Wenyue Wang, Shushu Zhao

- This assignment we include several .c files and .h files. For maze generation, we have created a structure for node and linked list which are included in header file. For maze solve, we created a queue structure and save several sub functions in queue.c file. Several important features we include here is randomly select which direction should we go to explore the next node, use linked list to store the generated route and write to a output file in forms of adjacent linked list, read in adjacent linked list, and perform BFS by storing partial route in a queue.

Generating maze

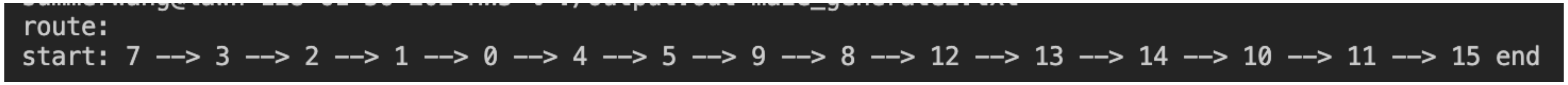
- To generate maze, we introduce a node and linked list structure, where define in a graph, which is a node’s current value and what is its next value as position. Then we apply DFS to generate the route, the general idea is that start from a randomly selected starting point, create an array to store all its neighbor nodes, and test whether they are visited or not. If all of them are visited, then we would backtrack to the former node, if not, we would randomly select a neighbor node that is unvisited. We would repeat this process until we have all points in our visited list and print out the last node as our end point. After that, we write a sub function to write out the size of maze, start and end points as well as the adjacent linked list into an output file for maze solving.

Solving maze

- To solve maze by finding a route from start point to end point, we create queue for BFS. Our general idea is that from the starting point, add every partial path we generate and pop out from the beginning, when we meet the end point, we would terminate the loop and printout the route. Since there is no default container for queue in C, and we need to pop out a partial queue from front and add new path to the queue at the end, so we manually create a struct class.

Evidence of correct operation

- We tested for several sizes of maze for both generating and solving, and the output is as follows:



```
[illegible]
route:
start: 7 --> 3 --> 2 --> 1 --> 0 --> 4 --> 5 --> 9 --> 8 --> 12 --> 13 --> 14 --> 10 --> 11 --> 15 end
```

Division of Labor:

- For this program, two of us contribute equally. Wenyue Wang did generating and Shushu Zhao did solving. And we write structures such as queue and linked list together.