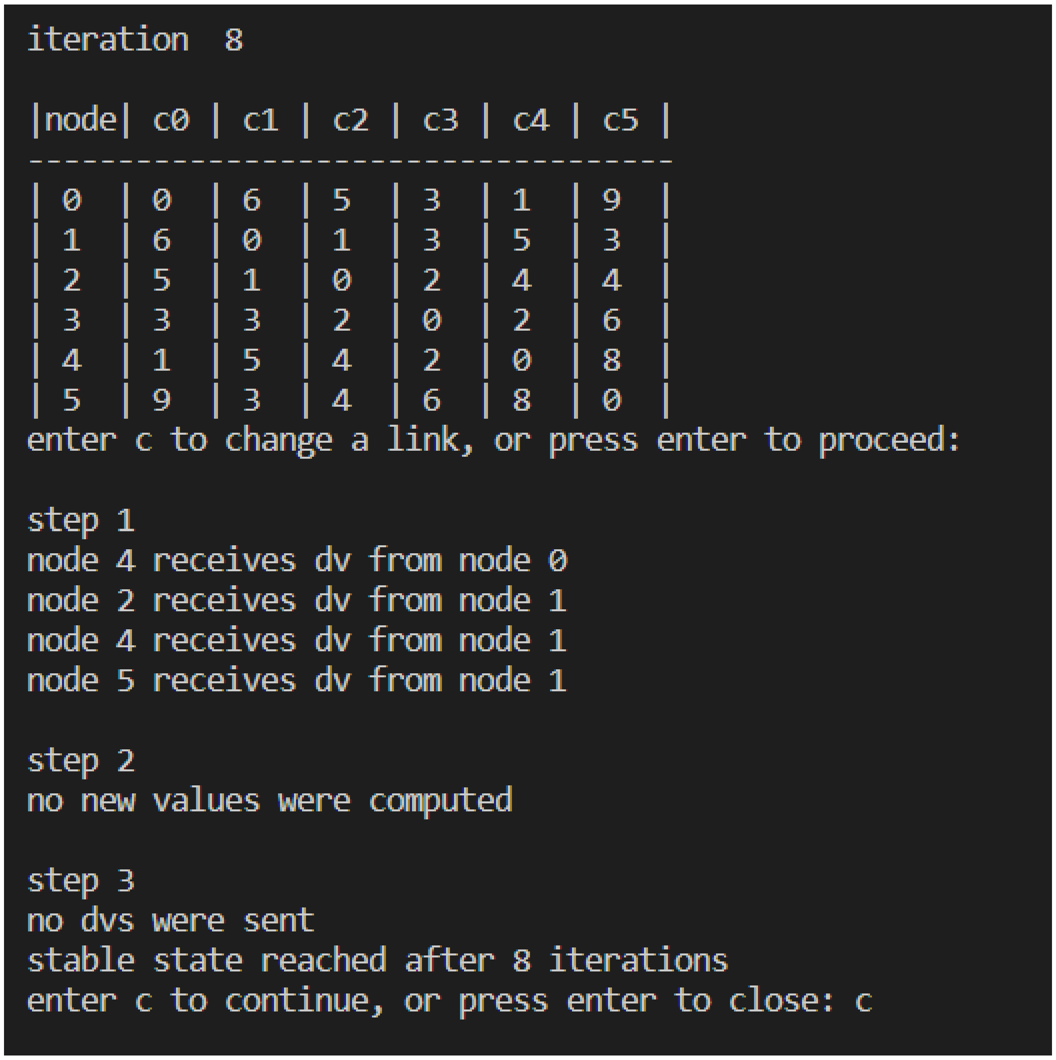
Iteration 7, after severing connection from file “1 2 7” and simulating line failure:



```
iteration  7

|node| c0 | c1 | c2 | c3 | c4 | c5 |
-------------------------------------
| 0  | 0  | 6  | 5  | 3  | 1  | 9  |
| 1  | 6  | 0  | 1  | 3  | 5  | 3  |
| 2  | 5  | 1  | 0  | 2  | 4  | 4  |
| 3  | 3  | 3  | 2  | 0  | 2  | 6  |
| 4  | 1  | 5  | 4  | 2  | 0  | 8  |
| 5  | 9  | 3  | 4  | 6  | 8  | 0  |
enter c to change a link, or press enter to proceed: c
enter number of first node: 0
enter number of second node: 1
enter cost, 0-16: 16
link updated
enter c to change a link, or press enter to proceed:
updated node info:

|node| c0 | c1 | c2 | c3 | c4 | c5 |
-------------------------------------
| 0  | 0  | 16 | 5  | 3  | 1  | 9  |
| 1  | 16 | 0  | 1  | 3  | 5  | 3  |
| 2  | 5  | 1  | 0  | 2  | 4  | 4  |
| 3  | 3  | 3  | 2  | 0  | 2  | 6  |
| 4  | 1  | 5  | 4  | 2  | 0  | 8  |
| 5  | 9  | 3  | 4  | 6  | 8  | 0  |

step 1
node 4 receives dv from node 0
node 2 receives dv from node 1
node 4 receives dv from node 1
node 5 receives dv from node 1

step 2
updated node 0 cost to node 1 from 16 to 6
updated node 1 cost to node 0 from 16 to 6

step 3
node 0 sends out updated dv to neighbors
node 1 sends out updated dv to neighbors
```

