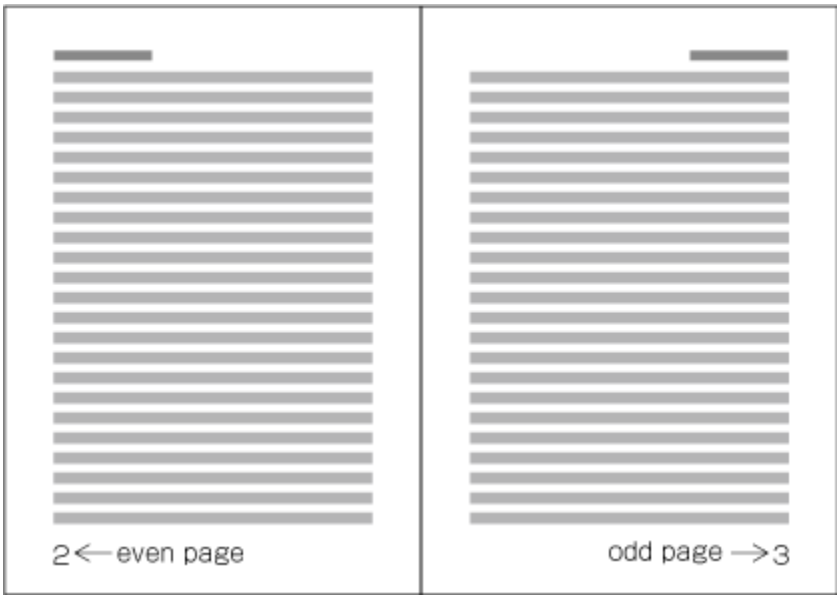
BIT Magazine (TRMAG)

The problem saw a lot of different solutions passing during the contest. The problem has 2 main solutions according to me one which is easy to think hard to code and the other that is hard to think but easy to code (thanks to Shilip to help me have an insight to that solution).



The figure shows that an odd page has the even page at its back.

Easy to think hard to code

The solution that strikes most of the coders initially is DP knapsack where you choose to remove R pages calculate the probabilities of their removal and go ahead calculating the expected total removal and subtract it from the sum of all the pages in the book to optimize you might choose the lesser of the removed pages or left pages in the book so that you can improve on the complexity as per the test case.

The recursive solution turns out to be.

expected(remain_removal,leaves_remain)=(1-p)*expected(remain_removal,leaves_remain-1)+p*(expected(remain_removal-1,leaves_remain-1)+removal);

p is the probability that the page will be removed which is equal to remain_removal/leaves_remain.

Hard to think easy to code

A complete mathematical solution where you calculate the sum of all the printed pages and then you just find the expected sum left by multiplying (leaves-removal)/leaves. The solution works as it considers all the possibilities of removal and not removal as that is the mathematical average of all the states possible considered with equal probabilities.