迭代一设计：

1. 迭代一读入文件“beijingSubway.txt”后需要掌握北京地铁的全部信息，因此读入的文件中应该有路线名称，每条路线上的站点名称，换乘站点名称。读入文件设计如下：第一行为路线名称，换行，第二行为该路线上所有的站点名称，每个名称之间用空格以区分，站点全部写完后换行，下一行又为路线名称，如此循环，即可将北京地铁的全部信息读入程序内部，而读入文件的接口需要根据空格和换行分别将路线名称和站点名称区分存储。
2. 路线名称和站点名称用string[]数组存储，每个路线都有一个标号index，站点名称对应的标号index用map<string, int>存储，路线和站点的包含关系用容数组器vector<int>[]存储，根据路线标号index将路线上的站点标号index存入容器数组中，实现站点和路线的关联。
3. 实现站点和路线的关联后，很容易就能在程序中找出换乘点，如果一个站点标号在两个线路中都存在，它就是换成站点。
4. 在搜寻路径时，bfs栈内的每个站点都需要调用它的邻居关系；为了实现换乘信息的显示，还需要记录点a到第i个邻居的地铁路线标号，等等。更多详情见代码74~112行的注释。
5. 全图遍历使用模拟退火算法，流程如下：  
   1.如果在某站点有未走过的邻居节点，优先走邻居，如果有多个邻居，就随机选择一个邻居；  
   2.如果邻居都走过了，就在全局检索，将所有未走过的节点列出来，并求出到他们的距离；  
   3.将节点按照距离从小到大依次排序，排在前的权重大，排在后的权重小；  
   4.按照一定的算法计算权重值（详情见代码），然后按照权重随机取一个节点作为下一跳站点；  
   5.输入起点终点进行bfs，并在bfs路径上进行访问标记；  
   6.计算步数，记录路径，如果还有未走节点就转到1；  
   7.如果没有未走节点，就将起始点作为终点进行bfs，得到最终的路径结果。
6. 因为在模拟退火算法中取权重是根据距离取的，如果考虑换乘开销只需要在有线路变更时距离+3即可。

迭代二设计：

1. 使用C++轻量级界面开发框架ImGUI实现图形界面
2. 为了实现地图的显示，需要设计一个新的输入文件“beijingSubwayMap.txt”以初始化各个站点的坐标信息，以及各个线路的颜色。
3. 显示遍历的路径只需要调用迭代一中的模拟退火算法得到路径结果，然后在图形化界面中将站点逐一显示即可。乘客已经经过的车站数在每次站点更新时+1即可。

迭代三设计：

1. 修改“beijingSubway.txt”和“beijingSubwayMap.txt”即可添加新的线路和新的站点。
2. 为了实现地图切换，加入新的输入文件“wuhanSubway.txt”（满足迭代一功能需要）和“wuhanSubwayMap.txt” （满足迭代二功能需要），“wuhanSubway.txt”初始化地铁全部信息，“wuhanSubwayMap.txt”用于图形化界面显示。
3. 在图形化界面添加一个按钮控件实现读入文件的切换，点击按钮后切换城市。
4. 大站快车可以通过修改“beijingSubway.txt”将原本一个路线修改为快线，在路径展示时不显示快线中不存在的站点，即只停部分大站。
5. 在图形化界面添加一个时间bar实现时段控制，在特定时段可以停运部分路线。停运该路线之后，无论是点到点的寻路还是全图遍历，都不会走停运路线。
6. 在时段更新后，停运的路线即停运路线中的站点将不再出现在用户可选的起点/目的地列表中。

迭代四设计：

1. 按照c++代码逻辑使用js重写代码实现exe到web程序的移植。
2. （尚未完工）