1.题目名称

公交线路提示

2.代码行数

290行

3.算法思想

excel表格的处理

这里我使用了python的pandas库进行处理，实现了站点以及线路中文与其对应id的映射。

寻找最少转站

这边采用Bfs以及DP这两种算法实现。

使用队列这一数据结构进行广度优先搜索，队列不为空时，取出队首元素，并进行公交车站->公交路线->公交车站的遍历方式，每次遍历时，插入队列，并判断是否符合条件，符合则更新对应数据结构。

为了防止重复插入队列，我使用了visited数组用于跟踪已访问的站点，初始化为未访问状态。

寻找最少经过站

这边采用非递归的Dfs以及DP这两种算法实现。

使用栈这一数据结构进行深度优先搜索，栈不为空时，取出栈顶元素，并进行公交车站->公交路线->公交车站的遍历方式，每次遍历时，压入栈，并判断是否符合条件，符合则更新对应数据结构。

为了防止重复插入栈，我使用了visited数组用于跟踪已访问的站点，初始化为未访问状态。

4.主要/核心函数分析

FindMinTransferRoute

```
vector<string> Graph::FindMinTransferRoute() {
    if (start == -1 || end == -1) {
        cout << "未设置起始站点和目标站点" << endl;
        return {};
    }

    unordered_set<int> visited;
    queue<StationRoad*> q;
    vector<string> Path(7000);
    vector<int> PathCount(7000,0);
    for(auto &it:Path){
        it= BusIdRev[start]+"->";
    }

    StationRoad* startStation = StationSum[start];
    q.push(startStation);
    while (!q.empty()) {    //Bfs算法遍历
        StationRoad* current = q.front();
        q.pop();

```

```
        int currentStation = current->id;
        vector<int> currentRoad = current->Road;

        if (currentStation == end) {
            return Path;
        }

        if (visited.count(currentStation) > 0) {
            continue;
        }

        visited.insert(currentStation);

        for (int bus : currentRoad) {       //站->线
            vector<int> stops = RoadSum[bus]->BusStation;
            for (int stop : stops) {        //线->站
                if (visited.count(stop)<=0) {
                    StationRoad* nextStation = StationSum[stop];
                    q.push(nextStation);
                    if(PathCount[stop]==0 || PathCount[stop]>PathCount[currentStation]+1){         //动规 min(stop经过的路线数 ,当前节点经过的路线数+1)
                        Path[stop]=Path[currentStation]+"("+RoadIdRev[bus]+")"+"->"+ BusIdRev[stop]+"->";
                        PathCount[stop]++;
                    }

                }
            }
        }
    }

    cout << "无法找到起始站点到目标站点的路线" << endl;
    return {};
}
```

这边采用Bfs以及DP这两种算法实现。

状态方程:stop经过的路线长=min(stop经过的路线数 ,当前节点经过的路线数+1)

使用队列这一数据结构进行广度优先搜索，队列不为空时，取出队首元素，并进行公交车站->公交路线->公交车站的遍历方式，每次遍历时，插入队列，并利用状态方程进行判断，符合则更新对应数据结构。

为了防止重复插入队列，我使用了visited数组用于跟踪已访问的站点，初始化为未访问状态。

FindMinThroughRoute

```
vector<string> Graph::FindMinThroughRoute() {
    if (start == -1 || end == -1) {
        cout << "未设置起始站点和目标站点" << endl;
        return {};
    }

    vector<bool> visited(7000, false);
    vector<string> Path(7000, "");
    vector<int> PathCount(7000, 0);
    for (auto &it : Path) {
```

```
        it = BusIdRev[start] + "->";
    }

    stack<int> stationStack;
    stationStack.push(start);

    while (!stationStack.empty()) {          //dfs遍历
        int currentStation = stationStack.top();
        stationStack.pop();

        if (currentStation == end)
            break;

        visited[currentStation] = true;

        StationRoad* current = StationSum[currentStation];
        vector<int> currentRoad = current->Road;

        for (int bus : currentRoad) {
            vector<int> stops = RoadSum[bus]->BusStation;
            for (int stop : stops) {
                if (!visited[stop]) {
                    stationStack.push(stop);
                    int k = PathCount[currentStation] + abs(RoadSum[bus]->nStation[stop] - RoadSum[bus]->nStation[currentStation]);
                    if (PathCount[stop] == 0 || PathCount[stop] > k) { // 动态规划：min(stop经过的站点数,当前节点经过的站点数+到该节点的站点数)
                        Path[stop] = Path[currentStation] + "(" + RoadIdRev[bus] + ")" + "->" + BusIdRev[stop] + "->";
                        PathCount[stop] = k;
                    }
                }
            }
        }
    }

    return Path;
}
```

