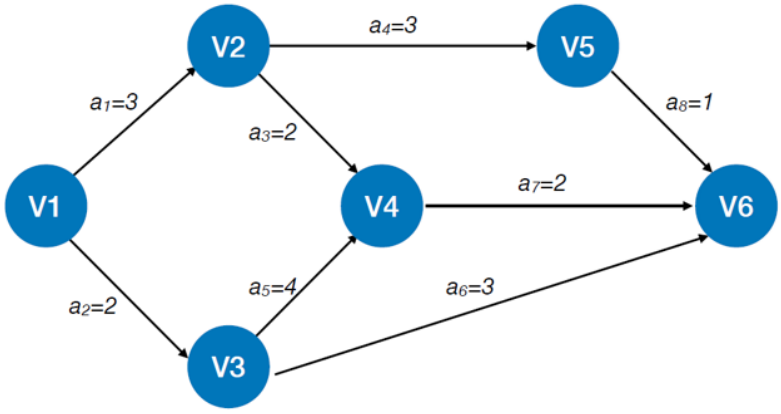
数据结构 2022秋 Lab8

吴逸昕 19302010013

说明

本次 Lab 中，你将通过图的拓扑排序求解关键路径。



提交

截止日期 2022年12月4日 23:59:59，只需提交 `CriticalPath.cpp` 文件，将代码打包为 zip，命名为 学号_姓名_Lab8.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。**请务必确保函数返回的列表元素的顺序与测试用例要求的相符！**

任务：通过拓扑排序求解关键路径

Edge.cpp

在 `Edge.cpp` 中，定义了图的 “边”。无需修改此文件。

```
    int id;         // Edge id
    int start;      // Start node id
    int end;        // End node id
    int weight;     // Edge weight, e.g activity time
```

属性 `id` 为该条边的编号，各条边的 `id` 均在 `[0, 总边数-1]` 范围内。

起止节点编号 `start` 与 `end` 均在 `[0, 总节点数-1]` 范围内，二者不相等，且不用考虑其他特殊情况。

CriticalPath.cpp

说明：

在 `CriticalPath.cpp` 中， **需实现拓扑排序，求出各 事件/活动 的 最早/最迟 发生时间，并找出关键路径** 。可自由添加其他辅助函数，并添加 STL 标准模板库，但请勿修改已给出的函数接口。

可认为用于初始化图的边都是合法的，有向图中不存在回路，两个节点间至多存在 1 条有向边，且不用考虑其他特殊情况。

输出：

在 `nodesEarliestTime()` `nodesLatestTime()` 函数输出时，输出结果需按 **节点 id 从小到大** 排序 。

在 `edgesEarliestTime()` `edgesLatestTime()` `timeDifference()` 函数输出时，输出结果需按 **边 id 从小到大** 排序 。

在 `getCriticalPath()` 函数输出时，输出结果需按 **从起点到终点经过这些边的顺序** 排序。

代码：

用图的所有边与图的总节点个数初始化 `CriticalPath` 对象：

```
    CriticalPath(vector<Edge> &graph, int nodeCount) {
        this->graph = graph;
        this->nodeCount = nodeCount;
    }
```

你共需实现以下 7 个函数（具体输出结构请参考代码文件）：

```
    /**
     * @brief Topological sort.
     * */
    vector<Edge> topologicalSort();
    /**
     * @brief Earliest time for 'events' (nodes) to take place.
     * */
    vector<int> nodesEarliestTime();
    /**
     * @brief Latest time for 'events' (nodes) to take place.
     * */
    vector<int> nodesLatestTime();
    /**
     * @brief Earliest time for 'activities' (edges) to take place.
     * */
    vector<int> edgesEarliestTime();
    /**
     * @brief Latest time for 'activities' (edges) to take place.
     * */
    vector<int> edgesLatestTime();
    /**
     * @brief Time difference (remaining time) for 'activities' (edges).
     * */
    vector<int> timeDifference();
    /**
     * @brief Critical path for graph, in START TO END order.
     * */
    vector<Edge> getCriticalPath();
```