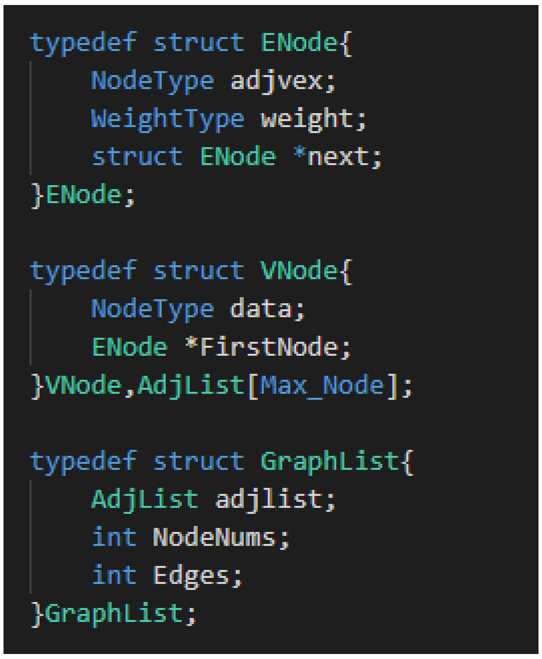
Part 1:

The data structures used in the program are graphs and stacks.

In the program, the graph is realized by the structure, and the stack is realized by the array. The structures are defined as below:



```c
typedef struct ENode{
    NodeType adjvex;
    WeightType weight;
    struct ENode *next;
}ENode;

typedef struct VNode{
    NodeType data;
    ENode *FirstNode;
}VNode,AdjList[Max_Node];

typedef struct GraphList{
    AdjList adjlist;
    int NodeNums;
    int Edges;
}GraphList;
```

Part 2:

Three kinds of algorithms are used in the program:

1. Tarjan
   For the first point I, Mark DFN [I] = LOW [I] = + + Index. Then mark point I visited and traverse all outgoing edges of point I. (Tarjan(I)).
   Define J as the next point,
      if J is not visited, apply Tarjan(J) and update LOW[I].
      if J has been visited and J is on the stack, update LOW[I].
2. Breadth First Search (BFS)
   By using the graph that create in function 1, this method will find the closest vertex to the starting vertex first, then the next closest, and then searching outwards. Finally, all the name of the reachable bus stop will be print out.
3. Dijkstra shortest path
   Define the global variable MAXVEX 100 as the number of initialized nodes.
   Define the global variable INFINITY 999 as an unreachable length.
   The structure Path [MAXVEX] is defined to hold the Path node.
   structure ShortPathTable [MAXVEX] holds the length of the shortest path (D).
   The structure Bus is defined to store the Bus information corresponding to the BusRoute.txt.
   First, determine whether the starting point to the end point is reachable, if it is reachable, then continue to execute.

Execute N times:
(1) Find the one with the shortest path among the points that have not been determined, and then execute the second step.
(2) Determine whether this point is the end point. If so, the path and length will be output directly, and the information of available bus will be output according to the path and exit the loop. If not, proceed to Step 3.
(3) Use this point as an intermediate point to find the path length of all other points that have not been determined and compare the path length saved. If it is less than the previously saved length, update it, otherwise keep the previously saved length. Then continue to step 1.
Part 3:
1. TarJan algorithm is O (m + n).
2. The BFS algorithm is O (m + n).
3. Dijkstra algorithm O(n^2).

So, Function 123 is O (m + n), and Function 4 is O (n^2).