**7205 HW5**

**Name: Xuebao Zhao NUID: 002108354**

**Q1:**

**Undirected graph drawing**

Select 16 buildings which makes the graph has four buildings from each one of the following numbering ranges:

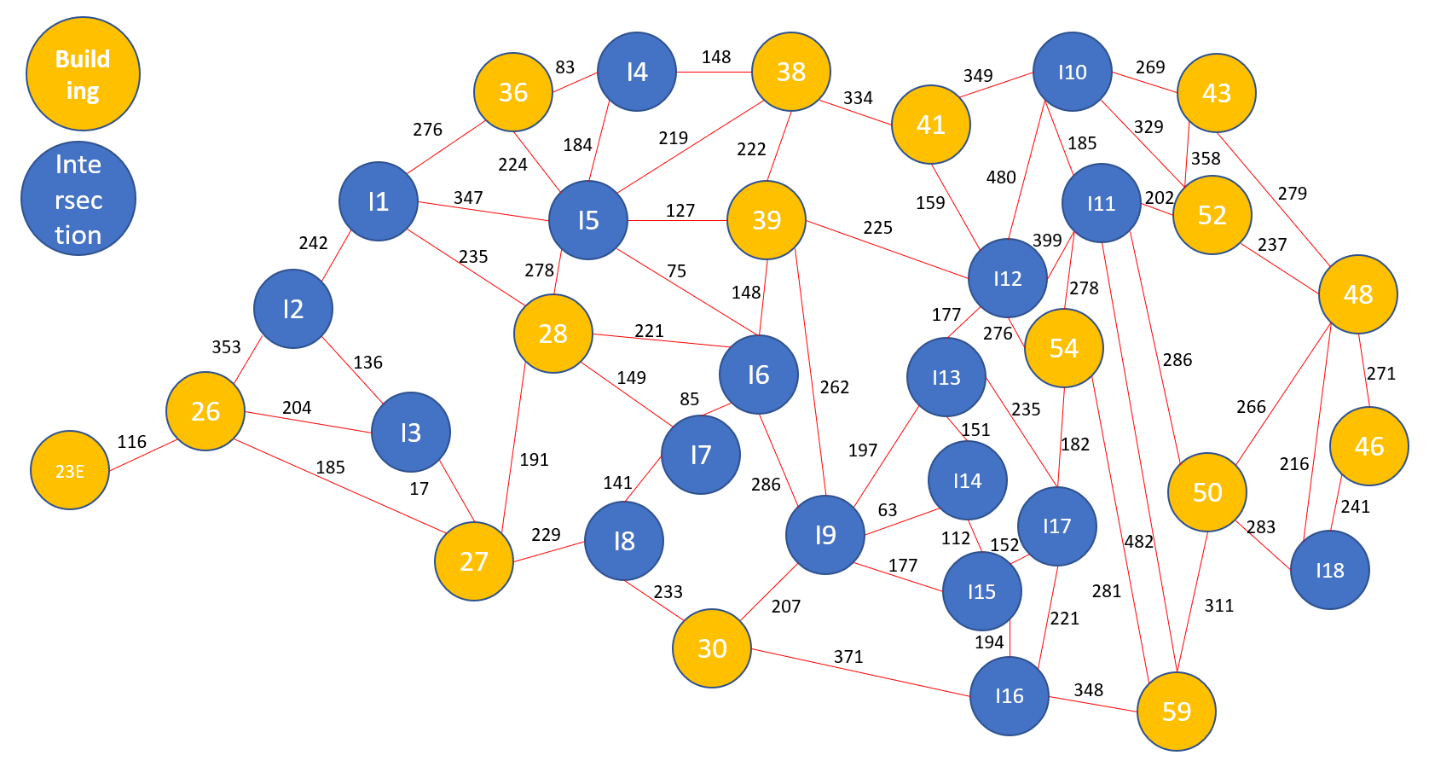
20’s: 23, 26, 27, 28

30’s: 30, 36, 38, 39

40’s: 41, 43, 46, 48

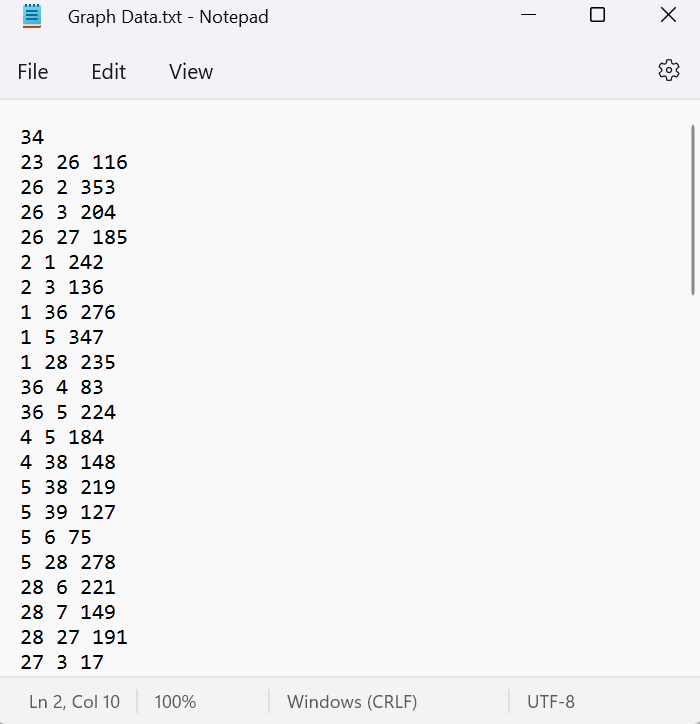
50’s: 50, 52, 54, 59

Use yellow color to mark the buildings and blue color to mark the intersections. And the undirected graph is drawn as follows:



**Create a text file**

The text file includes the total number of vertices followed by the data of the graph edges. For each edge, provide its <vertex1> <vertex2> <distance>. Because the numbers of buildings are all greater than 20, I use the number under 20 to the intersections which making sure that they are different than the used building numbers. And the text file is as follows:



**Store the vertices and edges**

The program reads the ‘.txt’ file and use an array to map the program indices to the user‐friendly building and intersections numbers as stored in the text file. The first number in the file is the total number of the vertices. The remaining integers in groups of three represent an edge. When the program reads the first two integers in a group, first determines whether it is a building or an intersection. Then the program determines whether the vertex has been recorded in the array. If it is already in the array, record the index. Else store the numbers of buildings in the first fifteen positions of the array and store the numbers of intersections in the rest of the array. Then record the index.

