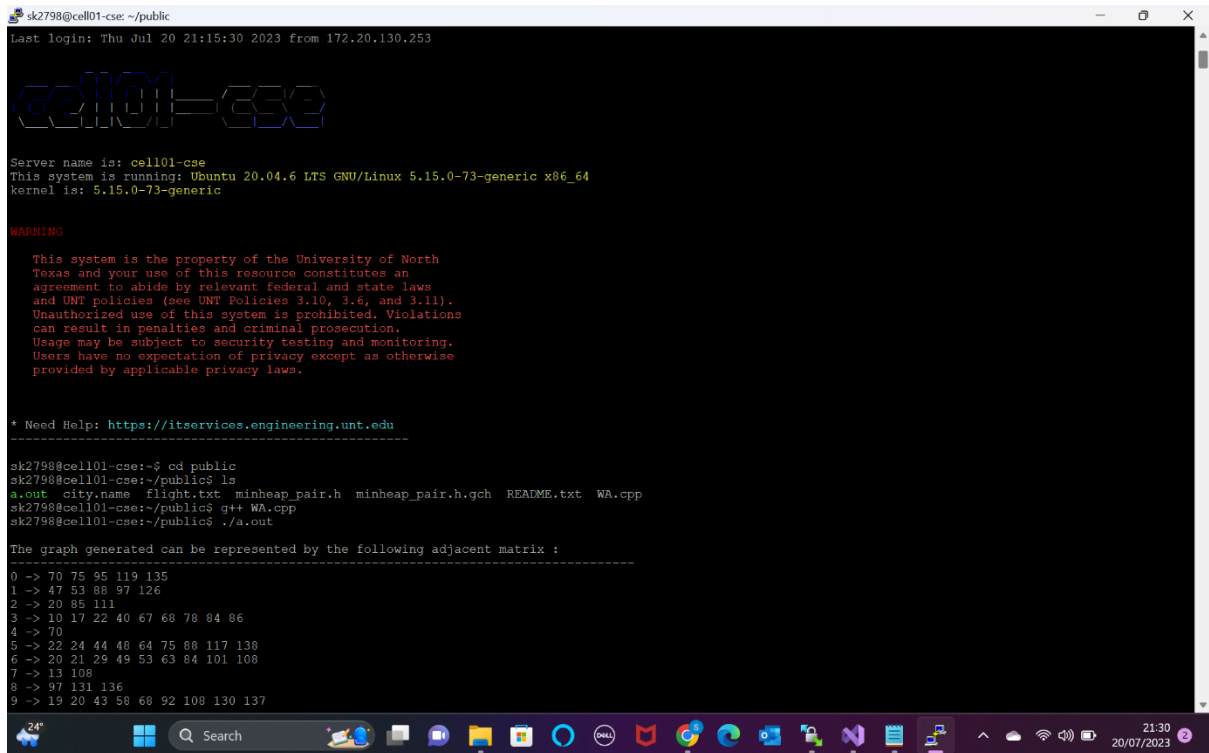


Assignment 4

Code Implementation:

1. Compilation Process:

The instructions that are used to compile the code are:



```
sk2798@cell01-cse: ~/.public
Last login: Thu Jul 20 21:15:30 2023 from 172.20.130.253

cell01-cse

Server name is: cell01-cse
This system is running: Ubuntu 20.04.6 LTS GNU/Linux 5.15.0-73-generic x86_64
kernel is: 5.15.0-73-generic

WARNING

This system is the property of the University of North
Texas and your use of this resource constitutes an
agreement to abide by relevant federal and state laws
and UNT policies (see UNT Policies 3.10, 3.6, and 3.11).
Unauthorized use of this system is prohibited. Violations
can result in penalties and criminal prosecution.
Usage may be subject to security testing and monitoring.
Users have no expectation of privacy except as otherwise
provided by applicable privacy laws.

* Need Help: https://itservices.engineering.unt.edu
-----
sk2798@cell01-cse:~$ cd public
sk2798@cell01-cse:~/public$ ls
a.out  city.name  flight.txt  minheap_pair.h  minheap_pair.h.gch  README.txt  WA.cpp
sk2798@cell01-cse:~/public$ g++ WA.cpp
sk2798@cell01-cse:~/public$ ./a.out

The graph generated can be represented by the following adjacent matrix :
-----
0 -> 70 75 95 119 135
1 -> 47 53 88 97 126
2 -> 20 85 111
3 -> 10 17 22 40 67 68 78 84 86
4 -> 70
5 -> 22 24 44 48 64 75 88 117 138
6 -> 20 21 29 49 53 63 84 101 108
7 -> 13 108
8 -> 97 131 136
9 -> 19 20 43 58 68 92 108 130 137
```