Line failure is simulated and new values are calculated for the connection between the nodes.

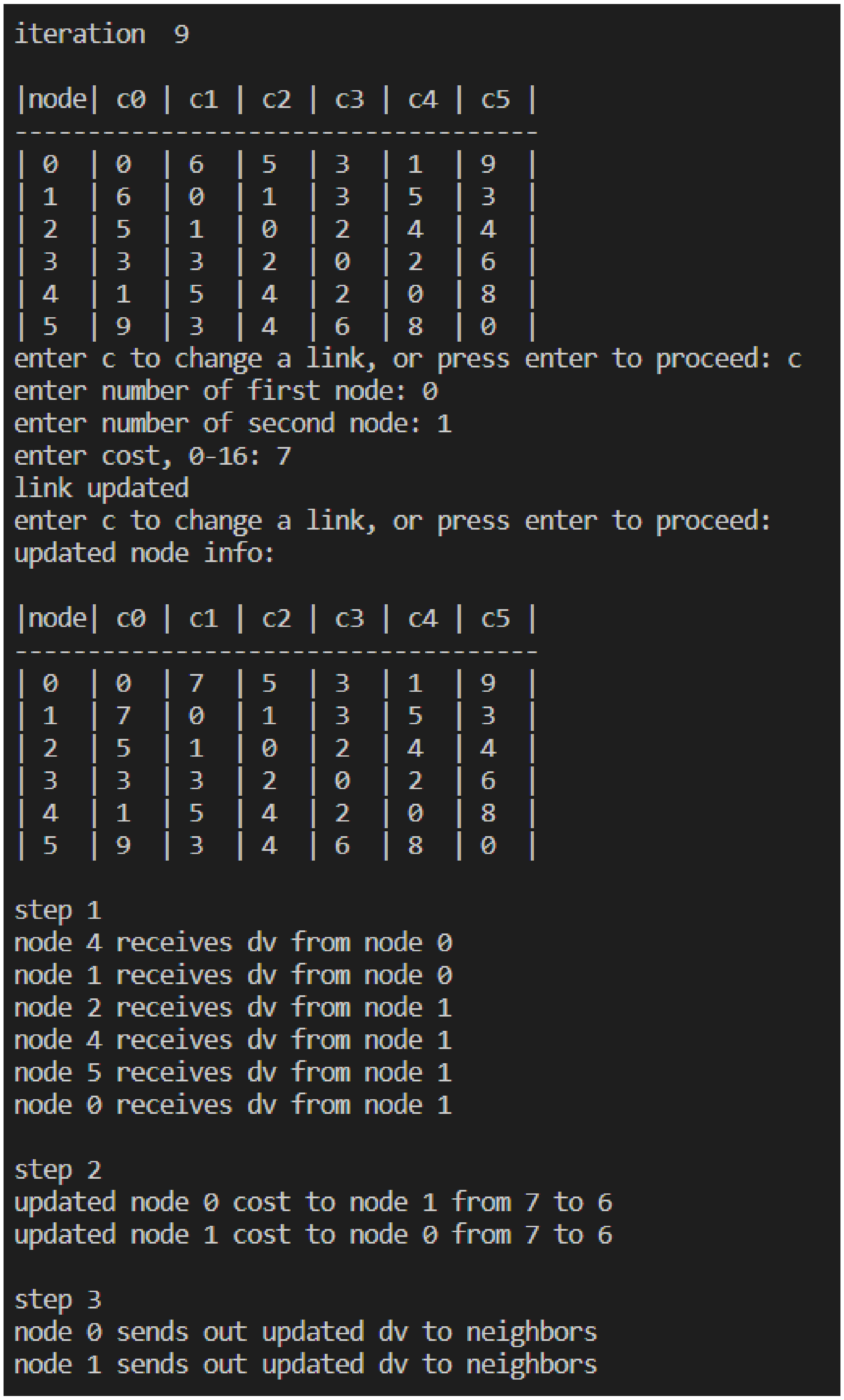
Iteration 8:



```
iteration  8

|node| c0 | c1 | c2 | c3 | c4 | c5 |
------------------------------------
| 0  | 0  | 6  | 5  | 3  | 1  | 9  |
| 1  | 6  | 0  | 1  | 3  | 5  | 3  |
| 2  | 5  | 1  | 0  | 2  | 4  | 4  |
| 3  | 3  | 3  | 2  | 0  | 2  | 6  |
| 4  | 1  | 5  | 4  | 2  | 0  | 8  |
| 5  | 9  | 3  | 4  | 6  | 8  | 0  |
enter c to change a link, or press enter to proceed:

step 1
node 4 receives dv from node 0
node 2 receives dv from node 1
node 4 receives dv from node 1
node 5 receives dv from node 1

step 2
no new values were computed

step 3
no dvs were sent
stable state reached after 8 iterations
enter c to continue, or press enter to close: c
```

