# 需求分析

1. 按距离最优二点寻路
2. 按时间最优二点寻路
3. 负载均衡寻路
4. 按距离最优途径多点寻路
5. 按时间最优途径多点寻路
6. 鼠标点击查看建筑物信息
7. 搜索附近建筑物，并展现路径及距离
8. 日志文件记录历史
9. 记录当前位置状态
10. 按时间推进
11. 多种交通工具
12. 逻辑位置和物理位置的映射
13. 行进过程中更改路线、导航策略
14. 校区间通行
15. 导入课表根据系统时间导航
16. 模糊搜索
17. 系统时间比例缩放真实时间，并支持修改

输入：

（所有输入都会配有模糊搜索，如果没有匹配地点，会弹出对话框提供若干个近似地点供用户选择）

1. 两点寻路：qt的line edit中输入中文后按pushbutton选择交通工具，最后弹出对话框展示按距离/时间的路径长度和耗时，以供用户选择。
2. 搜索建筑物：line edit+pushbutton
3. 搜素附近：按pushbuttom
4. 多点寻路：pushbutton后弹出窗口，在新对话框的line edit中输入一个建筑物，按pushbutton“继续输入”会清空line edit以便输入下一个建筑物；按pushbutton“开始寻路”，最后弹出对话框展示按距离/时间的路径长度和耗时，以供用户选择。
5. 课程表寻路：按pushbutton
6. 负载均衡寻路：导航终点为食堂时，会展示多个选项供选择

输出：

图形化辅以文字

文件：

1. 地图图片文件
2. 建筑物介绍图片文件
3. 存储建筑物图片的路径的文件
4. 邻接表文件
5. 顶点信息文件
6. 边的信息文件
7. 日志文件
8. 逻辑名映射物理名文件
9. 账号文件（包含课表）

类：

1. Dijkstra：用于二点寻路、范围搜索
2. Ant：用于多点寻路
3. MainWindow：主界面
4. Login：登陆界面
5. Clock：系统时钟
6. Current：当前状态
7. Account：账户信息
8. Vertex：顶点
9. Edge：边
10. Course：课程信息

数据结构：

MainWindow：

1. QVector<QMap<int,Edge\*>> adj\_map：邻接表
2. QVector<Vertex\*> vertices：顶点集
3. QMap<QString,QVector<int>> logic2phy：逻辑名物理名映射

Login：

1. QMap<QString,Account\*> accounts：账户账号和所有账户的映射

Dijkstra：

1. QVector<Vertex\*> dij\_path：返回的路径数组，应包括起点和终点
2. QVector<int> to\_visit：待访问的顶点集合，保持升序
3. QMap<int,int> in\_visited：已访问的顶点集，用map便于直接判断
4. QMap<int,int> parent：存储上一节点
5. QMap<int,QVector<Vertex\*>> paths：返回的搜索附近的所有顶点及对应路径
6. QMap<int,qreal> dis：顶点对应的离起点的距离
7. QMap<int,qreal> time：顶点对应的离终点的距离

Ant:

Login模块：

用户登陆界面，读取账户文件及匹配用户账号密码，登陆成功调用MainWindow

MainWindow模块：

用户操作主界面，centralWidget部分为地图，通过工具栏进行基本的导航操作，辅以弹出对话框来获取用户需求。

输出以图形化界面为主，同时辅以文字描述。

绘制及推进线路使用图形视图框架，由QTimer控制时间。

Dijkstra模块：

采用Dijkstra算法求二点之间最短距离和路径，当导航策略及交通工具不同时能通过将不同的项作为边的权值进行搜索。

同时可用于求出一个范围内所有顶点到起点的距离和路径。

由MainWindow调用。

KMP模块：

采用KMP算法进行子串匹配以进行模糊搜索，由MainWindow调用。

Ant模块：

采用蚁群算法进行指定的途经多点的最短路径计算，通过模拟蚂蚁的信息素近似计算路径。

由MainWindow调用