**项目说明文档**

**数据结构课程设计**

**——电网造价模拟系统**

### 作 者 姓 名 尹诚成

学 号 2351279

指 导 教 师 张 颖

学 院 专 业 计算机科学与技术学院 软件工程



二〇二三 年 十二 月 十三 日

**目录**

[1.项目分析 1](#_bookmark0)

* 1. .[项目背景分析 1](#_bookmark1)
  2. .[项目需求分析 1](#_bookmark2)
  3. .[项目功能分析 1](#_bookmark3)

1.3.1.建立Prim最小生成树功能 1

1.3.2.异常处理功能 1

[2.项目设计 1](#_bookmark12)

* 1. .[数据结构设计 1](#_bookmark13)
  2. .[结构体与类设计 2](#_bookmark14)

2.2.1 ClosestEdge结构体的设计 2

2.2.2 MinTreeNode类的设计 2

2.2.3 PowerGrid结构体的设计 2

* 1. .[项目主体架构设计 2](#_bookmark32)

[3.项目功能实现 3](#_bookmark33)

3.1.项目主体架构的实现 3

3.2.建立Prim最小生成树功能的实现 4

3.3.异常处理功能的实现 5

4.项目测试 5

4.1.输入电网节点个数功能的测试 5

4.2.输入电网之间距离功能的测试 5

4.3.建立Prim最小生成树功能的测试 6

1. **项目分析**
   1. 项目背景分析

随着城市化进程的加快，城市电网的建设成为了一个重要议题。在城市中，小区之间的电网建设需要优化，以确保电力的有效分配和经济效率。因此，开发一套电网建设造价模拟系统，以找到一种最低成本的电网连接方案，显得尤为重要。

