README

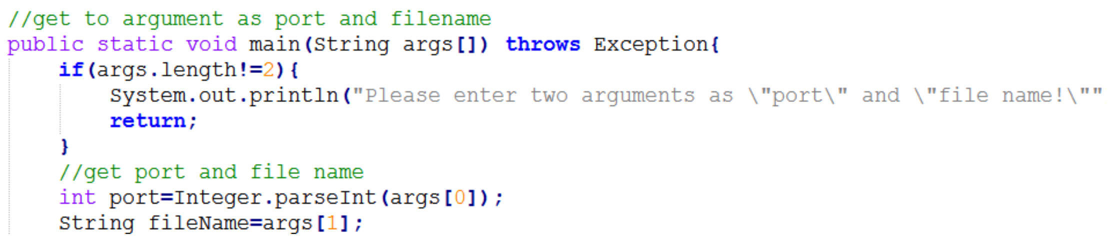
This program is written in java, a Makefile is provide

To start the program, two arguments are needed, first one is port at which UDP starts, 2nd one is filename which is in the same directory with program.

Like java router 8899 conf1.

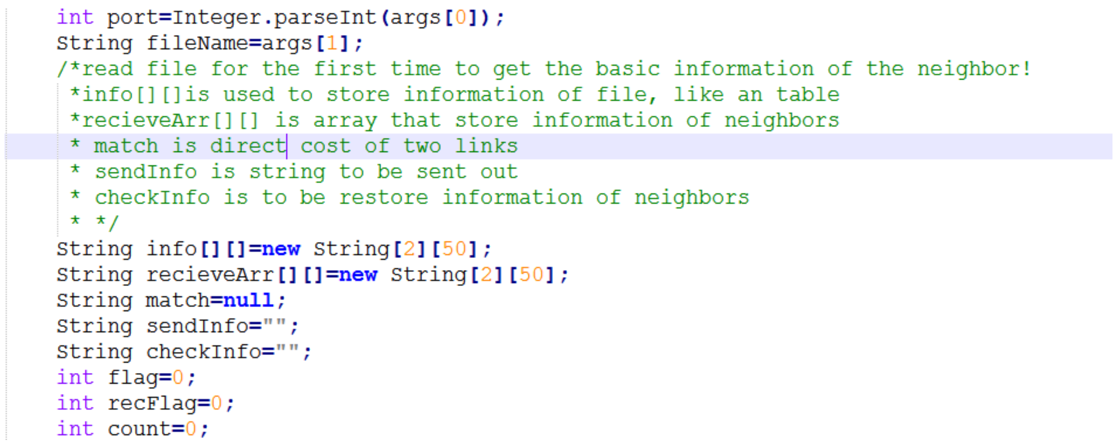
Main structure is listed as following:

1. get arguments from input:



```
//get to argument as port and filename
public static void main(String args[]) throws Exception{
    if(args.length!=2){
        System.out.println("Please enter two arguments as \"port\" and \"file name!\""
        return;
    }
    //get port and file name
    int port=Integer.parseInt(args[0]);
    String fileName=args[1];
```

2. most of the important variables in the program.



```
    int port=Integer.parseInt(args[0]);
    String fileName=args[1];
    /*read file for the first time to get the basic information of the neighbor!
     *info[][]is used to store information of file, like an table
     *recieveArr[][] is array that store information of neighbors
     * match is direct cost of two links
     * sendInfo is string to be sent out
     * checkInfo is to be restore information of neighbors
     * */
    String info[][]=new String[2][50];
    String recieveArr[][]=new String[2][50];
    String match=null;
    String sendInfo="";
    String checkInfo="";
    int flag=0;
    int recFlag=0;
    int count=0;
```

3. read file and store information into array

```
File file = new File(args[1]);
    if (file.isFile() && file.exists()){
    //if file exist, open and read it line by line
    InputStreamReader read = new InputStreamReader(new FileInputStream(file));
    BufferedReader bufferedReader = new BufferedReader(read);
    String line=null;
    String firstLine = bufferedReader.readLine();
    //sotre file in info[][],as the information table
    //and get sendInfo
    while((line = bufferedReader.readLine()) != null) {
        info[0][flag]=line.substring(0,line.indexOf(" "));
        info[1][flag]=line.substring(line.indexOf(" ")+1);
        sendInfo=sendInfo+line+"\n";
        flag++;
    }
    //send infomation to neighbors
        for(int i=0;i<flag;i++){
            InetAddress neighIP = InetAddress.getByName(info[0][i]);
            send(serverSocket,neighIP,sendInfo,port);
        }
read.close();
bufferedReader.close();

    }else{
        System.out.print("file not exist");
    }
```

4. compute the table and update it when changes are made.

```
//compare recieveArr to info[][], get a shortest path
int exist=0;
for(int i=0;i<recFlag-1;i++){
    exist=0;
    for(int j=0;j<flag;j++){
        if(recieveArr[0][i].equals(localhost)){
            exist=1;
            break;
        }
        if(recieveArr[0][i].equals(info[0][j])){
            if((Double.parseDouble(match)+Double.parseDouble(recieveArr[1][i]))<Double.parseDouble(info[1][j])){
                info[1][j]=Double.toString(Double.parseDouble(match)+Double.parseDouble(recieveArr[1][i]));
                exist=1;
                break;
            }else{
                exist=1;
            }
        }
    }
    if(exist==0){
        flag++;
        info[0][flag-1]=recieveArr[0][i];
        info[1][flag-1]=Double.toString(Double.parseDouble(recieveArr[1][i])+Double.parseDouble(match));
    }
}
```

5. this is a function that send information to neighbors.

```
    //function send, to send information to neighbors
    public static void send(DatagramSocket serverSocket,InetAddress IPAddress,String sendInfo,int port) throws IOException{
        byte[] sendData;
        sendData = sendInfo.getBytes();
        DatagramPacket sendPacket =new DatagramPacket(sendData, sendData.length, IPAddress, port);
        serverSocket.send(sendPacket);
    }
}
```