项目二：MapReduce 日志统计

需求分析：

有一份网站访问日志(含有脏数据)，请按照 ip 统计用户访问情况，要求显示结果如下格式所示：

ip starttime startpage lasttime lastpage pagecount timecount

一共 7 个字段，含义解释：

starttime 表示第一次访问时间，只输出时分秒，不要年月日

startpage 表示第一次访问的 url，不需要输出 http://www.baidu.cn/，只需要输出后面的内容

pagecount 表示一共访问多少个页面，不需要去重

timecount 表示一共停留多长时间，单位是分钟。如果不足 1 分钟，向上取整，显示 1 分钟。

思路：(只用了 MapReduce，解析字符串时未用正则表达式)

观察输入的结果，每个用户输出一个一行，意味着按照用户进行分类。参考 MapReduce 的执行原理：map 输入的每一行的原始记录，reduce 每输出一次就是一行结果。在 map 到 reduce 的过程中，框架会对 k2 进行分组。所以，用户可以作为 k2 出现，成为分组。v2 是什么哪？可以让每一行原始记录的 time、page 拼接到一起作为 v2。接下来，是 map 的伪代码：

```
void map(k1,  v1,  context){
//从 v1 中解析出 user、time、page
//k2 就是 user
```

```
//v2 就是 time 和 page 拼接的字符串
}
```

map 函数执行完之后，框架会对输出的<k2,v2>进行分组。
分组后，所有 user1 的<k2,v2>就分到一个组，所有 user2 的<k2,v2>分到一个组。
每个分组调用一次 reduce 函数，即一次 reduce 函数调用的时候，形参就是相同 k2 对应的所有 v2。接下来，看一下 reduce 的伪代码：

```
void reduce(k2,  v2s,  context){
   count = 0;
   int firstTime = Long.MAX_VALUE;
   String firstPage;
   int lastTime = Long.MIN_VALUE;
   String lastPage;

   //page 的次数、time 的差
   for(v2 : v2s){
      count++;
      if(v2.time<firstTime){
         firstTime = v2.time;
         firstPage = v2.page;
      }
      if(v2.time>lastTime){
         lastTime = v2.time;
         lastPage = v2.page;
      }
   }
}
```