对于给定的 N 本书和 M 个学生，每本书的页数已经按升序排列。我们的任务是分配这些书，使得分配给每个学生的最大阅读页数达到最小，并且每个学生需要阅读升序排列上连续的书。输出最小页数。

例子：输入：N=4，页数[] = {12, 34, 67, 90}, M = 2 输出：113

1. 实现思想

算法的实现思想是：对于给定的N本书和M个学生要找到最小页数，分析可知最小页数的范围必定在页数这个数组的最大值以及整个数组的和之中，所以只需要在该范围内使用分治思想进行二分查找即可，判断每个解是否符合分配给M个学生的要求，并最终找到满足的最小页数。（蛮力算法则是找到所有可能的解情况并判断哪个为最小页数，复杂度较二分来说高很多）

算法步骤为：

1.确定二分查找的左边界和右边界。左边界为书中的最大页数，右边界为所有书的总页数。

2.在这个范围内进行二分查找，找到一个中间值 mid。

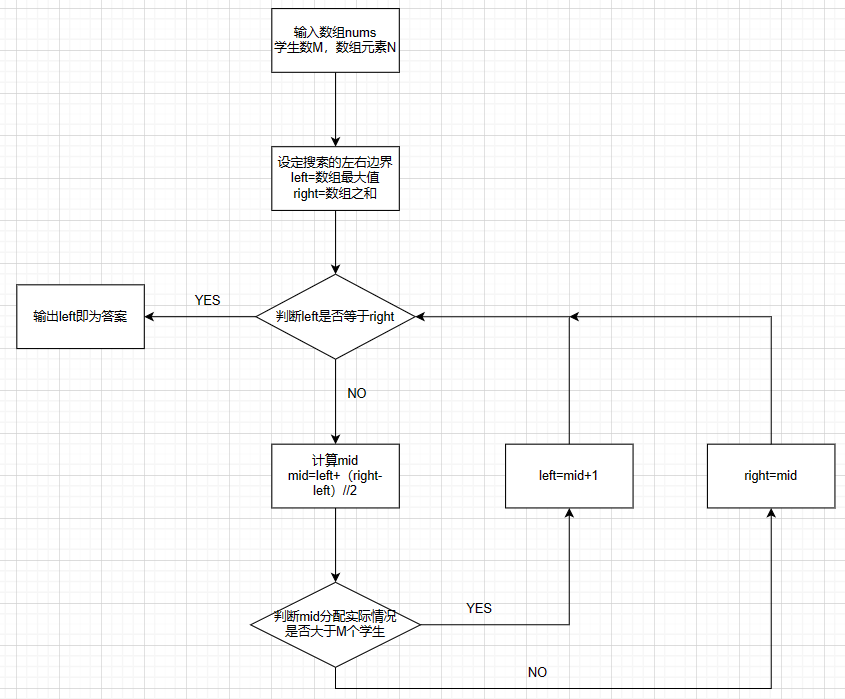
3.对于给定的 mid，模拟分配书的过程：.从第一本书开始，向后分配给学生，直到分配的页数超过 mid。如果超过 mid，就需要分配下一个学生，重置分配的页数，继续分配。继续这个过程，直到所有书都被分配完毕。

4.在模拟分配过程中，统计实际分配的学生数量。

5.根据实际分配的学生数量与给定的学生数量 M 进行比较，来确定二分查找的方向：如果实际分配的学生数量小于等于 M，则表示 mid 值偏大，应该在左半边继续查找。如果实际分配的学生数量大于 M，则表示 mid 值偏小，应该在右半边继续查找。

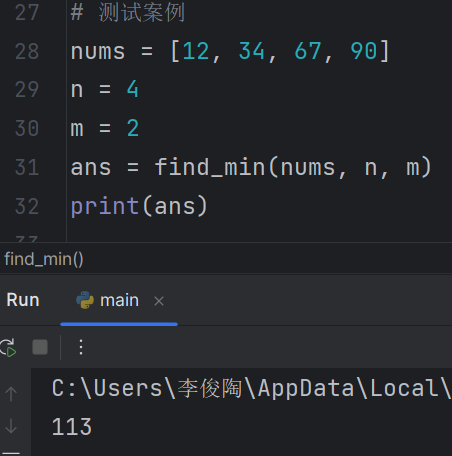
重复步骤 2 到步骤 5，直到左边界和右边界相遇。此时即为最小的最大阅读页数。

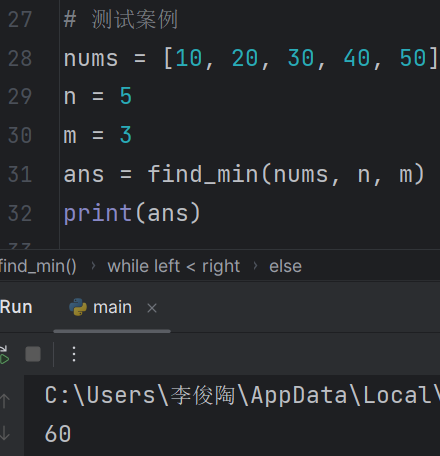
算法流程图如下：

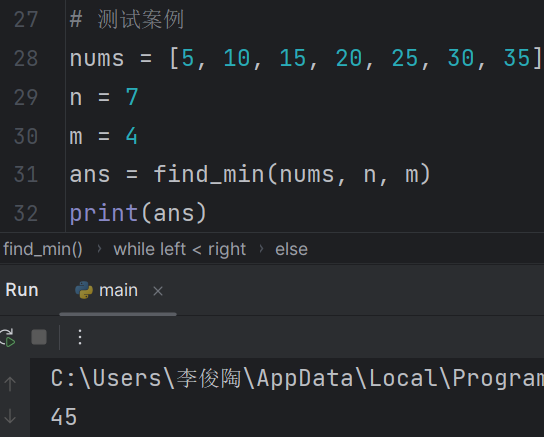


1. 输出截图

除了所给例子之外还自行设置了几个测试数据，检验算法的正确性：

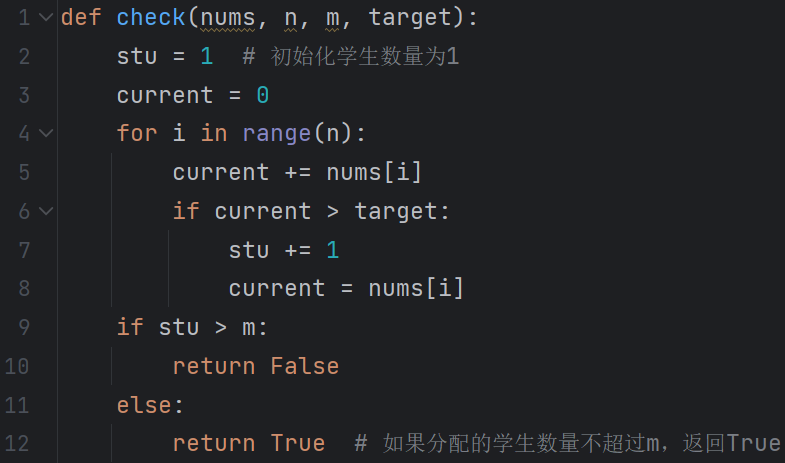






1. 源代码

检查是否符合条件的函数：



寻找最小页数以及测试样例：

