5137169011 陈伟业　Vorringer@sjtu.edu.cn

这个 lab 要求实现 DSDV 路由协议，其中使用不同进程模拟不同路由器。

对于路由器间的通信，也即进程间的通信，我封装了 socket 网络通信代码，为协议的逻辑层提供了三个接口：bind，send，receive，存放在 udp.h/cpp 文件中。逻辑层需要先 bind 端口得到套接字描述符 fd，再使用 send 和 receive 进行信息的发送与接收。

路由器的抽象表示：

```
Map<string, RouterEntry> routerTable;

struct RouterEntry
{
    string dst;          //目的地路由的名字
    string nextHop;      //下一跳
    double cost;         //到目的地的总长度
    int seqNum;          //序列号
};
```

逻辑层：

逻辑层由 router.cpp 构成，rtbl.h/cpp 文件为逻辑层提供服务接口。

主程序由两个线程构成，分别进行 send 和 receive。

Send 线程每 5 秒检测本地的信息表，若有路径信息的变化（设其另一个端点为 P），如路径长度变化、路径损坏等，则将本地路由的序列号（seqNum）+2，更新 table 条目中下一跳（nextHop）为 P 的 cost 为 cost + newLength - oldLength，并使其 seqNum+2。随后向所有连通的邻居发送更新包。

Receive 线程是一个循环不断检测是否有路由给自己发送了消息。若接收到消息，则立即进行路由表的更新。更新的策略是：若更新包中该条目的 seqNum 比当前 table 该条目的 seqNum 大，则使用更新包的数据。若相等，则比较当前 cost 与更新包 cost + dis(当前路由，发送方路由)，选择更小的一方。若小于当前 seqNum，则直接跳过，不进行更新。

ps： 我使用长度 10000 表示损坏的路径，这样可以与正常更新策略统一，便于程序的编写。