如：一个文件大小为10字节，分3个线程来下载，问：每个线程下载多少？

10 / 3 = 3，如果每个线程下载3个，则少1个字节，如下：

线程1：3字节

线程2：3字节

线程3：3字节

三个线程共下载9个字节，少了一个字节，所以每个线程需要下载4个字节，如下：

线程1：4字节

线程2：4字节

线程3：2字节

此结果如何通过公式计算？假设线程数为threadCount，文件大小为fileSize，公式如下：

val remainder = fileSize % threadCount // 计算余数

val average = fileSize / threadCount // 计算平均数

if (remainder > 0) average++ // 如果有余数，则平均数要加1

看到别人的公式为：val average = (fileSize + threadCount – 1) / threadCount 这个公式中fileSize + threadCount – 1其实原理就是等于判断如果fileSize / threadCount有余数则平均数加1是一个原理的。如果是分3个线程下载，如果拿文件大小直接除以3，则有可能平均数之后，会多出余数1或者2，总之余数会比3小1，不可能说余数是3，余数是3的话不就正好每个线程多分配1个字节吗？这是不可能的。假设余数为1，如何让除以3时的平均数多1呢？ 1 + 2 除3就会多1。核心思想就是如何让有余额的计算平均值能多1，假设要分10个线程下载，如果直接除10来计算平均值，则余数是1时，1 + 9再除10，就会多出1。那为什么要减1呢？道理也很简单，如果余数为0，你不减1的话，则计算的结果就不准了，如10字节分10线程下载，每线程分1字节。（10 + 10 – 1）/ 10 = 1，如果不减1的话就等于2了。

按上面公式的话，有些情况很奇怪，如文件大小为4字节，按3个线程下载，按公式算每个线程下载2字节，则第三个线程就不需要下载了，两个线程就够了。这种情况应该都是出现在文件很小的情况吧，不知道字节数多一点的会不会也出现这个情况。

5字节，4个线程下，则分为：2 2 1 0，6字节5线程下，则分为：2 2 2 0 0，按着这个规律，好像是文件大小跟线程数差不多的时候就会出现这种情况，如果文件大小比线程数多的多的话应该是不会出现这样的情况的吧？