1. **项目简述：**项目基于UDP网络协议，实现类似TCP中的可靠传输部分功能，主要由sender和receiver两个部分运行

Sender命令行参数：receiver\_host\_ip receiver\_port filename MWS MSS initialTimeout pDrop seed

Receiver命令行参数: receiver\_port filename MWS

1. **传输报文的格式：**
2. 数据报首部
   1. 判断位（1位）：判断数据段是否为空，空为0非空为1.
   2. Seq位（4位）：序列号
   3. Ack位（4位）：确认号
   4. 标志位（3位）：含有3个标志，分别为SYN、ACK、FIN，每个标志长1位
3. 数据报数据部分
4. **收发控制机制：**
5. 建立连接：建立连接采用的是简单的两次握手，第一次握手是发送方向接收方发送报文，将数据报的SYN标志位置为1，ACK，FIN置为0，并且事先约定一个序列号seq，作为请求。第二次握手是接收方对发送方的回应，SYN置为1，同时ACK也置为一，ack置为事先约定的响应序列号。发送方接收到第二次握手后，代表连接已经建立。
6. 正常收发：在两次握手后，发送方用固定窗口读入文件，并将其保存。然后把保存的文件加入传输中，发往接收方。接收方收到报文，即按照报文的序列号作出回应。发送方收到接收方的回应后，即按照相应序列号取消保存。窗口继续读入。
7. 丢包情况：发送模块采用的是随机发送机制，即人工模拟丢包。如果是丢包情况，会不发送该报文。于是接收方不会返回给发送方回应，该报文就仍被保存，就会在下一次再进行发送。
8. 延迟乱序情况：在接收完所有报文后，接收方根据唯一的seq序列号进行排序，确保报文信息顺序正确。且最后一次报文中ack域设置为data长度，以便准确读入数据
9. 断开连接：断开连接采取的也是两次握手，可以参考建立连接的过程，是针对标识位FIN，ACK和序列号seq，响应号ack做出判断。
10. **关键数据结构或类的说明：**
11. Segement类：报文类，设置报文的格式并提供与报文有关的方法
12. Receiver类：接收方，响应连接并接受sender传输的数据。为了简化测试，输入采用控制台输入的方式，并且需要设置对应的报文长度。过程包括establishConnection()和receiveData();
    1. 存储接受的报文

private HashSet<Segment> buffer = new HashSet<>();

采用HashSet保证延时报文不会重复

* 1. 写入接收的数据

private RandomAccessFile output;

采用RandomAccessFile加快写入大文件速度

1. Logger类：日志类，设置日志记录格式和提供记录方法
2. Sender类：发送方，发起连接和传输从文件读入的数据。为了简化测试，输入采用控制台输入的方式。

主要过程为：

* sender.init();
* sender.establishConnection();
* sender.transferFile();
* sender.close();
  1. toSendSegments待发送的报文，进入此集合的报文会进入发送模块

采用HashSet数据结构，避免重复

* 1. transferingSegment 正在发送（直至收到确认前）的所有报文的备份，与toSendSegments中不重复的报文内容相同，当收到相应报文确认时在这里删除相应备份
  2. sendModule发送模块

发送数据的线程，调用RandomModule的丢包延迟模拟发送报文，

* 1. receiveModule 接收模块

接收数据的线程，在收到报文前阻塞，若是确认报文则删除transfeingSegment中相应报文备份，若是握手报文则加入receiveSegments中等待其他线程调用

* 1. timerSegment 自建类，继承TimerTask，将报文和计时器绑定，通过timer.schedule函数来实现超时重发，重发时将自建类中的报文加入toSendSegments中