DESIGN REPORT

1. This program accepts the number of rows and columns of a square matrix as a command-line argument.
2. The current generation, denoted by the currentGen matrix, is populated with 0's and 1's randomly.
3. This input array is then copied from the host to the device and the kernel is called upon to start the operation.
4. The algorithm to find the alive status of the cells is simple and straightforward.
5. The current element is represented by the thread that is working in the kernel method.
6. The current element can be located anywhere in the matrix. So, the different cases have to be considered to calculate the alive neighbors carefully such that the index is not out of bounds.
7. First the four corner elements of the matrix have to be considered. Its neighbors are then calculated.
8. Then the 4 walls, excluding the corners, are considered and their neighbors are calculated.
9. Finally, the middle elements are considered and their 8 neighbors are calculated.
10. By this way, it can be made sure that the index never goes out of bounds.
11. At each step mentioned above while calculating the neighbors, the total alive neighbors of the current node is repetitively summed up.
12. Based on the current node's alive status and the number of alive neighbors, the current node's alive status for the next iteration is calculated according to the rules given in the question.

LIMITATION:

- This program is for square matrices as mentioned in the question.
- The minimum size of the matrix should be 10x10. If not, the program would not run properly.