Sample Program #4 (Scholarly Reader) Testing Strategy

I generated 25 test cases total. The first 3 were from the sample, where all three test cases tested a boundary (either add or subtract 1 more page to read and it changes the final answer).

The minimum case for this problem is n = 1, so I generated three test cases where n = 1, where you couldn't read enough pages, you could read exactly the right number and where you could read more than possible. These cases stress test the minimal boundary conditions and array out of bounds issues, since there's only 1 valid index into an array of size 1.

My next three cases used n = 10, and I rearranged my answers to range from 0 to 10, so the full bounds were tested. In these cases I used the same unsorted list as input.

My next six cases were still relatively small but randomly generated. I picked an end answer (# of books) and then counted the number of pages in those books (the smallest numbers), and then made the number of pages I would read one less than that, equal to that and one more than it, since this is precisely the boundary were answers change.

My last ten cases are large cases, with only the last 5 requiring longs. The first five random cases used n = 100000 books with at most 10,000 pages each, randomly generated.

My first two cases that tested longs used n = 100000 with each book being exactly 10^9 pages long and a page limit of 10^{14}-1 and 10^{14}, respectively, testing the absolute max case of the program at the boundary. The last three cases were random.

All of the generated data was made with the functions in mydata.py.

Note: the key idea behind the testing here is that you solve the problem by sorting and summing values going from left to right, so it was important to make target values (maximum # of pages to read) that spanned the whole spectrum along the possible sums.