Question 1: Making use of the Algorithm and the K-Maps depicted, reduce the following function into a Majority Logic function. Each of the three functions (f₁, f₂, f₃) will be only from the Library of K-Map patterns depicted above.

- n = \overline{a}b\overline{c} + \overline{a}b\overline{c} + \overline{a}b\overline{c} + ab\overline{c} + ab\overline{c}
- Function needs to be broken in the form n = Maj (f₁, f₂, f₃)
- Find an admissible pattern for f₁ from the above library.
- For finding f₂, set \( \Psi_1 \) is obtained as follows: if a minterm of n is not a minterm of f₁, add this minterm to \( \Psi_1 \).
- Similarly, for finding f₃, set \( \Psi_0 \) is obtained as follows: if a maxterm of n is not a maxterm of f₁, add this maxterm to \( \Psi_0 \).
- A suitable pattern for f₂ is then determined using new \( \Psi_1 \) and \( \Psi_0 \) (from the above library).
- Furthermore, to determine f₃, \( \Psi_1 \) and \( \Psi_0 \) are updated again as follows: if a minterm (maxterm) of node n is not a minterm (maxterm) of both f₁ and f₂, add this minterm (maxterm) to \( \Psi_1 \) (\( \Psi_0 \)).
Question 2: Perform the AND/OR mapping of the same expression \( n = \overline{a.b.c} + \overline{a.b.c} + \overline{a.b.c} + a.b.c \). Then see the difference in the number of majority gates used for K-map method and AND/OR method.