Text

Description automatically generated

表格

描述已自动生成

Store the edges in a two-dimensional array ‘edge’, such as ‘edge[i][j]’ represents distance from ‘vertice I’ to ‘vertice j’

Text

Description automatically generated

**Dijkstra**

When calculating the shortest path in the graph by Dijkstra, the starting point ‘start’ needs to be specified.

Furthermore, three arrays P, S, and D are introduced. The role of P is to record the vertices for which the shortest path has been found (and the corresponding shortest path length), while S is to record the vertices for which the shortest path has not been found. And D is to record the distance from the vertex to the starting point).

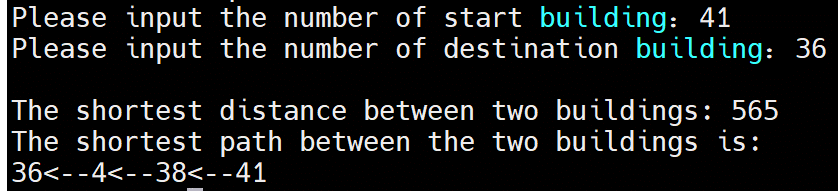
Initially, there is only the starting point ‘start’ in P; there are vertices other than ‘start’ in S, and the path of the vertices in D is "the path from the starting point s to this vertex". Then, find the vertex with the shortest path from S and add it to P; then, update the vertex in S and the path corresponding to the vertex. Then, find the vertex with the shortest path from S and add it to P; then, update the vertex in S and the path corresponding to the vertex. ... repeat this operation until all vertices have been traversed.