Stable state reached, now a line repair will be simulated.

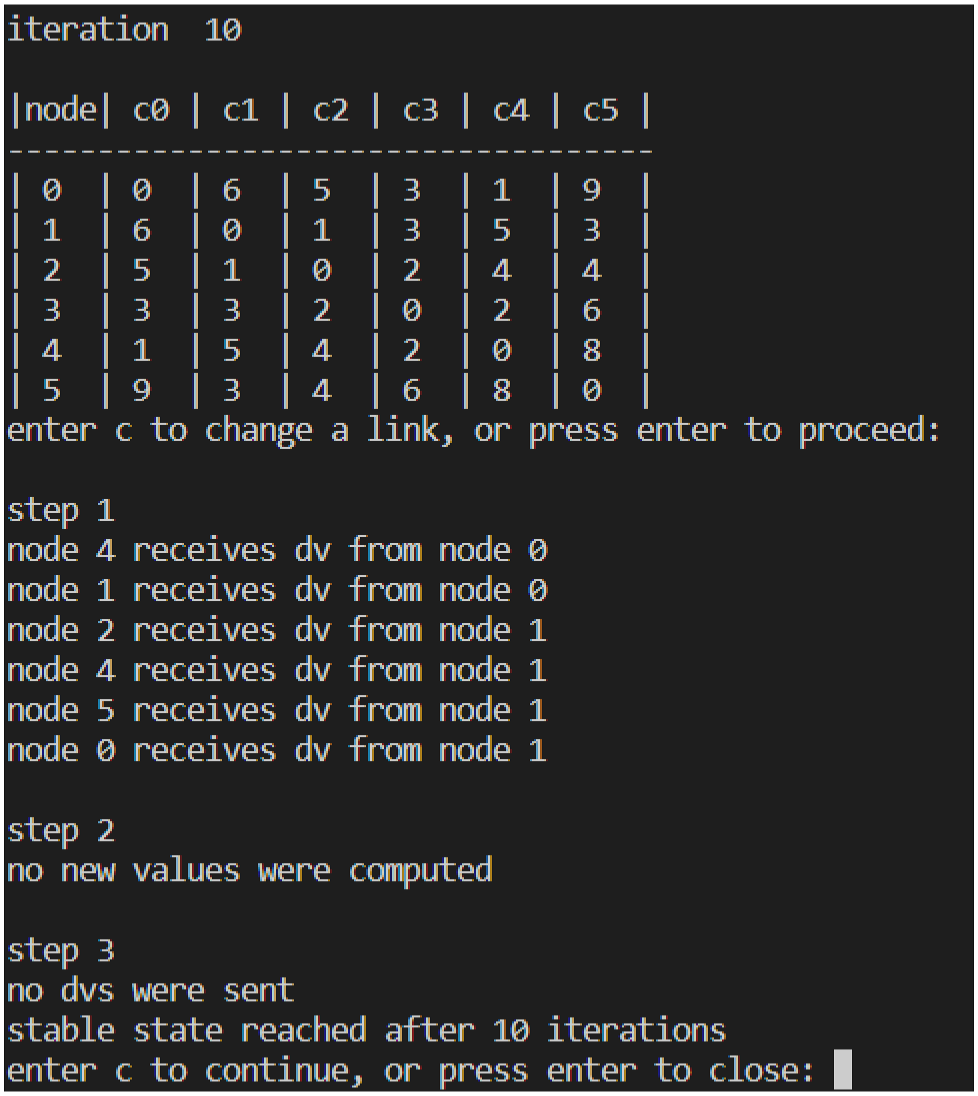
Iteration 9, after repairing line failure:



```
iteration  9

|node| c0 | c1 | c2 | c3 | c4 | c5 |
------------------------------------
| 0  | 0  | 6  | 5  | 3  | 1  | 9  |
| 1  | 6  | 0  | 1  | 3  | 5  | 3  |
| 2  | 5  | 1  | 0  | 2  | 4  | 4  |
| 3  | 3  | 3  | 2  | 0  | 2  | 6  |
| 4  | 1  | 5  | 4  | 2  | 0  | 8  |
| 5  | 9  | 3  | 4  | 6  | 8  | 0  |
enter c to change a link, or press enter to proceed: c
enter number of first node: 0
enter number of second node: 1
enter cost, 0-16: 7
link updated
enter c to change a link, or press enter to proceed:
updated node info:

|node| c0 | c1 | c2 | c3 | c4 | c5 |
------------------------------------
| 0  | 0  | 7  | 5  | 3  | 1  | 9  |
| 1  | 7  | 0  | 1  | 3  | 5  | 3  |
| 2  | 5  | 1  | 0  | 2  | 4  | 4  |
| 3  | 3  | 3  | 2  | 0  | 2  | 6  |
| 4  | 1  | 5  | 4  | 2  | 0  | 8  |
| 5  | 9  | 3  | 4  | 6  | 8  | 0  |

step 1
node 4 receives dv from node 0
node 1 receives dv from node 0
node 2 receives dv from node 1
node 4 receives dv from node 1
node 5 receives dv from node 1
