Design manual

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.

Among them, the graph is defined as follows:

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: There are three kinds of algorithms used in the program:

1. Tarjan algorithm is described as follows:

Iterate through each point (skip if it has already been visited) :

For each unaccessed point, do the following:

TarJan(i):

Mark DFN [I] = LOW [I] = + + Index;

I'm going to push I;

Mark accessed;

Traverse all outgoing edges of each point;

Let's define j as the next point;

If the point is not accessed:

TarJan(i);

Update the LOW [I];

If the point has been accessed and is on the stack:

Update the LOW [I];

If DFN[I] = Low [I]:

Push all elements from I to the top of the stack

Results: Every set of elements out of the stack is a strongly connected component

2. Breadth First Search (BFS) algorithm is described as follows:

10, Breadth First Search, referred to as BFS.It is a "blanket" search strategy, finding the closest to the starting vertex first, then the next closest, and then searching outwards.A two-dimensional array is established by reading the name of bus stops in the file busStop.txt, and an adjacency list is created by reading the edge of distance.txt to realize the storage of the directed graph. The breadth first algorithm is used to traverse the reachable points through the starting point of input, and finally the name of the BusStop is output through the subindex of the two-dimensional array.

3. Dijkstra algorithm is described as follows:

Define the global variable MAXVEX 46 as the number of initialized nodes.

Define the global variable INFINITY 999 as an unreachable length.

The Path[MAXVEX] structure is defined to hold the Path node precursor, instantiated as P.

ShortPathTable[MAXVEX] holds the length of the shortest path (D).

The structure Bus is defined to store the Bus information corresponding to the route,Bus Bus []. Data is the Bus information,Bus Bus []. Index1 is the precursor node of this section, and Bus Bus []. Index2 is the successor point of this section

D[v] represents the distance from v0(starting point) to v; P[v] is the precursor node of v, which is used to output the path; final[v] indicates whether the shortest path at this point has been confirmed; 1 means it has been determined.

First, determine whether the starting point to the end point is reachable, if it is reachable, then continue to execute

D[v]=INFINITY, D[v]= g.rc [v0][I], P[v]=0, final[MAXVEX]=0

From the beginning of the first execution, the precursor of initialization starting point P[v0] is -1, and final[v0] is 1

Execute N times:

(1) Find the one with the shortest path among the points that have not been determined, modify it as confirmed, 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 before updating. If it is less than the previously saved length, then update;Otherwise keep, continue with step 1

The third part: complexity analysis

1. TarJan algorithm is O (m + n);

2. The BFS algorithm is O (n ^ 2);

3.Dijkstra algorithm O(n^2);