这边采用非递归的Dfs以及DP这两种算法实现。

状态方程:stop经过的站点数=min(stop经过的站点数,当前节点经过的站点数+到该节点的站点数)

使用栈这一数据结构进行深度优先搜索，栈不为空时，取出栈顶元素，并进行公交车站->公交路线->公交车站的遍历方式，每次遍历时，压入栈，并利用状态方程进行判断，符合则更新对应数据结构。

为了防止重复插入栈，我使用了visited数组用于跟踪已访问的站点，初始化为未访问状态。

5.测试数据(规模,测试次数)

规模:南京任意两个公交站点

测试次数:2

测试用例:自行输入

6.运行结果

test1

```
F:\data_structure\Must\question_6\cmake-build-debug\question_6.exe
Requirement already satisfied: pandas in
c:\users\asusa\appdata\local\programs\python\python310\lib\site-packages
(2.1.1)
Requirement already satisfied: numpy>=1.22.4 in
c:\users\asusa\appdata\local\programs\python\python310\lib\site-packages
(from pandas) (1.23.4)
Requirement already satisfied: python-dateutil>=2.8.2 in
c:\users\asusa\appdata\local\programs\python\python310\lib\site-packages
(from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
c:\users\asusa\appdata\local\programs\python\python310\lib\site-packages
(from pandas) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in
c:\users\asusa\appdata\local\programs\python\python310\lib\site-packages
(from pandas) (2023.3)
Requirement already satisfied: six>=1.5 in
c:\users\asusa\appdata\local\programs\python\python310\lib\site-packages
(from python-dateutil>=2.8.2->pandas) (1.16.0)
Python文件运行完毕
Input your start and end
建康路·夫子庙
玄武门地铁站南
转车次数最少的乘车路线
建康路·夫子庙->(1路(建康路·夫子庙--南堡公园))->玄武门地铁站南
经过站点最少的乘车路线
建康路·夫子庙->(202路(雨花台南大门--灵谷寺公园))->中华门城堡->(Y2路夜间(应天大街·雨花
路--大瓜园))->市口腔医院->(25路(南理工科技园--北崮山))->玄武门地铁站南

进程已结束,退出代码0
```

test2

```
F:\data_structure\Must\question_6\cmake-build-debug\question_6.exe
Requirement already satisfied: pandas in
c:\users\asusa\appdata\local\programs\python\python310\lib\site-packages
(2.1.1)
Requirement already satisfied: numpy>=1.22.4 in
c:\users\asusa\appdata\local\programs\python\python310\lib\site-packages
(from pandas) (1.23.4)
Requirement already satisfied: python-dateutil>=2.8.2 in
c:\users\asusa\appdata\local\programs\python\python310\lib\site-packages
(from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
c:\users\asusa\appdata\local\programs\python\python310\lib\site-packages
(from pandas) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in
c:\users\asusa\appdata\local\programs\python\python310\lib\site-packages
(from pandas) (2023.3)
Requirement already satisfied: six>=1.5 in
c:\users\asusa\appdata\local\programs\python\python310\lib\site-packages
(from python-dateutil>=2.8.2->pandas) (1.16.0)
Python文件运行完毕
Input your start and end
建康路·夫子庙
中华门地铁站
```

```
转车次数最少的乘车路线
建康路·夫子庙->(1路(建康路·夫子庙--南堡公园))->新街口北->(16路(新和源装饰城--南京西站))->中华门地铁站
经过站点最少的乘车路线
建康路·夫子庙->(202路(雨花台南大门--灵谷寺公园))->中华路·长乐路->(Y2路夜间(应天大街·雨花路--大瓜园))->应天大街·雨花路->(2路(中华门地铁站--长途东站))->中华门地铁站

进程已结束,退出代码0
```

7.时间复杂度分析

`FindMinTransferRoute`:该函数的主要时间复杂度为O(m*k*n)，其中m为公交车数量，k为每个公交车平均经过的线路数量。n为每个线路平均经过的站点数量。

`FindMinThroughRoute`:该函数的主要时间复杂度为O(m*k*n)，其中m为公交车数量，k为每个公交车平均经过的线路数量。n为每个线路平均经过的站点数量。

8.结果截屏图片

9.心得体会

该题对于图有了更深入的理解，并且对于图的遍历更加熟悉。同时，这道题贴近现实生活，让我明白可以把一些现实生活中的问题抽象成图的问题。