Text

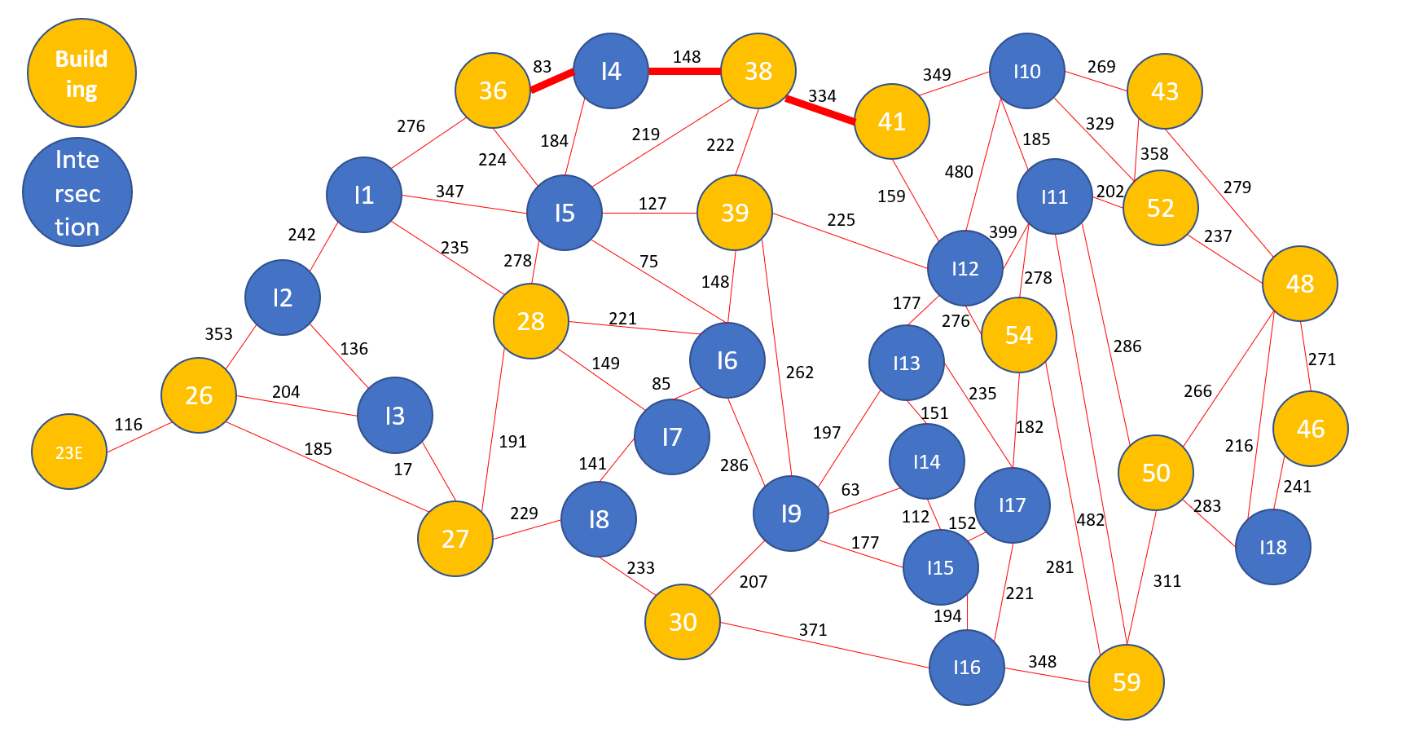
Description automatically generated

**Result**

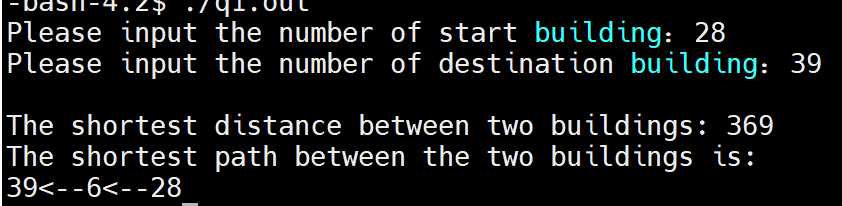
