Background. We consider combinational Boolean logic networks in the form of And-Inverter Graphs (AIGs). An AIG consists of one-input or two-input nodes representing logical conjunction, and edges optionally containing markers indicating logical negation. We define the fanout of a node \( n \), \( \text{fanout}(n) \), as the number of nodes (including internal and output nodes) that node \( n \) directly drives. We further define the maximum fanout of an And-Inverter Graph \( G \), \( \max_{\text{fanout}}(G) \), as the largest fanout of the internal nodes (nodes that are neither inputs nor outputs) in \( G \). We define the size of an AIG as its number of internal nodes (a standalone buffer or inverter also counts as one node). We define the level of an AIG as the maximum number of internal nodes (a standalone buffer or inverter contributes to one level) among all paths from an input to an output.

Task. You are given an input AIG \( G_0 \) and a positive integer number \( \text{MAX}_\text{FO} \), where \( \max_{\text{fanout}}(G_0) > \text{MAX}_\text{FO} \). Please write a computer program to transform \( G_0 \) into a functionally equivalent AIG \( G_1 \) that is legalized with regard to the maximum fanout constraint (i.e., \( \max_{\text{fanout}}(G_1) \leq \text{MAX}_\text{FO} \)), and (2) the quality-of-result (QoR) of \( G_1 \) is maximized (defined next).

Score. The score \( S \) of each testcase is calculated based on the following criteria: (1) The score is zero if \( G_1 \) exceeds the maximum fanout constraint (i.e., \( \max_{\text{fanout}}(G_1) > \text{MAX}_\text{FO} \)); (2) otherwise, the score is computed as the sum of the relative gain of \( G_1 \) in size and level with regard to \( G_0 \), that is, \( S = \frac{\text{size}(G_0)}{\text{size}(G_1)} + \frac{\text{level}(G_0)}{\text{level}(G_1)} \).

We will evaluate each team’s submission using a hidden set of testcases, where we enforce a timeout of 5 minutes per testcase. A team wins a testcase if its submission achieves the highest score among all the submissions for that particular testcase. The final ranking of the teams are ordered by the number of winning testcases that each team receives. In the case of a tie, the teams are further ranked by the total runtime across all testcases.

Example. Consider the following testcase \( G_0 \) with \( \text{MAX}_\text{FO} = 2 \). The output of node \( a \) has a fanout of 3, which exceeds the fanout constraint. A straightforward approach to legalize \( G_0 \) is to add a buffer after node \( a \), as shown in \( G_1 \). Another solution is shown in \( G_1' \), which has better size and level than \( G_1 \).

Format and submission. Please submit your binary or source code. Your program should have a clearly stated interface that takes (1) an input AIG in .blif format, and (2) an integer representing the maximum fanout constraint. Your program should output a .blif file representing your solution. You are encouraged to write a brief summary of your algorithm. You are also encouraged to share your source code with the IWLS community, but it is not required or has any effect on the rating.

Hint: one potential approach can be re-doing eliminate, collapse and kerneling under the maximum fanout constraint. This can help restructure the circuit in a legal way while minimizing the QoR loss.