数据结构 2022秋 Lab6

吴逸昕 19302010013

说明

本次 Lab 中，你将通过 Kruskal 算法实现最小生成树。

提交

截止日期 2022年11月13日 23:59:59，只需提交 `Kruskal.cpp` 文件，将代码打包为 zip，命名为 学号_姓名_Lab6.zip，提交于 elearning。

评分

将基于测试用例通过情况评分，测试时将使用的测试用例均已给出。请提交前务必确保能够通过编译，且不会产生无效的内存引用等问题。

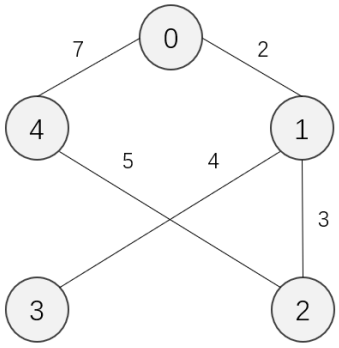
任务：Kruskal 算法实现最小生成树

Edge.cpp

在 `Edge.cpp` 中，定义了图的 “边”。它仅有 `v1` `v2` `len` 三个私有属性与相应的获取方法。无需修改此文件。

`v1` `v2` 为这条边的两个顶点，可以认为某个图中必定含有 0~顶点个数-1 中的所有顶点编号，即：如果该图共有 5 个顶点，则该图中每条边的 `v1` `v2` 仅能为 0、1、2、3、4。

注：不用考虑顶点编号不连续、不从 0 开始等情况；某条边中 `v1` 必定不等于 `v2`，但小于或大于皆可；用于初始化图的第一条边不一定经过顶点 0；可能存在多条边连接两个顶点。



Kruskal.cpp

在 `Kruskal.cpp` 中，需使用 Kruskal 算法，**实现** `findMST()` **函数并返回最小生成树所使用的边**。可自由添加其他辅助函数，但请勿修改已给出的函数接口。

用图的所有边与总顶点数初始化 `Kruskal` 对象。

`findTotalLength()` 函数用于求出最小生成树的总边长，实现已给出，无需修改此函数。

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include "Edge.cpp"

using namespace std;

class Kruskal {
private:
    vector<Edge> edges;     // All edges of the graph
    int vertexCount;        // Total number of vertexes in graph

public:
    Kruskal(vector<Edge> &edges, int vertexCount) {
        this->edges = edges;
        this->vertexCount = vertexCount;
    }

    int findTotalLength() {
        int total = 0;
        for (Edge e: findMST()) {
            total += e.getLen();
        }
        return total;
    }

    vector<Edge> findMST() {
        // TODO

    }
};
```