数据结构大作业2

任务

根据给定的 XM 航空公司的现有航线数据，构造该公司的航线图。在此基础上，设计一个简易的航线规划系统。该系统可根据用户提出的不同需求，返回航班线路的解决方案(诸如航班 ID 顺序表)。

航线数据见project/data/flight-data.csv

	Flight ID	Departure date	Intl/Dome	Flight NO.	Departure airport	Arrival airport	Departure Time	Arrival Time	Airplane ID	Airplane Model	Air fares
0	1	5/5/2017	Dome	346	48	50	5/5/2017 12:20	5/5/2017 15:10	30	1	666
1	2	5/7/2017	Dome	346	48	50	5/7/2017 12:20	5/7/2017 15:10	40	1	1350
2	3	5/8/2017	Dome	346	48	50	5/8/2017 12:20	5/8/2017 15:10	31	1	1656
3	4	5/5/2017	Dome	347	49	48	5/5/2017 21:00	5/5/2017 23:45	59	2	1197
4	5	5/6/2017	Dome	346	48	50	5/6/2017 12:20	5/6/2017 15:10	40	1	1671
...	...	...	...	...	...	...	...	...	...	...	...
2341	2342	5/5/2017	Intl	490	11	49	5/5/2017 6:30	5/5/2017 8:55	106	2	1663
2342	2343	5/6/2017	Intl	490	11	49	5/6/2017 6:30	5/6/2017 8:55	109	2	1371
2343	2344	5/7/2017	Intl	490	11	49	5/7/2017 6:30	5/7/2017 8:55	18	2	1702
2344	2345	5/8/2017	Intl	490	11	49	5/8/2017 6:30	5/8/2017 8:55	102	2	1489
2345	2346	5/6/2017	Dome	65	49	65	5/6/2017 7:10	5/6/2017 9:55	90	2	915

2346 rows × 11 columns

功能要求

1. 给定机场和出发时间，执行深度优先遍历和广度优先遍历，并输出遍历结果(机场编号序列)。遍历时每一个机场只到达一次, 按照出发时间、目标机场编号的顺序遍历。对应函数, query_dfs, query_bfs。
2. 给定机场A和B, 仅限直飞或 1 次中转, 求出从A到B的航线或者输出没有航线。对应函数 query_connectivity。
3. 给定机场A和B以及起始结束时间, 求出从A到B的最小时间开销航线或者输出没有航线。对应函数 query_shortest_path。
4. 给定机场A和B以及起始结束时间, 求出从A到B的最小价格开销航线或者输出没有航线。对应函数 query_minimum_cost_path。
5. 给定机场A和B以及起始结束时间, 仅限直飞或 1 次中转, 求出从A到B的所有航线。对应函数 query_all_paths.

```cpp
// project/include/planner.hpp

struct Planner {
    DFSResult query_dfs(int airport_id, std::string start_time);
    BFSResult query_bfs(int airport_id, std::string start_time);
    ConnectivityResult query_connectivity(int airport_1, int airport_2);
    ShortestPathResult query_shortest_path(
        int airport_1, int airport_2, int start_time, int end_time);
    MinimumCostPathResult query_minimum_cost_path(
        int airport_1, int airport_2, int start_time, int end_time);
    AllPathsResult query_all_paths(
        int airport_1, int airport_2, int start_time, int end_time);
};
```

输入输出

请自行设计输入输出格式，要求可以从命令行读入用户请求并输出查询结果。

(注：实现功能即可，不用美观)

```cpp
// project/src/main.cpp
int main() {
    Planner planner;

    while (true) {
        std::string query;
        std::getline(std::cin, query);
        /*
        code
        */
    }
    return 0;
}
```

实验要求

使用miniSTL中实现的Vector, List实现Planner中的功能(可以额外使用priority_queue, tuple, array).