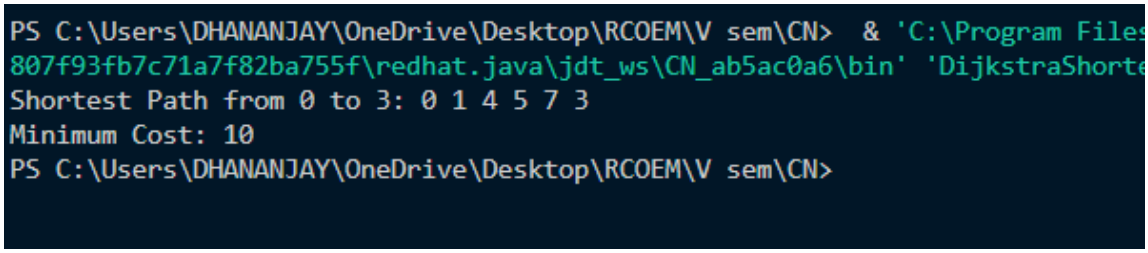
Code

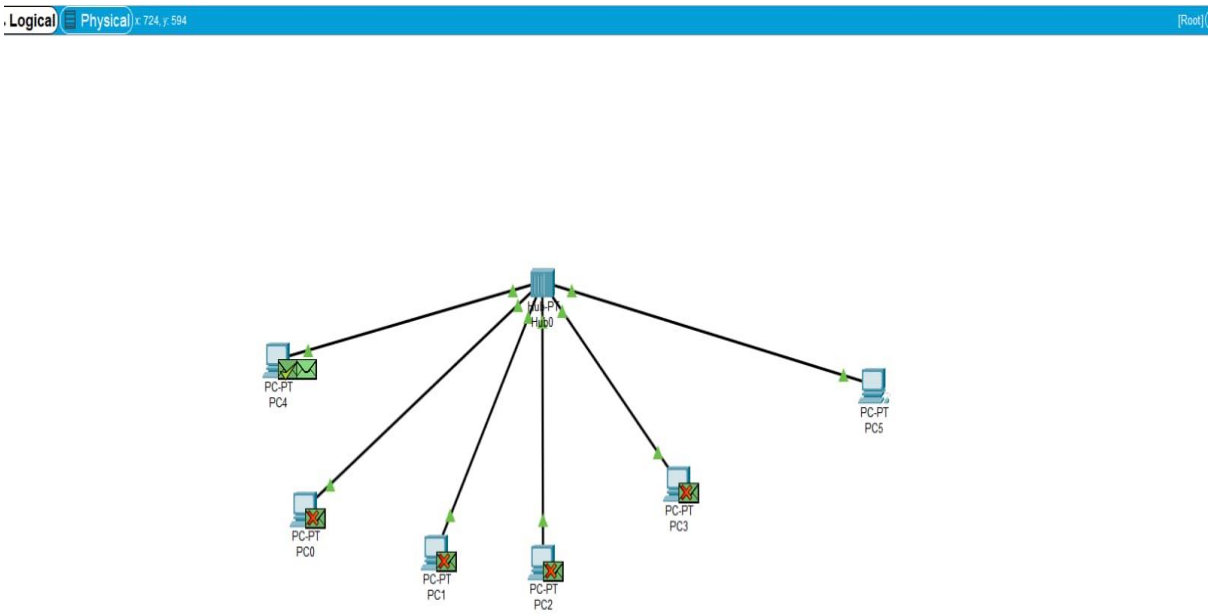
```
public class DijkstraShortestPath {
   private static final int MAX_NODES = 8;
   private static final int INFINITY = 1000000000;

   private static int n = 8;
   private static int[][] graph = {
         {0, 2, INFINITY, INFINITY, INFINITY, INFINITY, 6, INFINITY},
         {2, 0, 7, INFINITY, 2, INFINITY, INFINITY, INFINITY},
         {INFINITY, 7, 0, 3, INFINITY, 3, INFINITY, INFINITY},
         {INFINITY, INFINITY, 3, 0, INFINITY, INFINITY, INFINITY, 2},
         {INFINITY, 2, INFINITY, INFINITY, 0, 2, 1, INFINITY},
         {INFINITY, INFINITY, 3, INFINITY, 2, 0, INFINITY, 4},
         {6, INFINITY, INFINITY, INFINITY, 1, INFINITY, 0, 4},
         {INFINITY, INFINITY, INFINITY, 2, INFINITY, 2, 4, 0}
   };

   public static void main(String[] args) {
      int source = 0;
      int destination = 3;
      int[] path = new int[MAX_NODES];
      int[] mincost = new int[1];

      shortestPath(source, destination, path, mincost);

      System.out.print("Shortest Path from " + source + " to " + destination + ": ");
      for (int i = 0; i < n; i++) {
         System.out.print(path[i] + " ");
         if(path[i]==destination){
            break;
         }
      }

      System.out.println("\nMinimum Cost: " + mincost[0]);
   }

   private static void shortestPath(int s, int t, int[] path, int[] mincost) {
      class State {
         int predecessor;
         int length;
         boolean label; // true for permanent, false for tentative
```