- `>>cd public`-command is used to change the current directory to the public directory. By doing this, you can view and interact with files in the "public" directory without constantly giving the complete path.
- `>>ls`-It will list all the files present in the public folder.
- `>>g++ WA.cpp`- is used to compile the code
- `>>./a.out`- is used to run the code

2. Code Description:

The `routeSearch_2` function implements Task 2, which seeks to determine the route from city "A" to city "D" through cities "B" and "C" with the fewest connections. The graph is the class that represents the graph, and `city_A`, `city_B`, `city_C`, and `city_D` are integer IDs of the cities in the graph. The function is declared with the signature `void routeSearch_2(Graph graph, int city_A, city_B, city_C, and city_D)`. The function begins by setting up a few variables:

The number of cities in the graph overall, as determined by `graph.get()`.

The Dijkstra algorithm uses the following arrays to hold the shortest distances and parent nodes from city "A" to cities "B" and "C," accordingly.

`d_CD[n]`, `p_CD[n]`: Arrays to hold, throughout the Dijkstra algorithm, the shortest distances and parent nodes from cities "B" and "C" to city "D," respectively.

The ultimate pathways from city "A" to cities "B" and "C," and from cities "B" and "C" to city "D," are each stored in a vector, ans_AB and ans_CD. In order to determine the shortest routes from city "A" to cities "B" and "C" (using Dijkstra(graph, city_A, d_AB, p_AB)) and from cities "B" and "C" to city "D" (using Dijkstra(graph, city_D, d_CD, p_CD)), the function makes two calls to the Dijkstra function. There may be routes from city "A" to city "B" and from city "B" and "C" to city "D," respectively, if the two shortest distances, d_AB[city_B] and d_CD[city_C], are not equal to INT_MAX.

After that, the method reconstructs the routes from city "A" to city "B" and from cities "B" and "C" to city "D" using the parent arrays p_AB and p_CD.

The two paths (ans_AB and ans_CD) are combined into the final route by inserting the items from ans_CD after ans_AB once the paths have been rebuilt. The final output of the function includes the total number of connections along with the ultimate route from city "A" to city "D" via cities "B" and "C." If any of the shortest distances, d_AB[city_B] or d_CD[city_C], is INT_MAX, then there isn't a path connecting city "A" and city "D" via cities "B" and "C." In this situation, the function outputs "No such route."

3. Output:

```

sk2798@cel101-cse: ~/public
Riyadh, Saudi Arabia: 93
Rome, Italy: 20
San Francisco, United States: 34
San Jose, Costa Rica: 129
San Juan, Puerto Rico: 35
Santiago, Chile: 86
Santo Domingo, Dominican Republic: 121
Sao Paulo, Brazil: 35
Seattle, United States: 97
Seoul, South Korea: 1
Shanghai, People's Republic of China: 19
Shenzhen, People's Republic of China: 44
Singapore, Singapore: 16
Sofia, Bulgaria: 107
St. Louis, United States: 99
St. Petersburg, Russia: 11
Stockholm, Sweden: 36
Sydney, Australia: 18
Taipei, Taiwan: 27
Tallinn, Estonia: 91
Tehran, Iran: 120
Tel Aviv, Israel: 23
Tianjin, People's Republic of China: 123
Tokyo, Japan: 2
Toronto, Canada: 47
Tunis, Tunisia: 128
Vancouver, Canada: 51
Vienna, Austria: 22
Vilnius, Lithuania: 111
Warsaw, Poland: 59
Washington, United States: 78
Wellington, New Zealand: 100
White Plains, United States: 43
Winston-Salem, United States: 119
Zagreb, Croatia: 74
Zurich, Switzerland: 8
Please choose the type of the questions:
1: From city 'A' to city 'B' with less than x connections?
2: Route with the smallest number of connections from city 'A' to city 'D' through city 'B' and 'C'?
2
Please enter the city A: 78
Please enter the city D: 109
Please enter the city B: 77
Please enter the city C: 44
Route from City 78 to Cities 109 through Cities 77 and 44: city77 -> city44 -> city78 -> city109
Total connections: 2
sk2798@cel101-cse:~/public$ ./a.out
The graph generated can be represented by the following adjacent matrix :
  