* 1. 项目需求分析

基于以上背景分析，本项目需要实现需求如下：

(1)设计一个电网建设造价模拟系统，能够输入多个小区间的电网连接成本；

(2)使用算法优化这些连接，以达到总成本最低，同时保证每个小区间的电网连通；

(3)提供友好的用户界面，方便用户输入数据和查看结果；

(4)系统需要具备良好的稳定性和安全性，能够处理非法输入等异常情况。

* 1. 项目功能分析

本项目旨在通过使用Prim算法建立最小生成树，并考虑用户界面设计，实 现电网建设造价模拟系统。下面对项目的功能进行详细分析。

* + 1. 建立Prim最小生成树功能

Prim 算法是一种用于在带权无向图中构建最小生成树的算法。最小生成树是指覆盖图中所有顶点并使边的总权重最小的树形结构。

Prim 算法适用于本项目，原因如下：

(1)Prim 算法能够确保找到全局最优的最小生成树，即总成本最低的电网构建方案；

(2)相比其他算法如Kruskal算法，Prim算法在处理稠密图时更加高效；

(3)该算法易于与邻接矩阵结合，提高了整个项目的运行效率。

通过Prim 算法，程序将计算出最小成本的电网构建方案。该算法可以有效处理复杂网络，找到连接所有小区所需的最低成本。

* + 1. 异常处理功能

实现异常处理机制，处理用户可能输入的非法信息，确保系统的稳定性和安全性。

1. **项目设计**

2.1.数据结构设计

基于项目分析，在使用Prim算法建立最小生成树的过程中，项目采用邻接矩阵的数据结构来表示图，原因如下：

(1)邻接矩阵适用于稠密图的场景，即图中大部分顶点之间都有边相连，类似于本项目中的城市电网场景；

(2)便于实现Prim算法中的关键操作，如查找最小权重的边；

(3)简化算法的实现，使代码更加易于理解和维护。

2.2.结构体与类设计

2.2.1. ClosestEdge结构体的设计

ClosestEdge 结构体是一个用于辅助Prim算法设计的结构体，用于表示图中每一个顶点中与其相连的权值最小的边。其数据成员定义及含义如下：

int lowestVertex：最小权值边的另一个顶点。

int lowestCost：最小权值边的权值。

2.2.2. MinTreeNode结构体的设计

MinTreeNode 结构体用于储存最小生成树中的所有边的信息，其数据成员定义及含义如下：

char vertex1, vertex2：边的两个顶点。

int cost：边的权值。

2.2.3.PowerGrid 类的设计

PowerGrid 类是电网建设造价模拟系统的核心组成部分，专门用于构建和处理最小生成树，从而实现电网建设的成本最优化。该类采用 Prim 算法，适用于模拟城市小区之间的电网连接，以寻找连接所有小区的最低成本方案。它提供了一系列功能，包括电网结构的初始化、权重设置、最小生成树的构建和打印，以及异常处理机制。通过这个类，用户可以有效地模拟和分析 给定电网结构的最优连接路径，从而在确保电网互通的前提下，最小化建设成本。其数据成员、构造函数、析构函数、公有成员函数定义及含义如下：

int\*\* costMatrix：存储电网中各小区间的连接成本

int vertexNum：顶点数量

int edgeNum：当前边的数量

int maxEdgeNum：电网中边的最大数量

MinTreeNode\* minSpanningTree：表示图中每一个顶点中与其相连的权值最小的边

ClosestEdge\* closestEdges：储存最小生成树中的所有边的信息

PowerGrid(int V)：构造函数，接受顶点数量作为参数，初始化图结构和相关数组

~PowerGrid()：析构函数，释放动态分配的内存资源

bool insertEdge(int v1, int v2, int cost)：设置图中两个顶点间的边权重

void createMinimalSpanningTree(int startNode)：Prim算法，从指定的顶点开始生成最小生成树

void printMinimalSpanningTree()：打印最小生成树

2.3.项目主体架构设计

项目主体架构设计为：

(1)创建电网；

(2)输入电网节点个数；

(3)输入任意两个电网节点之间的距离；

(4)输入建立最小生成树的起始节点；

(5)建立Prim最小生成树；

(6)打印Prim最小生成树；

(7)退出程序。

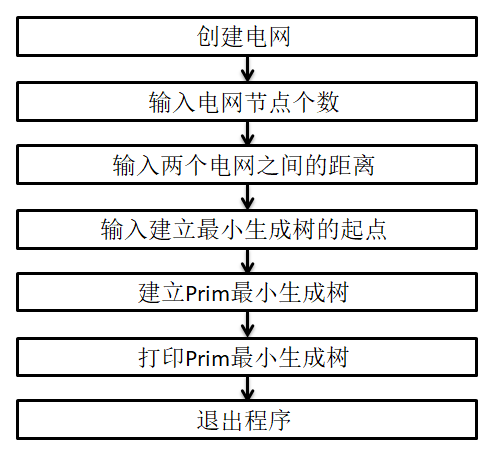


图 2.3 项目主体架构设计流程图

**3.项目功能实现**

3.1.项目主体架构的实现

项目主体架构实现思路为：

(1)创建电网：在程序的开头，通过打印欢迎信息向用户介绍系统，为用户创建电网的过程提供指引；

(2)输入电网节点个数：使用inputInteger函数提示用户输入电网的节点个数。这个函数确保用户输入的是一个有效的整数，并且在指定的范围内。用户输入的数值决定了电网中的小区数量；

(3)输入任意两个电网节点之间的距离：进入一个双层循环，每次循环调用 inputInteger 函数来输入两个节点间的距离或成本。这一步骤对于每一对节 都会重复，直到所有可能的节点对的距离都被输入。距离数据存储在PowerGrid类的实例中；

(4)输入建立最小生成树的起始节点：通过inputInteger函数获取用户输入的起始节点。这个函数处理用户的输入，确保它是一个有效的起始点；

(5)建立 Prim 最小生成树：调用PowerGrid类的createMinimalSpanningTree函数，使用Prim算法从指定的起始节点开始构建最小生成树。该方法负责处理整 个图结构，并找出总成本最低的连接方式；