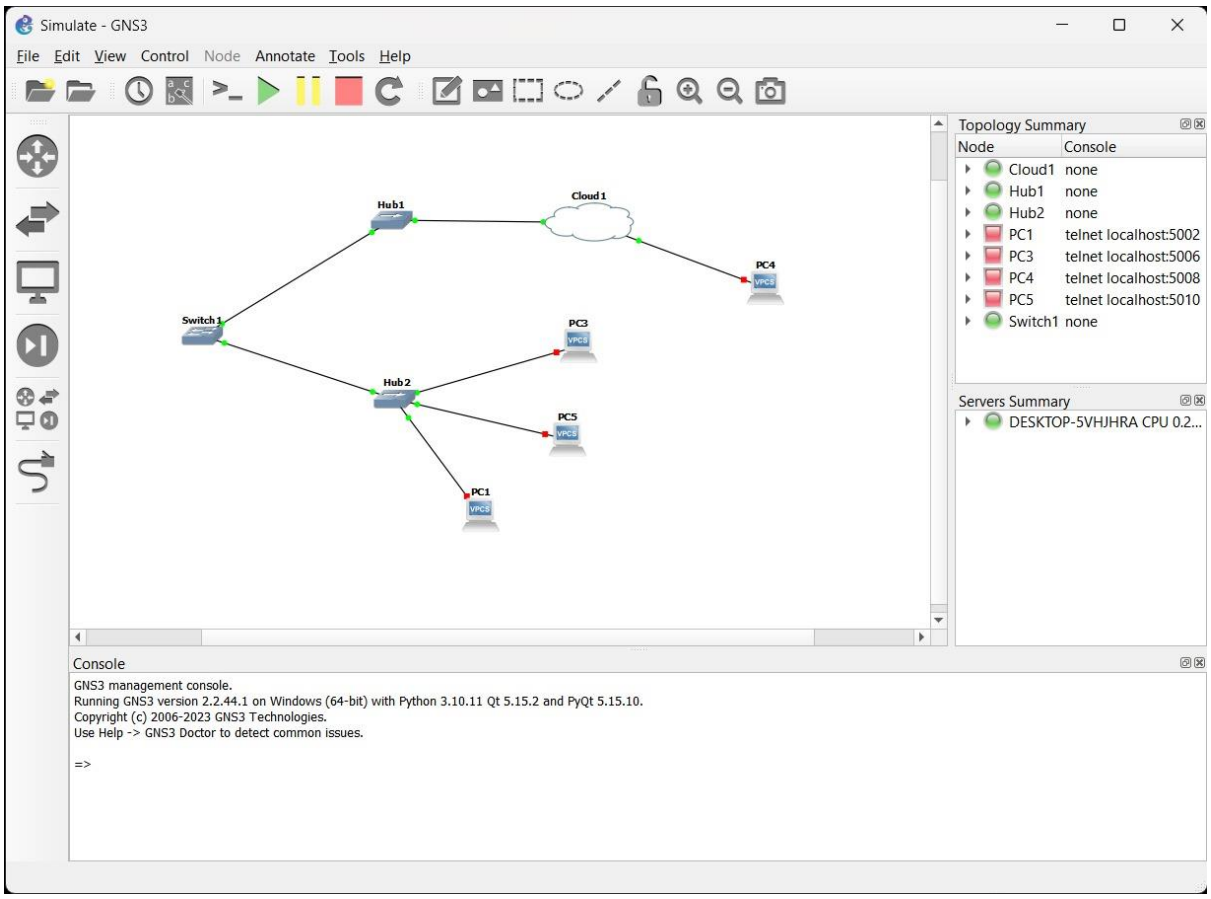
```
}

State[] state = new State[MAX_NODES];

for (int i = 0; i < n; i++) {
   state[i] = new State();
   state[i].predecessor = -1;
   state[i].length = INFINITY;
   state[i].label = false;
}

state[t].length = 0;
state[t].label = true;

int k = t;
do {
   // Update tentative distances
   for (int i = 0; i < n; i++) {
      if (graph[k][i] != 0 && !state[i].label) {
         if (state[k].length + graph[k][i] < state[i].length) {
            state[i].predecessor = k;
            state[i].length = state[k].length + graph[k][i];
         }
      }
   }

   // Find the tentatively labeled node with the smallest label
   k = 0;
   int min = INFINITY;
   for (int i = 0; i < n; i++) {
      if (!state[i].label && state[i].length < min) {
         min = state[i].length;
         k = i;
      }
   }
   // Mark the node as permanent
   state[k].label = true;
} while (k != s);
// Copy the path into the output array
int i = 0;
k = s;
do {
   path[i++] = k;
   k = state[k].predecessor;
```

```
        } while (k >= 0);
        // Calculate and return the minimum cost
        mincost[0] = state[s].length;
    }
}
```

OUTPUT




Logical
Physical
x: 724, y: 594
[Root]
Hub-PT
Hub0
PC-PT
PC4
PC-PT
PC5
PC-PT
PC3
PC-PT
PC0
PC-PT
PC1
PC-PT
PC2


Simulate - GNS3
File Edit View Control Node Annotate Tools Help
Hub1
Cloud1
PC4
Switch1
PC3
Hub2
PC5
PC1
Topology Summary
Node
Console
Cloud1 none
Hub1 none
Hub2 none
PC1 telnet localhost:5002
PC3 telnet localhost:5006
PC4 telnet localhost:5008
PC5 telnet localhost:5010
Switch1 none
Servers Summary
DESKTOP-5VHJHRA CPU 0.2...
Console
GNS3 management console.
Running GNS3 version 2.2.44.1 on Windows (64-bit) with Python 3.10.11 Qt 5.15.2 and PyQt 5.15.10.
Copyright (c) 2006-2023 GNS3 Technologies.
Use Help -> GNS3 Doctor to detect common issues.
=>