node 0 receives dv from node 1

step 2
updated node 0 cost to node 1 from 7 to 6
updated node 1 cost to node 0 from 7 to 6

step 3
node 0 sends out updated dv to neighbors
node 1 sends out updated dv to neighbors
```

Line repair simulated by adding back in connection that was severed.

Iteration 10:



```
iteration  10

|node| c0 | c1 | c2 | c3 | c4 | c5 |
------------------------------------
| 0  | 0  | 6  | 5  | 3  | 1  | 9  |
| 1  | 6  | 0  | 1  | 3  | 5  | 3  |
| 2  | 5  | 1  | 0  | 2  | 4  | 4  |
| 3  | 3  | 3  | 2  | 0  | 2  | 6  |
| 4  | 1  | 5  | 4  | 2  | 0  | 8  |
| 5  | 9  | 3  | 4  | 6  | 8  | 0  |
enter c to change a link, or press enter to proceed:

step 1
node 4 receives dv from node 0
node 1 receives dv from node 0
node 2 receives dv from node 1
node 4 receives dv from node 1
node 5 receives dv from node 1
node 0 receives dv from node 1

step 2
no new values were computed

step 3
no dvs were sent
stable state reached after 10 iterations
enter c to continue, or press enter to close:
```

Stable state reached with line repair, dv table now in state it originally was before line failure.