(6)打印Prim最小生成树：printMinimalSpanningTree方法被调用以打印出构建的最小生成树。这个步骤在屏幕上展示了每个节点间的连接及其相应的成本。

(7)退出程序。

3.2.建立Prim最小生成树功能的实现

建立Prim 最小生成树功能的函数为PowerGrid 类的成员函数createMinimalSpanningTree，建立Prim最小生成树功能实现的思路为：

(1)初始化关键数据结构：在primMST函数开始时，首先初始化关键的数据结构，包括closestEdges数组、minSpanningTree数组。closestEdges数组用于存储到达每个顶点的最小权重边的权重和边的另一个顶点，minSpanningTree数组用于存储所有最小边的信息；

(2)选择起始顶点：将closestEdges 数组，将其lowestVertex成员设置为起始顶点startNode。

(3)构建最小生成树：对于电网中的每个顶点，执行以下步骤：

①寻找closestEdges中最小的边，找到最小边的索引。

②将找到的这条边的信息保存到最小生成树中，包括两个节点和边的权重。

③更新closestEdges：对于所有节点，再次检查与当前选择的最小边相关的边，更新其权重值，如果发现更小的权重，则更新该边的相关信息。

(4)最小生成树结果输出：使用printMinimalSpanningTree方法打印最终的最小生成树。这一步骤通过遍历minSpanningTree数组来实现，展示从起始顶点到其他所有顶点的最低成本路径。

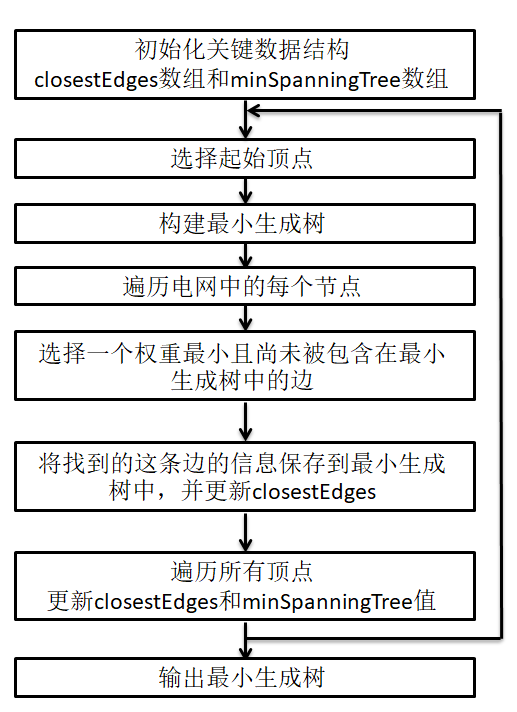


图 3.2 建立Prim最小生成树功能实现流程图

3.3.异常处理功能的实现

在进行MinimumSpanningTree类中私有数据成员等的动态内存申请时，程序使用new(std::nothrow)来尝试分配内存。new(std::nothrow)在分配内存失败时不会引发异常，而是返回一个空指针（NULL或nullptr），代码检查指针是否为空指针，如果为空指针，意味着内存分配失败，这时程序将执行以下操作：