```

In this case, the shortest route from city 78 to city 109 goes through cities 77 and 44, with a total of 2 connections between the cities. The output provides a clear representation of the route and the number of connections taken to reach the destination city 109 from the starting city 78 through cities 77 and 44. This line shows the path that travels from city 78 to city 109, and it specifies that cities 77 and 44 are on the path. The order of the cities along the route is depicted by this line. The path departs from city 77 and travels to city 44, city 78, and lastly city 109.

```
sk2798@cel101-cse: ~/public
San Jose, Costa Rica: 129
San Juan, Puerto Rico: 35
Santiago, Chile: 86
Santo Domingo, Dominican Republic: 121
Sao Paulo, Brazil: 33
Seattle, United States: 97
Seoul, South Korea: 1
Shanghai, People's Republic of China: 19
Shenzhen, People's Republic of China: 44
Singapore, Singapore: 16
Sofia, Bulgaria: 107
St. Louis, United States: 99
St. Petersburg, Russia: 11
Stockholm, Sweden: 36
Sydney, Australia: 18
Taipei, Taiwan: 27
Tallinn, Estonia: 91
Tehran, Iran: 120
Tel Aviv, Israel: 23
Tianjin, People's Republic of China: 123
Tokyo, Japan: 2
Toronto, Canada: 47
Tunis, Tunisia: 128
Vancouver, Canada: 51
Vienna, Austria: 22
Vilnius, Lithuania: 111
Warsaw, Poland: 59
Washington, United States: 78
Wellington, New Zealand: 100
White Plains, United States: 43
Winston-Salem, United States: 119
Zagreb, Croatia: 74
Zurich, Switzerland: 8
Please choose the type of the questions:
1: From city 'A' to city 'B' with less than x connections?
2: Route with the smallest number of connections from city 'A' to city 'D' through city 'B' and 'C'?
2
Please enter the city A: 45
Please enter the city D: 7
Please enter the city B: 46
Please enter the city C: 40
Route from City 45 to Cities 7 through Cities 46 and 40: city46 -> city40 -> city3 -> city60 -> city13 -> city7
Total connections: 4
sk2798@cel101-cse:~/public$ ./a.out

The graph generated can be represented by the following adjacent matrix :
-----
0 -> 70 75 95 119 135
1 -> 47 53 88 97 126
```

In these two screenshots, the shortest distance obtained is 4 connections and for the other connections route is not found.

```
sk2798@cel101-cse: ~/public
Riyadh, Saudi Arabia: 93
Rome, Italy: 20
San Francisco, United States: 34
San Jose, Costa Rica: 129
San Juan, Puerto Rico: 35
Santiago, Chile: 86
Santo Domingo, Dominican Republic: 121
Sao Paulo, Brazil: 33
Seattle, United States: 97
Seoul, South Korea: 1
Shanghai, People's Republic of China: 19
Shenzhen, People's Republic of China: 44
Singapore, Singapore: 16
Sofia, Bulgaria: 107
St. Louis, United States: 99
St. Petersburg, Russia: 11
Stockholm, Sweden: 36
Sydney, Australia: 18
Taipei, Taiwan: 27
Tallinn, Estonia: 91
Tehran, Iran: 120
Tel Aviv, Israel: 23
Tianjin, People's Republic of China: 123
Tokyo, Japan: 2
Toronto, Canada: 47
Tunis, Tunisia: 128
Vancouver, Canada: 51
Vienna, Austria: 22
Vilnius, Lithuania: 111
Warsaw, Poland: 59
Washington, United States: 78
Wellington, New Zealand: 100
White Plains, United States: 43
Winston-Salem, United States: 119
Zagreb, Croatia: 74
Zurich, Switzerland: 8
Please choose the type of the questions:
1: From city 'A' to city 'B' with less than x connections?
2: Route with the smallest number of connections from city 'A' to city 'D' through city 'B' and 'C'?
2
Please enter the city A: 43
Please enter the city D: 28
Please enter the city B: 69
Please enter the city C: 59
No such route.
sk2798@cel101-cse:~/public$ ./a.out

The graph generated can be represented by the following adjacent matrix :
-----
```