**Case1: 41 -> 36**

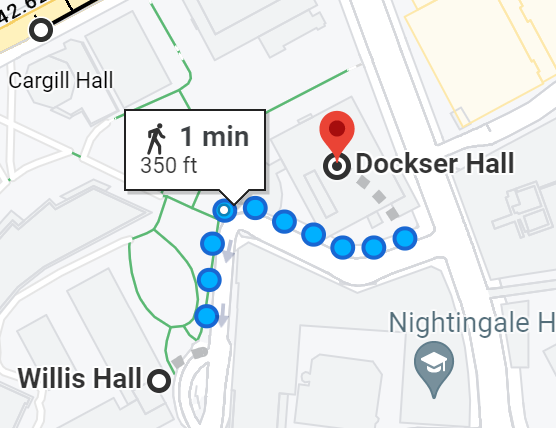
****

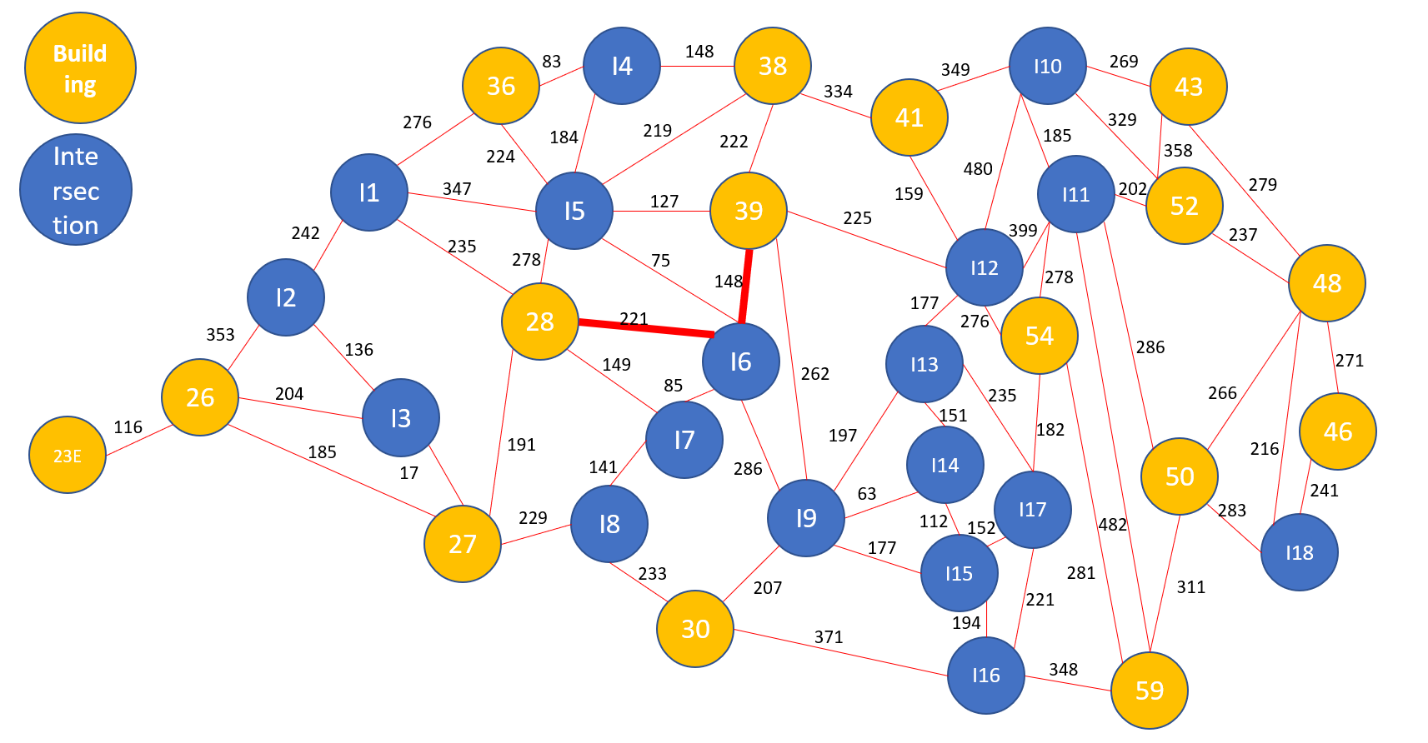
****

****

**Case2: 28 -> 39**

****

****

****