(1)向标准错误流std::cerr输出一条错误消息"Error: Memory allocation failed."，指出内存分配失败；

(2)调用exit函数，返回错误码MEMORY\_ALLOCATION\_ERROR（通过宏定义方式定义为-1），用于指示内存分配错误，并导致程序退出。

**4.项目测试**

4.1.输入电网节点个数功能测试

分别输入超过上下限的整数、浮点数、字符、字符串，可以验证程序对输入非法的情况进行了处理。

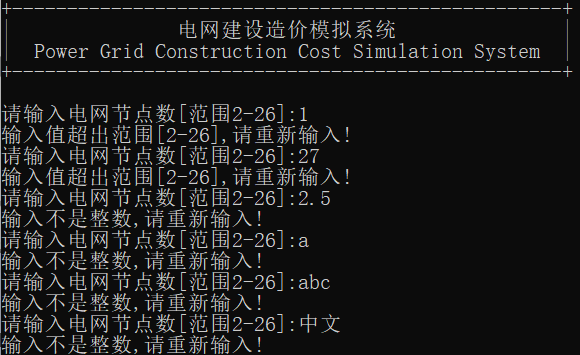


图 4.1.1 输入电网节点个数功能测试（输入非法）

当输入合法时，程序继续运行。

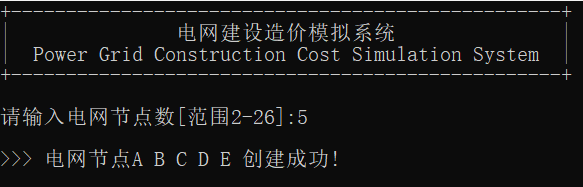


图 4.1.2 输入电网节点个数功能测试（输入合法）

4.2. 输入任意两个电网节点之间的距离功能测试

分别输入超过上下限的整数、浮点数、字符、字符串，可以验证程序对输入非法的情况进行了处理。

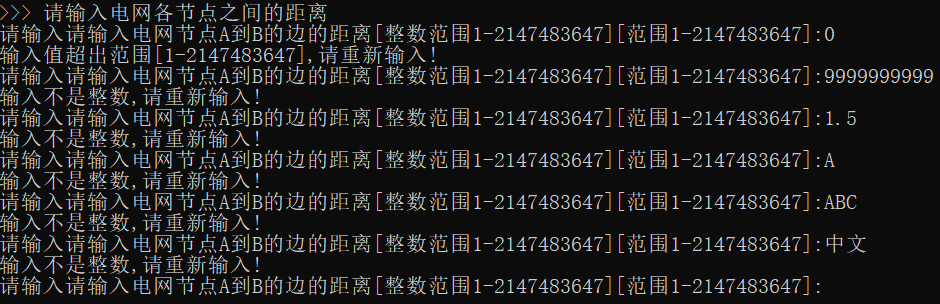


图 4.2.1 输入任意两个电网节点之间距离功能测试（输入非法）

当输入合法时，程序继续运行。

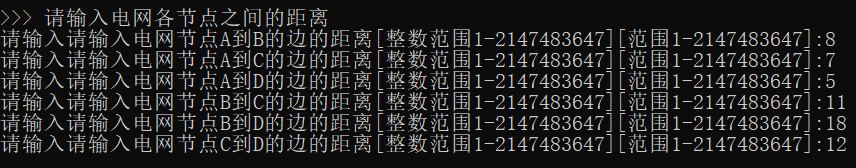


图 4.2.2 输入任意两个电网节点之间距离功能测试（输入合法）

4.3.建立Prim最小生成树功能测试

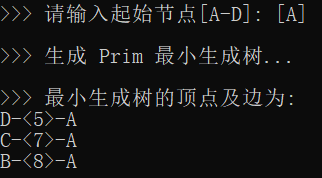


图 4.3.1 建立Prim最小生成树功能测试（起始节点为A）

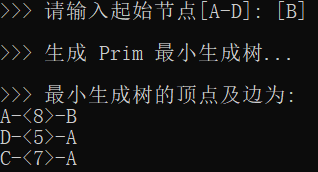


图 4.3.2 建立Prim最小生成树功能测试（起始节点为B）

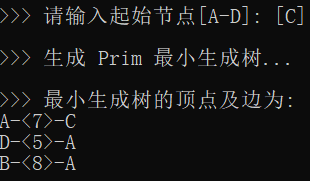


图 4.3.3 建立Prim最小生成树功能测试（起始节点为C）

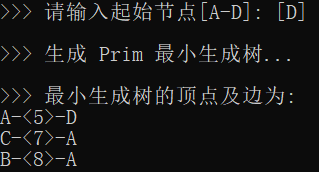


图 4.3.4 建立Prim最小生成树功能测试（起始节点为D）