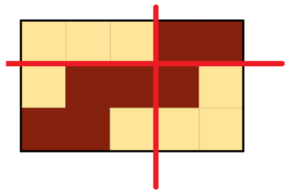
ABC 159E Dividing Chocolate

Given a chocolate bar, in which each square is black or white. You can cut the bar horizontally or vertically, but you must cut all the way through. What is the minimum number of cuts so that every block has or less white squares?

For example, the input below should be cut with 2 cuts:

3 5 4

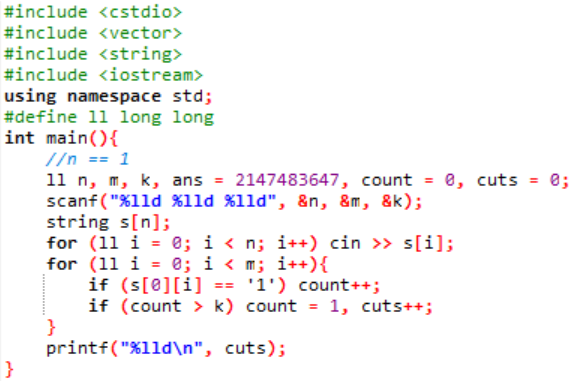
11100

10001

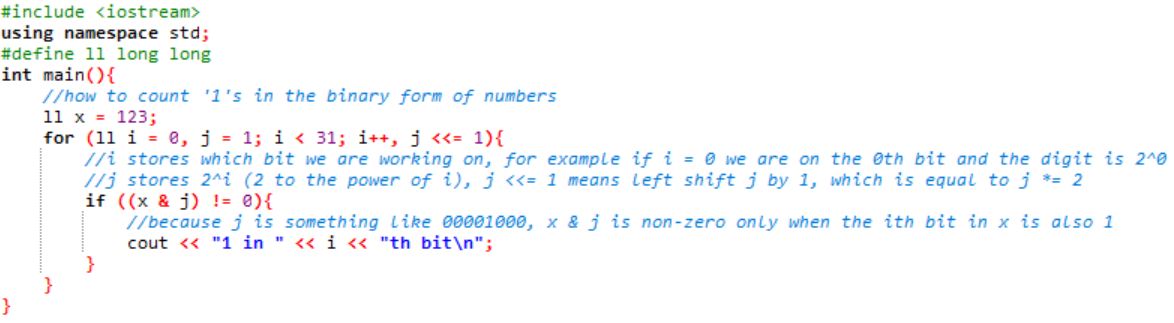
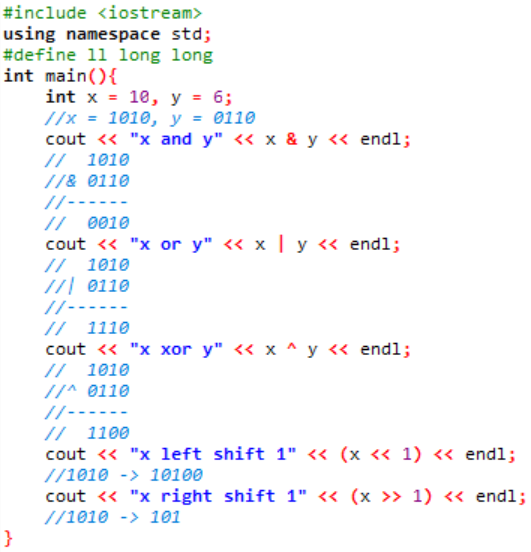
00111

You can see that each block has white squares.

Firstly, let’s consider the case where . We can use the greedy approach to solve this problem. We should always choose to cut where the length of the current block is maximized without having more than white squares. If you choose to cut before that point, the number of white squares in the current block will always stay same or decrease. That means you left 0 or more white squares to future blocks when you could have dealt with them immediately. Therefore, greedy works in this case. This line of thinking can also be applied to other problems, where you think if doing stuff before is always suboptimal. If that’s the case, you can always pick the latest time to do stuff, and that will be optimal.



Now we have to consider the horizontal cuts. We observe that , so a brute force approach may pass. How do we brute force? We can see that there are cutting points (as there are rows), and for each cutting point, we can choose to cut or not cut. Therefore, there are ways to cut horizontally. Moreover, we can apply the same trick above, so we can obtain the minimum number of vertical cuts for each way to cut horizontally in . The total time is , which should pass the time limit.

However, we now face a big problem: how do we implement this? Sure, we can use a recursive function to decide to cut or not to cut for each cutting point, but that would be annoying to code. Instead of that, we can use bitmask! Bitmask is a technique that involves manipulating bits and iterating though stuff. Let’s learn some bitwise operations:

Implementation:

